ODYSSEYPROCESSREUSE: UMA METODOLOGIA PARA ENGENHARIA DE LINHA DE PROCESSOS DE SOFTWARE BASEADA EM COMPONENTES

Eldânae Nogueira Teixeira

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia de Sistemas e Computação.

Orientadora: Cláudia Maria Lima Werner

Rio de Janeiro

Março de 2016

ODYSSEYPROCESSREUSE: UMA METODOLOGIA PARA ENGENHARIA DE LINHA DE PROCESSOS DE SOFTWARE BASEADA EM COMPONENTES

Eldânae Nogueira Teixeira

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Examinada por:

_____
Profa. Cláudia Maria Lima Werner, D.Sc.

_____
Prof. Toacy Cavalcante de Oliveira, D.Sc.

_____
Prof. Uirá Kulesza, D.Sc.

_____
Prof. Leonardo Gresta Paulino Murta, D.Sc.

_____
Profa. Renata Mendes de Araujo, D.Sc.

RIO DE JANEIRO, RJ - BRASIL
MARÇO DE 2016

Teixeira, Eldânae Nogueira

OdysseyProcessReuse: Uma Metodologia para Engenharia de Linha de Processos de Software baseada em Componentes/ Eldânae Nogueira Teixeira. – Rio de Janeiro: UFRJ/COPPE, 2016.

XX, 359 p.: il.; 29,7 cm.

Orientadora: Cláudia Maria Lima Werner.

Tese (doutorado) – UFRJ/ COPPE/ Programa de Engenharia de Sistemas e Computação, 2016.

Referências Bibliográficas: p. 257-275.

1. Reutilização de Processos de Software. 2. Linha de Processos de Software 3. Variabilidade de Processos de Software 4. Componentes de Processos de Software I. Werner, Cláudia Maria Lima. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

"Digo o que penso, com esperança. Penso no que faço, com fé. Faço o que devo fazer, com amor. Eu me esforço para ser cada dia melhor, pois bondade também se aprende. Mesmo quando tudo parece desabar, cabe a mim decidir entre rir ou chorar, ir ou ficar, desistir ou lutar. Porque descobri, no caminho incerto da vida, que o mais importante é o decidir."

Cora Coralina

AGRADECIMENTOS

À energia suprema do universo, luz que me faz vencer os medos, representa o significado da minha trajetória, e me guia na consolidação do meu caráter, sabedoria e fé.

A minha mãe, essa guerreira e eterna companheira, que sempre está ao meu lado e dedica todo amor e confiança em minha essência, na minha capacidade e na minha transformação para concretização dos meus objetivos. Agradeço a paciência pelos momentos de angústia e ansiedade, e pelo discernimento e ensinamento transmitidos para que eu possa persistir e buscar o verdadeiro significado que a vida representa para mim.

Ao meu filhotinho e melhor amigo, Hachi, razão do meu sorriso e de cada alegria que me faz esquecer qualquer tristeza e a buscar sempre o melhor em tudo, inclusive em mim mesma.

Ao meu avô, Nogueira, que sempre se orgulhou de cada conquista.

À orientadora Cláudia Werner, pelos direcionamentos ao longo de minha formação acadêmica, desde a Iniciação Científica, que com a convivência e o tempo, me motivou a novos desafios, mas sem interferir no meu modo de ser "introspectivo". Agradeço à confiança, o incentivo e as diversas oportunidades que contribuíram de forma essencial para o meu amadurecimento profissional e pessoal.

À Aline Vasconcelos, pela coorientação na elaboração deste trabalho. Agradeço pela parceria em diferentes momentos e pelas conversas com troca de ideias e experiências, e perspectivas de colaborações futuras.

Aos professores da UFRJ e da COPPE que participaram do meu aprendizado e formação e confiaram em minha capacidade.

À instituição como um todo, que viabilizou minha formação e deu todo suporte, em especial, ao CNPq, pelo apoio financeiro, e ao PESC.

Aos antigos e novos colegas dessa trajetória, dos grupos de pesquisa da COPPE, da COPPETEC e, em especial, do grupo de Reutilização de Software da COPPE/UFRJ, por todas as contribuições para o desenvolvimento pessoal e troca de conhecimentos.

Aos professores Leonardo Murta, Renata Araujo, Toacy Oliveira e Uirá Kulesza, por terem aceitado fazer parte desta banca.

A todos que participaram das avaliações conduzidas ao longo do desenvolvimento deste trabalho.

Por último, agradeço à vida. "*C'ést La Vie!*"

"A beleza de ser um eterno aprendiz" (Gonzaguinha)

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

ODYSSEYPROCESSREUSE: UMA METODOLOGIA PARA ENGENHARIA DE LINHA DE PROCESSOS DE SOFTWARE BASEADA EM COMPONENTES

Eldânae Nogueira Teixeira

Março/2016

Orientadora: Cláudia Maria Lima Werner

Programa: Engenharia de Sistemas e Computação

A definição de processo de software pode ser considerada uma tarefa complexa. Espera-se que tal atividade, quando suportada por uma abordagem sistemática de reutilização, forneça os mesmos benefícios alcançados em reutilização de software, como melhorias na produtividade e qualidade associadas com reduções de custos e esforços. A escolha de quais informações podem ser reutilizadas, garantindo que fatores específicos de projeto e da organização sejam atendidos, é um aspecto que pode ser melhor suportado dependendo dos mecanismos usados para representar e organizar um domínio de processos de software. Desta forma, o objetivo desta tese é definir uma metodologia que suporte o engenheiro de domínio de processos a definir uma estrutura de processos de software reutilizável combinando técnicas de reutilização. Essa tese envolve a definição de um processo para reutilização, de uma representação para modelagem de um domínio de processos de software por meio da definição de um metamodelo e notação, que trate as propriedades de variabilidade em diferentes níveis de abstração, e de procedimentos que direcionem a definição, mapeamento e agrupamento de artefatos do domínio. Um suporte ferramental está sendo desenvolvido e técnicas de verificação semântica dos modelos criados também foram concebidas. Foram realizados estudos para analisar a viabilidade da metodologia proposta. Os resultados dessas avaliações apresentaram indícios da viabilidade de aplicação dos diferentes elementos que compõem a metodologia proposta.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

ODYSSEYPROCESSREUSE: A METHODOLOGY FOR COMPONENT BASED SOFTWARE PROCESS LINE ENGINEERING

Eldânae Nogueira Teixeira

March/2016

Advisor: Cláudia Maria Lima Werner

Department: Systems Engineering and Computer Science

The software process definition can be considered a complex task. It is expected that this activity, if supported by a systematic reuse approach, can provide the same benefits reached in software reuse for products as improvements in productivity and quality, associated with effort and cost reductions. An important aspect to promote the reusability in process definition is the representation and organization of the software process domain. So, the goal of this PhD thesis is to define a methodology to support the process domain engineer to define a software process reusable structure combining reuse techniques. This proposal involves the definition of a process for reuse, a software process domain modelling representation through a metamodel and notation, which threats variability properties in different abstraction levels, and a set of procedures to guide the definition, mapping and grouping of domains artefacts. Also, a supporting environment is being implemented and techniques for models semantic verification were conceived. To analyze the viability of the proposed methodology, studies were conducted. Their results provide initial evidence of the application feasibility of the elements that compose the proposed methodology.

Sumário

Capítulo 1 – Introdução 1

1.1. Contexto 1

1.2. Caracterização do Problema 4

1.3. Objetivos 8

1.4. Metodologia de Pesquisa 9

1.5. Organização da Tese 11

Capítulo 2 – Revisão da Literatura de Reutilização de Processos de Software 13

2.1. Introdução 13

2.2. Fundamentação teórica 15

2.3. Revisão da Literatura e Identificação de Abordagens de LPrS 26

2.3.1. Resultados retornados pela execução da revisão – Abordagens de LPrS Identificadas 30

2.3.2. Descrição das Abordagens de Linha de Processos de Software Identificadas 33

2.3.3. Análises das Abordagens de LPrS Identificadas segundo os critérios definidos 40

2.4. Variabilidade de Processos de Software nas Abordagens de LPrS 45

2.5. Componentes de Processos de Software nas abordagens de LPrS 55

2.6. Considerações Finais 61

Capítulo 3 – *OdysseyProcessReuse* 67

3.1. Introdução 67

3.2. Engenharia de Linha de Processos de Software (ELPrS) 71

3.2.1. Engenharia de Domínio de Processos de Software (EDPS) 71

3.2.1.1. Análise do Domínio de Processos de Software (ADPS) 72

3.2.1.2. Projeto do Domínio de Processos de Software 87

3.2.2. Engenharia de Aplicação de Processos de Software (EAPS) 92

3.3. Considerações Finais..... 96

Capítulo 4 – Modelagem de Variabilidades de LPrS 98

4.1. Introdução..... 98

4.2. Modelagem de Variabilidades de Linha de Processos de Software 99

4.2.1. Modelagem de Características..... 103

4.2.2. Modelagem de Componentes..... 109

4.2.3. Estudo Comparativo de Modelagem de Variabilidade de LPrS 112

4.2.3.1. Taxonomia..... 116

4.2.3.2. Relacionamentos..... 125

4.2.3.3. Restrições de Dependência e Mútua Exclusividade..... 128

4.3. Heurísticas de Mapeamento..... 131

4.3.1. Exemplo de Aplicação das Heurísticas de Mapeamento do Modelo de Características para o Modelo de Componentes de Processo de Software..... 135

4.3.2. Estudo de Viabilidade das Heurísticas de Mapeamento..... 138

4.3.2.1. Planejamento do Estudo 138

4.3.2.2.Execução do Estudo 141

4.3.2.3. Análise dos resultados obtidos 142

4.4. Critérios de Agrupamento 147

4.4.1. Critério de Agrupamento por meio da Análise de Escopos do Domínio Definidos 150

4.4.2. Critérios de Agrupamento por meio da Análise de Componentes com relações entre si 150

4.4.2.1. Análise pelo grau de responsabilidade dos componentes 151

4.4.2.2. Análise pelo grau de dependência dos componentes..... 156

4.4.2.3. Análise pelo grau de instabilidade de um componente 160

4.4.3. Outras considerações 163

4.4.4. Estudo de Viabilidade de Critérios de Agrupamento 166

4.4.4.1. Planejamento do Estudo 167

4.4.4.2. Execução do Estudo 169

4.4.4.3. Análise dos resultados obtidos 170

4.4.4.4. Considerações combinadas dos estudos de viabilidade das heurísticas e dos critérios de agrupamento 173

4.5. Inspeção da modelagem do domínio 175

4.5.1. Base de conhecimento para desenvolvimento das técnicas de verificação de processos de software 176

4.5.1.1. Process Variability Modeling Checking (PVMCheck) 178

4.5.1.2. Process Component Variability Modeling Checking (PCVMCheck) 181

4.5.2. Estudo de Viabilidade de PVMCheck 185

4.5.2.1. Planejamento do Estudo 186

4.5.2.2. Execução do Estudo 189

4.5.2.3. Análise dos resultados obtidos 190

4.6. Considerações Finais 192

Capítulo 5 – Avaliações da Abordagem *OdysseyProcessReuse* 195

5.1. Introdução 195

5.2. Estudo de Caso - Avaliação do Resultado de Aplicação da Abordagem de Desenvolvimento de LPrS Proposta 198

5.2.1. Planejamento do Estudo 198

5.2.2. Execução do Estudo 206

5.2.3. Análise dos perfis dos participantes 206

5.2.4. Análise dos resultados obtidos 210
5.2.4.1. Análises dos resultados das tarefas básicas 210
5.2.4.2. Análises dos resultados das tarefas intermediárias 213
5.2.4.3. Análises dos resultados das tarefas de assimilação 216
5.2.4.4. Sumarização dos resultados das tarefas 218
5.2.5. Análise da avaliação pós-estudo 219
5.3. Considerações finais 225
Capítulo 6 – Implementação – Ambiente de Reutilização 226
6.1. Introdução 226
6.2. Estrutura Inicial do ambiente de apoio à reutilização 227
6.3. Estrutura do ambiente de apoio ao desenvolvimento de LPrS (EDPS) 228
6.3.1. Nível de Escopo 229
6.3.2. Nível de Características 230
6.3.3. Nível de Componentes 240
6.4. Heurísticas de Mapeamento 245
6.5. Considerações Finais 247
Capítulo 7 – Conclusão 248
7.1. Epílogo 248
7.2. Contribuições 249
7.3. Limitações 253
7.4. Considerações Finais e Oportunidades Futuras 255
Referências 257
Apêndice I - Protocolo Revisão Sistemática 276

Apêndice II – Aplicação de Heurísticas de Mapeamento do Modelo de Características para o Modelo de Componentes de Processo de Software 281

Apêndice III – Casos Discrepantes *PVMCheck* 286

Apêndice IV – Modelagem da LPrS de Medição de Software com *OdysseyProcess-FEX* 311

Apêndice V – Estudo de Viabilidade das Heurísticas de Mapeamento . 336

Apêndice VI – Estudo de Viabilidade dos Critérios de Agrupamento de Componentes de Processos de Software 345

Apêndice VII – Formulários da Avaliação do Resultado de Aplicação da Abordagem de Desenvolvimento de LPrS Proposta 353

Índice de Figuras

Figura 2.1 - Frequência de elementos individuais em que variações são aplicadas (Adaptado de: (MARTÍNEZ-RUIZ *et al*., 2012)) 20
Figura 2.2 - Correspondência entre operações de processos e mecanismos de processos ricos em variantes (Adaptado de: (MARTÍNEZ-RUIZ *et al*., 2012)) 22
Figura 2.3 - Classificação proposta para variabilidades de processos de software (ALEIXO, 2013) 23
Figura 2.4 – Etapas do Processo de Revisão Sistemática (BRERETON *et al*., 2007) 30
Figura 2.5 - Número de artigos por Máquina de Busca 30
Figura 2.6 – Distribuição dos trabalhos por Ano 33
Figura 2.7 – Distribuição dos trabalhos por Veículos de Publicação 33
Figura 2.8 – Distribuição dos trabalhos por fase da Engenharia de LPrS 41
Figura 2.9 – Distribuição dos trabalhos por fase da Engenharia de LPrS 45
Figura 2.10 - Elementos de Processo apresentados para Modelagem de Variabilidade46
Figura 2.11 - Descrição dos elementos gráficos notação BARRETO (2011) - Fonte (CARDOSO, 2012) 50
Figura 2.12 – Metamodelo eSPEM (ALEGRÍA E BASTARRICA, 2012) 51
Figura 2.13 - Ícones gráficos vSPEM (MARTÍNEZ-RUIZ *et a*l., 2008) 51
Figura 2.14 - Estereótipos e meta-atributos do *SMartySPEMProfile* (PAZIN, 2012) 53
Figura 2.15 - LPrS hipotética extraída de (PAZIN, 2012) 54
Figura 3.1 - Elementos Principais do desenvolvimento da *OdysseyProcessReuse* 67
Figura 3.2 – Fases da ELPrS: EDPS e EAPS 68
Figura 3.3 - Visão Geral de uma ELPrS 71
Figura 3.4 – Etapas da EDPS (ADPS e PDPS) 72
Figura 3.5 - Atividades de Análise do Domínio de Processos de Software 74
Figura 3.6 – Atividade de Captura do Conhecimento do Domínio de Processos de Software 75
Figura 3.7 - Análise de Similaridades e Variabilidades no Domínio de Processos de Software 76
Figura 3.8 - Atividades da Modelagem de Análise do Domínio de Processos de Software 78

Figura 3.9 – Exemplo de Visão Estrutural de um Modelo de Características de LPrS de Gerência de Portfólio 79
Figura 3.10 – Atividades da Modelagem de Características do Domínio (visão estrutural e comportamental) 80
Figura 3.11 – Atividades da Modelagem Contextual do Domínio 82
Figura 3.12 – Exemplo Modelo de Contexto utilizando notação UbiFEX 84
Figura 3.13 – Abordagem em Níveis para Definição de Processos (Fonte: BERTOLLO, 2006) 86
Figura 3.14 – Estrutura de Subescopos do Domínio 87
Figura 3.15 - Atividades da Etapa de Projeto do Domínio de Processos de Software ... 88
Figura 3.16 - Atividades da Modelagem de Componentes de Processos de Software .. 89
Figura 3.17 - Atividades da Modelagem de Componentes de Processos de Software .. 89
Figura 3.18 – Principais conceitos para tratar medidas associadas a componentes de processos (Adaptado de BARRETO, 2011) 90
Figura 3.19 – Visão Geral EAPS 92
Figura 3.20 – Etapas da EAPS (AAPS e PAPS) 93
Figura 3.21 - Atividades da EAPS 94
Figura 4.1- Metamodelo *OdysseyProcess-FEX*: Pacote Principal (*Process Features View*) (TEIXEIRA *et al*., 2016) 104
Figura 4.2- LPrS de Planejamento de Projetos de Software – Modelo de Características Visão Estrutural 108
Figura 4.3 - LPrS de Planejamento de Projetos de Software – Modelo de Características Visão Comportamental 108
Figura 4.4 - Metamodelo do Nível de Componentes do *OdysseyProcess-FEX (Process Components View)*: Pacote Principal 110
Figura 4.5 – Modelo de Componentes da LPrS de Planejamento de Projetos de Software 112
Figura 4.6 – LPrS para Medição no Contexto Organizacional (CARDOSO, 2012) 113
Figura 4.7 – Visão Estrutural da Atividade *Planejar Medição* 118
Figura 4.8 – Visão Estrutural da Atividade *Planejar Medição usando o método GQM* 119
Figura 4.9 – Modelo de Componentes - Atividade *Planejar Medição* e variantes 120

Figura 4.10 – Modelos da Atividade *Planejar medição utilizando o método GQM* 124
Figura 4.11 – Arquitetura Interna do Componente de Processo: *Planejar medição utilizando o método GQM* (CARDOSO, 2012) 128
Figura 4.12 – Atividade *Reportar e armazenar as análises de medição* modelada usando a notação *OdysseyProcess-FEX* 130
Figura 4.13 – Atividade *Reportar e armazenar as análises de medição* modelada usando a notação *OdysseyProcess-FEX* 130
Figura 4.14 – Visão Comportamental do Modelo de Componentes da LPrS de Planejamento de Projetos de Software 137
Figura 4.15 – Exemplo de Análise de Mapeamento por Componente Esperado 143
Figura 4.16 – Gráfico de Dispersão de Eficiência e Efetividade na aplicação das heurísticas de mapeamento 145
Figura 4.17 – Visão de Responsabilidades dos componentes da LPrS de Planejamento de Projetos de Software 156
Figura 4.18 – Visão de Dependências dos componentes da LPrS de Planejamento de Projetos de Software 160
Figura 4.19 – Visão de Instabilidades dos componentes da LPrS de Planejamento de Projetos de Software 161
Figura 4.20 – Modelo Final de Componentes Agrupados da LPrS de Planejamento de Projetos de Software 163
Figura 4.21 – Análise tempo nos estudos..... 174
Figura 4.22 – Distribuição de Eficiência e Efetividade nos grupos A e B..... 191
Figura 5.1 – Análise do perfil dos participantes com relação ao nível de experiência com processos de software 207
Figura 5.2 – Análise do perfil dos participantes com relação ao nível de experiência com reutilização de processos de software 209
Figura 5.3 – Análise do perfil dos participantes pelos diferentes critérios..... 210
Figura 5.4 – Análise respostas Tarefas Básicas..... 211
Figura 5.5 – Análise do Tempo Total das Tarefas Básicas 213
Figura 5.6 – Análise do Tempo por tarefa básica de cada participante..... 213
Figura 5.7 – Análise das respostas das Tarefas Intermediárias..... 215

Figura 5.8 – Análise do Tempo Total das Tarefas Intermediárias 215
Figura 5.9 – Análise do Tempo por tarefa intermediária de cada participante 216
Figura 5.10 – Análise respostas das Tarefas de Assimilação 217
Figura 5.11 – Análise do Tempo Total das Tarefas de Assimilação 218
Figura 6.1 - Infraestrutura ambiente Odyssey 227
Figura 6.2 – Tela Principal do ambiente Odyssey adaptado 228
Figura 6.3 – Ambiente de Modelagem de LPrS e seus níveis de abstração 229
Figura 6.4 – Ambiente de Modelagem de LPrS – Nível de Esocpo (*Scope View*) 230
Figura 6.5 – Ambiente de Modelagem de LPrS – Nível de Características (*Features View*) 231
Figura 6.6 – Ambiente de Modelagem de LPrS – Painel de Edição de Característica - Nível de Características (*Features View*) 231
Figura 6.7 – Ambiente de Modelagem de LPrS – Painel de Edição de Característica Ponto de Variação 232
Figura 6.8 – Ambiente de Modelagem de LPrS – Diagramadores (Visão Estrutural e Visão Comportamental) 233
Figura 6.9 – Ambiente de Modelagem de LPrS – Diagramadores Visão Estrutural 234
Figura 6.10 – Ambiente de Modelagem de LPrS – Diagramadores Visão Estrutural ... 234
Figura 6.11 – Ambiente de Modelagem de LPrS – Diagramadores Visão Estrutural – Perspectiva *Role Process Feature Diagram* 235
Figura 6.12 – Ambiente de Modelagem de LPrS – Diagramadores Visão Comportamental 236
Figura 6.13 – Edição de Regras de Composição no ambiente 237
Figura 6.14 – Modelagem de Contexto no ambiente Odyssey 238
Figura 6.15 – Exemplo de notificação de inconsistência gerada pelo mecanismo de verificação 239
Figura 6.16 – Tela Principal de Edição de Componentes no ambiente Odyssey Nível Projeto 240
Figura 6.17 – Tela Principal de Edição de Componentes Abas Componentes internos e variantes internas 241

Figura 6.18 – Tela Principal de Edição de Componentes - Aba Componentes internos 241
Figura 6.19 –Componentes internos – Diagrama criado após seleção de componentes 242
Figura 6.20 – Tela Principal de Edição de Componentes – Elementos Internos 242
Figura 6.21 – Tela Principal de Edição de Componentes – Elementos Internos rastreabilidade com os elementos do modelo de características 243
Figura 6.22 – Tela de Edição Medidas de Componentes 244
Figura 6.23 – Tela de Edição *Baselines* de Componentes 244
Figura 6.24 – Tela de Edição de Interfaces 245
Figura 6.25 – Tela Inicial do Procedimento de Mapeamento do Modelo de Características para o Modelo de Componentes 245
Figura 6.26 – Segunda Tela do Procedimento de Mapeamento do Modelo de Características para o Modelo de Componentes – Edição dos nomes dos pacotes 246
Figura 6.27 - Mapeamento de um Modelo de Características para um Modelo de Componentes 246
Figura 6.28 – Resultado do Mapeamento de um Modelo de Características para um Modelo de Componentes 247

Índice de Tabelas

Tabela 2.1- Resumo dos elementos de processos presentes em diferentes propostas de representação 18
Tabela 2.2 – Revisões Sistemáticas da Área de Processos 29
Tabela 2.3 - Publicações selecionadas organizadas por ano 31
Tabela 2.4 - Análise dos elementos de processos passíveis de variação e a notação utilizada nas abordagens de LPrS 46
Tabela 2.5 - Análise de Abordagens de Definição de Processos de Software baseada em Componentes 56
Tabela 2.6 – Sumarização de análise das abordagens de LPrS identificadas 64
Tabela 4.1 - Categorias de características *OdysseyProcess-FEX* 105
Tabela 4.2 - Relacionamento do metamodelo *OdysseyProcess-FEX* Nível Análise Modelo de Características 105
Tabela 4.3 – Modelagem de Informação Contextual do Domínio (informações que guiam a seleção de artefatos reutilizáveis) 114
Tabela 4.4 – Modelagem Elementos de Processo do Domínio (Unidade de Trabalho) 116
Tabela 4.5 – Modelagem Elementos de Processo do Domínio (Papel, produto de trabalho, ferramenta) 121
Tabela 4.6 – Modelagem Relacionamentos entre Elementos de Processo do Domínio 126
Tabela 4.7 – Modelagem Relações de Dependência e Mútua Exclusividade do Domínio 129
Tabela 4.8- Heurísticas de Mapeamento de Modelo de Características para Modelo de Componentes de uma LPrS 132
Tabela 4.9 – Heurísticas aplicadas na LPrS de Planejamento de Projetos de Software 136
Tabela 4.10 – Perfil dos Participantes 143
Tabela 4.11 – Sumarização dos resultados de análise 144
Tabela 4.12 – Sumarização dos resultados de avaliação das heurísticas pelos participantes 146

Tabela 4.13 – Sumarização dos resultados de aplicação dos critérios de agrupamento de componentes pelos participantes 171
Tabela 4.14 – Sumarização dos resultados de avaliação dos critérios de agrupamento pelos participantes..... 172
Tabela 4.15 – Sumarização dos resultados de avaliação das heurísticas e critérios de agrupamento 175
Tabela 4.16 – Tipos de defeitos adaptados de (TRAVASSOS *et al*., 1999) 178
Tabela 4.17- *PVMCheck* - *Checklist* para Inspeção do Modelo de Caraterísticas de LPrS 180
Tabela 4.18- *PCVMCheck* Passo 1: *Checklist* para Inspeção de Componentes de Processos de Software..... 182
Tabela 4.19- *PCVMCheck* Passo 2: *Checklist* para Inspeção da Arquitetura da LPrS ... 184
Tabela 4.20 – Sumarização dos resultados de análise 191
Tabela 4.21- Comparativo do apoio oferecido pelas abordagens de LPrS e a *OdysseyProcessReuse* 194
Tabela 5.1 – Perfil dos Participantes quanto ao conhecimento em reutilização de processos de software..... 208
Tabela 5.2 – Perfil dos Participantes segundo todos os critérios analisados..... 209
Tabela 5.3 – Respostas Esperadas das Tarefas Básicas..... 211
Tabela 5.4 – Respostas Esperadas das Tarefas Intermediárias..... 214
Tabela 5.5 – Respostas esperadas Tarefas de Assimilação..... 216
Tabela 5.6 – Resumo dos resultados do estudo..... 219
Tabela 5.7 – Análise dos benefícios esperados com a adoção da abordagem 220
Tabela 5.8 – Análise do nível de dificuldade esperado com a adoção da abordagem 220

Lista de Acrônimos

ADPS Análise do Domínio de Processos de Software

DPBC Desenvolvimento de Processos Baseados em Componentes

EAPS Engenharia de Aplicação de Processos de Software

EDPS Engenharia de Domínio de Processos de Software

ELPrS Engenharia de Linha de Processos de Software

LPS Linha de Produtos de Software

LPrS Linha de Processos de Software

PDPS Projeto do Domínio de Processos de Software

Capítulo 1 – Introdução

1.1. CONTEXTO

Diante da ampla competitividade e dinamicidade de mercado, o desenvolvimento de software é um aspecto de grande relevância na criação de diferencial e agregação de valor ao negócio. A Engenharia de Processos de Software tem recebido maior atenção de organizações de software que buscam melhorar a qualidade de seus produtos, aumentar sua produtividade e atender à flexibilidade necessária no provimento de seus serviços. Considerando a forte relação entre a qualidade desses produtos e a qualidade dos processos de desenvolvimento adotados para construí-los (OSTERWEIL, 1987, FUGGETTA, 2000), as organizações têm investido na melhoria de seus processos. Acredita-se que uma organização bem gerenciada, com processos bem definidos, tem maior probabilidade de desenvolver produtos que sigam as exigências do cliente dentro do cronograma e do orçamento, quando comparada a uma organização mal gerenciada e sem processos definidos (SOLINGEN, 2004).

A definição de processos de software auxilia na avaliação e entendimento tanto do processo como da própria organização. A definição explícita de processos representa um papel chave nas principais iniciativas de melhoria de processos de software (RUIZ-RUBE *et al*., 2015). Modelos de referência de processos, tais como o CMMI (CHRISSIS *et al*., 2006), a ISO/IEC 12207 (ISO/IEC, 2008) e o MPS.Br (SOFTEX, 2016), foram criados como forma de auxiliar no gerenciamento orientado a processos e na agregação de boas práticas de desenvolvimento de software reconhecidas. Tais modelos incluem a definição de processos como uma das áreas relevantes dentro do desenvolvimento de software, que visa definir um conjunto de ativos de processo organizacional e padrões do ambiente de trabalho usáveis e aplicáveis às necessidades de negócio da organização (SOFTEX, 2016).

No entanto, a diversidade de projetos, características e objetivos organizacionais, modelos e padrões de referência de desenvolvimento de software, aspectos específicos de produtos, restrições de processos e recursos existentes são fatores que desafiam a definição de processos e aumentam a complexidade de tal atividade. Não existe um modelo de desenvolvimento único e genérico que possa ser

aplicado a qualquer projeto de desenvolvimento, sem a necessidade de realizar adaptações para contemplar as especificidades e particularidades do contexto em que tal processo será executado. Para lidar com esta diversidade, há uma crescente necessidade por parte da indústria de software pela rápida e efetiva definição de processos de software (ALEIXO *et al*., 2010a, HANSSON *et al*., 2006). Isto envolve apoio à definição de processos alinhada à realidade dos projetos e organizações e, principalmente, pela aplicação de conhecimento adquirido na condução de projetos anteriores, de forma a minimizar a complexidade e o esforço envolvidos. Neste cenário, o acúmulo de conhecimento e experiência tornam-se aspectos fundamentais dos recursos envolvidos nesta atividade e em todo o ciclo de vida da Engenharia de Processos. Gerentes de projetos e equipes de desenvolvimento de processos experientes acabam agregando responsabilidades, ao realizar tais atividades e tomadas de decisão mentalmente. Entretanto, como um modelo mental não pode ser diretamente reutilizado por gerentes menos experientes, faz-se necessária uma representação explícita para o conhecimento tácito dos gerentes experientes (BARROS, 2001), pois se espera que caso o conhecimento dos engenheiros de processo experientes seja explicitado, formalizado e disponibilizado para outros profissionais, seja possível reutilizar esse conhecimento de forma mais efetiva (BARRETO, 2007) para apoiar a definição de novos processos. Neste cenário, a reutilização de processos de software vem se destacando como uma das formas de lidar com esses desafios, pois visa reutilizar o conhecimento que organizações de Engenharia de Software adquirem durante a realização de projetos, prevenindo a ocorrência dos mesmos erros e fornecendo melhores práticas (ROUILLÉ *et al*., 2013a).

Aplicando a analogia existente entre processos e produtos de software (OSTERWEIL, 1987), KELLNER (1996) destaca que o conhecimento de técnicas de reutilização de produtos de software, tais como: arquiteturas povoadas com componentes reutilizáveis; gerenciamento de repositórios para armazenar, catalogar, buscar, acessar, etc. ativos reutilizáveis; gerência de configuração de ativos reutilizáveis, entre outros, poderiam ser aplicados a processos de software. Neste contexto, técnicas de reutilização de software, como a Engenharia de Domínio (ED) (ARANGO E PRIETO-DIAZ, 1991) e, uma de suas vertentes, a Linha de Produtos de

Software (LPS) (NORTHROP, 2002), e ainda o Desenvolvimento Baseado em Componentes (DBC) (SAMETINGER, 1997) têm sido adaptadas ao contexto de definição de processos de software (KELLNER, 1996, ROMBACH, 2006).

Uma das técnicas de reutilização de processos, denominada Linha de Processos, surgiu com base nos conceitos provenientes de LPS. O termo foi proposto em diversas abordagens (JAUFMAN E MÜNCH, 2005, ROMBACH, 2006, WASHIZAKI, 2006, ARMBRUST *et al*., 2009, ALEIXO *et al*., 2010a, BARRETO *et al*., 2011, HURTADO ALEGRÍA *et al*., 2013, SIMMONDS *et al*., 2015, MAGDALENO *et al*., 2015) - para citar algumas, que trabalham o conceito de LPS cujos produtos são processos. Uma Linha de Processos de Software (LPrS) pode ser definida como "um conjunto de processos de software que compartilham um conjunto de características comuns e variáveis, e são desenvolvidas a partir de artefatos (*core assets*), que podem ser reutilizados e combinados entre si, segundo regras de composição e recorte, para compor e adaptar processos de software" (NUNES *et al*., 2010, MAGDALENO, 2010, TEIXEIRA, 2011). Abordagens de LPrS visam aplicar uma iniciativa proativa para reutilização, onde uma organização prepara ativamente seus processos de software para um certo número de necessidades previstas (ARMBRUST *et al*., 2008). Alguns objetivos podem ser apontados com o uso de Linha de Processos: aumento da produtividade da atividade de definição de processos, diminuindo o esforço necessário para realizá-la; aumento da qualidade e da adequação dos processos gerados, por meio da reutilização do conhecimento de especialistas e de dados sobre utilização; aumento do potencial de reutilização por meio da representação de variabilidades; e redução dos riscos de uma definição inadequada de processo (BARRETO *et al*., 2009, JAUFMAN e MÜNCH, 2005, ROMBACH, 2006).

Outro campo dentro da reutilização de processos trata da Definição de Processos Baseada em Componentes de processos (DPBC). Existem diferentes abordagens que trabalham o conceito de componente de processos (GARY E LINDQUIST, 1999a, FUSARO *et al*., 1998, RU-ZHI *et al*., 2005, BHUTA *et al*., 2005, AOUSSAT *et al*., 2014). Um componente de processo pode ser visto como um encapsulamento de informações e comportamento de processo em um dado nível de granularidade (GARY E LINDQUIST, 1999a). Desta forma, o processo é derivado a partir

de blocos de construção menores que devem ser estruturados por meio de um arcabouço estrutural denominado arquitetura de processos.

Resumindo, o propósito da área de reutilização de processos de software é facilitar a definição de processos, diminuindo o custo e o esforço associados à atividade, além de possivelmente aumentar a qualidade dos processos gerados, inclusive tornando a realização da atividade acessível a profissionais menos experientes (BARRETO, 2007). A dificuldade na reutilização de processos de software reside no fato de que a variabilidade dos requisitos dos projetos frequentemente implica em variabilidade dos processos de software (ROUILLÉ *et al*., 2013a) e, para tornar a reutilização efetiva, é necessário desenvolver métodos para capturar elementos comuns e variáveis de processos e criar definições de processos reutilizáveis (HOLLENBACH E FRAKES, 1996).

1.2. CARACTERIZAÇÃO DO PROBLEMA

Como se pode observar, existem diferentes informações de processos de software que estão envolvidas em sua definição. A escolha de quais informações podem ser reutilizadas em novos processos, garantindo que fatores específicos do projeto e da organização sejam atendidos, é um aspecto que pode ser melhor suportado dependendo dos mecanismos usados para produzir e organizar a informação reutilizável. O apoio ao engenheiro de processos de software, para desenvolver a estrutura de processos reutilizável, é de grande importância. Tal importância lida com a disponibilização da maior quantidade de informação passível de reutilização e de forma inteligível por parte dos envolvidos com a definição de processos específicos de projetos aplicando esta informação.

Esse apoio envolve a definição de quais informações de processos podem ser reutilizadas, quais dessas informações representam aspectos comuns ou variações em um domínio de processos, e como essas variações impactam outras no domínio. A forma de representação de maneira explícita das variações e com o gerenciamento da complexidade envolvida é outro fator a ser considerado. A abstração é um conceito essencial em reutilização, pois para que um determinado artefato de software possa ser reutilizado, ele precisa antes ser compreendido (KRUEGER, 1992). Isso é

diretamente aplicável para processos, onde o uso de diferentes níveis de abstração é uma das formas de lidar com a possibilidade de desenvolvimento gradual e complementar das informações que compõem um domínio, representando desde informações conceituais do nível de definição até informações de nível de instanciação e execução (agentes, dados de utilização, lições apreendidas, dentre outras). No entanto, a escolha de utilizar níveis de abstração e diferentes modelos necessita de mecanismos de suporte de mapeamento entre os artefatos e manutenção da rastreabilidade entre eles. Vale ressaltar que perspectivas de visualização da informação também são importantes, pois processos lidam com questões tanto estruturais, como a definição de elementos e relações entre eles, quanto comportamentais, descrevendo fluxos de execução das unidades de trabalho. Fatores contextuais que impactam a definição de processos e sua instanciação e execução devem ser especificados de forma relacionada aos demais artefatos do domínio.

A definição de processos via composição por unidades menores como subprocessos é apontada como um aspecto relevante por modelos de maturidade como o MPS.Br (SOFTEX, 2016). Desta forma, a organização da informação do domínio nesses blocos de construção, nesta tese denominados de componentes de processo, é uma atividade importante que pode ser suportada por mecanismos que orientem o seu agrupamento mantendo a consistência com as propriedades definidas. Além disso, a análise da possibilidade de agrupamento desses componentes em elementos com tratamento do acoplamento em função da comunicação por meio de suas interfaces ou por informações a respeito da semântica do domínio é importante para melhorar a compreensão, manutenibilidade e, sobretudo, a reutilização do domínio. De forma complementar, o suporte à representação de dados que indiquem a estabilidade, desempenho e capacidade dos componentes, de forma que o processo definido seja adequado ao contexto do projeto em que será utilizado (BARRETO, 2011) é importante para suportar organizações com objetivos estratégicos claramente definidos e que almejem alcançar maiores níveis de maturidade no mercado. Estruturas que representem medidas e valores de referência são importantes para entender melhor a capacidade de utilização de determinado componente do domínio frente a uma

perspectiva particular como, por exemplo, o seu potencial de colaboração (MAGDALENO *et al*., 2015).

Trabalhos na área de reutilização de processos de software vêm sendo utilizados como forma de lidar com a complexidade da atividade de definição de processos. As técnicas de LPrS e de DPBC são algumas dessas abordagens.

A área de LPrS é recente e não está ainda totalmente consolidada na literatura (BARRETO *et al.*, 2009). Segundo DE CARVALHO *et al*. (2014), a área ainda pode ser considerada imatura, já que não existe nenhuma taxonomia bem definida e a avaliação de qualidade das abordagens propostas precisa ser melhorada em termos de avaliação empírica. Portanto, apesar dos esforços na área, não há uma abordagem única ou um consenso sobre como realizar a adaptação de processo de uma maneira controlada e consistente (MARTÍNEZ-RUIZ *et al*., 2012) (PILLAT *et al*., 2015).

A expressividade da representação para especificar a variabilidade de uma LPrS é um ponto considerado relevante para o sucesso de sua adoção (SIMMONDS *et al*., 2013). No entanto, apenas alguns trabalhos apresentam uma notação especial para representação de variabilidade em processos, e o desenvolvimento de uma notação que atenda aos requisitos para modelagem de variabilidades é uma necessidade real (MARTÍNEZ-RUIZ *et al*., 2012). Não existe consenso corrente para uma notação padrão que atenda necessidades de adaptação (PILLAT *et al*., 2015) e falta uma capacidade para captura da flexibilidade requerida por processos de software devido à falta de habilidade para expressar flexibilidade usando as linguagens de modelagem de processos de hoje (KUHRMANN, 2014). Abordagens de LPrS, em geral, não contemplam uma representação explícita e completa de um conjunto de elementos de processos (atividade, tarefa, papel, produto de trabalho, ferramentas e relações) que podem sofrer variação tanto a nível de opcionalidade quanto a nível de configuração por alternativas. Além disso, falta a combinação desta representação com possíveis restrições de dependência e mútua exclusividade e variações nos fluxos de controle e de dados.

Outro ponto a ser destacado como campo a ser explorado é a combinação das técnicas de LPrS e gerenciamento de variabilidades com os conceitos da área de DPBC. Seguindo a analogia com a reutilização de software e com base em trabalhos (BRAGA,

1999, BLOIS, 2006) que estudaram a oportunidade de promover a reutilização de artefatos pela combinação das áreas de ED e DBC, pode-se analisar a mesma oportunidade no campo de processos de software. Se por um lado a ED visa identificar aspectos variáveis e invariáveis de um domínio para a sua reutilização, por outro o DBC propõe soluções reutilizáveis para o processo de desenvolvimento de software, disponibilizando aos usuários tais variabilidades por meio de componentes e interfaces (BROWN, 2000). Desta forma, analogias de tais benefícios devem ser buscadas na área de reutilização de processos de software. No entanto, a aplicação de conceitos de DPBC ainda é explorada por poucas abordagens de LPrS. O próprio conceito de componente ainda não apresenta consenso na literatura. Não existe consenso sobre qual informação deve existir em um componente de processo e qual o nível de detalhe que este deve possuir (BARRETO *et al*., 2011). A comunicação entre componentes precisa ser melhor descrita por meio da definição do conceito de interfaces e conectores. A definição de conectores também não possui consenso sobre sua interpretação na literatura (AOUSSAT *et al*., 2014).

A definição de processos baseada em blocos de construção menores e modulares apoiada por uma sistemática que gerencie a variabilidade e opcionalidade inerentes a uma família de processos tende a ser promissora. No entanto, apesar das abordagens existentes, analisando as oportunidades de pesquisa apontadas, o suporte à reutilização de processos de software ainda é um desafio.

Diante do exposto, a questão que norteia esta pesquisa consiste em: Como apoiar um engenheiro de processos a disponibilizar a informação passível de reutilização de um domínio de processos de software de forma a ser de fácil identificação por parte dos envolvidos com o uso dessa informação para definir processos específicos de projetos?

A hipótese definida nesta tese é que com a combinação das técnicas de LPrS e DPBC com: (i) mecanismos para representação explícita e gerenciamento de variabilidade e opcionalidade de um domínio de processos de software em diferentes níveis de abstração e (ii) procedimentos que direcionam a definição, mapeamento e agrupamento de artefatos do domínio, seja possível apoiar a disponibilização de informação reutilizável que apoie a identificação de alternativas de definição de novos

processos em um contexto específico. A facilidade de identificação pode ser observada pela diminuição do esforço e pelo aumento da capacidade de identificar os elementos necessários, de forma precisa e com maior grau de cobertura, próximos a definição esperada, independente do nível de experiência do responsável pela definição.

1.3. OBJETIVOS

Com base na questão definida, o objetivo central deste trabalho é definir uma metodologia de apoio à representação e organização do conhecimento de domínios de processo de software de forma a aumentar o potencial de reutilização com fácil identificação de alternativas para definições de processos de software específicos de projeto.

Busca-se com a metodologia proposta guiar o procedimento de construção de LPrSs, de maneira que profissionais com diferentes níveis de experiência possam alcançar os mesmos resultados esperados de organização e representação do domínio. Ao mesmo tempo, visa-se que o resultado da aplicação da metodologia gere domínios com a mesma facilidade de identificação da informação a ser reutilizada e de seus pontos de configuração por profissionais com diferentes níveis de experiência.

Para alcançar tal objetivo é necessário: (1) gerenciar a representação explícita de variabilidade e opcionalidade de um domínio de processos de software; (2) modularizar tal conhecimento em blocos de construção denominados componentes; e (3) definir procedimentos para guiar a construção dos artefatos de processos de software reutilizáveis e (4) especificar técnicas de verificação dos artefatos criados.

Para atender cada item, uma metodologia de reutilização de processos de software, denominada *OdysseyProcessReuse,* foi definida visando propor:

- Uma Engenharia de LPrS que visa sistematizar as principais atividades de desenvolvimento e uso de uma LPrS, com foco no Processo de Desenvolvimento *para* Reutilização que disponibiliza orientação efetiva para a execução das atividades referentes a todas as fases inerentes à Engenharia de Domínio de Processos de Software (EDPS);
- Uma representação de LPrS para modelagem do domínio de processos de software por meio de dois níveis de abstração correspondentes à análise do domínio (modelo de características) e ao projeto do domínio

(modelo de componentes). Para suportar tal representação, foram definidos metamodelo, notação e perspectivas visuais para tratar os diferentes tipos de elementos de processos que sofrem variações no domínio, os relacionamentos e restrições entre eles e aspectos comportamentais de processos de software (fluxos de controle). Busca-se aumentar a expressividade semântica de representação de LPrS, de maneira a tratar a complexidade dos artefatos e a quantidade de informação com diferentes níveis de abstração complementares;

- Heurísticas de mapeamento entre os artefatos dos dois níveis de abstração que compõem a representação do domínio, estabelecendo a rastreabilidade necessária entre os elementos e mantendo a consistência de propriedades de variabilidade e opcionalidade no domínio, relacionamentos e regras de composição entre artefatos;
- Um conjunto inicial de critérios para apoiar o agrupamento de componentes em elementos mais modulares, visando reduzir o acoplamento e tratar a granularidade dos blocos de construção de processos. Tais critérios analisam o agrupamento de elementos, considerando a estrutura de interações entre os componentes modelados, com o propósito de reorganizar o domínio e aumentar a modularidade;
- Uma técnica para inspeção baseada em *checklist* dos modelos construídos, visando identificar defeitos na fase de desenvolvimento dos artefatos e evitar a propagação de erros para etapa de derivação de processos;
- Um apoio ferramental como infraestrutura de utilização da abordagem proposta, como extensão do ambiente Odyssey (ODYSSEY, 2016), uma infraestrutura de reutilização baseada em modelos de domínio que utiliza técnicas de LPS e DBC.

1.4. METODOLOGIA DE PESQUISA

A metodologia para desenvolvimento da abordagem foi especificada segundo um conjunto de passos, descritos a seguir:

- Definição do Problema de Pesquisa;
- Realização de uma revisão da literatura nas principais áreas envolvidas no desenvolvimento deste trabalho de pesquisa;
- Definição do objetivo da pesquisa;
- Elaboração da metodologia *OdysseyProcessReuse*; e
- Análise da viabilidade da abordagem proposta.

Os três primeiros passos ocorreram de forma iterativa. O tema de reutilização de processos de software foi definido inicialmente como foco da pesquisa, em particular a análise das técnicas de LPrS e DPBC.

Uma fundamentação teórica foi realizada para entendimento do ciclo de vida de processos de software, da atividade de modelagem de processos de software e conceituação dos termos de variabilidade e componentes para processos de software. Nesta etapa, uma das questões analisadas consistiu em: *Qual é o conjunto de informações de processos de software que podem ser disponibilizados para reutilização?* Esta questão inclui a análise de quais elementos são mais relevantes na modelagem resultante da etapa de definição de processos e identificação. Dentre tais elementos identificados, buscou-se analisar quais podem sofrer algum tipo de variação. O tema de reutilização foi então pesquisado por meio de uma revisão da literatura, com o objetivo de entender como abordagens de LPrS estavam sendo tratadas e combinadas com elementos de DPBC, do ponto de vista de pesquisadores, no contexto industrial e acadêmico. Para uma análise controlada, um conjunto de critérios foi definido como forma de auxiliar na identificação dos principais tópicos envolvidos na pesquisa, incluindo a análise de quais modelos foram usados para representar uma família de processos que deixasse explícito as variações no domínio; identificação de quais mecanismos foram disponibilizados para agrupar os elementos de processos reutilizáveis em componentes; identificação de quais atividades foram definidas para apoiar o engenheiro de processo na especificação de uma LPrS; identificação de métodos ou técnicas de verificação dos modelos reutilizáveis, dentre outras.

A metodologia proposta visou tratar o objetivo especificado por meio do desenvolvimento de partes complementares:

- Especificação da representação da LPrS, pela definição dos modelos em diferentes níveis de abstração com gerenciamento dos conceitos de opcionalidade e variabilidade (análise – Modelo de características e projeto – Modelo de Componentes do Domínio). Nesta etapa, um metamodelo e uma notação foram definidos;
- Definição dos procedimentos de mapeamento entre os diferentes artefatos do domínio e dos dois diferentes níveis de abstração especificados (análise e projeto);
- Definição de procedimentos para tratar o acoplamento e a granularidade dos componentes de processo por meio de critérios de agrupamentos;
- Definição de técnicas de inspeção dos modelos de uma LPrS; e
- Especificação do método de Engenharia de LPrS, pela descrição dos fluxos de atividades do desenvolvimentos dos artefatos da LPrS, com foco em conceitos de DPBC;
- Implementação do ferramental de apoio.

A viabilidade de utilização da metodologia foi analisada ao longo do desenvolvimento do trabalho com a realização de pequenos estudos de viabilidade das partes que o compõem.

1.5. ORGANIZAÇÃO DA TESE

Este trabalho está organizado em sete capítulos. O **Capítulo 1**, de Introdução, apresenta a motivação e o problema, os objetivos a serem alcançados e a organização do texto.

O **Capítulo 2** apresenta uma breve Fundamentação Teórica, com a descrição dos principais conceitos sobre processos de software e variabilidade em processos de software. Uma revisão *quasi*-sistemática da literatura na área de reutilização de processos de software, com foco em abordagens de LPrS, é apresentada. O objetivo é analisar tais abordagens segundo um conjunto de critérios estabelecidos, visando entender qual o apoio fornecido às etapas de desenvolvimento e uso de LPrS, a identificação de quais etapas do ciclo de vida de processos de software são consideradas durante a reutilização, identificação de quais possibilidades de

representação de variabilidades são providas nas abordagens de LPrS identificadas e como o conceito de componentes de processos são aplicados nestas abordagens.

Os **Capítulos 3** e **4** tratam da metodologia proposta. O Capítulo 3 apresenta uma visão geral dos elementos que a compõem e descreve o método de Engenharia de Linha de Processos de Software baseado em componentes proposto. O metamodelo e a notação para modelagem da LPrS em dois níveis de abstração, heurísticas de mapeamento entre os níveis e critérios para apoiar o agrupamento de elementos de processos de software em componentes acoplados são apresentados no Capítulo 4, em conjunto com a técnica de inspeção proposta. Associado a cada um desses elementos, estudos de análise da viabilidade de aplicação são apresentados.

O **Capítulo 5** apresenta o planejamento e os resultados alcançados com o último estudo de viabilidade realizado neste trabalho.

O apoio ferramental é apresentado no **Capítulo 6**, com a descrição das funcionalidades implementadas no ambiente Odyssey (ODYSSEY, 2016) para apoiar a utilização da metodologia proposta.

O **Capítulo 7** apresenta as conclusões, principais contribuições alcançadas com o desenvolvimento deste trabalho e limitações. Além disso, a enumeração de possíveis trabalhos futuros descreve desdobramentos da pesquisa.

O Apêndice I apresenta o protocolo da revisão *quasi*-sistemática apresentada no Capítulo 2.

O Apêndice II descreve um exemplo de aplicação das heurísticas de mapeamento entre os dois níveis de abstração definidos (análise – Modelo de características e projeto – Modelo de Componentes do Domínio).

O Apêndice III descreve os casos discrepantes identificados para apoiar a definição dos itens do *checklist* da técnica de inspeção do modelo de características.

Os demais apêndices (IV, V, VI e VII) apresentam materiais desenvolvidos durante o planejamento dos estudos aplicados.

Capítulo 2 – Revisão da Literatura de Reutilização de Processos de Software

2.1. INTRODUÇÃO

O termo reutilização de processos de software (*Software Process Reuse*) define uma ampla área de estudo e utilização prática relacionada aos diferentes aspectos envolvidos com a reutilização do conhecimento adquirido na condução de projetos de software anteriores, prevenindo a ocorrência de erros anteriores e fornecendo melhores práticas (REIS, 2002, ROUILLÉ *et al*., 2013a). Além da reutilização de processo ser vista como a reutilização da descrição de processo na criação de outra descrição de processo, é importante considerar a reutilização de artefatos, procedimentos e documentações dentro desses processos (HENNINGER, 1998). Assim, a reutilização de processos de software inclui não apenas a reutilização de definições de processos, mas também a reutilização de informações relacionadas à utilização de processos (conhecimento e experiência adquiridos) (RU-ZHI *et al*., 2005).

Esse tópico recebeu bastante atenção de pesquisadores e da indústria na segunda metade da década de 1990 e, atualmente, mecanismos aperfeiçoados vêm sendo pesquisados e experimentados para o armazenamento e a definição de elementos reutilizáveis de processos (BARRETO, 2007). Seguindo um breve histórico, a reutilização de processos era, inicialmente, tratada por meio de um mecanismo de cópia e modificação, sem tratar a relevância do contexto de aplicação. Embora “copiar e modificar” seja importante, pois minimiza o esforço na construção de um novo modelo, essa prática é menos útil para o aperfeiçoamento da organização e dos seus processos (JØRGENSEN, 2001) *apud* (REIS, 2002), já que, por meio dela, não são registradas informações que permitiriam acompanhar a evolução dos processos e seus componentes.

No entanto, a criação de um processo capaz de ser utilizado em diversos projetos é uma tarefa difícil. Um método efetivo para capturar os elementos comuns e variantes de processos específicos de projetos é necessário, além da criação de definições de processos que possam ser aplicadas em uma variedade de situações, ou seja, que são reutilizáveis (HOLLENBACH e FRAKES, 1996). Neste cenário, a reutilização

de processos de software vem se destacando como uma das principais práticas para prover a melhoria contínua de processos, por aproveitar as informações (tecnológicas e gerenciais) produzidas durante desenvolvimentos de software passados, com o objetivo de reduzir o esforço necessário para novos desenvolvimentos (BARRETO, 2007).

Aplicando a analogia existente entre produto de software e processo de software (OSTERWEIL, 1987) no campo de reutilização, podemos notar o uso do conhecimento de técnicas de reutilização de produtos de software na área de reutilização de processos (KELLNER, 1996). Abordagens como técnicas de reutilização que trabalham o conceito de componentização de processos (DPBC) e técnicas que utilizam conceitos de famílias de processos denominadas Linhas de Processos de Software (LPrS) podem ser citadas. WASHIZAKI (2006a) define uma LPr como "um conjunto de processos de um determinado domínio de problema, ou com um determinado propósito, que têm características em comum e é construído baseado em ativos reutilizáveis de processos". Alguns objetivos podem ser apontados para o uso de Linha de Processos: aumento da produtividade da atividade de definição de processos, diminuindo o esforço necessário para realizá-la; aumento da qualidade e da adequação dos processos gerados, pela reutilização do conhecimento de especialistas e de dados sobre utilização; aumento do potencial de reutilização por meio da representação de variabilidades; e redução dos riscos de uma definição inadequada de processo (BARRETO *et al*., 2009, JAUFMAN e MÜNCH, 2005, ROMBACH, 2006). O gerenciamento de variabilidades é um dos pontos relevantes neste tipo de abordagem. LPrS buscam prover técnicas e mecanismos para a modelagem de similaridades e variabilidades em uma família de processos de software (OLIVEIRA JUNIOR *et al*., 2013).

Inicialmente, este capítulo apresenta na Seção 2.2, os fundamentos e conceitos básicos sobre processos, as diferentes etapas do ciclo de vida que compõem sua engenharia, questões envolvidas na modelagem resultante da etapa de definição de processos e como o conceito de variabilidade é tratado pela área de processos como variações de definições de processos propostas para atender os múltiplos aspectos dos diferentes projetos que apoiam. O foco do capítulo consiste na análise da área de reutilização de processos de software, principalmente de abordagens de Linha de

Processos de Software (LPrS). Um conjunto de critérios foi definido para a análise dos trabalhos relacionados, buscando identificar as contribuições e pontos de melhorias das abordagens propostas. A Seção 2.3 apresenta o procedimento para realização de uma revisão *quasi*-sistemática da literatura e os resultados da identificação de quais abordagens trabalham a reutilização de processos por meio de LPrS. Uma revisão sistemática é usada como uma metodologia de pesquisa específica desenvolvida para coletar e avaliar as evidências disponíveis de um tópico específico (BIOLCHINI *et al*., 2005). Tais revisões quando aplicadas para caracterização de um tópico de pesquisa (revisões iniciais da área), onde comparações não são realizadas e meta-análises não são viáveis de aplicação, são denominadas como revisões *quasi*-sistemáticas (TRAVASSOS *et al*., 2008). Os resultados gerais obtidos com a revisão são descritos na Subseção 2.3.1. As abordagens de LPrS são apresentadas na Subseção 2.3.2 e analisadas, segundo os critérios definidos, na Subseção 2.3.3. A Seção 2.4 foca em uma análise de conceitos e técnicas para a modelagem de variabilidade de processos de software. A Seção 2.5 apresenta uma análise dos trabalhos relacionados à outra área correlata a esta pesquisa: Componentes de Processos de Software e Arquitetura de Linha de Processos de Software. Por fim, na Seção 2.6, são feitas algumas considerações finais a partir dos conceitos e análises das abordagens de LPrS identificadas.

2.2. FUNDAMENTAÇÃO TEÓRICA

Um processo de software pode ser definido como um conjunto coerente de políticas, estruturas organizacionais, tecnologias, procedimentos e artefatos necessários para conceber, desenvolver, implantar e manter um produto de software (FUGGETTA, 2000). O ciclo de vida de um processo de software abrange as atividades de engenharia de um processo. As atividades deste ciclo são chamadas de meta-atividades, e o ciclo de vida é chamado de metaprocesso de software (DERNIAME *et al*., 1999). Existem diversas propostas de ciclo de vida para processos de software. Em REIS (2003), um ciclo de vida é descrito contendo as seguintes atividades: provisão de tecnologia; análise de requisitos; projeto do processo; instanciação do processo; simulação do processo; execução do modelo de processo; e avaliação do processo. A definição de um processo de software de acordo com um conjunto de requisitos se

encontra nas fases iniciais do ciclo de vida. A definição visa auxiliar a condução de projetos de desenvolvimento de software, explicitando o conjunto de tarefas a serem seguidas por meio de uma representação que permita um entendimento comum dos envolvidos. Um processo definido permite o acompanhamento de seu comportamento e ações de melhorias, assim como maior suporte ao gerenciamento e controle. No entanto, a atividade é complexa e apresenta diversas formas para ser conduzida como com base em normas e modelos de referência, com base em objetivos estratégicos ou na cultura organizacional ou na necessidade de clientes. Normas e modelos de referência, tais como o CMMI (CHRISSIS *et al*., 2006), a ISO/IEC 12207 (ISO/IEC, 2008) e o MPS.Br (SOFTEX, 2016), foram criados como forma de auxiliar no gerenciamento orientado a processos e na agregação de boas práticas de desenvolvimento de software reconhecidas. Esses modelos tratam de descrições genéricas de processos de software, especificando seus propósitos e resultados esperados dentro de um determinado grau de maturidade para esses processos dentro de uma organização. Em geral, essas normas e modelos orientam para a definição de um processo padrão organizacional e regras de adaptação a serem aplicadas em situações previamente definidas. Um processo padrão é um ativo de processo organizacional que compreende um conjunto de definições de processos básicos para guiar todos os processos da organização, ou seja, devem ser adaptados para projetos para criar os processos definidos (ISO/IEC, 2004). No contexto de organização de alta maturidade, os modelos de maturidade estabelecem requisitos adicionais para a que realização da definição de processo para um projeto possibilite o processo atender seus objetivos de qualidade e de desempenho com base em técnicas estatísticas e outras técnicas quantitativas.

Como resultado, o processo é especificado por meio de um modelo, uma estrutura complexa, que inter-relaciona diferentes aspectos tecnológicos, organizacionais e sociais para descrever o processo de desenvolvimento de software (REIS, 2002). Um modelo especifica como as atividades de engenharia de software serão executadas, quais papéis estão envolvidos, os recursos consumidos, as ferramentas utilizadas, os artefatos consumidos e produzidos pelas tarefas, o produto desenvolvido, assim como os mecanismos de comunicação entre papéis e tarefas

(ZAMLI E LEE, 2001). Alguns objetivos específicos e benefícios da modelagem de processos de software foram apresentados por CURTIS *et al*. (1992): facilidade de entendimento e comunicação; controle e suporte ao gerenciamento de processos; fornecimento de orientações automatizadas para um melhor desempenho de processos; fornecimento de suporte para execução automatizada de processos; suporte a melhoria de processos.

Para representar um modelo de processo de software é frequentemente adotada uma Linguagem de Modelagem de Processos (*Process Modeling Language* – PML), a qual deve oferecer recursos para descrever e manipular os elementos do processo. Linguagens de processo de software devem suportar uma notação visual como forma de suprir a dificuldade de entendimento de representações textuais. Devem possibilitar o entendimento do trabalho de outros papéis envolvidos no processo e o conhecimento do próprio e de outros processos, permitindo um maior apoio a trabalhos colaborativos e um maior suporte ao processo (ZAMLI e LEE, 2001).

Existem várias classificações sobre os principais elementos de um modelo de processo (FEILER E HUMPHREY, 1993, LONCHAMP, 1993). Vários tipos de dados são integrados em um modelo de processo para indicar quem, quando, onde, como e por que os passos são realizados (LONCHAMP, 1993). A Tabela 2.1 apresenta um resumo dos elementos encontrados em algumas representações propostas na literatura de modelagem de processos.

Apesar de diferentes representações analisadas (Tabela 2.1), podemos observar que um conjunto de elementos de processo está sempre presente e pode ser considerado como um conjunto dos elementos essenciais: (1) Unidade de trabalho - representado por diferentes granularidades, variando desde um processo como todo até uma atividade ou ainda uma tarefa, que pode ser especificada por um conjunto de passos que descrevem sua forma de execução; (2) Artefato – representado por produtos de trabalho requeridos ou produzidos para ou pela execução de uma unidade de trabalho; (3) Papel - conjunto de habilidades, competências e responsabilidades assumidas pelos participantes de uma unidade de trabalho; e (4) Recursos como ferramentas que atuam como sistemas de apoio e/ou podem automatizar parte ou toda a execução de uma unidade de trabalho. Associados a esses

elementos principais, ainda podemos identificar outros conceitos importantes como procedimentos que podem especificar diretrizes, identificando o que deve ser realizado em uma unidade de trabalho sem especificar como realiza-lo. Diretrizes podem ser consideradas como roteiros ou normas. Métodos são procedimentos mais sistemáticos e especificam os passos que devem ser executados para alcançar resultados, determinando que conjunto de atividades deve compor uma atividade maior. Técnicas descrevem aspectos gerais que devem ser considerados para a condução de tarefas ou atividades. Regras e políticas especificam restrições que podem impactar a execução de uma tarefa. *Templates* descrevem estruturas de documentos e artefatos gerados pelas tarefas.

Tabela 2.1 - Resumo dos elementos de processos presentes em diferentes propostas de representação

Elementos de Processo / Abordagem	Unidade de Trabalho	Responsabilidade Habilidade Competência Papel	Produto de Trabalho	Recursos	Procedimento: determina como atividades são realizadas, gerando variações na forma de execução e, consequentemente, no produto esperado
(ACUÑA E FERRÉ, 2001)	Atividade	Papel	Artefato (entrada/ saída)	Ator (humano/ ferramentas)	Procedimentos, Regras, Políticas, Objetivos
(STOROLLI *et al.*, 2009)	Processo é composto por Atividades (sequência) compostas por Tarefas	Habilidades	Artefatos (produzidos, modificados e utilizados)	Atores (pessoas, sistemas e equipamentos) utilizam Recursos	Políticas (aspectos de qualidade, regras e critérios); Métricas e Controle (verificação de aplicação das políticas e métricas)
(FALBO E BERTOLLO, 2005)	Atividade (Composição: Super/Sub atividades) (Dependências e ordem de execução: Pré/Pós-atividades)	Agente Humano	Artefatos (Produto/ Insumo)	Agentes: Hardware; Software (Procedimentos automatizados e Sistema de Apoio); Humano	Procedimento: Técnica/Método/ Diretriz Adequação a Paradigma e Tecnologia de Desenvolvimento
APSEE-Reuse (REIS, 2002)	Atividades	Papéis e Habilidades	Artefatos	Ferramentas	Políticas
(LONCHAMP, 1993)	Atividades e Tarefas	Papéis	Artefatos e Entregas de Produtos	Recursos Humanos e Computadorizados	Restrições: regras, procedimentos, políticas ou disciplinas.
(LANNA, 2009)	Processos, Atividades e Tarefas	Papéis, habilidades, conhecimento dos membros	Artefatos	Ferramentas e Linguagens	Restrições (custo, tempo, pessoal); políticas e metas
SPEM (OMG, 2008)	Processos, Atividades, Tarefa	Papel	Artefatos (produzidos, consumidos ou modificados)	Ferramentas	Procedimentos: *Checklist*, Métricas, *Guidelines*: técnicas, regras, diretrizes; método; *templates* especificação de Uso de ferramenta

Um conjunto de variações para combinação desses elementos presentes na definição de processos de software é decorrente da necessidade de atender a uma série de características específicas de projetos, produtos e organizações. Características como paradigma de desenvolvimento; o tipo de software a ser desenvolvido; a tecnologia de desenvolvimento a ser adotada; o nível de maturidade da organização; as restrições de prazo, custo e pessoal; as políticas organizacionais, dentre outros, podem impactar na aplicação de procedimentos e especificações de elementos de processo que acabam por influenciar a atividade de definição de processos de software. Esta condição pode ser denominada de variabilidade. Para a área de produtos de software, o termo variabilidade pode ser entendido como a possibilidade de configuração, ou ainda, como a habilidade que um sistema ou artefato de software possui de ser alterado, customizado, ou configurado para um contexto em particular (BOSCH, 2004). Um paralelo pode ser feito para um processo de software que possui a capacidade de derivar variações de acordo com as condições inerentes a diferentes requisitos de projetos. Portfólios de processos evoluem devido a fatores internos relacionados à evolução de processos e/ou a fatores externos ou combinações e aquisições em que diferentes processos, podendo ter partes comuns, precisam ser integrados (ROLLAND E NURCAN, 2010). Como todos os componentes de processo (atividades, produtos, agentes, ferramentas) e suas interações (fluxo de informação, fluxo de artefatos, controle, comunicação, tempo, dependências, concorrência) podem variar, processos irão variar, mesmo que estejam no mesmo nível, escopo e objetivo (LINDVALL E RUS, 2000).

Um estudo recente (MARTÍNEZ-RUIZ *et al*., 2012) define a variabilidade de processo como a capacidade de definir, estabelecer o escopo, e executar alterações de uma maneira sistemática e consistente, com o intuito de adaptar modelos de processos a características únicas de um projeto em um dado contexto. Neste estudo, diferentes trabalhos foram analisados para identificar o estado da arte das iniciativas de adaptação de processos de software. A análise busca requisitos de variabilidade que são indicados como importantes pelos mecanismos de adaptação. Um conjunto de categorias de elementos de processos foi adotado para analisar quais desses elementos de processos são considerados durante a atividade de adaptação:

elementos de trabalho (atividades, tarefas; subprocessos, passos); papéis; artefatos; recursos; e técnicas e práticas. Os elementos mais apontados como variações são os elementos de trabalho; artefatos e papéis (Figura 2.1). Variações em recursos também devem ser consideradas, já que processos são executados por papéis que usam diferentes ferramentas e tais elementos variam de um projeto para outro. Algumas relações foram identificadas, caracterizando o impacto de uma variação de um tipo de elemento em outro, cuja consistência é mantida por relações de variabilidades entre tais elementos, identificando que variações secundárias são geradas a partir de uma variação primária a ser executada:

- Papéis variam no contexto de variação de atividades;
- Tarefas variam no contexto de variação de atividades;
- Passos variam no contexto de variação de tarefas;
- Artefatos variam no contexto de variação de atividades e tarefas;
- Recursos variam no contexto de variação de atividades, tarefas, artefatos e papéis;
- Subprocessos variam no contexto de variação de atividades.

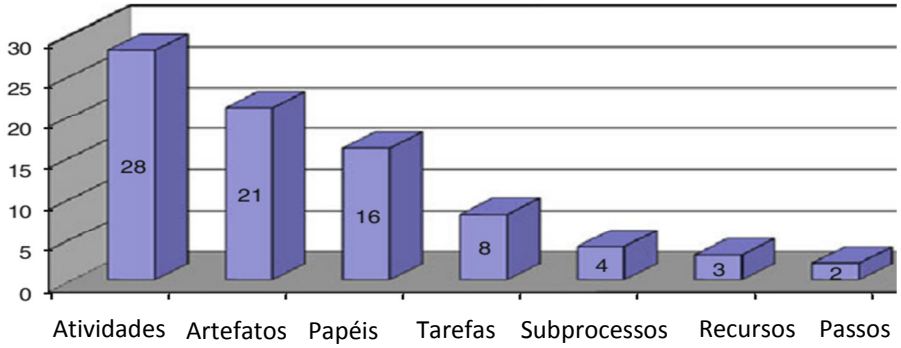


Figura 2.1 - Frequência de elementos individuais em que variações são aplicadas (Adaptado de: (MARTÍNEZ-RUIZ *et al*., 2012))

Além de variações em elementos de processos, alguns tipos de variação ocorrem nas relações entre esses elementos, como a variabilidade em fluxos de controle, definida em RAZAVIAN E KHOSRAVI (2008) como a variação na forma de diferenciação entre as sequências que são estabelecidas entre as atividades, ou seja, diferentes caminhos de execução de atividades.

Algumas formas de adaptações de processo, por meio da variação dos elementos que o compõem, foram identificadas em MARTÍNEZ-RUIZ *et al*. (2012):

- Operações diretas foram identificadas como: (1) *inserção*; (2) *remoção*; (3) *modificação* (alteração de propriedades dos elementos ao invés de substituí-los); (4) *substituição de elementos*; e (5) definição das relações e restrições entre elementos. Essas ações são aplicadas no momento de adaptar modelos de processos para contextos específicos realizando alterações diretamente nos elementos que os compõem.
- Operações indiretas: (1) *padrões*: compostos por variações bem definidas aplicáveis a um conjunto de requisitos a ser satisfeito; (2) *parametrização*: atribuir determinado valor a certos parâmetros previamente definidos de um processo no momento de adaptação; (3) *herança:* adaptação do processo superior pela definição de processos inferiores que estendem as propriedades do processo superior, de acordo com contextos particulares; (4) encapsulamento de atividades: definição de grupos de atividades que são tratados em conjunto para realizar a adaptação do processo, ou nós de decisão compostos por condições, que são usados para modificar fluxos entre atividades (fluxo de controle) ou modificar o uso de artefatos (fluxo de artefatos). Tais operações expressam formas de adaptações especificadas previamente e podem auxiliar no momento de derivação de um processo específico. No entanto, ainda são operações menos aplicadas nas abordagens analisadas pelos autores.
- Aplicação de Conceitos de Linha de Processos: tendências atuais mostram a adaptação sendo gerenciada por meio do conceito de processos ricos em variantes, onde a variabilidade é definida como pontos de variação e variantes. A variabilidade pode ser definida em três tipos: (1) *opcional*: elementos que podem ou não estar presentes em processos; (2) *mandatório*: elemento que deve estar presente em um processo; (3) *alternativo*: referente a pontos de variação que possuem diferentes alternativas para ser configurados. Nesta tese, esses tipos de mecanismos foram os aplicados no tratamento de variabilidade de processos de software junto ao tratamento por meio de encapsulamento de elementos de processos em módulos, que preservam as propriedades de variabilidade aqui descritas. Ou seja, o tratamento é feito também considerando o paradigma de desenvolvimento baseado em componentes. Esses mecanismos podem ser

aplicados a elementos individuais de composição de processo e algumas equivalências com as operações diretas podem ser estabelecidas. Desta forma, um paralelo foi realizado para estabelecer relações entre as operações diretas e os tipos de variabilidades descritos (Figura 2.2).

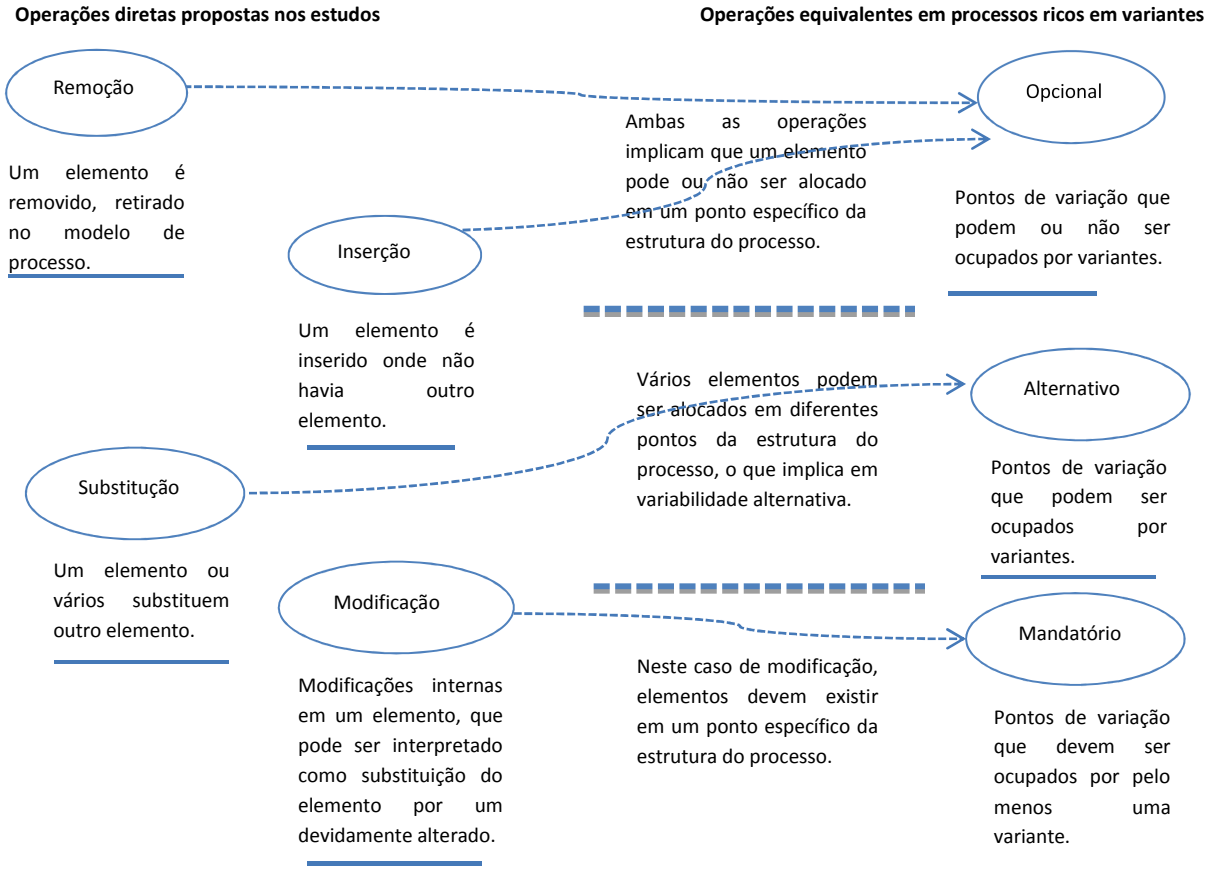


Figura 2.2 - Correspondência entre operações de processos e mecanismos de processos ricos em variantes (Adaptado de: (MARTÍNEZ-RUIZ *et al.*, 2012))

Em ALEIXO (2013), uma proposta de categorização de variabilidades de processos é apresentada. Tal categorização divide as variabilidades de processos em duas dimensões: (1) variabilidades de processos de software no espaço de solução e (2) variabilidades de processos de software no espaço de problema (Figura 2.3).

As variabilidades de processos de software no espaço de solução são classificadas de acordo com os seguintes critérios: (i) *relativas aos elementos de processo* – que especificam e definem de forma explícita quem é de fato o processo, tais como, atividades, tarefas, papéis, práticas, roteiros, etc. Este tipo de variabilidade identifica se os elementos de processos podem ou não ser incluídos em uma dada instância, pela definição de elementos opcionais ou alternativos; (ii) *granularidade* –

que dizem respeito ao tipo de refinamento de granularidade grossa ou fina que é realizado sobre elementos de processo. Uma variabilidade de granularidade grossa determina a inclusão de elementos de processo, associada a possível alteração de elementos já existentes, e uma variabilidade de granularidade fina determina modificações apenas em propriedades de elementos de processos já existentes; (iii) *natureza de especificação* – que indica se a variabilidade ocorre de forma pontual (localizada) ou dispersa (transversal) a um dado elemento de processo; e (iv) *tempo de binding* – que diz respeito a se a variabilidade pode ser aplicada ao processo, no momento de sua *modelagem/definição* – durante o início da execução de um projeto, *implantação* – preparação do mesmo para ser implantado em alguma ferramenta que promova o acompanhamento do processo, tais como, as ferramentas de gerenciamento de projeto, gerência de mudanças ou que agregue ambas as funcionalidades; e *execução* – que diz respeito a variabilidades realizadas no processo durante a execução de um projeto.

Nível de Abstração	Critério	Classificação
Espaço de Solução	Elementos de processo	(1) papel; (2) atividade, (3) tarefa, (4) artefato e (5) orientação
	Granularidade	(6) fina e (7) grossa
	Natureza de especificação	(8) pontual e (9) transversal
	Tempo de *binding*	(15) modelagem, (16) implantação e (17) execução
Espaço de Problema	Técnicas e tecnologias associadas à disciplina de processo	(10) requisitos, (11) projeto (*design*), (12) implementação, (13) testes, (14) gerenciamento de projetos
	Nível de modelo de maturidade de processos	(20) MPS.Br G, (21) MPS.Br F, (22) MPS.BR E, (23) MPS.Br D (...)

Figura 2.3 - Classificação proposta para variabilidades de processos de software (ALEIXO, 2013)

As variabilidades de processos de software no espaço de problema indicam situações que podem impactar a seleção dos elementos que farão parte de um processo específico de projeto (modelagem contextual) e são classificadas de acordo com os seguintes critérios: (i) relativas às *técnicas e tecnologias associadas às disciplinas de processo*, tais como, requisitos, projeto, implementação, testes e gerência de projeto; e (ii) relativas aos *níveis de modelos de maturidade de processos* – indica se um dado processo derivado precisa atender a um determinado nível de um modelo de maturidade, como por exemplo, os níveis G ou F do MPS.Br (SOFTEX, 2016).

Nesta tese, os dois espaços de solução são tratados. O espaço de solução é trabalhado na modelagem da LPrS com foco para o tempo de modelagem/definição de processos. O espaço de problema pode ser apoiado pela modelagem contextual descrita no Capítulo 3, que permite a definição dos pontos descritos acima.

Diante de tal variabilidade em processos de software, a diversidade de organizações e projetos, suas características e objetivos a tarefa de definição de processos de software torna-se não trivial. Em decorrência do fato de que organizações são diferentes, e ainda que dois projetos dentro da mesma organização também podem ser diferentes (BERGER, 2003), não existe um processo de software que possa ser genericamente aplicado a todos os projetos (MACHADO, 2000). Dependendo das características do projeto, um processo aplicado com sucesso em um projeto pode ser um fracasso em outro (PEDREIRA *et al*., 2007). Essa complexidade da tarefa de definição de processos levou à necessidade de caracterizar, usar e gerenciar as similaridades e diferenças entre os processos (SUTTON e OSTERWEIL, 1996). Neste sentido, diversos autores apontam que a sistematização da reutilização pode ser também aplicada a processos (MONTERO *et al*., 2007). O objetivo é explorar as melhores práticas e o conhecimento capitalizado de modelagens de processos e experiências de execução prévias.

No campo de reutilização, podemos notar o uso do conhecimento de técnicas de reutilização de produtos de software na área de reutilização de processos (KELLNER, 1996), tais como: arquiteturas povoadas com componentes reusáveis; gerenciamento de repositórios para armazenar, catalogar, procurar, acessar, etc. ativos reusáveis; e gerência de configuração de ativos reusáveis poderiam ser aplicados a processos de software. Arquitetura de processos pode ser entendida como um conjunto padrão de unidades ou passos de processo principais com regras que os descrevem e relacionam (HUMPHREY, 1989). Já um *template* de processo consiste em um modelo de processo genérico e reutilizável que estabelece um ponto de início para a construção de um novo modelo de processo (REIS, 2002). Um padrão de processo descreve uma abordagem ou série de ações bem sucedidas para o desenvolvimento de software (AMBLER, 1998). Um componente de processo pode ser visto como um encapsulamento de informações e comportamento de processo em um dado nível de

granularidade (GARY e LINDQUIST, 1999a). Maiores informações sobre componentes de processos são apresentadas na Seção 2.5.

Outra abordagem, denominada Linha de Processos de Software (LPrS), surgiu como uma técnica sistemática de reutilização de processos e foi apresentada em abordagens (BARRETO, 2011, JAUFMAN e MÜNCH, 2005, ROMBACH, 2006, WASHIZAKI, 2006a) que aplicam a ideia de LPS em processos. WASHIZAKI (2006a) define uma Linha de Processo como "um conjunto de processos de um determinado domínio de problema, ou com um determinado propósito, que têm características em comum e é construído baseado em ativos reutilizáveis de processos. Alguns objetivos podem ser apontados com o uso de Linha de Processos: aumento da produtividade da atividade de definição de processos, diminuindo o esforço necessário para realizá-la; aumento da qualidade e da adequação dos processos gerados, pela reutilização do conhecimento de especialistas e de dados sobre utilização; aumento do potencial de reutilização por meio da representação de variabilidades; e redução dos riscos de uma definição inadequada de processo (BARRETO *et al.,* 2009, JAUFMAN e MÜNCH, 2005, ROMBACH, 2006). Maiores detalhes sobre LPrS são descritos nas seções a seguir, ao longo deste capítulo.

Com base em trabalhos na área de reutilização de software (BRAGA, 1999, BLOIS, 2006) que estudam a oportunidade de promover a reutilização de artefatos pela combinação das áreas de Engenharia de Domínio (ED) (ARANGO E PRIETO-DIAZ, 1991) e Desenvolvimento Baseado em Componentes (DBC) (SAMETINGER, 1997), e aplicando a analogia entre software e processos de software (OSTERWEIL, 1987), podemos identificar a possibilidade de mesma oportunidade no campo de processos de software. Se por um lado a ED visa identificar aspectos variáveis e invariáveis de um domínio para a sua reutilização, por outro, o DBC propõe soluções reutilizáveis para o processo de desenvolvimento de software, disponibilizando aos usuários tais variabilidades por meio de componentes e interfaces (BROWN, 2000). Neste sentido, um desafio da área de ED é promover o uso de abordagens de DBC, visando a construção de Arquiteturas de Referência (ARs) baseadas em componentes, e como desafio da área de DBC, podemos citar a modelagem de variabilidades por meio de componentes, representando de forma consistente tais aspectos de domínios (BLOIS,

2006). Transferindo para a área de processos de software, identificamos os conceitos de Engenharia de Domínio aplicados em abordagens de LPrS e os conceitos de DBC em abordagens que visam realizar a Definição de Processos Baseada em Componentes (DPBC).

Desta forma, uma análise dos trabalhos relacionados a estas áreas foi realizada e é apresentada nas seções a seguir. Uma revisão *quasi*-sistemática da literatura foi conduzida com o intuito de identificar as principais abordagens de LPrS e observar o suporte fornecido à reutilização provido por elas.

2.3. Revisão da Literatura e Identificação de Abordagens de LPrS

As áreas de Linha de Processos de Software (LPrS) e de Definição de Processo de Software Baseada em Componentes (DPBC) servem de base para a construção da sistemática de reutilização de processos de software proposta nesta pesquisa. Desta forma, o objetivo da revisão foi coletar e analisar publicações científicas sobre processos de software, com o propósito de caracterizar abordagens (ex.: técnicas, métodos, processos, ferramentas), com relação à utilização de técnicas de LPrS, do ponto de vista de pesquisadores, no contexto industrial e acadêmico. Deste conjunto de abordagens identificadas, foi analisado se o conceito de componente é tratado e como é aplicado. Para uma análise controlada, um conjunto de critérios foi definido como forma de auxiliar na identificação dos principais tópicos envolvidos na área de LPrS.

Dentro da área de reutilização de software, o arcabouço de comparação utilizado no estudo de viabilidade do método FODA (KANG *et al*., 1990) apresenta três principais aspectos que, segundo os autores, devem ser considerados na avaliação de um método: a) *a existência de um processo*; b) *os produtos gerados pelo método*, como são representados e aplicáveis ao desenvolvimento de um domínio de aplicações e; c) *as ferramentas para automatizar o processo*. Assim, um aspecto a ser observado nos trabalhos consistiu na identificação da existência de processos que especifiquem como conduzir a técnica proposta e a especificação dos produtos a serem gerados pela técnica. Tal sistemática pode ser alcançada mediante a definição de estratégias de definição de **processos *para* e *com* reutilização**. Essas estratégias

devem definir o conjunto de atividades e resultados esperados que viabilizem uma reutilização efetiva. Assim, um dos critérios de análise consistiu em analisar qual das fases de Engenharia de LPrS são suportadas pela abordagem analisada: análise da existência de uma especificação de um processo de desenvolvimento *para* reutilização de processos de software (Engenharia de Domínio de Processos de Software), definindo as etapas envolvidas na construção de artefatos de processos de software reutilizáveis, e de um processo de desenvolvimento *com* reutilização (Engenharia de Aplicação de Processos de Software) definindo as etapas envolvidas na descrição de processos específicos de projetos, que utilizam os artefatos de processo reutilizáveis gerados. De forma complementar buscou-se identificar quais atividades do ciclo de vida de processos de software são suportados pela abordagem (definição, simulação, implantação, execução e melhoria de processos). Atividades como apoio à melhoria e evolução da LPrS, apoio à etapa de execução dos processos derivados a partir da linha, e apoio à verificação e validação dos artefatos reutilizáveis desenvolvidos foram observadas quanto ao suporte fornecido em cada abordagem.

Uma análise de **ferramentas de apoio às diferentes etapas da abordagem** foi realizada. Além disso, a análise da existência de uma infraestrutura de reutilização de apoio integrado às diferentes etapas de construção e uso de uma LPrS é importante, uma vez que, do ponto de vista de reutilização, os artefatos gerados em todas as fases do processo estão relacionados. Assim, caso não exista integração entre os diferentes apoios ferramentais, a reutilização dos artefatos poderá ser prejudicada, limitando-se a alguma fase ou atividade específica de um processo.

Outros aspectos foram observados como identificação de tipos de exemplos práticos aplicando a abordagem; identificação de como a abordagem foi avaliada (exemplo, estudo de caso, aplicação na indústria, comparação, *survey*, avaliação por especialista ou outro); e identificação da aplicação de critérios ou métricas específicas para avaliação da abordagem.

Mais um critério definido consistiu na identificação da existência de uma proposta de **representação para modelagem de variabilidades e opcionalidades de uma LPrS**. Os elementos de processos classificados como variáveis e os tipos de variações permitidas são aspectos importantes a serem observados. Além disso, a

definição de uma notação para representação da LPrS deve ser analisada. A possibilidade do uso de diferentes níveis de abstração para representar a LPrS, como ocorre em abordagens de reutilização de software, deve ser observada em conjunto com a identificação de procedimentos de mapeamento entre os diferentes artefatos utilizados para a manutenção da consistência dos modelos e dos processos a serem derivados a partir da linha definida. O uso de diferentes níveis de abstração é importante pela complexidade de modelagem de LPrS e possibilidade de desenvolvimento gradual e complementar das informações que a compõe. Essa análise foi complementada com critérios específicos de modelagem de variabilidades em LPrS, apresentados na Seção 2.4.

Outra área de reutilização de processos de software relevante consiste na Definição de Processo de Software Baseada em Componentes (DPBC). Desta forma, esta pesquisa tem como propósito unir a sistemática da LPrS a ser definida a conceitos de DPBC por meio da especificação de uma Arquitetura de Componentes de LPrS. A **análise do uso de princípios de DPBC** nas abordagens analisadas visa identificar o quanto de componentização tem sido trabalhado de forma conjunta a técnica de LPrS, como forma de promover um aumento dos benefícios para a reutilização. Esta análise é apresentada na Seção 2.5. Além disso, foi analisado se métodos de desenvolvimento de LPrS e DPBC fornecem sistemáticas para a organização de seus artefatos em elementos arquiteturais, os quais compõem uma arquitetura de componentes de LPrS. Para isso, é esperado o uso de critérios, métricas ou outras técnicas para apoio a criação destes elementos arquiteturais.

A análise dos resultados de outras revisões sistemáticas foi inicialmente analisado para o estabelecimento dos objetivos e critérios desta análise da literatura da área de LPrS. A Tabela 2.2 apresenta o conjunto de revisões sistemáticas consideradas relacionadas. Pode-se observar que poucos trabalhos lidam com a análise do estado da arte em LPrS.

Outras duas revisões sistemáticas recentemente conduzidas em teses de doutorado (ALEIXO, 2013; BARRETO, 2011) que tratam do tema de reutilização de processos de software também foram observadas. A revisão de ALEIXO (2013) teve como objetivo identificar as abordagens existentes de gerência de variabilidades de

processos de software, indicando que técnicas elas utilizam, de que forma vem sendo avaliadas e com que tipos de exemplos de variabilidades tais estudos são realizados. A revisão de BARRETO (2011) teve como objetivo analisar relatos de experiência e publicações científicas sobre a definição de processos de software, com o propósito de caracterizar abordagens (ex.: técnicas, métodos, processos, ferramentas), com relação à utilização de técnicas de reutilização, do ponto de vista de pesquisadores, no contexto industrial e acadêmico. A segunda revisão é mais abrangente do que o escopo estipulado neste trabalho, incluindo outras técnicas de reutilização de software, como padrões de processos de software.

Tabela 2.2 – Revisões Sistemáticas Relacionadas da Área de Reutilização de Processos

Referência	Período	# Estudos Primários	Área Investigada
DE CARVALHO *et al*., 2014	1996-2013	40	LPrS
ROCHA E FANTINATO, 2013	2003-2012	63	LPS aplicadas à Gerenciamento de Processos de Negócios
MARTÍNEZ-RUÍZ *et al*., 2012	1990-2009	32	Adaptação de Processos
PEDREIRA *et al*., 2007	Até 2006	28	Adaptação de Processos

As revisões de base possuíam critérios focados no contexto do trabalho em que estava sendo desenvolvido e, desta forma, precisaram ser estendidos nesta proposta de acordo com os objetivos de pesquisa. A maior razão para a realização de uma revisão sistemática ao invés de uma seleção *ad-hoc* da literatura relacionada é melhorar a qualidade do resultado de análise dos estudos coletados na área pesquisada com a aplicação de uma sequência de passos estabelecidos e sistematizados (Figura 2.4). Parte da revisão foi realizada por um trabalho de parceria com uma equipe de pesquisadores do Rio Grande do Norte (UFRN) e do Paraná (UEM). O protocolo estabelecido foi estruturado pela definição das questões de pesquisa, procedimentos de execução da revisão sistemática (procedimento de coleta, análise e síntese dos trabalhos relacionados), e avaliação do resultado coletado a ser publicado. Tal protocolo encontra-se descrito no Apêndice I. Os principais resultados retornados com a busca realizada são apresentados na Seção 2.3.1, com a lista de abordagens de LPrS identificadas. A Seção 2.3.2 apresenta uma descrição de cada abordagem identificada e, por última, a Seção 2.3.3 sumariza a análise dos critérios especificados.

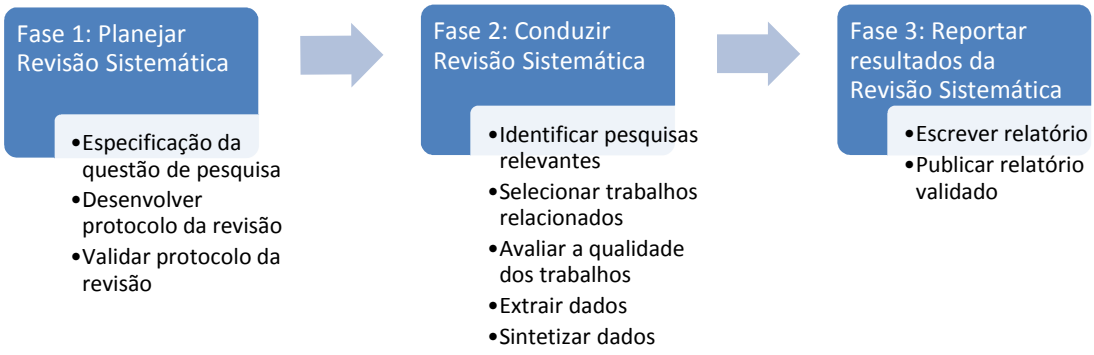


Figura 2.4 – Etapas do Processo de Revisão Sistemática (BRERETON *et al.*, 2007)

2.3.1. Resultados retornados pela execução da revisão – Abordagens de LPrS Identificadas

O objetivo principal desta revisão é caracterizar o estado da arte de abordagens de Linhas de Processos de Software. Este objetivo envolve coletar e caracterizar publicações científicas sobre processos de software com o propósito de caracterizar abordagens relacionadas ao uso de LPrS do ponto de vista de pesquisadores no contexto acadêmico e industrial. Como questão principal foi definida: Quais abordagens de LPrS podem ser identificadas e qual suporte é fornecido por essas abordagens para a reutilização de software? Dez questões secundárias foram especificadas para complementar a análise de acordo com os critérios definidos. O protocolo completo encontra-se descrito no Apêndice I.

A identificação dos estudos foi realizada com a aplicação da *string* de busca nas três bases selecionadas. Tal atividade foi realizada em Novembro de 2014 com a *string* sendo aplicada a campos específicos das publicações: título, palavra-chave e resumo. Um total de 261 publicações forma retornadas (*Scopus: 170; IeeeXplore: 32; Compendex: 59*). Publicações duplicadas foram identificadas e 75 duplicações foram excluídas do total de 261 artigos retornados.

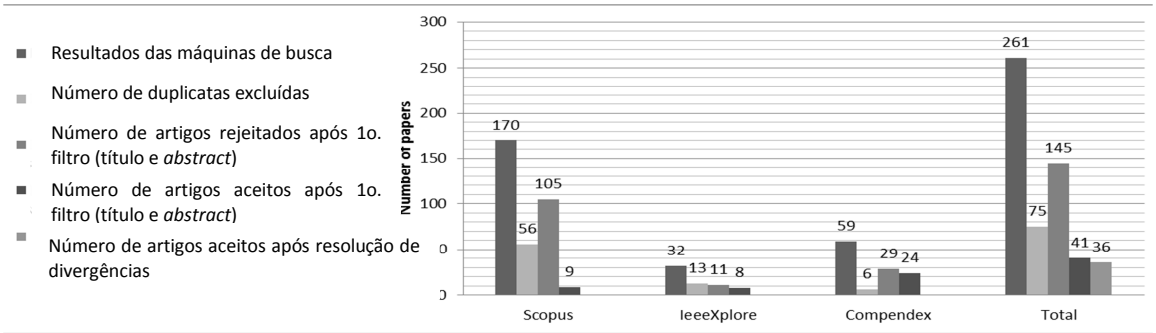


Figura 2.5 - Número de Artigos por Máquina de Busca

Para cada publicação retornada, os dados como título, autor, resumo, palavras-chave e ano de publicação forma extraídos e carregados na ferramenta de apoio à revisão sistemática StArt (*State of the Art through Systematic*) (HERNANDES *et al*., 2012). O uso da ferramenta auxiliou a etapa de seleção, que foi inicialmente conduzida por dois pesquisadores, individualmente, aplicando os critérios de inclusão e exclusão nos dados extraídos das publicações. Após a análise, nove divergências para exclusão foram identificadas e resolvidas por um terceiro pesquisador. Do conjunto de 186 artigos para seleção, 36 estudos foram selecionados para a próxima etapa de análise, com a recuperação do conteúdo completo das publicações e leitura para extração de dados (Figura 2.5). No entanto, não foi possível recuperar o conteúdo completo de três publicações. Além disso, durante a leitura dos 33 restantes, mais oito estudos foram removidos por se aplicarem ao critério de exclusão 1, não apresentando abordagens relacionadas ao escopo desta revisão. Como resultado da busca e seleção de abordagens, vinte e cinco artigos foram selecionados, conforme pode ser observado na Tabela 2.3.

Tabela 2.3 - Publicações selecionadas organizadas por ano

Publicação	Ano
Hollenbach, C. R. (1996, June). Experiences in process domain engineering at PRC Inc. In *International Software Process Workshop,* (pp. 78-78). IEEE Computer Society.	1996
Sutton, S. M., & Osterweil, L. J. (1996). Product families and process families. In *Software Process Workshop, 1996. Proceedings of the 10th International Process Support of Software Product Lines.* (pp. 109-111). IEEE.	1996
Durán, A., Benavides, D., & Bermejo, J. (2004). Applying system families concepts to requirements engineering process definition. In *Software Product-Family Engineering* (pp. 140-151). Springer Berlin Heidelberg.	2004
Jaufman, O. (2005). Emergent process design. In *Proceedings of the 27th International Conference on Software engineering* (pp. 653-653). ACM.	2005
Jaufman, O., & Münch, J. (2005). Acquisition of a project-specific process. In: *Product Focused Software Process Improvement* (pp. 328-342). Springer Berlin Heidelberg.	2005
Rombach, D. (2006). Integrated software process and product lines. In *Unifying the Software Process Spectrum* (pp. 83-90). Springer Berlin Heidelberg.	2006
Washizaki, H. (2006). Building software process line architectures from bottom up. In *Product-Focused Software Process Improvement* (pp. 415-421). Springer Berlin Heidelberg.	2006
Washizaki, H. (2006). Deriving project-specific processes from process line architecture with commonality and variability. In IEEE *International Conference on Industrial Informatics, 2006* (pp. 1301-1306). IEEE.	2006
Simidchieva, B. I., Clarke, L. A., & Osterweil, L. J. (2007). Representing process variation with a process family. In *Software Process Dynamics and Agility* (pp. 109-120). Springer Berlin Heidelberg.	2007

Publicação	Ano
Thränert, M., & Werner, A. (2007). A Process Family Approach for the reuse of development processes. In *Innovations and Advanced Techniques in Computer and Information Sciences and Engineering* (pp. 309-313). Springer Netherlands.	2007
Armbrust, O., Katahira, M., Miyamoto, Y., Münch, J., Nakao, H., & Ocampo, A. (2009). Scoping software process lines. *Software Process: Improvement and Practice*, *14*(3), 181-197.	2009
Martínez-Ruiz, T., García, F., & Piattini, M. (2009). Process Institutionalization using Software Process Lines. In *ICEIS (3)* (pp. 359-362).	2009
Ternité, T. (2009, August). Process lines: a product line approach designed for process model development. In *Software Engineering and Advanced Applications, 2009. SEAA'09. 35th Euromicro Conference on* (pp. 173-180). IEEE.	2009
Aleixo, F. A., Freire, M. A., dos Santos, W. C., & Kulesza, U. (2010). A Model-driven Approach to Managing and Customizing Software Process Variabilities. In *ICEIS (3)* (pp. 92-100).	2010
Aleixo, F. A., Freire, M. A., dos Santos, W. C., & Kulesza, U. (2010). An Approach to Manage and Customize Variability in Software Processes. In *2010 Brazilian Symposium on Software Engineering* (pp. 118-127).	2010
Barreto, A., Duarte, E., Rocha, A. R., & Murta, L. (2010). Supporting the Definition of Software Processes at Consulting Organizations via Software Process Lines. In *Seventh International Conference on the Quality of Information and Communications Technology (QUATIC), 2010 (*pp. 15-24). IEEE.	2010
Barreto, A., Murta, L. G. P., & da Rocha, A. R. C. (2011). Software Process Definition: a Reuse-based Approach. *J. UCS*, *17*(13), 1765-1799.	2011
Hurtado Alegría, J. A., Bastarrica, M. C., Quispe, A., & Ochoa, S. F. (2011). An MDE approach to software process tailoring. In *Proceedings of the 2011 International Conference on Software and Systems Process* (pp. 43-52). ACM.	2011
Martínez-Ruíz, T., García, F., Piattini, M., & Munch, J. (2011). Applying AOSE concepts to model crosscutting variability in variant-rich processes. In *37th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA), 2011* (pp. 334-338). IEEE.	2011
Martínez-Ruiz, T., García, F., Piattini, M., & Munch, J. (2011). Modelling software process variability: an empirical study. *Software, IET*, *5*(2), 172-187.	2011
Jafarinezhad, O., & Ramsin, R. (2012). Development of Situational Requirements Engineering Processes: A Process Factory Approach. In *IEEE 36th Annual Computer Software and Applications Conference (COMPSAC), 2012* (pp. 279-288). IEEE.	2012
Magdaleno, A. M., de Araujo, R. M., & Werner, C. (2012). COMPOOTIM: An Approach to Software Processes Composition and Optimization. In *CIbSE* (pp. 42-55).	2012
Rouillé, E., Combemale, B., Barais, O., Touzet, D., & Jézéquel, J. M. (2012). Leveraging CVL to manage variability in software process lines. In *19th Asia-Pacific Software Engineering Conference (APSEC), 2012* (Vol. 1, pp. 148-157). IEEE.	2012
Hurtado, J. A., Bastarrica, M. C., Ochoa, S. F., & Simmonds, J. (2013). MDE software process lines in small companies. *Journal of Systems and Software*,*86*(5), 1153-1171.	2013
Simmonds, J., Bastarrica, M. C., Silvestre, L., & Quispe, A. (2013). Variability in software process models: Requirements for adoption in industrial settings. In *4th International Workshop on Product Line Approaches in Software Engineering (PLEASE), 2013* (pp. 33-36). IEEE.	2013

As publicações estão distribuídas no período entre 1996 e 2013, com aumento do interesse na área a partir de 2004. Os anos que mais concentraram publicações foram 2006, 2009, 2010, 2011 e 2012, com destaque para 2011 (Figura 2.6).

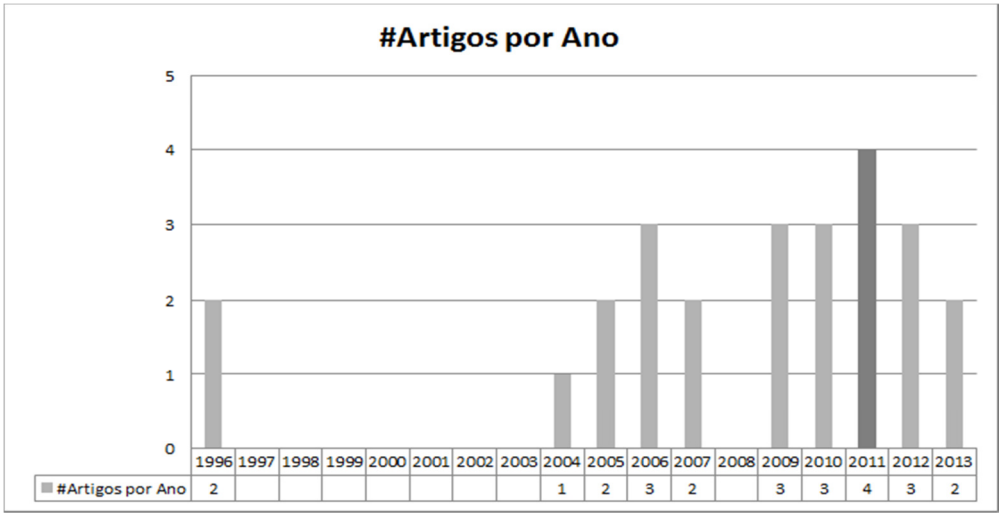


Figura 2.6 – Distribuição dos Trabalhos por Ano

A maioria das publicações se concentrou em conferências (17 de 25 - 68%) e apenas 16% (4 de 25) foram publicadas em revistas, indicando uma área ainda imatura e com poucos trabalhos consolidados pelos pares (Figura 2.7).

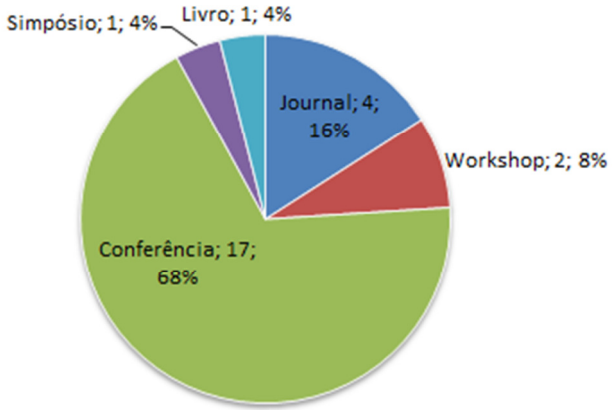


Figura 2.7 – Distribuição dos Trabalhos por Veículos de Publicação

2.3.2. Descrição das Abordagens de Linha de Processos de Software Identificadas

Os primeiros trabalhos, em 1996, são introdutórios a alguns conceitos da área, como SUTTON E OSTERWEIL (1996) que discutem a noção de uma família de processos implícita na noção de uma família de produtos. HOLLENBACH (1996) também não trata especificamente de uma abordagem em si, mas da descrição breve da experiência da prática de Engenharia de Domínio de Processos de Software em uma organização. Tem o objetivo de levantar indícios de que o esforço de Engenharia de Domínio de

Processos pode trazer benefícios para organizações. Um trabalho que foca na descrição de conceitos essenciais envolvidos em Linhas de Processo é o de JAUFMAN E MÜNCH (2005), que propõem um método de aquisição de processos emergentes. A técnica apresentada foca na fase de derivação de um processo específico de projeto com apoio à adaptabilidade e alterações sucessivas para customização do processo de acordo com objetivos e características de contexto. Um ferramental de apoio é mencionado, mas não é de fato apresentado pelos autores. Este trabalho foca no uso de linha de processos, mas não descreve como esta deve ser desenvolvida. O trabalho de ROMBACH (2006) também pode ser citado pela descrição de aspectos importantes da área, onde o conceito de estabelecer Linhas de Produtos e Processos de Software integradas é descrito, destacando que processos de software são variáveis entre projetos e podem ser beneficiados se gerenciados de maneira sistemática, como artefatos de software em LPSs.

DURÁN *et al*. (2004) propõem a aplicação da abordagem de família de processos como um meio para definir um processo central que pode ser adaptado a necessidades específicas das companhias envolvidas de uma maneira controlada e pré-definida. A variabilidade foi analisada nas técnicas, papéis e documentos envolvidos no domínio de Engenharia de Requisitos e foi representada por meio de descrições adicionais em cenários e relações de herança aplicadas em diagramas de classes UML. A abordagem apresenta uma proposta inicial para a fase de desenvolvimento *para* reutilização, mas sem se preocupar em especificar um processo. Não discute o conceito de componentes de processos e o agrupamento dos elementos identificados. Não apresenta continuidade da pesquisa em publicações mais recentes.

WASHIZAKI (2006a) (2006b) define Linha de Processos como um conjunto de processos similares dentro de um domínio em particular ou para um propósito em particular, que possui características comuns e é construída baseada em artefatos de processos comuns e reutilizáveis. De forma similar à Engenharia de Linha de Produtos, a Engenharia de Linha de Processos é dividida em duas fases: (i) ED, que corresponde a coleta de similaridades e variabilidades de requisitos da linha de processos, utilizados no projeto e na implementação da Arquitetura da linha e suas variantes; e (ii) EA, em que um processo específico de projeto é construído por meio da Arquitetura da linha e

suas variantes. O apoio à construção do diagrama de características é limitado e manutenção da consistência entre os modelos não é garantida, não é definido como criar as relações de impacto entre os modelos. A abordagem não discute o conceito de componentes de processos e o agrupamento dos elementos identificados. Não apresenta continuidade da pesquisa em publicações mais recentes.

SIMIDCHIEVA *et al*. (2007) trata variações de processos pela especificação de mudanças de comportamentos dos agentes associados a tarefas; descrição detalhada de tarefas descritas em níveis mais abstratos e alterações na estrutura dos artefatos. Usa a linguagem de modelagem Little-JIL e uma base de ativos para descrever as variantes. A técnica de seleção de variantes mais apropriada é usada de forma manual pelo usuário da abordagem. O trabalho aponta a necessidade de automação para apoio a essa atividade. Outro ponto importante constitui a definição do que seria de fato uma variante de um processo base. Aponta o desenvolvimento de métricas como um meio para definir o grau de similaridade entre os processos e definir o que seria uma variante viável de um processo base. O trabalho não apresenta continuidade de pesquisa em publicações atuais.

THRÄNERT E WERNER (2007) apresenta um conceito que permite reutilizar conhecimento de processos baseado na abordagem de família de processos. É uma publicação de apresentação de um trabalho ainda introdutório que aponta várias melhorias e continuações como conclusão.

ARMBRUST *et al*. (2008) (2009) trabalham o conceito de LPrS como um conjunto de processos de software com um conjunto de características que satisfazem necessidades específicas de uma organização em particular e que são desenvolvidas a partir de um conjunto comum de processos, de acordo com uma forma prescrita. A abordagem descreve um conjunto de atividade para construção e uso da LPrS. A atividade inicial consiste na Definição do Escopo de uma LPrS e é o foco do trabalho. A abordagem SCOPE (ARMBRUST, 2010) é apresentada com maiores detalhes, visando identificar as demandas de processo de produtos e projetos correntes e futuros, análise dos processos existentes para seleção do grupo de processo com melhor benefício para apoio a essas demandas da organização. O trabalho não apresenta continuidade de pesquisa em publicações atuais, não trata com detalhes questões de

modelagem de variabilidades, representação de componentes de processo e provimento de ferramental de apoio às etapas da abordagem.

Alguns dos autores participam de grupos de pesquisa na área de processos de software, como os autores Martínez-Ruiz, García, Piattini que atuam no ALARCOS *Research Group*[1] da *University of Castilla* - La Mancha, Espanha. MARTÍNEZ-RUIZ *et al.* (2013) aplicam o paradigma de processos rico em variantes no cenário de desenvolvimento de software global. Um ciclo de quatro etapas (adaptação, execução, análise e padronização) para transformação e inclusão de processos como novos e efetivos ativos é proposto. O grupo propõe uma representação para a modelagem de processos de software denominada vSPEM, uma extensão do SPEM v2.0 para permitir a especificação de variabilidades em modelos de processos de software. A abordagem não discute o conceito de componentes de processos e o agrupamento dos elementos identificados. Não apresenta continuidade da pesquisa em publicações mais recentes.

A abordagem proposta por TERNITÉ (2009) apresenta conceitos da técnica de reutilização baseada em LPrS. A definição de variabilidades é feita por meio de operações de adaptação: adição/remoção de elementos opcionais, e operações de mudanças pré-estabelecidas em um metamodelo que define os possíveis elementos e as operações passíveis de realização. Neste trabalho, uma LPrS é definida como um conjunto de processos que captura as similaridades e variabilidades controladas. O conceito de arquitetura é definido como arcabouço arquitetural que prove mecanismos para variações controladas. A arquitetura deve permitir a configuração de características mandatórias, opcionais e alternativas que implicam os efeitos definidos por tipos de variabilidades positiva, negativa, extensão e substituição. A abordagem não discute o conceito de componentes de processos e o agrupamento dos elementos identificados. Não apresenta continuidade da pesquisa em publicações mais recentes.

Uma abordagem anotativa foi utilizada para gerência de variabilidades em processos de software e derivação automática de processos, oriundos da customização automática de especificações de famílias de processos de software em ALEIXO *et al*. (2010a) (2010b) (2012) (2013). A abordagem parte da modelagem e definição de uma linha de processo de software, utilizando ferramentas existentes de especificação de

[1] ALARCOS http://alarcos.esi.uclm.es/index_en

processos. Em seguida, concentra-se na gerência automatizada de variabilidades dos elementos da linha de processo sendo modelada, pela especificação dos modelos de variabilidades (modelo de características, modelo de configuração e modelo de processo). Na etapa seguinte, com o auxílio de uma ferramenta de derivação, são selecionadas as características relevantes para um processo, possibilitando a geração automática de processos de software customizados que lidem com cenários e necessidades específicas de um dado projeto. A abordagem também visa especificar o processo customizado como workflow para execução em máquinas de workflow. Uma abordagem que foca em aspectos técnicos que podem dificultar o uso pelo engenheiro de processos ou gerente de projeto devido como, por exemplo, a necessidade de introduzir as anotações diretamente nos arquivos fontes. A abordagem não discute o conceito de componentes de processos e o agrupamento dos elementos identificados. Mais recentemente, o autor atua em parceria no desenvolvimento da abordagem *SMartySPEM* (OLIVEIRA JUNIOR *et al*., 2013) para modelagem de variabilidades de processos. Não apresenta continuidade da pesquisa em publicações mais recentes.

BARRETO *et al*. (2010)(2011) propõem uma abordagem para reutilizar processo de software utilizando componentes de processos. Neste contexto, um componente de trabalho é definido como uma unidade básica de composição de processos, considerada relevante para: (i) ser reutilizada em outras definições de processos; (ii) ter sua estabilidade e desempenho analisado; (iii) ser versionada; e (iv) ter vários tipos de rastreabilidade estabelecido; entre outros. Uma Linha de Processos de Software é definida como uma arquitetura de processos que possui variabilidades ou opcionalidades (BARRETO *et al*., 2009). Uma arquitetura de processos é similar a fluxos de trabalhos, e pode ser composta por componentes de processos, atividades ou qualquer combinação entre eles. Define um “esqueleto” que o processo deve possuir, determinando os principais elementos e como estes se relacionam, sem necessariamente definir como será o detalhamento desses elementos principais. Este trabalho usa o conceito de características de processos como um aspecto, qualidade ou característica que um processo precisa ser compatível (BARRETO *et al*., 2010). Características de processo são usadas apenas como um mecanismo de alto nível para seleção de componentes. O autor menciona que poucos mecanismos de

relacionamentos entre características foram utilizados pela abordagem. Desta forma, a modelagem de variabilidades de uma Linha de Processos de Software é realizada por meio de um modelo de componentes que descreve componentes de processo mandatórios e opcionais, além de representar pontos de variação e variantes. A variabilidade também é trabalhada com a definição de dois tipos de componentes: componentes concretos, que não permitem mais configurações, e componentes abstratos, com definições incompletas que permitem configurações por meio da implementação por componentes concretos. A visualização da LPrS e dos componentes poderia permitir identificar mais facilmente a estrutura interna de um componente e as variantes de um ponto de variação. A abordagem ainda inclui estratégias de definição de processos para e com reutilização. A abordagem não trabalha a variabilidade em elementos internos a um componente de processo como papéis, produtos de trabalho e ferramentas. A abordagem não oferece mecanismos para direcionar a organização e o agrupamento de componentes de processos. Não apresenta continuidade da pesquisa em publicações mais recentes.

Hurtado Alegría, Simmonds, Bastarrica e Silvestre atuam no Projeto Adapte[2] de universidades do Chile. O projeto trata de uma estratégia de adaptação de processo que implica em mover parte do esforço da adaptação para etapa de definição de processo organizacional. O mecanismo proposto aplica os conceitos de LPrS e Engenharia Dirigida por Modelos (MDE – *Model Driven Engineering*). Em HURTADO ALEGRÍA *et al*. (2013), uma abordagem baseada em modelo para adaptação de processos de software é apresentada para suportar a geração automática de processos específicos de projetos com base em processos organizacionais e contextos de projetos. Mais recentemente, o grupo tem desenvolvido ferramental de apoio à definição dos modelos de decisão de variação (SILVESTRE *et al*., 2014) e aplicado o conceito de *megamodeling* para lidar com alguns desafios da aplicação da abordagem na prática (BASTARRRICA *et al*., 2014) (SIMMONDS *et al*., 2015). A abordagem não discute o conceito de componentes de processos e o agrupamento dos elementos identificados. O grupo parece ativo, dando continuidade na pesquisa.

[2] ADAPTE (*Adaptable Domain And Process Transformation Engineering*) http://www.adapte.cl/

JAFARINEZHAD E RAMSIN (2012) apresentam uma abordagem de fábrica de processo baseada em *features* para processos de Engenharia de Requisitos a serem derivados baseados em características de projeto (situações de projetos). Um modelo de *features,* denominado de modelo do domínio de solução*,* é utilizado para representar similaridades e variações entre processos de Engenharia de Requisitos existentes. Um mecanismo de mapeamento entre as situações e as *features* correspondentes é especificado. Os elementos de processo trabalhados na abordagem são fases, estágios e tarefas. A abordagem é proposta para o domínio de Engenharia de Requisitos. O trabalho não apresenta continuidade de pesquisa em publicações atuais, não trata de questões de representação de componentes de processo e provimento de ferramental de apoio às etapas da abordagem.

MAGDALENO *et al*. (2012) apresentam a abordagem COMPOOTIM para apoiar decisões de gerentes de projetos na seleção e combinação de componentes de processos para derivação de um processo específico de projeto. Para uma reutilização e composição de processo sistemática, uma LPrS baseada em contexto (NUNES *et al*., 2010) foi definida como um conjunto de componentes de processo, organizados para representar partes variáveis e comuns dentro de um domínio específico que podem ser reutilizados e combinados, de acordo com regras de composição, para compor e adaptar dinamicamente abordagens de processos que apoiam decisões de gerentes de projetos sobre a seleção e composição de componentes de processo e otimiza a sugestão de processos para um contexto de projeto particular. O foco do trabalho está na fase de derivação de um processo para um projeto específico (MAGDALENO, 2013) . Usando a abordagem contextual, a abordagem é capaz de oferecer informações adicionais a LPrS para apoiar a decisão do gerente de projeto. Para isso, informações de contexto de projetos de software (por exemplo, informações sobre a estrutura organizacional; informações de projeto: tamanho, complexidade, duração, estabilidade; informações sobre equipe: experiência, proximidade geográfica, etc.) devem ser identificadas e registradas junto ao impacto de seleção de variações do domínio de processos analisado. O uso de mecanismos de otimização visa identificar soluções para a derivação de um processo específico de projeto (MAGDALENO *et al*.,

2015). O trabalho apresenta a representação de LPrS utilizando a notação *OdysseyProcess-FEX* (TEIXEIRA, 2011) e componentes de processos de software.

O trabalho de ROUILLÉ *et al.* (2013a) consiste em definir LPrS e os processos a serem reutilizados de acordo com requisitos de projetos. Os processos são associados a componentes que automatizem unidades de trabalho em geral recorrentes e manuais. O trabalho de ROUILLÉ *et al.* (2012) analisa a aplicação de *Common Variability Language*[3] (CVL) no contexto de processos, para a modelagem de variabilidades nos requisitos e suas ligações com o processo. Desta forma, a abordagem propõe que o modelo de variabilidade seja representado separadamente do modelo de processo e relações entre os dois sejam especificadas para definir qual variação deve ser aplicada no modelo de processo base. Em geral, as abordagens de LPr trabalham de forma dependente a um metamodelo de processos, com adaptações deste para incluir mecanismos de variabilidade. Apesar da abordagem proposta alcançar independência de uma metamodelo de processo para modelar a variabilidade, aumenta a complexidade para edição e manutenção de LPrS. A visualização e o entendimento dos pontos de variação versus requisitos de projetos não é tão acessível pela separação dos modelos. Já a associação dos processos aos componentes para automatizá-los, o trabalho (ROUILLÉ *et al.,* 2013b) que associa os componentes a LPrS para identificar os seus contextos de uso. Esta associação permite automatizar um processo derivado a partir da linha.

2.3.3. Análises das Abordagens de LPrS Identificadas segundo os critérios definidos

Considerando as duas grandes etapas de uma Engenharia de LPrS, a Engenharia de Domínio e a Engenharia de Aplicação, desenvolvimento *para* e *com* reutilização, respectivamente, pode-se observar que apenas 40% (10 dos 25) dos estudos apresentam suporte combinando as duas etapas. A etapa de Engenharia de Domínio é priorizada pelos trabalhos selecionados, estando presente em todas as abordagens analisadas.

Da mesma forma que a ED é priorizada, a etapa do ciclo de vida de processos de software mais apoiada é a definição de processos, sendo apontada como foco de

[3] http://www.omgwiki.org/variability/

todas as abordagens analisadas. Apenas poucas abordagens mencionam outras fases como instanciação, execução e melhoria de processos. No entanto, apenas duas abordagens descrevem com um pouco mais de detalhe o apoio à instanciação e execução dos processos derivados. Uma das abordagens aponta um ferramental de apoio à fase de execução (ALEIXO *et al*., 2010b) e a outra indica como tais fases influenciam a evolução das variações de processos registradas e os contextos de aplicação (MARTÍNEZ-RUÍZ *et al*., 2009).

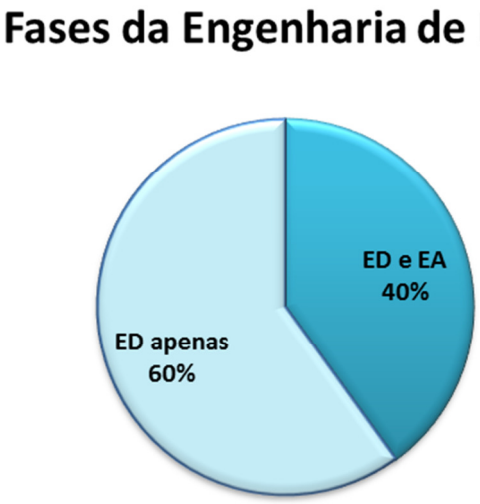


Figura 2.8 – Distribuição dos trabalhos por fase da Engenharia de LPrS

De forma complementar a essa análise, pode-se observar que não há apoio significativo a verificações sintáticas e semânticas das linhas produzidas. O trabalho de BARRETO *et al*. (2010) apresenta um *checklist* de apoio para verificar se a linha definida está adequada aos requisitos estabelecidos.

A definição de um ferramental de apoio visa viabilizar a utilização prática da abordagem, fornecendo maior possibilidade de aplicação em cenários reais. No entanto, das dezesseis abordagens identificadas, apenas sete mencionam algum apoio ferramental a etapas da abordagem proposta. A maioria não trata da implementação de ferramental próprio para disponibilização de um ambiente de reutilização integrado. A maior parte promove a adaptação de ferramentas existentes para apoiar alguma parte da abordagem.

ARMBRUST *et al*. (2009) apontam o uso da ferramenta SPEARMINT™[4] para apoio à modelagem das variações dos processos. Uma cadeia de ferramentas é

[4] Spearmint, http://www.iese.fhg.de/fhg/iese/research/quality/pam/index.jsp.

apontado por HURTADO ALEGRÍA *et al*. (2011) para apoio à abordagem com o uso do *Eclipse Modeling Framework* – EMF 3.4 e ATL *plug-in* 2.0. SIMMONDS *et al*. (2013) mencionam o uso de EPF Composer[5] para apoio a modelagem de processo usando SPEM e SPLOT[6] na modelagem de *features*. Mais recentemente, tem sido desenvolvido um ferramental de apoio à definição dos modelos de decisão de variação na abordagem proposta pelo grupo de HURTADO ALEGRÍA, visando diminuir a dificuldade de interação com a abordagem de transformações de modelo e linguagens de baixo nível de abstração como ATL (SILVESTRE *et al*., 2014). BARRETO *et al*. (2010) apontam um sistema WEB como apoio para definir componentes de processos, associar características, definir LPrS e derivar processos a partir das linhas definidas. Este ferramental é um dos mais próximos a um ambiente de reutilização integrado em uma única ferramenta. ALEIXO *et al.* (2010b) utilizam uma cadeia de ferramentas para validar a proposta: (i) os processos de software são especificados usando EPF (*Eclipse Process Framework*)[7], (ii) o gerenciamento de variabilidades é implementado por uma extensão de uma ferramenta de LPS denominada GenArch[8], e (iii) linguagens de transformação como ATL e Acceleo são adotados para transformar modelos de processo para a linguagem jPDL, para possibilitar a execução do processo derivado usando a máquina de processo JBPM[9]. ROUILLÉ *et al*. (2013a) apresentam a descrição de uma ferramenta construída para apoiar a utilização da abordagem. O EMF é utilizado para construir três editores dos modelos de processo. Além disso, uma máquina para derivação da CVL para modelar variações também foi desenvolvida. Ferramentas para associar os componentes às unidades de trabalho descritas na LPrS e apoiar a derivação do processo para execução também foram desenvolvidas. Uma limitação é a dependência de implementação dos componentes aos processos ao metamodelo usado para definir a LPrS. O trabalho de MAGDALENO *et al*. (2012) apresenta a ferramenta Odyssey[10] para modelagem da LPrS e apoio à etapa de

[5] EPF Composer: http://www.eclipse.org/epf

[6] SPLOT: http://www.splot-research.org/

[7] Eclipse Foundation, Eclipse Process Framework (EPF) Composer 1.0 Architecture Over-view (2010), http://www.eclipse.org/epf/composer_architecture/.

[8] GenArch Plugin. A model-based Product Line Derivation Tool (2009), http://www.teccomm.les.inf.puc-rio.br/genarch/

[9] JBPM (2009), http://labs.jboss.com/jbossjbpm/

[10] ODYSSEY: Odyssey SDE Homepage, http://reuse.cos.ufrj.br

Engenharia de Domínio. Outra ferramenta apoia a complementação da definição da linha e apoio à etapa de Engenharia de Aplicação com aplicação dos algoritmos de otimização para derivação de processos específicos de projetos a partir da linha derivada.

As avaliações das abordagens podem ser classificadas como exemplos, estudos de caso, aplicação prática na indústria, comparação, *survey*, avaliação de especialista, entre outros. Os tipos mais comuns encontrados nas abordagens identificadas são exemplos e estudos de caso, apresentando a avaliação da abordagem em contextos específicos e controlados. Das publicações analisadas, dezoito das vinte e cinco ilustram a abordagem proposta por meio de exemplos em diferentes cenários: (i) desenvolvimento de processos de software; (ii) foco em processos de Engenharia de Requisitos; (iii) processo de *hardware* e *software co-design*; (iv) foco na disciplina de gerenciamento de projetos, dentre outros.

A ilustração de aplicação da abordagem na indústria é um tipo importante de avaliação que está começando a ser utilizada mais recentemente, demonstrando uma evolução da área de pesquisa e aumento da maturidade das abordagens. A abordagem proposta por HURTADO ALEGRÍA *et al*. (2011) foi aplicada em processos de Engenharia de Requisitos de uma organização chilena de médio porte e em duas organizações de pequeno porte (HURTADO ALEGRÍA *et al*., 2013). Tais organizações proveem seus processos organizacionais como parte do projeto Tutelkán[11]. O projeto trata de uma estratégia comunitária de melhoria de processos de software em organizações chilenas para compartilhamento de recursos para projeto e adaptação de artefatos de processos de software reutilizáveis (ALEGRÍA *et al*., 2010).

O trabalho de ROUILLÉ *et al*. (2012) aplica a abordagem em uma organização chamada Sodifrance[12] para gerenciar as variações de processos de software simplificado de projetos de desenvolvimento utilizando Java.

O trabalho desenvolvido por MAGDALENO *et al*. (2012) foi aplicado no contexto de uma grande organização de petróleo e gás brasileira, avaliando a composição de

[11] http://www.tutelkan.info
[12] http://www.sodifrance.fr/

processos em três projetos de desenvolvimento de software (MAGDALENO *et al*., 2015).

O trabalho de JAUFMAN E MÜNCH (2005) apresenta uma análise comparativa com o método de adaptação V Model. Três critérios são usados para análise quantitativa e qualitativa (esforço – tempo necessário para aplicação da abordagem; qualidade dos produtos derivados; satisfação da equipe envolvida).

Uma análise de quatro abordagens para especificação e gerenciamento de variabilidades é apresentada em (SIMMONDS *et al.,* 2013). Sendo este um aspecto importante para adoção de LPrS, os autores analisam os benefícios e dificuldades envolvidas no uso de primitivas de propósito geral (modelo de *features* e OVM) e específicas de domínio (SPEM e vSPEM) para tratamento de variabilidade. Cinco critérios são analisados: expressividade da notação, capacidade de entendimento da notação, usabilidade e disponibilidade de ferramenta para apoio da modelagem, e nível de interoperabilidade da ferramenta com outras usadas por uma organização. Os autores chegam a conclusão que nenhuma das abordagens consegue satisfazer completamente os requisitos listados e analisados pelos critérios.

BARRETO *et al.* (2011) aplicou um *survey* para coletar a opinião de engenheiros de processo sobre o conjunto de benefícios e dificuldades esperadas ao aplicar abordagens de reutilização de processo de software. Linhas de Processos de Software foram apontadas como uma das abordagens mais promissoras neste cenário e a maior preocupação está na falta de suporte ferramental.

Os resultados analisados apresentam indícios do aumento da maturidade da área pesquisada. Extensões de pesquisa e desafios estão sendo trabalhados, como interação com projetos reais e análise da viabilidade de aplicação do conceito de reutilização de processos na indústria. Ainda pode-se observar que alguns aspectos importantes ainda não foram completamente atendidos como a questão da representação e gerenciamento de variabilidades e aplicação dos conceitos de componentes de processos de software. Esses itens são analisados nas seções a seguir.

2.4. VARIABILIDADE DE PROCESSOS DE SOFTWARE NAS ABORDAGENS DE LPRS

Assim como em Linha de Produtos de Software (LPS), o conceito de variabilidade está fortemente presente na técnica de Linha de Processos de Software (LPrS). A variabilidade em uma família de processos visa explicitar as similaridades presentes e os pontos em que tais processos se diferenciam entre si. O conceito de opcionalidade, que indica a necessidade ou não da presença de um elemento em processo específico, também é de extrema relevância na identificação e representação das variabilidades de uma família de processos.

As abordagens de LPrS identificadas foram analisadas segundo quatro critérios focados no apoio à representação de variabilidades em famílias de processos: (1) identificação do uso de alguma notação existente ou de alguma proposta de notação; (2) identificação dos elementos de processos que foram considerados pela abordagem como passíveis de sofrer variação; (3) identificação do tipo de variação utilizado, principalmente, opcionalidade e variabilidade; e (4) identificação do tipo de restrições e regras de consistência que foi definido.

As notações de modelagem mais referenciadas foram (Figura 2.9): modelos de características (8 das 16 abordagens); SPEM ou alguma derivação (5 de 16 abordagens); e UML (2 de 16 abordagens). Quatro abordagens não mencionam nenhuma notação para modelagem da LPrS (JAUFMAN, 2005, JAUFMAN E MÜNCH, 2005, HOLLENBACH, 1996, ROMBACH, 2006, ARMBRUST *et al*., 2009). Sete abordagens apresentam outra representação como um modelo de componentes ou notações próprias como Little-JIL (SIMIDCHIEVA *et al*., 2007). O uso de BPMN ou derivação foi apontado por apenas uma abordagem (ALEIXO *et al*., 2010a). Sete abordagens usam combinações de propostas de representação para a modelagem de uma linha.

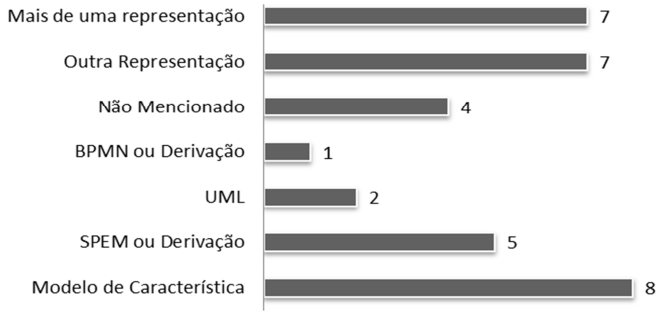


Figura 2.9 – Distribuição dos trabalhos por fase da Engenharia de LPrS

Os elementos de processos mais citados como passíveis de sofrer variação foram: atividades (15 citações); tarefas (10 citações); papéis (9 citações) e produtos de trabalho (10 citações) (Figura 2.10). Ferramentas são citadas como elementos em apenas três abordagens (MARTÍNEZ-RUÍZ *et a*l., 2011a e 2011b; MAGDALENO *et al*., 2012; ROUILLÉ *et al*., 2012) e passo em apenas uma (ALEIXO *et al*., 2010a e 2010b).

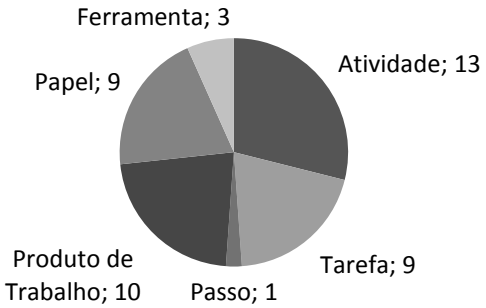


Figura 2.10 - Elementos de Processo apresentados para Modelagem de Variabilidade

Uma análise mais detalhada dos elementos de processos tratados ou ao menos mencionados pelas principais abordagens que tratam esse aspecto é apresentada na Tabela 2.4.

Apenas seis abordagens apresentam algum suporte para representação de fluxo de controle, em geral, utilizando SPEM (ou alguma derivação) ou UML (diagrama de atividades). No entanto, apesar de em fluxos de dados e controles apresentarem a possibilidade de assumir aspectos opcionais em uma família de processos, as abordagens, em sua maioria, não fornecem tratamentos dessa opcionalidade em suas representações. O foco maior é o tratamento de variações em atividades e tarefas.

Dois tipos principais de classificação de variabilidade podem ser observados: (1) elementos classificados como opcionais (nove) ou elementos classificados como pontos de variação (dez). Elementos opcionais tratam da não obrigatoriedade de determinado elemento no domínio. Ponto de variação descreve pontos de configuração no domínio onde alternativas podem ser selecionadas.

Tabela 2.4 - Análise dos elementos de processos passíveis de variação e a notação utilizada nas abordagens de LPrS

Abordagem	Elementos de Processos que podem sofrer variações	Notação
DURÁN *et al.* (2004)	Atividades; papéis e *templates* de documentação	Utilizando-se de descrições adicionais em formulários de cenários. Uso de diagramas de classes UML para expressar as variantes possíveis (Uso de herança).

Abordagem	Elementos de Processos que podem sofrer variações	Notação
WASHIZAKI (2006a)(2006b)	Atividades, artefatos de entrada e saída, e papéis.	Extensões do SPEM 1.1. Modelagem de características baseada no FODA. Não descreve as adaptações das notações adotadas.
SIMIDCHIEVA *et al*. (2007)	Agentes; tarefas/passos; artefatos.	Little-JIL
THRÄNERT E WERNER (2007)	Atividade e Tarefa	Modelagem de características
ARMBRUST *et al.* (2008) (2009)	A modelagem de variabilidades não é o foco do trabalho.	Ferramenta gráfica de modelagem de processos de software SPEARMINT™.
TERNITÉ (2009)	Atividades, artefatos e papéis. Relações entre elementos de processos.	Metamodelo V-Modell XT 1.3
ALEIXO *et al*. (2010a)(2010b)	Atividades, tarefas, passo, artefatos de entrada e saída, e papéis.	Modelo de características, Variação do SPEM e uso da linguagem jDPL
BARRETO *et al*. (2010) (2011)	Componente de processo (subprocesso e atividade) e tarefa	Notação própria para modelagem de LPrS por meio de modelo de componente.
HURTADO ALEGRÍA *et al*. (2011) (2013) **ALEGRÍA E BASTARRICA (2012)** **SIMMONDS et al. (2011) (2012)** **SIMMONDS E BASTARRICA (2011)**	Atividade, tarefa, papel e produto de trabalho.	SPEM 2.0 para representar o processo organizacional (eSPEM) e modelagem de características sem notação específica. O modelo de processo geral utiliza a notação SPEM 2.0 com a ferramenta EPFC (*Eclipse Process Framework Composer*). A variabilidade é representada por meio um modelo de características com a ferramenta SPLOT. Como forma ilustrativa da abordagem, a variabilidade do processo geral foi demarcada manualmente. O terceiro modelo que trata da consistência entre os modelos anteriores utiliza a ferramenta Modisco/AMW.
MARTÍNEZ-RUÍZ *et al*. (2008) (2011b)	Atividade, tarefa, produto de trabalho e papel.	vSPEM - Extensão do SPEM com elementos para tratamento de variabilidade. Modelagem de variabilidades usando modelo de características.
JAFARINEZHAD E RAMSIN (2012)	Atividade, tarefa e papel	Modelo de situação e modelagem de características
MAGDALENO *et al*. (2012)	Atividade, tarefa, produto de trabalho, papel e ferramenta	Modelagem de Característica (*OdysseyProcess-FEX*), modelagem de componentes e de contexto.
ROUILLÉ *et al*. (2012)	Atividade, tarefa, produto de trabalho, papel e ferramenta	CVL – *Common Variability Language*, mapeamento entre requisitos de projeto e elementos de processo (VRM) e *feature model* (RM)

Relações de dependência e mútua exclusividade são restrições importantes em um domínio de processos para caracterizar quando elementos devem ser selecionados em conjunto ou não podem ser utilizados na mesma derivação, respectivamente. Este tipo de restrição foi citado em dez abordagens. No entanto, a maioria das abordagens não menciona suporte a representação de restrições ou regras para verificação de consistência entre elementos da linha modelada. Modelos de contexto são apresentados como representações complementares em duas abordagens (MAGDALENO *et al*., 2012, HURTADO ALEGRÍA *et al*., 2013).

Algumas notações foram propostas especificamente para tratar da representação de variabilidades em processos de software. O metamodelo SPEM (*Software & Systems Process Engineering MetaModel 2.0*) (OMG, 2008) foi proposto pela OMG (*Object Management Group*) com os meios necessários para modelar, documentar, apresentar, gerenciar e instanciar métodos e processos de desenvolvimento. A especificação separa conteúdo de métodos reutilizáveis da sua aplicação em processos. A representação de variabilidade é tratada por mecanismos de extensão com quatro tipos de relacionamentos: (1) Contribuição (*contributes*); (2) Substituição (*replaces*); (3) Extensão (*extends*); e (4) Extensão-substituição (*extends-replaces*). Essa abordagem apresenta um conjunto de limitações apontados por (MARTÍNEZ-RUIZ *et al*., 2008), como a não possibilidade de representar os pontos comuns a todos os processos de uma linha, ou seja, as características do núcleo da linha não podem ser especificadas. Desta forma, não é possível garantir a manutenção dos objetivos dos processos.

PILLAT *et al*. (2012) apresentam uma proposta para representar mecanismos de adaptação de processos de software no metamodelo BPMN 2.0 (OMG, 2011). A extensão, denominada BPMNt, acrescenta mecanismos de adaptação de processos similares aos apresentados pelo SPEM 2.0 sem modificar a semântica original da especificação BPMN (PILLAT *et al*., 2015). O objetivo é acrescentar a área de processos de software uma notação, apesar de não focada na modelagem desse tipo de processo, com a vantagem de possuir ferramentas e técnicas para otimização, execução e simulação de processos. A proposta trabalha com algumas das operações

diretas descritas na Seção 2.2, suportando a habilidade para remover, substituir e adicionar elementos de processos. Desta forma, a um modelo de processo base modelado segundo a especificação BPMN 2.0, são acrescentados as adaptações possíveis em um segundo modelo, que utiliza os conceitos definidos pela BPMNt. Por fim, um conjunto de regras de ações são aplicadas para interpretar as relações de adaptação e gerar o modelo de processo adaptado final. Uma ferramenta foi desenvolvida para suportar a especificação de adaptação usando BPMNt, que estende o projeto MDT/BPMN2[13]. A modelagem proposta busca identificar diretrizes de adaptação por meio da aplicação de ações de remoção, adição e substituição de elementos em um modelo base. Desta forma, cada especificação representa uma variante do processo base a ser derivada.

Outra modelagem proposta na abordagem de BARRETO (2011) representa uma LPrS por meio de uma arquitetura de processos, um "esqueleto" de processo com seus principais elementos e suas relações. A modelagem é realizada por um modelo de componentes que descreve componentes de processo mandatórios e opcionais, além de representar pontos de variação. A variabilidade é trabalhada com a definição de componentes concretos, que não configuráveis, e componentes abstratos, que permitem configurações pela implementação por componentes concretos. O conjunto de elementos usados na modelagem e a notação são mostrados na Figura 2.11 (CARDOSO, 2012). Características de processo são associados como um mecanismo para seleção de componentes, podendo ser entendidas como um conjunto de regras (necessidades de processos) que guiam a definição de processos.

Essa abordagem apresenta uma representação explícita dos conceitos de variabilidades e opcionalidades, mas apenas para um grupo de elementos de processos (alto nível de granularidade – componentes e tarefas). A análise de opcionalidades e variabilidades em elementos como papéis, artefatos e ferramentas não é tratada nem expressa graficamente. Relações de dependência e mútua exclusividade não são amplamente exploradas.

[13] MDT/BPMN2 Project. http://wiki.eclipse.org/MDT/BPMN2.

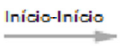
Notação Utilizada	Nome do Elemento	Descrição
	Componente de processo concreto obrigatório	Representação gráfica de elementos não configuráveis (variante).
	Componente de processo concreto opcional	Representação gráfica da opcionalidade de elementos não configuráveis (variante).
	Componente de processo abstrato obrigatório	Representação gráfica de pontos de variação.
	Componente de processo abstrato opcional	Representação gráfica da opcionalidade de pontos de variação.
	Atividade obrigatória	Representação gráfica de uma atividade obrigatória em uma arquitetura de processos.
	Atividade opcional	Representação gráfica de uma atividade opcional em uma arquitetura de processos.
	Tipo de conexão opcional entre elementos de processo	Representação gráfica do relacionamento opcional entre elementos de processos. Nesse caso, uma relação do tipo "Fim-Início".
	Tipo de conexão obrigatória entre elementos de processo	Representação gráfica do relacionamento obrigatório entre elementos de processos. Nesse caso, uma relação do tipo "Início- Início".
	Item início da arquitetura	Representação gráfica do ponto de início dentro de uma arquitetura.
	Item fim da arquitetura	Representação gráfica do ponto de encerramento dentro de uma arquitetura.

Figura 2.11 - Descrição dos elementos gráficos notação BARRETO (2011) - Fonte (CARDOSO, 2012)

A abordagem CASPER (HURTADO ALEGRÍA *et al*., 2013) propõe a representação das informações da LPrS em três modelos: modelo contextual, modelo de características e modelo de processos com variabilidades. O modelo contextual representa informações do contexto de diferentes projetos relacionados à linha. O modelo define atributos de elemento de contexto com valores associados. O modelo de característica é definido por meio de um metamodelo que contempla os conceitos da notação de CZARNECKI E ANTKIEWICZ (2005) com os estereótipos do SPEM 2.0. O modelo de variabilidade é uma representação do SPEM 2.0 com mecanismo alternativo e de opcionalidade introduzido aos elementos de processo atividade, tarefa, papel e produto de trabalho (Figura 2.12). Nenhuma representação gráfica é descrita, nem tratamento de variações nos comportamentos dos processos em termos de fluxos de controle e de dados.

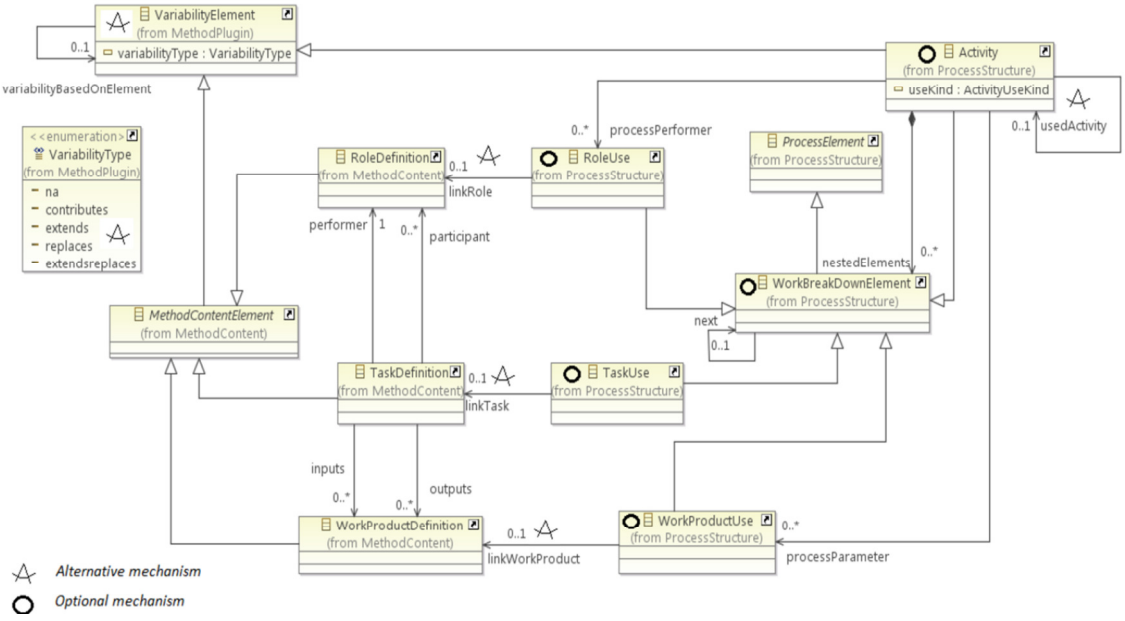


Figura 2.12 – Metamodelo eSPEM (ALEGRÍA E BASTARRICA, 2012)

A abordagem vSPEM (MARTÍNEZ-RUIZ *et al*., 2008, MARTÍNEZ-RUIZ *et al*., 2011b) apresenta uma extensão do SPEM v2.0 para permitir a especificação de variabilidades em modelos de processos de software. Para isso, a abordagem acrescenta ao metamodelo SPEM um pacote denominado *ProcessLineComponents,* que trata da criação de elementos para trabalhar variabilidade nos modelos. As relações entre elementos classificados como pontos de variação e variantes são realizadas por uma relação denominada Ocupação. Essa relação pode ser opcional, mandatória ou alternativa e possui um conjunto de restrições associadas para determinar o tipo de elemento variante que pode vir a ocupar um ponto de variação. Dependências podem ser expressas por relações entre os elementos ditos participarem de uma LPrS (pontos de variação e variantes). Essa relação pode ser inclusiva ou exclusiva. Um conjunto de novos ícones é proposto à notação SPEM para viabilizar a representação explícita de tais conceitos (Figura 2.13).

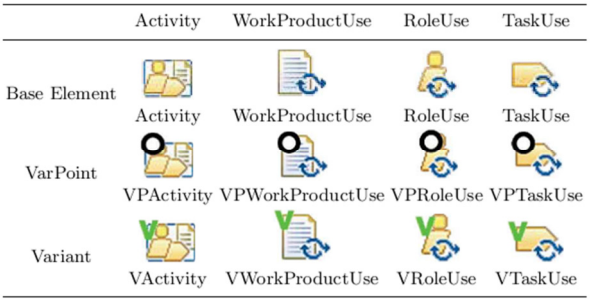


Figura 2.13 - Ícones gráficos vSPEM (MARTÍNEZ-RUIZ *et a*l., 2008)

Desta forma, as variações entre os diferentes processos que compõem a linha são gerenciadas segundo a criação de pontos de variação e tratamento da ocupação por variantes. As classificações de opcionalidades apesar de atribuídas a diversos elementos não possui representação gráfica explícita. Definições de relações de dependência e mútua exclusividade ficam restritas a elementos pertencentes a relacionamentos de variabilidade. O tratamento do impacto de variações no comportamento do processo por meio dos fluxos de controle e de dados não é mencionado na proposta.

Outras limitações dessa abordagem foram apontadas em (PAZIN, 2012), que propõe a abordagem SMartySPEM para dar suporte ao gerenciamento de variabilidades em LPrS. Baseada nas abordagens SMarty (OLIVEIRA JUNIOR *et al*., 2010; FIORI *et al*., 2012) e SPEM 2.0, propõe extensões que permitem representar variabilidades em processos de software. Tal abordagem é composta por um perfil UML, denominado *SMartySPEMProfile,* e um conjunto de diretrizes que guiam o usuário para identificar e representar variabilidade em elementos de processos de software (atividade, tarefa, passo, papel e artefato).

O SMartySPEMProfile define um conjunto de estereótipos e meta-atributos para representar variabilidades em modelos de LPrS (Figura 2.14). O conceito de variabilidade é tratado como um estereótipo (<<*variability*>>) utilizando uma extensão da metaclasse *Comment*. Possui um conjunto de atributos que permite relacionar o número mínimo e máximo de variantes que podem solucionar um ponto de variação; o momento em que a variabilidade deve ser resolvida; e o conjunto de variantes associadas. O conceito de ponto de variação, que estabelece locais específicos de elementos de processos onde ocorre variação, pode ser atribuído a nós de decisão; papéis; tarefas e artefatos. O conceito de variante pode ser atribuído a atividades; papéis; tarefas; passos e artefatos. Um pouco diferente do conceito tradicional em que variante é designada apenas como alternativa para ponto de variação, a abordagem classifica variante como elementos de processos que promovam algum tipo de variação na derivação de processos específicos a partir de uma LPrS. Variantes podem ter como classificação adicional o tratamento de opcionalidade por meio dos estereótipos <<*mandatory*>>: para designar elementos de processos obrigatórios que

devem ser parte de todos os processos específicos derivados de uma LPrS; e <<*optional*>>: para designar elementos de processos opcionais que podem ou não ser parte de processos específicos derivados de uma LPrS. Para variantes associadas à resolução de pontos de variação, os estereótipos <<*alternative_OR*>> e <<*alternative_XOR*>> podem ser utilizados para especificar que diferentes combinações de elementos de processos podem ser aplicadas ou que apenas um único elemento de um conjunto associado pode ser aplicado, respectivamente. O relacionamento de dependência entre dois elementos de processos é representado pelo estereótipo <<*requires*>> e o relacionamento de mútua exclusividade entre dois elementos de processos é representado pelo estereótipo <<*mutex*>>.

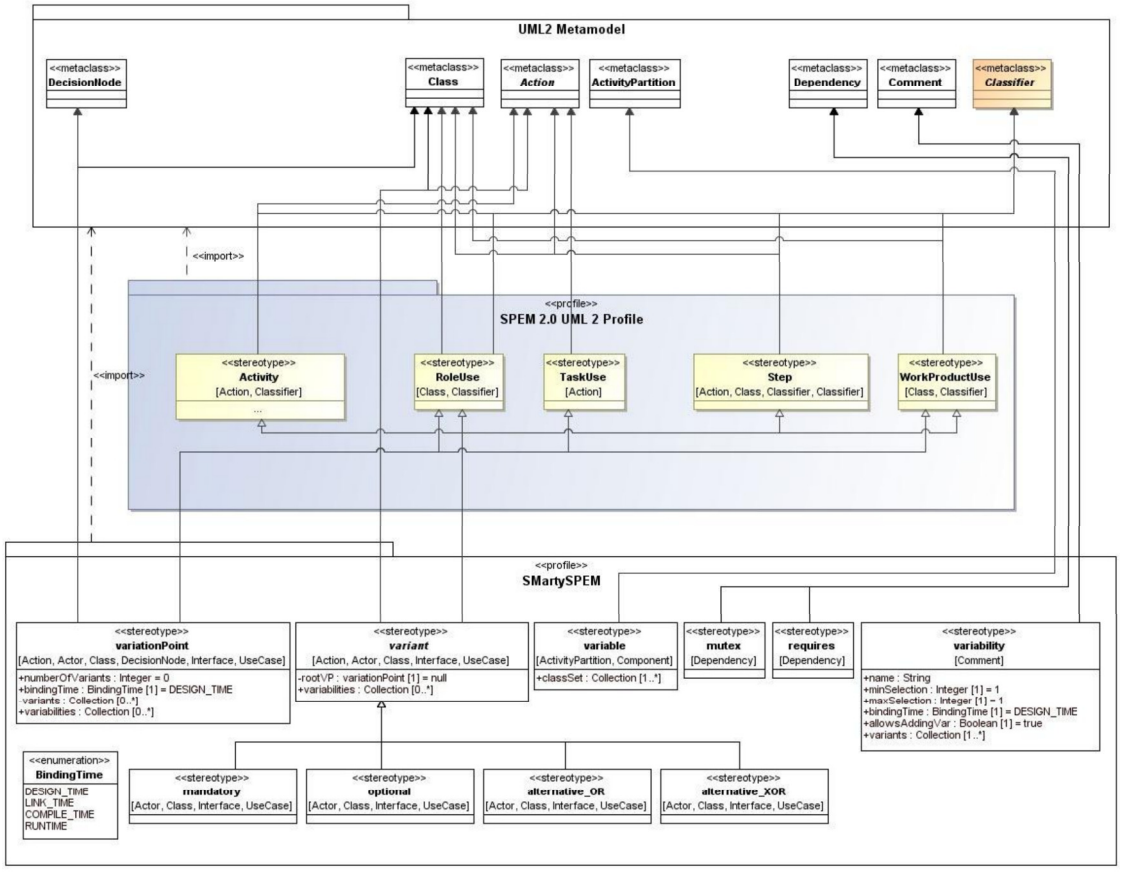


Figura 2.14 - Estereótipos e meta-atributos do *SMartySPEMProfile* (PAZIN, 2012)

Um conjunto de diretrizes é definido com o objetivo de auxiliar na identificação e representação de variabilidades em LPrS. Um primeiro conjunto de diretrizes foca no tratamento de variabilidades com diagramas de atividades UML. Trabalha a variação em termos de atividades que podem ser consideradas variantes dentro do domínio de aplicação de um processo de software. O segundo conjunto trata das variabilidades

dentro da modelagem com SPEM 2.0. Um exemplo do uso da abordagem na representação de uma LPrS hipotética pode ser encontrada na Figura 2.15.

Esta notação não trabalha os conceitos de opcionalidade e variabilidade de forma ortogonal. A classificação de opcionalidade não é atribuída a elementos classificados como alternativas a pontos de variação na LPrS. Algumas representações gráficas tiveram seu conceito inicial modificado, como é o caso do ponto de decisão representando um elemento de processo como uma atividade, o que pode causar alguma dificuldade no entendimento e aplicabilidade. O elemento passo, que indica uma ação, também pode ser classificado como ponto de variação, atribuindo variabilidade a um baixo nível de granularidade. No entanto, não fica claro como os múltiplos comportamentos do processo serão afetados por tantos níveis de variação. O elemento ferramenta não foi contemplado na abordagem.

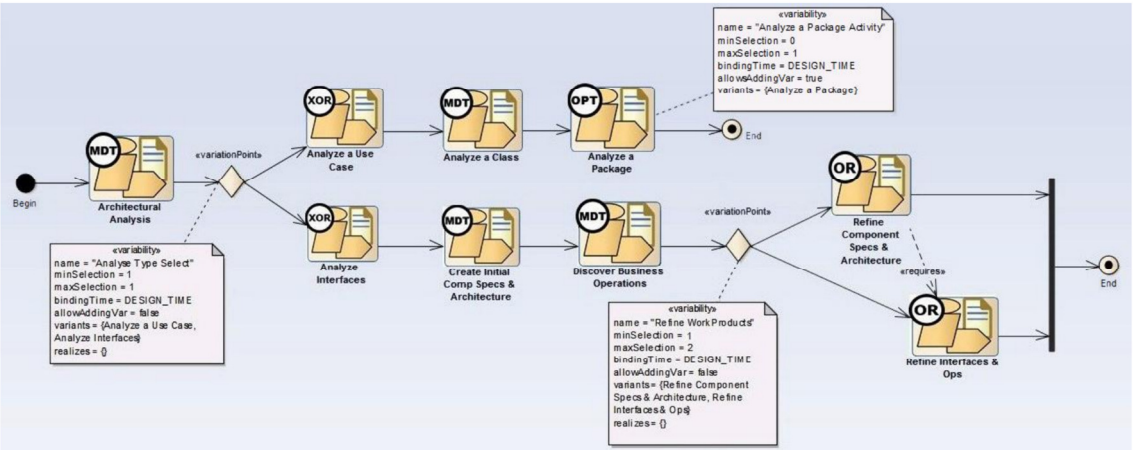


Figura 2.15 - LPrS hipotética extraída de (PAZIN, 2012)

Um estudo recente (MARTÍNEZ-RUIZ *et al*., 2012) mostrou a necessidade de uma notação para representar variabilidades em processos, devido à falta de representações com capacidade de tratar a riqueza semântica envolvida em uma família de processos e o conjunto de variações que a constituem.

Resumindo, nenhuma das representações identificadas nas abordagens analisadas fornece um apoio adequado a modelagem e variabilidades considerando os diferentes elementos de processos que podem sofrer variação; a representação das variações de forma explícita; representação de regras de composição e consistência; considerando variações em termos estruturais (elementos e relações) e comportamentais (fluxos de dados e controle) de processos de software.

2.5. COMPONENTES DE PROCESSOS DE SOFTWARE NAS ABORDAGENS DE LPRS

Segundo HUMPHREY (1989), uma organização geralmente necessita de diferentes tipos de processos. Essa família de processos deve incluir uma arquitetura de processos, uma biblioteca de definições de processos reutilizáveis e vários *templates*, padrões e formas. A arquitetura de processo deve relacionar todos os membros dessa família de processos e deve conter as seguintes seções: definições de elementos de processos maiores e suas funções; formatos de padrões de processos; especificações de interface e regras de composição e adaptação. A biblioteca de elementos de processos reutilizáveis deve ser usada como fonte de blocos de construção comuns para construir múltiplas definições de processos. Esses processos devem ser construídos de forma consistente com a arquitetura.

A construção de modelos de processos como conjuntos de componentes que interagem permite interoperabilidade por meio do encapsulamento da semântica de representações existentes. Uma abordagem baseada em componentes evita uma profunda integração de modelos semânticos; controla a complexidade natural de processos de software; responde a dinamicidade de processos de software e facilita a reutilização (GARY *et al.,* 1998). Assim como em reutilização de produto, não é possível reutilizar planos de projetos inteiros em múltiplos projetos, dado um conjunto de parâmetros de projetos. No entanto, se elementos discretos, como componentes de software, são construídos com estratégias de reutilização, passa a ser possível integrá-los para criar planos de desenvolvimento de software para múltiplos projetos (BHUTA *et al*., 2005).

Componentes de processo devem ser compreensíveis, ou seja, devem possuir um conjunto de informações que descrevam suas funcionalidades, o valor ou resultados providos, sua forma de execução e os recursos necessários para sua execução. Componentes também devem ser independentes, podendo ser executados sem outros elementos. Além disso, devem ser robustos, sendo tolerantes a falhas (BHUTA *et al*., 2005).

Do conjunto de abordagens de LPrS analisado, apenas duas abordagens descritas em três artigos apresentaram conceitos relacionados a área de DPBC

(BARRETO *et al.*, 2010)(BARRETO *et al.*, 2011)(MAGDALENO *et al.* 2012). O conceito de Arquitetura de LPrS foi citado em (ALEIXO *et al*., 2010b)(WASHIZAKI, 2006a)(WASHIZAKI, 2006b).

Em WASHIZAKI (2006a), uma Arquitetura de LPrS é definida como uma estrutura de processo que reflete a similaridade e variabilidade em uma coleção de processos que constrói uma linha de processo a partir da perspectiva de uma otimização total. O conceito de componente não é abordado.

MAGDALENO *et al*. (2012) definem uma arquitetura de processo o esqueleto que um processo deve ter, estabelecendo os principais componentes, suas propriedades internas e interfaces, e como estes devem estar relacionados. Um componente de processo organiza as partes comuns e variáveis dentro de um domínio específico que pode ser reutilizado e combinado, de acordo com regras de composição, para dinamicamente compor e adaptar processos.

BARRETO *et al*. (2011) definem um componente como um elemento de processo com uma arquitetura interna, composto por outros componentes que podem não estar completamente definidos para permitir algum tipo de adaptação (variabilidade). Acrescenta que um componente é considerado algo válido para reutilização, análise de estabilidade ou versionamento de desempenho. Nesta abordagem, uma LPrS é considerada um tipo de arquitetura de processo quando é capaz de gerar diferentes definições de processos, pois possui componentes abstratos e elementos opcionais (BARRETO *et al*., 2010)

Uma análise de trabalhos que apresentam uma abordagem de definição de processos de software baseada em componentes foi realizada para identificar como o conceito de componente está sendo trabalhado (Tabela 2.5).

Tabela 2.5 - Análise de Abordagens de Definição de Processos de Software baseada em Componentes

Abordagem	Abordagem para Composição de Processos de Software
BARRETO *et al*. (2009) (2010) (2011)	Abordagem composta por 4 atividades: (1) Definir componentes de processos; (2) Definir características de processo; (3) Definir uma LPrS ou arquitetura; (4) Aprovar inclusão de elementos reutilizáveis na biblioteca de componentes.
***Open Process Component* (OPC) GARY *et al.* (1998) GARY E LINDQUIST (1999a) (1999b)**	*Open Process Components (OPC) framework* Divide a informação da abordagem de composição de processos em três maneiras: Esquema de processo (possíveis entidades e relações); estados de processo (transições do comportamento dinâmico do componente: executando, suspenso e abortado) e implementação de processo (representação executável de um componente).

Abordagem	Abordagem para Composição de Processos de Software
RU-ZHI *et al*. (2003) (2005) (2010)	Definição de um arcabouço para melhoria de processo baseado em reutilização. Os componentes devem ser recuperados e, posteriormente, adaptados para o contexto atual.
BHUTA *et al*. (2005)	Descreve atividades para construção e uso de elementos de processos, incluindo a definição de atributos de qualidade dos elementos e meios para recuperação dos elementos apropriados a um contexto. Cada elemento de processo construído é descrito usando o modelo 3C (Conceito, Conteúdo e Contexto) e a abordagem de Similaridade e Variabilidade com a classificação de variante e invariante das atividades.
EPC **DAI E LI (2007a) (2007b)** **DAI *et al.*(2008)**	Uma abordagem *top-down* para composição de processos baseada em Arquitetura de Processos foi definida em cinco passos: (1) Elicitação e Análise de Requisitos de Processos; (2) Construção da Arquitetura de Processos; (3) Construção dos Componentes de Processos que compõem a Arquitetura; (4) Composição de um Processo Específico e (5) Simulação do Processo e Execução.
SEGRINI (2009)	A reutilização pode ser vertical, utilização de componentes do nível padrão e especializados na definição de processos no nível projeto. Ou ainda, a reutilização pode ocorrer em um mesmo nível de abstração, pela definição de componentes de maior nível de granularidade a partir de componentes de granularidade mais fina (reutilização horizontal).
LANNA E PIETROBON (2010)	Uma estratégia foi especificada seguindo cinco etapas: (1) Partindo dos requisitos do novo processo de software, pesquisar o repositório em busca de componentes; (2) Qualificá-los de modo a permitir selecionar os componentes que serão efetivamente reusados; (3) Adaptá-los de modo a corrigir possíveis inconsistências entre eles; (4) Implantá-los no ambiente de execução de processo; (5) Realizar a atualização de um componente de processo por outro.
Abordagem Pynode **AVRILIONIS *et al*. (1996)**	A modularização de processos é realizada pela definição de visões, que permitem a decomposição de processos de acordo com características bem definidas (papel, atividade ou produto). Neste trabalho, foi introduzido o conceito de visão de execução (*execution view*). Uma visão de execução contém um conjunto de componentes reusáveis. A abordagem de modularização consiste em uma decomposição *top-down* de modelos de processos, combinada a uma abordagem *bottom-up*, que considera experiências anteriores, políticas pré-definidas, estratégias de gerenciamento de processo específicas, ou seja, um conjunto de informações reusáveis empacotadas em componentes. Cada visão de execução pode então ser formada por componentes e sub-visões.
COULETTE *et al*. (2000) (2001)	Um conjunto de fatores de qualidades de componentes de processo foram definidos: (a) separação entre especificação e implementação: uma especificação pode ter várias implementações, representando um papel importante na evolução estática do componente e gerenciamento de versões; (b) autodescrição: um componente deve encapsular informações que o descrevem; (c) autonomia: permite a reutilização de um componente independente de outros, ou pelo menos deve descrever claramente de quais outros componentes depende; (d) evolutivo: gerenciamento evolução estática e dinâmica; (e) consistência; (f) visões de descrição múltiplas.
IIDA E TANAKA (2002)	Este trabalho apresenta um *framework* para modelagem de processos de software com componentes capazes de serem conectados com outros por meio de classes de conexões de componentes definidas como padrões composicionais. Um Padrão composicional é definido como um conjunto de elementos de processos compostos e fluxos de produtos entre eles (relações de conexões). Esses padrões são usados para busca

Abordagem	Abordagem para Composição de Processos de Software
	de descrições de processos já desenvolvidos e estabelecidos como elementos de processos compostos. Esses padrões podem ser entendidos como *templates* de processos. Cada processo composto possui um gerenciador de composição que gerencia a composição de elementos de processo baseado no padrão composicional especificado.
TORTORELLA E VISAGGIO (1997) **FUSARO *et al*. (1998)**	O método é dividido em quatro passos: (1) Construção de um Modelo de Processo; (2) Validar Modelo de Processo; (3) Modificar o Modelo de Processo; e (4) Caracterizar o Componente de Processo. Para analisar e avaliar componentes de processo é necessário formalizar o conhecimento obtido sobre o componente. Nesta abordagem, o formalismo de modelagem utilizado é o Modelador Conceitual de Processo (*Process Conceptual Modeler* – PCM), uma ferramenta no PROMETHEUS (*PROcess Model Evolution Through Experience Unfolded Systematically*).

A análise dos diferentes trabalhos permitiu observar que ainda não há um consenso da definição do termo componente de processo. Por muitas vezes, componente de processo é citado como um fragmento de processo e visto como o encapsulamento de um conjunto de informações e comportamentos, mas sem especificar a granularidade e como os diversos elementos de processo essenciais são trabalhados. De forma geral, o conteúdo de um componente de processo é formado por uma unidade de trabalho, variando de tarefas a processos. A comunicação por meio de interfaces é precariamente especificada, sem definir exatamente o que são os pontos de entrada e saída do componente e como são estabelecidas as relações de fluxos de trabalho na definição de componentes mais complexos formados pela composição e adaptação de estruturas menores. A representação dos componentes, tanto gráfica como descritiva, é pouco discutida e só é efetivamente abordada em alguns trabalhos que apresentam alguma proposta de representação e classificação de componentes.

A maioria dos trabalhos não aborda nem trata a questão de variabilidades e opcionalidades dos diferentes elementos de processo, nem como isso se reflete no componente como um todo. Ou seja, as abordagens tratam apenas a visão do componente como unidades de trabalho, sem considerar que os outros elementos associados precisam ser identificados e analisados dentro de contextos de aplicação.

Apesar de o uso de componentes para composição de processos aparentar fornecer muitas vantagens, partir de unidades tão pequenas para compor grandes processos parece ainda ser insuficiente. Caso se analise a reutilização no contexto de produtos de software, é possível observar que uma das lições aprendidas com os

esforços para se alcançar a reutilização nas últimas décadas foi a de que a reutilização *bottom-up*, ou seja, a composição de componentes arbitrários para construir sistemas, não funciona bem na prática (BARRETO, 2007). Programas de reutilização bem sucedidos devem empregar, também, uma abordagem *top-down*, ou seja, componentes são desenvolvidos de forma a se encaixarem em uma estrutura de alto nível definida por uma arquitetura de software (BOSCH, 2000). Assim, a definição de processos por meio de componentes pode ser potencializada se combinada com uma estrutura maior que define as diretrizes de combinação e organização das unidades fundamentais.

A Arquitetura de Processo consiste em um arcabouço conceitual para incorporar, relacionar e adaptar elementos de processos em instâncias de processos (HUMPRHEY, 1989). Essa estrutura geralmente é requerida quando há a necessidade de relacionar processos com outros já existentes ou processos futuros, como nos casos de reutilização de elementos e adaptação de processos. Segundo o CMMI-DEV (CHRISSIS *et al.*, 2006), arquitetura de processos é definida como a ordenação, interfaces, interdependências, e outros relacionamentos entre os elementos de processo em um processo padrão ou processos externos. Podemos citar como propósitos de uma arquitetura de processo: (1) a descrição de componentes relevantes, estrutura, relacionamentos, interfaces internas e externas; e (2) guiar a seleção, composição e adaptação de componentes (DAI *et al*., 2008).

Desta forma, uma abordagem para definição de processo de software a partir de componentes deve possuir uma estrutura básica para direcionar a definição de processos específicos. A definição de uma arquitetura possui esse propósito. No entanto, uma arquitetura deve ser representada por elementos arquiteturais e os relacionamentos entre esses elementos. A definição de elementos arquiteturais deve ser suportada por sistemáticas para a organização de artefatos de processo reutilizáveis em elementos arquiteturais, os quais compõem uma arquitetura de componentes. Para isso, é esperado o uso de critérios, métricas ou outras técnicas para apoio a criação destes elementos arquiteturais. Esse suporte não foi encontrado nas diferentes abordagens estudadas (Tabela 2.5).

Para que a abordagem de definição de processos baseada em reutilização atinja seus objetivos de facilitar a definição de processo por meio da composição de processos a partir de componentes reutilizáveis, é fundamental que exista uma biblioteca desses componentes. Além disso, a manutenção dessa biblioteca deve ser cuidadosa e realizada com procedimentos definidos para garantir que o conteúdo da biblioteca seja confiável e útil para a definição de processos (BARRETO, 2007).

Os repositórios de componentes de software podem ser vistos como bases de componentes que os armazenam, centralizando-os, a fim de facilitar a reutilização (SEGRINI, 2009). Eles são a principal ligação entre o desenvolvimento de componentes e o desenvolvimento com componentes (SAMETINGER, 1997). Na definição de processos baseada em componentes, a existência de repositório também e essencial (SEGRINI, 2009). Assim, uma estrutura para armazenamento dos artefatos de processos reutilizáveis desenvolvidos deve ser mantida. ROMBACH (2006) cita o repositório de processos como uma das características principais da Engenharia de Linha de Processos de Software. Neste repositório, processos reutilizáveis em todos os níveis de abstração devem estar disponíveis. GARY *et al.* (1999b) definem um repositório de componentes de processos como um container logicamente centralizado de componentes de processos (referências para localização dos componentes) e informações de processos específicas de domínio (usuários e dados).

Técnicas de armazenamento e busca devem ser implementadas de forma a aprimorar o reúso e apoiar a seleção de artefatos necessários a determinado contexto. O grupo de informações agrupadas em um repositório pode variar, mas é importante observar que dados relativos ao contexto de utilização e informações de desempenho de execução são relevantes para apoiar a seleção e viabilizar melhorias nos artefatos. Assim, um repositório pode armazenar descrições de elementos de processos, documentos relacionados, dados e métricas de execução, diretrizes de adaptação, dentre outros artefatos. Além disso, informações de possíveis dependências requeridas devem ser mantidas.

A recuperação de componentes de processos deve levar em consideração um conjunto de informações associados a um componente, desde metadados que qualificam o componente de acordo com seu contexto de aplicação, como dados que

descrevem seu conteúdo e sua estrutura. Métodos de classificação podem ser utilizados para auxiliar o mecanismo de busca. RU-ZHI *et al*. (2005) e DAI *et al.* (2008) utilizam mecanismos de classificação de componentes de processos em facetas. Um algoritmo de busca de componentes candidatos, realizado com o cálculo de similaridade entre a questão de busca e a descrição do componente, ambas representadas em formato de árvore foi proposto em RU-ZHI *et al.* (2005). Essa etapa visa identificar candidatos reutilização. Uma posterior etapa consiste na qualificação de componentes. Nessa etapa, os componentes candidatos ao reúso serão avaliados detalhadamente em diferentes aspectos, de modo a identificar vantagens e desvantagens em sua reutilização. O trabalho de FUSARO *et al*. (1998) define uma seção de avaliação que compara componentes que satisfazem requisitos e define o nível do risco a adotar um dos componentes, fornecendo diretrizes para seleção do componente mais apropriado. Esta seção é composta por três categorias: (1) técnica: descreve características tecnológicas de um componente; (2) formalismo: define o rigor de representação do componente; (3) qualidade: expressa a qualidade do componente de processo e dos objetos produzidos. LANNA (2009) também especifica um conjunto de três fases para essa qualificação e escolha de componentes apropriados.

2.6. CONSIDERAÇÕES FINAIS

Existe uma diversidade de fatores que podem afetar o contexto no qual processos precisam ser definidos e, desta forma, um conjunto de variantes de processos pode surgir em uma organização visando atender os múltiplos cenários e métodos de desenvolvimento. Apesar da existência de diferentes abordagens visando auxiliar a definição de processos via reutilização, mais especificamente, linhas de processos de software, alguns pontos ainda podem ser considerados oportunidade de trabalho na área.

Tendo em vista as abordagens de LPrS anteriormente citadas, suas limitações quanto ao suporte ao DPBC e modelagem de variabilidades, e critérios aplicados em métodos de ED com apoio a DBC (BLOIS, 2006), um conjunto de requisitos de apoio à reutilização com aplicação de conceitos de DPBC no contexto de LPrS é proposto:

- **Apoio às etapas da LPrS (EDPS e EAPS):** descrição de processos com as atividades e os artefatos envolvidos no desenvolvimento *para* e *com* reutilização de processos de software.
- **Especificação de apoio à modelagem de variabilidade de LPrS:** especificação da representação a ser utilizada na modelagem da LPrS de forma a permitir o entendimento das variações do domínio. Tal representação deve atender a um conjunto de requisitos para modelagem de variabilidade: (1) a representação deve ser gráfica e considerar os principais elementos que compõem processos (atividade, tarefa, papel, produto de trabalho e ferramenta); (2) a representação deve explicitar os elementos de processos fixos (invariantes), os elementos de processos configuráveis (pontos de variação) e suas alternativas de configuração (variantes); (3) a representação deve explicitar o grau de obrigatoriedade e opcionalidade na seleção de diferentes tipos de elementos e relações ao instanciar um processo específico de projeto (classificação de opcionalidade); (4) a representação deve explicitar os relacionamentos entre elementos; (5) a variabilidade comportamental deve ser modelada com alternativas de execução nos fluxos de controle e dados; e (6) a representação deve explicitar relações de dependência e mútua exclusividade entre elementos.
- **Modelagem da LPrS em diferentes níveis de abstração:** A abstração é um conceito essencial em reutilização, pois para que um determinado artefato de software possa ser reutilizado, ele precisa antes ser compreendido (KRUEGER, 1992). LPrS são, em geral, estruturas complexas que envolvem a representação de diferentes tipos de informações envolvidas em um domínio de processos de software. Desta forma, a modelagem de variabilidades e opcionalidades deve prever a representação das informações em níveis de abstração diferentes e complementares e por meio de perspectivas de visualização.
- **Diretrizes para apoiar o mapeamento entre os artefatos com manutenção das ligações e consistência**: A representação de uma LPrS com artefatos diferentes requer a manutenção da ligação entre os elementos que compõem esses artefatos de forma a manter a consistência. Além disso, propriedades de

variabilidades e opcionalidades devem ser mantidas ao longo do desenvolvimento da LPrS. Desta forma, mecanismos devem ser especificados para guiar o mapeamento entre os elementos dos diferentes níveis, descrevendo como alcançar determinado elemento de um nível de abstração.

- **Uso de princípios de DPBC na modelagem do domínio:** É indicado o uso de uma abordagem de DPBC por métodos de LPrS. Tal indicação se deve ao fato de que a organização de elementos de processos reutilizáveis por meio de componentes permite outra forma de organização do conhecimento em módulos de construção de processos. Essa modularização apoia características técnicas que promovem a reutilização, se destacando o acoplamento entre componentes de um domínio e a coesão de um componente em relação ao seu domínio de componentes. Desta forma, conceitos como componentes, interfaces e arquitetura devem ser especificados. Além disso, a definição de mecanismos para organização dos elementos em componentes e técnicas para agrupamento de componentes visando melhorar o acoplamento no domínio devem ser consideradas.
- **Modelagem e Reutilização de Artefatos e Modelos por meio de um ferramental apropriado:** Este requisito sugere a necessidade de uma infraestrutura de reutilização que possa dar apoio integrado ao ferramental de apoio aos critérios anteriores. É importante destacar esta necessidade de integração, uma vez que, do ponto de vista de reutilização, os artefatos gerados em todas as fases do processo estão relacionados a um conjunto de requisitos do domínio, que por sua vez poderá ser reutilizado para um dado processo específico de projeto. Assim, caso não exista esta integração entre as diferentes ferramentas, a reutilização dos artefatos poderá ser prejudicada, limitando-se a alguma fase ou atividade da Engenharia de LPrS.
- **Técnicas de verificação dos artefatos criados**: Um único artefato de domínio mal especificado pode levar à propagação de defeitos nos diversos produtos resultantes de uma linha. Desta forma, técnicas de verificação dos artefatos criados devem ser desenvolvidas de forma a verificar a consistência e a corretude sintática e semântica dos modelos criados.

As abordagens anteriormente identificadas na Seção 2.3.1 foram analisadas segundo os requisitos definidos (Tabela 2.6). O atendimento a cada requisito pelas abordagens foi classificado como SIM, NÃO ou PARCIAL. No caso da classificação ser parcial, o requisito é atendido de forma incompleta, possivelmente, com conceitos sendo apenas apresentados mas não descritos de forma detalhada a ponto de permitir a aplicação da proposta analisada.

Tabela 2.6 – Sumarização de análise das abordagens de LPrS identificadas

Publicação	Apoio às etapas da ELPrS (EDPS e EAPS)	Apoio à modelagem de variabilidade de LPrS	LPrS em diferentes níveis de abstração	Diretrizes para mapea-mento entre os artefatos	Uso de princípios de DPBC	Ferramen-tal apropriado	Técnicas de verifica-ção dos artefatos criados
Durán *et al*. (2004)	Parcial	Parcial	Não	Não	Não	Não	Não
Washizaki (2006a)(2006b)	Parcial	Parcial	Sim	Parcial	Não	Não	Não
Simidchieva *et al*. (2007)	Parcial	Parcial	Não	Não	Não	Não	Não
Thränert e Werner (2007)	Parcial	Parcial	Não	Não	Não	Não	Não
Armbrust *et al*. (2009)	Parcial	Parcial	Não	Não	Não	Não	Não
Martínez-Ruíz *et al*. (2009)(2011a) (2011b)	Sim	Parcial	Não	Não	Não	Não	Não
Ternité (2009)	Sim	Parcial	Não	Não	Não	Não	Não
Aleixo *et al*. (2010a)(2010b)	Sim	Parcial	Sim	Sim	Não	Sim	Não
Barreto *et al*. (2010) (2011)	Sim	Parcial	Sim	Sim	Parcial	Sim	Sim
Hurtado *et al*. (2011)(2013) Simmonds *et al*. (2013)	Sim	Parcial	Sim	Sim	Não	Sim	Não
Jafarinezhad e Ramsin (2012)	Parcial	Parcial	Não	Não	Não	Não	Não
Magdaleno *et al*. (2012)	Sim	Parcial	Sim	Não	Parcial	Sim	Não
Rouillé *et al*. (2012)	Sim	Parcial	Sim	Parcial	Não	Sim	Não

A especificação de uma sistemática de reutilização que contemple processos de desenvolvimento *para* e *com* reutilização combinando a técnica de LPrS com conceitos de DPBC, apoio à modelagem de variabilidades da LPrS em diferentes níveis de abstração, integrados em uma infraestrutura com suporte ferramental a diferentes

etapas do processo de reutilização para a definição de processos de software específicos de projetos é um diferencial ainda a ser trabalhado. A combinação das áreas de LPrS e DPBC ainda não foi amplamente explorada nas abordagens analisadas.

O uso de diferentes modelos para representação de LPrS foi utilizado em alguns trabalhos. No entanto, não fica clara a divisão em diferentes níveis de abstração. Falta um método para a definição dos rastros e ligações semânticas e sintáticas, de forma a garantir o tratamento da consistência entre os elementos e a complementariedade das informações a serem definidas, evitando redundâncias e retrabalho para o engenheiro do processo de software.

A variabilidade é mais amplamente tratada e modelada utilizando linguagens não elaboradas para o contexto de reutilização de processos. O foco de análise dos trabalhos encontra-se em variações de elementos estruturais de processos de software. Desta forma, as abordagens analisadas só contemplam parcialmente os requisitos listados para modelagem de variabilidades. Ainda há campo de estudo para definir os tipos de variações possíveis em comportamentos de processos designados pela variabilidade nos fluxos de execução e de dados. Assim, um tópico ainda a ser melhor explorado consiste na modelagem de variabilidade de LPrS, combinando o aspecto estrutural (elementos como unidades de trabalho, papéis e artefatos) e o aspecto comportamental (fluxos de execução e de dados) de processos de software.

O apoio ferramental é um ponto importante a ser trabalhado, pensando em uma infraestrutura de reutilização para prover o apoio necessário ao desenvolvimento e uso de uma abordagem de linha de processos de software. Tal infraestrutura deve fornecer meios capazes de guiar o engenheiro de processo e prover ferramentas para modelagem e projeto da linha com apoio pela aplicação de técnicas nas diferentes etapas envolvidas, visando minimizar o esforço envolvido e prover mecanismos de verificação da consistência dos ativos a serem reutilizados.

Mesmo analisando abordagens da área de DPBC, pode-se concluir que não é fornecido suporte de como realizar a organização da informação de processo reutilizável em componente, assim como não há consenso de uma definição sobre o que seria um componente de processo. Como foi colocado por BARRETO *et al*. (2011), não existe consenso de qual informação deve existir em um componente de processo

e que nível de detalhe este deve ter. Em geral, as abordagens utilizam definições próprias de acordo com suas necessidades. Em termos gerais, um componente é definido como elemento de processo reutilizável, mas não há explicações claras sobre sua granularidade ou que tipos de elementos podem ser organizados em um componente. Faltam diretrizes ou mecanismos de apoio de como compor um bloco de construção de processo reutilizável, lidando com as propriedades de opcionalidade e variabilidade dos diferentes tipos de elementos que descrevem um processo. Além disso, a forma de comunicação entre componentes também é um assunto pouco explorado. Como colocado por AOUSSAT *et al.* (2010), mesmo que as abordagens existentes avancem para uma mesma definição para o conceito de componente de processo de software, nenhum consenso ou metamodelo descrevendo as características de um componente de processo de software é alcançado.

Como uma abordagem para definição de processo de software a partir de componentes deve possuir uma estrutura básica para direcionar a definição de processos específicos, a definição de elementos arquiteturais deve ser suportada por sistemáticas para a organização de artefatos de processo reutilizáveis em elementos arquiteturais, os quais compõem uma arquitetura de componentes. Para isso, é esperado o uso de critérios, métricas ou outras técnicas para apoio a criação destes elementos arquiteturais. Esse suporte não foi encontrado nas diferentes abordagens estudadas.

Uma metodologia de reutilização de processos de software foi definida visando tratar os tópicos e limitações identificados ao longo deste capítulo com destaque para os requisitos listados nesta seção.

Capítulo 3 – *OdysseyProcessReuse*

3.1. INTRODUÇÃO

Conforme foi observado no Capítulo 2, as abordagens de LPrS presentes na literatura ainda apresentam oportunidades de trabalho para atender os requisitos definidos na Seção 2.6. Tais requisitos representam aspectos importantes a serem tratados por uma proposta de reutilização de processos de software. A sistemática deve prover suporte à representação e organização da informação reutilizável por meio de uma infraestrutura que permita a identificação e entendimento dos aspectos variáveis e específicos de um domínio e os modularize por meio de componentes, refletindo na melhoria do potencial de reutilização dos artefatos do domínio.

A metodologia proposta ***OdysseyProcessReuse*** consiste em uma sistemática de reutilização de processos de software, combinando técnicas e conceitos de LPrS e DPBC. Tal metodologia é composta por cinco elementos principais (Figura 3.1): (1) definição de um método para Engenharia de LPrS (ELPrS) com foco no processo de desenvolvimento *para* reutilização; (2) definição de níveis de abstração complementares para representação de LPrS com especificação da modelagem de variabilidades; (3) estabelecimento de mecanismos de mapeamento entre os artefatos dos diferentes níveis de abstração; (4) definição de procedimentos para suportar o agrupamento de componentes de processos da LPrS; e (5) definição de técnicas de verificação sintática e semântica dos artefatos criados. Uma infraestrutura de reutilização foi desenvolvida como apoio a diferentes etapas da metodologia proposta.

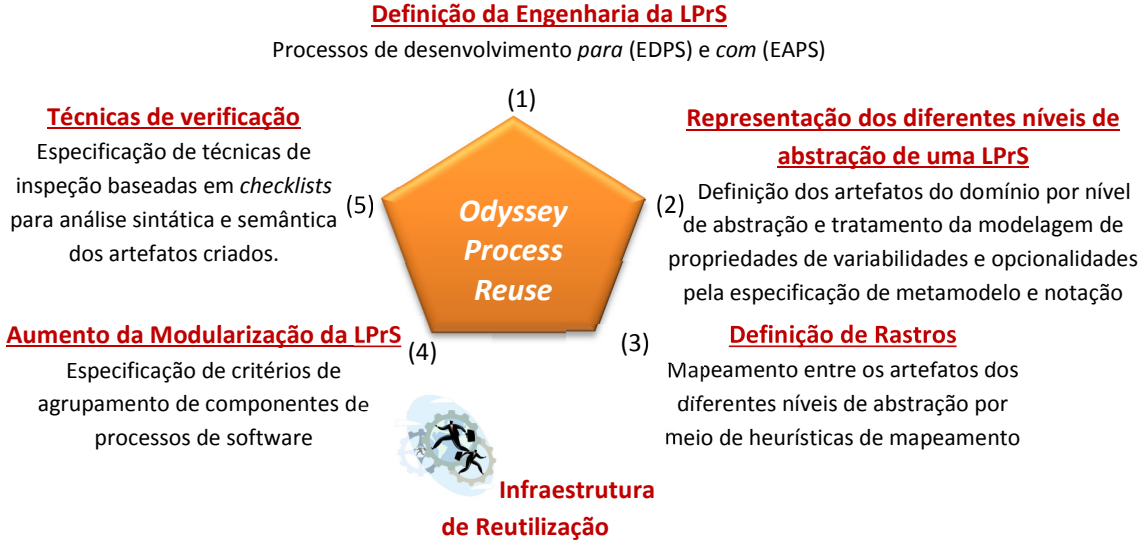


Figura 3.1 - Principais Elementos do Desenvolvimento da *OdysseyProcessReuse*

O primeiro elemento da metodologia, a Engenharia de LPrS (ELPrS), consiste no **método** que descreve as etapas que constituem o ciclo de desenvolvimento e aplicação de uma LPrS. É constituída de duas grandes fases (Figura 3.2): a Engenharia de Domínio de Processos de Software (EDPS), que consiste no processo de desenvolvimento *para* reutilização e a Engenharia de Aplicação de Processos de Software (EAPS), também denominada de processo de desenvolvimento *com* reutilização. O foco deste trabalho consiste em detalhar a fase de EDPS, descrevendo os artefatos que devem compor uma LPrS para promover a reutilização de processos de software e como representá-los de forma a organizar o conhecimento do domínio.

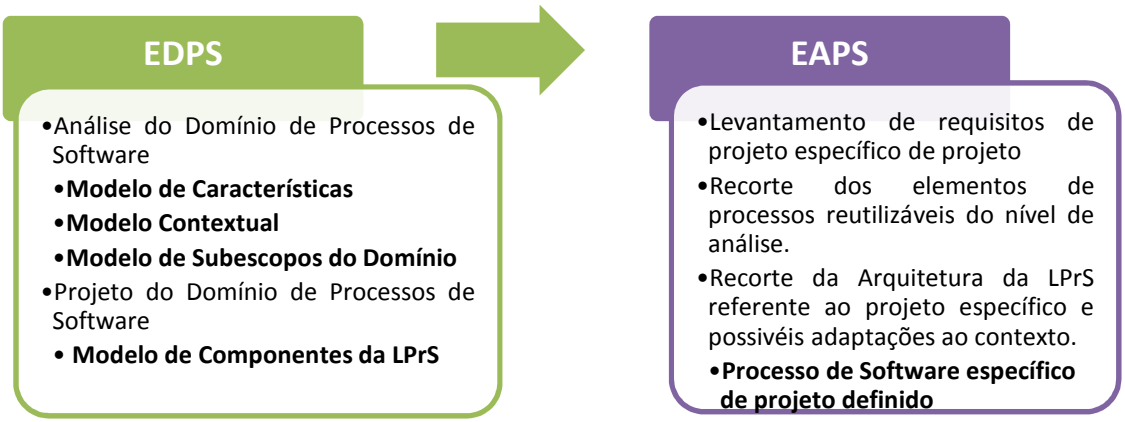


Figura 3.2 – Fases da ELPrS: EDPS e EAPS

Baseados na definição de domínio de software e estabelecidas analogias com processos de software, um **domínio de processos** pode ser entendido como uma coleção ou família de processos que compartilham um conjunto de aspectos comuns, ou seja, apresentam um conjunto de artefatos ou elementos de processos similares ou com objetivos similares, mas que também apresenta pontos de configuração e opcionalidades. Os artefatos do domínio de processos de software podem ser representados em diferentes níveis de abstração complementares para melhor organização do conhecimento a ser reutilizável. A **abstração** representa um papel central na reutilização e pode ser entendida como uma forma de trabalhar conceitos que permitam aos usuários do domínio localizar, entender, comparar e selecionar artefatos apropriados a partir do conjunto de elementos representado. Desta forma, a utilização de mais de um nível de abstração e de diferentes modelos para representar uma LPrS visa tratar a complexidade e a grande quantidade de informação a ser reutilizada de forma a aumentar a compreensão e a comunicação com a organização de conceitos por visões. Cada nível visa tratar um conjunto de informações de

elementos de processos que compõem o domínio e como estes se relacionam. Desta forma, o segundo elemento que compõe a metodologia consiste na definição de quais níveis de abstração devem ser contemplados e como a informação contida em cada nível pode ser modelada.

Neste trabalho, dois níveis de abstração foram definidos para representação da LPrS. O primeiro nível de abstração indica uma visão da LPrS representada por um modelo de domínio que agrega os elementos básicos de processos de software reutilizáveis (unidades de trabalho, papéis, produtos de trabalho e ferramentas) e analisa suas propriedades de opcionalidade e variabilidade no domínio sem especificar detalhes de instanciação ou implementação de tais elementos. Uma das formas de representação deste nível consiste em um **modelo de características**, um modelo amplamente utilizado em LPS e que visa representar o conhecimento do domínio sem a necessidade de explicitar os detalhes de mais baixo nível de abstração. Além disso, os possíveis comportamentos esperados para processos a serem derivados são especificados por meio de fluxos e sequenciamentos de execução. Um **modelo de contexto** pode ser também especificado para caracterizar entidades e situações que apoiem a seleção de artefatos de processos reutilizáveis segundo regras de contexto estabelecidas. A modelagem contextual visa especificar fatores e requisitos que apoiam a tomada de decisão de como combinar e selecionar os artefatos para atender uma cenário específico. Por exemplo, este modelo permite representar aderências a determinados padrões ou paradigmas de desenvolvimento, níveis de maturidade de processos de software, dentre outros aspectos. De forma complementar, um modelo de subescopos permite categorizar as características identificadas e organizar o domínio. O segundo nível visa estruturar a LPrS por meio de uma visão modular, organizando a informação de forma encapsulada em **componentes**, blocos de construção que combinam elementos de processo fortemente relacionados e que se comunicam por meio de interfaces. Neste nível, informações de instanciação e execução podem ser relacionadas aos componentes especificando aspectos de projeto para novos direcionamentos na definição de processos específicos de projetos durante a EAPS.

A construção progressiva de uma LPrS com seus diferentes níveis de abstração necessita de um conjunto de procedimentos para realizar mapeamentos entre os

artefatos, mantendo os rastros das propriedades de variabilidade, opcionalidade, relacionamentos, regras e restrições existentes de um artefato do domínio para o outro. Este mapeamento será suportado por heurísticas definidas para a organização dos elementos de processos de software especificados no nível de análise em componentes de processos de software, terceiro elemento que compõe a metodologia proposta.

Tais componentes podem ser ainda agrupados de forma a reduzir o acoplamento, aumentando a modularidade dentro do domínio. Critérios de agrupamento foram propostos com esse objetivo de melhor estruturar os componentes de uma LPrS e consistem no quarto elemento da metodologia.

Além das técnicas de modelagem especificadas, mecanismos de verificação devem ser providos de forma a auxiliar a equipe de engenharia do domínio na avaliação dos modelos que estão sendo desenvolvidos. Os meios propostos nesta metodologia para apoio à verificação foram a definição de um metamodelo para as representações propostas com regras de boa formação e técnicas de inspeção sintática e semântica dos modelos de características e de componentes baseadas em *checklists,* que constituem o quinto elemento da metodologia.

A especificação de um ambiente de reutilização visa prover o suporte necessário para a construção, integração e utilização dos artefatos envolvidos. Este projeto de pesquisa é baseado na adaptação do ambiente Odyssey (ODYSSEY, 2016), uma infraestrutura de reutilização de software com base em modelos de domínio desenvolvida pelo Grupo de Reutilização de Software da COPPE/UFRJ. O apoio ferramental deve suportar o desenvolvimento dos diferentes níveis de abstração de uma LPrS e gerenciar os rastros entre eles.

Neste Capítulo, o método proposto para a ELPrS é apresentado, com foco nas atividades envolvidas na fase de EDPS e na construção da infraestrutura para guiar o desenvolvimento de LPrS com conceitos de DPBC (Seção 3.2.1). Uma descrição preliminar de uma EAPS para seleção dos elementos que irão definir um processo a ser derivado a partir da linha também é disponibilizada na Seção 3.2.2. Os artefatos envolvidos em cada atividade são também especificados.

Os demais quatro elementos da metodologia proposta são apresentados nos capítulos posteriores. Desta forma, a modelagem de variabilidades de LPrS proposta e

os mecanismos de mapeamento entre os níveis de abstração definidos são descritos no Capítulo 4. Os critérios para suportar o agrupamento de componentes de processos da LPrS com o objetivo de aumentar a modularidade dos elementos da LPrS são apresentados também no Capítulo 4. O desenvolvimento da infraestrutura de reutilização provida como suporte ferramental às etapas da metodologia proposta é descrito no Capítulo 6 desta tese.

3.2. Engenharia de Linha de Processos de Software (ELPrS)

Uma visão geral de uma ELPrS pode ser vista na Figura 3.3. Conforme dito anteriormente, as duas grandes fases consistem na EDPS e EAPS. O resultado final consiste em um Processo Específico de Projeto Definido. Desta forma, de acordo com o ciclo de vida de processos de software, este resultado atende à atividade de definição de processos. No entanto, após esta atividade, é necessário atribuir informações específicas de um projeto para instanciar o processo e, possivelmente, transformá-lo em uma representação usando uma linguagem de modelagem de processos com suporte à execução do processo derivado. Estas atividades encontram-se fora do escopo desta tese e podem ser tratadas como trabalhos futuros para completar o ciclo de vida de processos de Software até a atividade de monitoramento, controle e melhoria de processos.

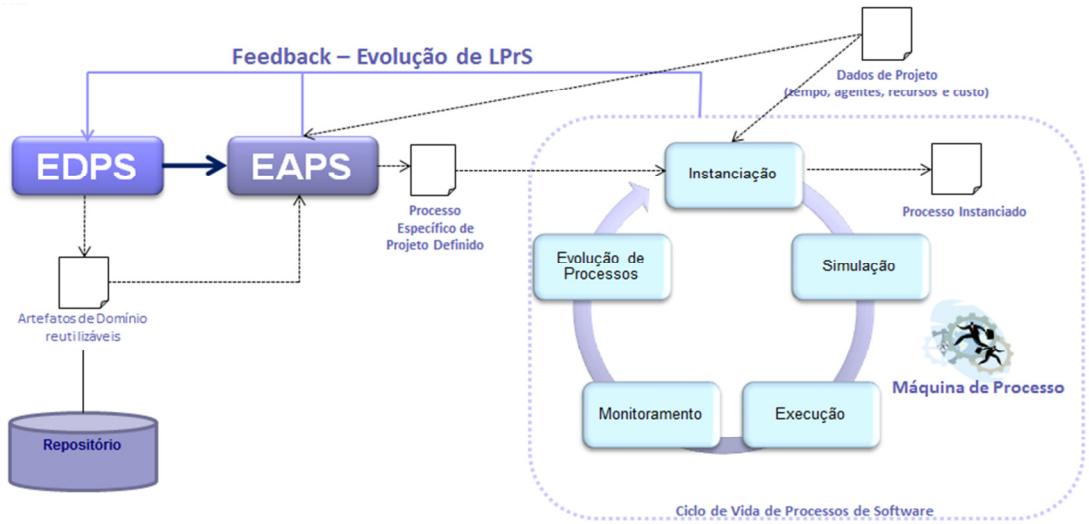


Figura 3.3 - Visão Geral de uma ELPrS

3.2.1. Engenharia de Domínio de Processos de Software (EDPS)

O processo da fase EDPS é composto por duas grandes etapas (Figura 3.4): (1) *Análise do Domínio de Processos de Software* (**ADPS**) e o (2) *Projeto do Domínio de*

Processos de Software (**PDPS**). Tal processo deve prover a possibilidade de realizar revisões entre as atividades para manter a consistência e possibilitar refinamentos do domínio. Os artefatos são gerados com o avanço do processo por dois diferentes níveis de abstração, partindo do mais alto nível (ADPS), que trata da representação com entendimento mais conceitual da LPrS, utilizando-se de um modelo de características e um modelo contextual, até o mais baixo nível (PDPS) com os artefatos descritos por componentes organizando a linha por meio de módulos. À medida que os artefatos vão sendo construídos, rastros entre os dois níveis de abstração devem ser estabelecidos. A metodologia propõe heurísticas de mapeamento entre os dois níveis de abstração para apoiar essa atividade e estabelecer os rastros entre os artefatos dos modelos.

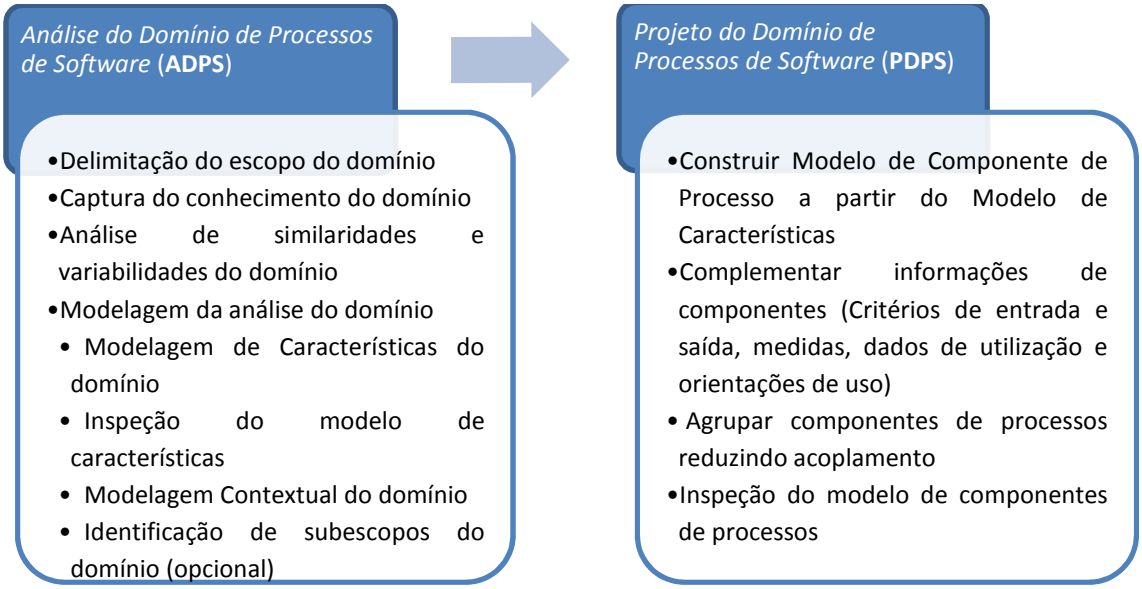


Figura 3.4 – Etapas da EDPS (ADPS e PDPS)

3.2.1.1. Análise do Domínio de Processos de Software (ADPS)

A etapa de ADPS se propõe a identificar e representar o conhecimento referente ao domínio em um alto nível de abstração, representando o resultado da análise de suas similaridades, variabilidades e opcionalidades. O domínio é modelado com informações que envolvem a descrição das diferentes alternativas de unidades de trabalho e de seus produtos de trabalho, papéis e ferramentas associados. Além disso, as propriedades e atributos de cada elemento são definidos, assim como, as possibilidades de fluxo de execução entre as unidades de trabalho são estabelecidas, indicando sequencialidade, paralelismo, sincronismo e fluxos de decisão.

As atividades iniciais da ADPS consistem em *Delimitação do Escopo do Domínio de Processos de Software, Captura do Conhecimento do Domínio de Processos de Software, Análise de Similaridades e Variabilidades do Domínio de Processos de Software, e Modelagem da Análise do Domínio de Processos de Software* (Figura 3.5). As atividades enumeradas devem ser realizadas sequencialmente. No entanto, existe a possibilidade de retorno à atividade anterior como forma de complementar a descrição dos elementos do domínio, visando ampliar o escopo, capturar mais informação sobre o domínio ou melhorar a análise do conhecimento adquirido.

As três primeiras atividades não serão tratadas no escopo desta tese, e, por isso, serão brevemente descritas a seguir, apontando algumas referências na literatura que podem apoiá-las. No entanto, vale ressaltar que tais atividades podem sofrer variações dependendo da abordagem de definição de processos *para* reutilização (*bottom-up* ou *top-down*, e até, possivelmente, uma combinação das duas) a ser aplicada. A abordagem *bottom-up* é aplicada quando a fonte de informações para construção da linha são processos legados, ou seja, processos pré-existentes dentro da organização ou o conhecimento recuperado de instâncias executadas dos projetos de desenvolvimento de software. Na ausência desta fonte, aplica-se a abordagem *top-down* para tratar um conjunto bem definido de necessidades de processos a ser atendido na organização. A abordagem *top-down* pode ser utilizada em conjunto com a abordagem *bottom-up*, ampliando a fonte de conhecimentos e trabalhando necessidades identificadas na organização, mas que ainda não foram aplicadas em seus processos atuais, provocando melhorias no conjunto de processos futuros a serem definidos a partir da linha criada.

A atividade de Delimitação do Escopo do Domínio de Processos de Software visa situar o domínio identificando suas fronteiras dentro da organização, considerando as oportunidades de reutilização versus o potencial e capacidade de reutilizar. O escopo visa cobrir as necessidades atuais da organização e do cenário em que esta atua, mas deve considerar as perspectivas futuras do planejamento estratégico da organização, analisando seus objetivos de negócio e seu desempenho atual, monitorado via requisitos de negócio e de qualidade. Esta atividade se comporta como um estudo de viabilidade do domínio investigando oportunidades de reutilização de ativos de processo junto a seus respectivos potenciais de reutilização e alinhando a

seleção do domínio com os objetivos que os processos visam atender. O objetivo principal é identificar um domínio adequado com o escopo delimitado de acordo com o tempo e recursos disponíveis versus os resultados esperados.

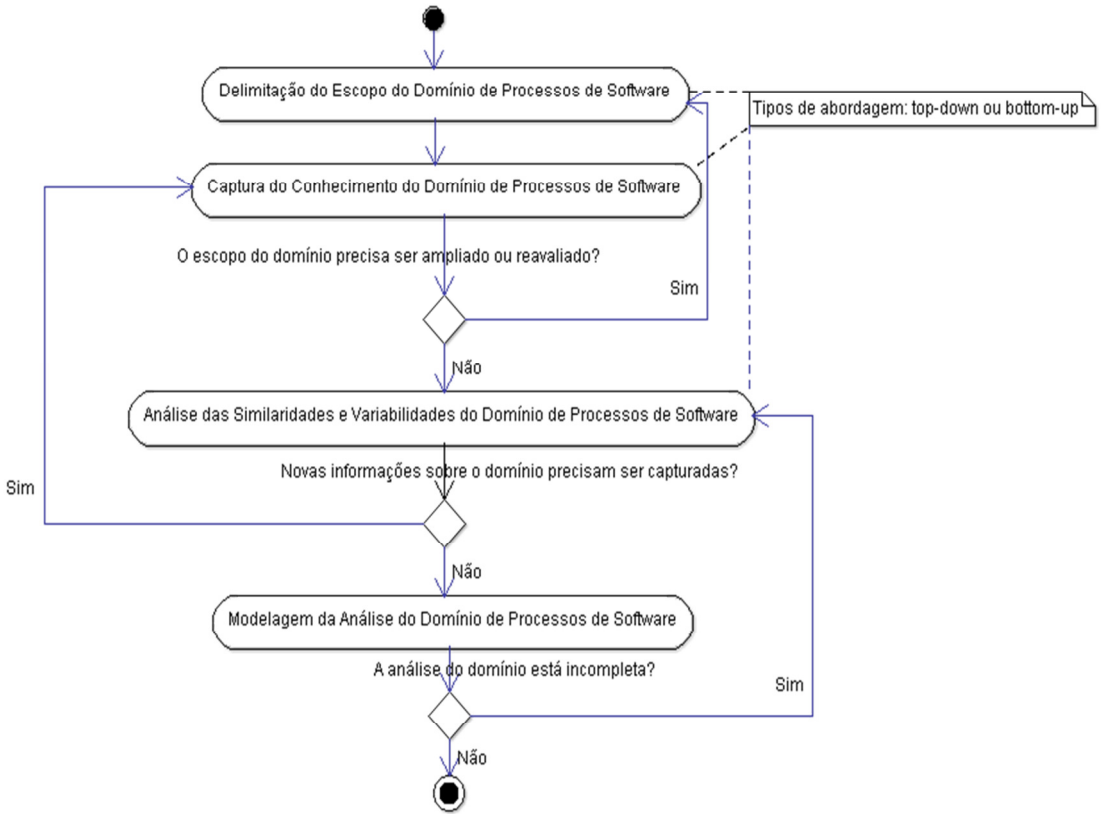


Figura 3.5 - Atividades de Análise do Domínio de Processos de Software

Uma abordagem para definição do escopo de uma LPrS foi proposta em ARMBRUST *et al*. (2008)(2009). O escopo deve delimitar o domínio para que suporte o desenvolvimento de produtos e a execução de projetos de acordo com os objetivos organizacionais e minimize o esforço do gerenciamento de processos. Determina o que incluir em uma LPrS baseado na análise de atributos de produtos (e.g., tamanho, complexidade, criticidade e estabilidade de requisitos), de projetos (e.g., grau de distribuição, pressão de cronograma, pessoal disponível e experiência da equipe), de processos, dentre outros aspectos.

No caso, de uma abordagem *bottom-up*, o conjunto de processos legados pode ser considerado como um escopo inicial e permite analisar a maturidade da organização dentro do domínio de processos considerado, com aquisição de informações sobre lições aprendidas e desvios durante a execução dos processos. No entanto, tais informações precisam estar disponíveis ou passíveis de serem capturadas.

A atividade ***Captura do Conhecimento do Domínio de Processos de Software*** visa capturar informações e conhecimento dentro de um domínio de processos de software. Encontra-se dividida nas seguintes tarefas: (1) *Identificação da(s) fonte(s) de conhecimento sobre o domínio*; (2) *Seleção da(s) técnica(s) de captura do conhecimento do domínio*; e (3) *Captura e Armazenamento do conhecimento do domínio* (Figura 3.6).

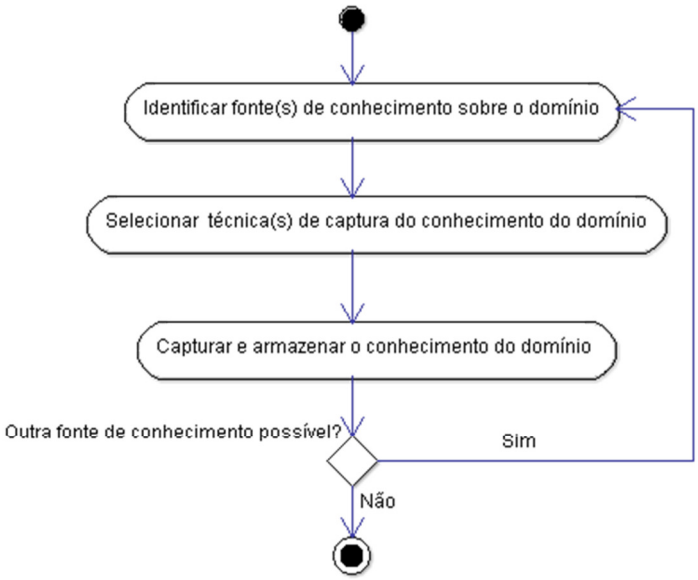


Figura 3.6 – Atividade de Captura do Conhecimento do Domínio de Processos de Software

Informações de processos podem ser capturadas de múltiplas fontes, das quais podemos citar: (1) Modelos de referências, tais como CMMI (CHRISSIS *et al.*, 2006), MPS.Br (SOFTEX, 2016) e ISO/IEC 12207 (ISO/IEC, 2008); (2) Metodologia Ágil de Desenvolvimento, tais como XP (BECK, 2004) e Scrum (SCHWABER, 2004); (3) Modelos de processos legados dentro de uma organização ou dados de execução de instâncias de projetos de desenvolvimento de software; e (4) entrevistas com especialistas do domínio (por exemplo, gerentes de projeto, engenheiros de processos, etc.); dentre outros. No caso de processos legados (abordagem *bottom-up*), técnicas para "reengenharia de processos" podem ser aplicadas para captura da informação, como técnicas de mineração de processos (*process mining*) (AALST e GIINTHER, 2007, AALST *et al*., 2004). A mineração de processos aplica engenharia reversa de dados reais da execução de processos, possivelmente em repositórios de ambientes de gerência de tarefas e/ou gerência de projetos. Um *log* de eventos pode ser obtido a partir de qualquer sistema de informação que armazene estes dados (BPMS, ERP, CRM ou um sistema próprio desenvolvido pela organização). Nesta área, técnicas similares às

aplicadas na Engenharia Reversa de software podem ser utilizadas, como *clustering* e reconhecimento de padrões (GRECO *et al.*, 2008; BOSE e AALST, 2009), e técnicas de reengenharia para apoio à Engenharia de Domínio de Produtos (VASCONCELOS E WERNER, 2011). No entanto, adaptações devem ser realizadas para avaliar o tratamento de informações referente a execução de processos conduzidos em projetos anteriores. No caso, de modelos de processos existentes e processos padrões definidos, essa informação deve ser capturada junto às diretrizes de adaptação. As diretrizes de adaptação do processo padrão da organização às necessidades específicas de projetos definem critérios e guias para adaptação. Tais diretrizes descrevem como os processos padrão e os ativos de processo organizacional devem ser utilizados para criar os processos definidos.

Após essa atividade, deve ser conduzida uma ***Análise das Similaridades e Variabilidades do Domínio de Processos de Software***, que consiste na identificação de aspectos comuns e variáveis dentro do domínio de processos de software (Figura 3.7). A atividade de *Detecção de Similaridades* tem por objetivo identificar os elementos que representam similaridades (aspectos comuns) dentro do domínio. O resultado desta atividade é insumo para a *Detecção de Opcionalidades*, que visa especificar os elementos que são considerados mandatórios e aqueles considerados opcionais dentro do domínio. O engenheiro de domínio pode determinar qual o número mínimo de processos aos quais elementos equivalentes devem pertencer para serem considerados mandatórios no domínio. A atividade de *Detecção de Variabilidades* determina pontos que são configuráveis dentro do domínio e suas alternativas de configuração.

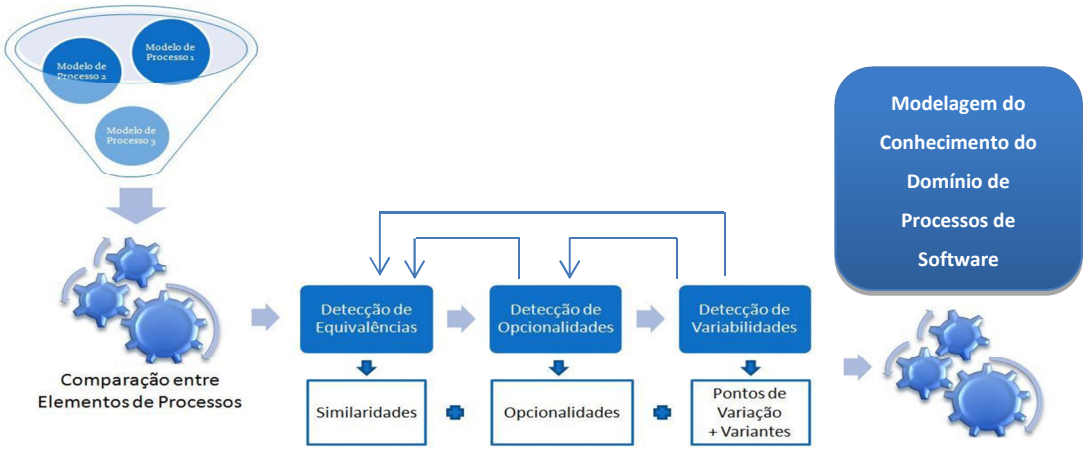


Figura 3.7 - Análise de Similaridades e Variabilidades no Domínio de Processos de Software

Existem técnicas para realizar esta análise dentro da área de LPS, como exemplo a abordagem ArchToDSSA (KÜMMEL, 2007), que propõe a comparação de arquiteturas de aplicações, visando à detecção de suas similaridades, opcionalidades e variabilidades, para que, com base nestas informações, seja possível apoiar o engenheiro de domínio na criação de uma DSSA (*Domain Specific Software Architecture*). Essa abordagem utiliza como elementos um dicionário de sinônimos, a detecção semiautomática de variabilidades, e o apoio à definição dos elementos para compor a DSSA. Esta abordagem foi adaptada para o contexto de LPrS e a descrição das etapas para construir um exemplo de LPrS pode ser encontrada em MAGDALENO *et al*. (2011).

Outra técnica foi desenvolvida por OCAMPO *et al*. (2005), que descreve a possibilidade de uma comparação entre processos de forma manual ou apoiada por regras, com a utilização de suporte ferramental. O papel responsável pela análise consiste no engenheiro de processo, que deve comparar as entidades dos modelos de processos recuperados, entrevistar os proprietários dos processos (desenvolvedores e líderes de projetos) e identificar indícios de similaridades e diferenças. Diferentes técnicas de comparação podem ser usadas e requerem identificar as partes que devem ser comparadas e como analisar seus elementos (critérios de comparação).

O conhecimento capturado e analisado deve ser modelado explicitando as particularidades do domínio, suas similaridades e variabilidades (Figura 3.8). Dentre os modelos de mais alto nível de abstração de domínio, o mais conhecido na literatura de LPS é o modelo de características, o qual descreve a teoria do domínio (ARANGO e PRIETO-DIAZ, 1991). No contexto desta proposta, na etapa de análise do domínio, o conhecimento é representado por meio de um Modelo de Características e de um Modelo Contextual do Domínio de Processos de Software. De forma opcional, um modelo contemplando a organização do domínio segundo subescopos pode ser definido. Dependendo da abrangência do domínio, a definição de subáreas dentro do escopo total pode ser importante para estabelecer nichos de elementos e categorizar os elementos segundo alguns critérios como elementos que pertencem a processos padrão da organização ou escopos especializados que representam diferentes domínios de aplicação de produtos de software desenvolvidos a partir dos processos derivados; paradigmas de desenvolvimento; etc.

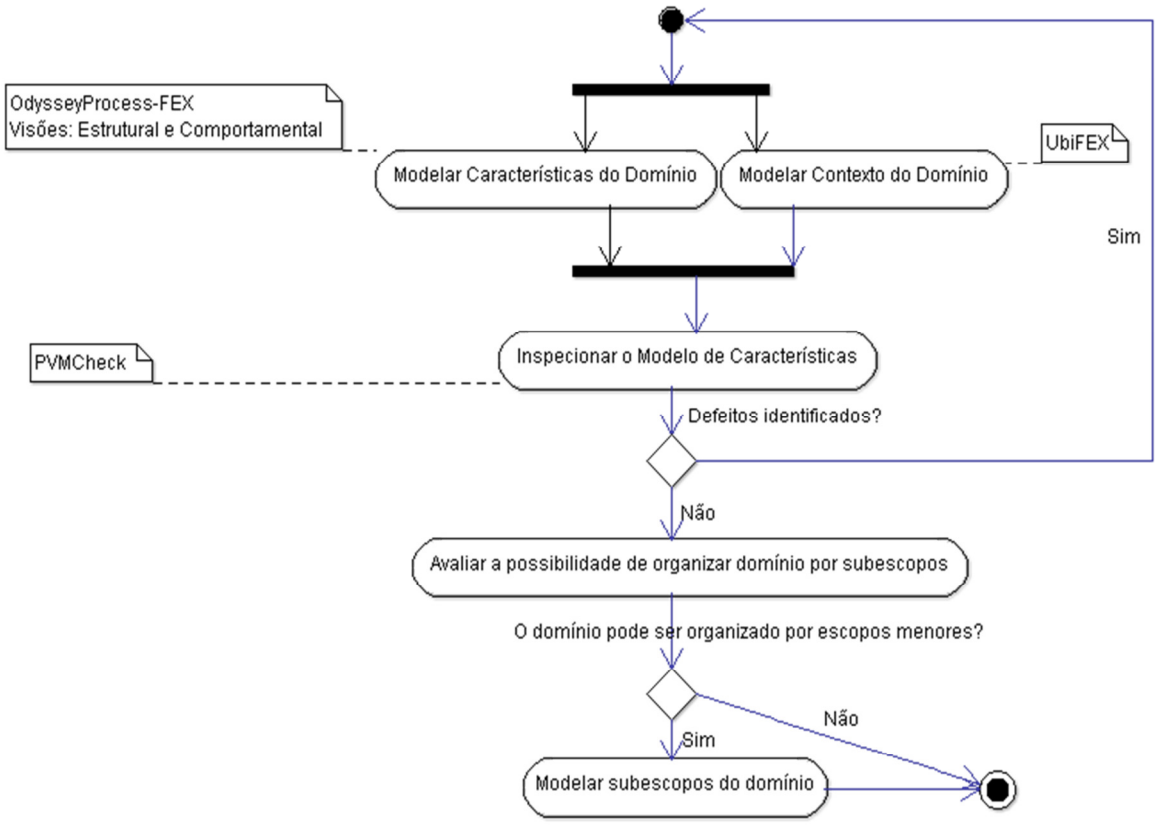


Figura 3.8 - Atividades da Modelagem de Análise do Domínio de Processos de Software

A modelagem de características inclui a identificação dos elementos de processo que compõem o domínio com a especificação de suas propriedades de variabilidade e opcionalidade (Figura 3.10). Uma **característica do domínio** representa os seguintes possíveis elementos: unidades de trabalho, ou seja, atividades e suas tarefas; produtos de trabalho consumidos, produzidos ou modificados pelas unidades de trabalho; papéis responsáveis pela execução das unidades de trabalho e ferramentas de apoio a essa execução.

Os relacionamentos estruturais entre tais elementos devem ser especificados, indicando ligações entre atividades e suas tarefas associadas; unidades de trabalho e seus produtos de trabalho (entrada ou saída), seus papéis (com respectiva responsabilidade assumida) e possíveis ferramentas; possíveis ligações entre produtos de trabalho, entre papéis e entre produtos de trabalho e papéis; ligações entre pontos de variação e as variantes correspondentes. Por último, regras de composição representando relações de dependência e mútua exclusividade devem ser especificadas. Esta descrição compreende a *visão estrutural* do modelo de características. Um exemplo desta visão modelada pode ser vista na Figura 3.9, que

descreve parte de uma LPrS de Gerência de Portfólios com os elementos de processos necessários à seleção de projetos que devem compor um portfólio de projetos aderentes aos objetivos estratégicos de uma organização.

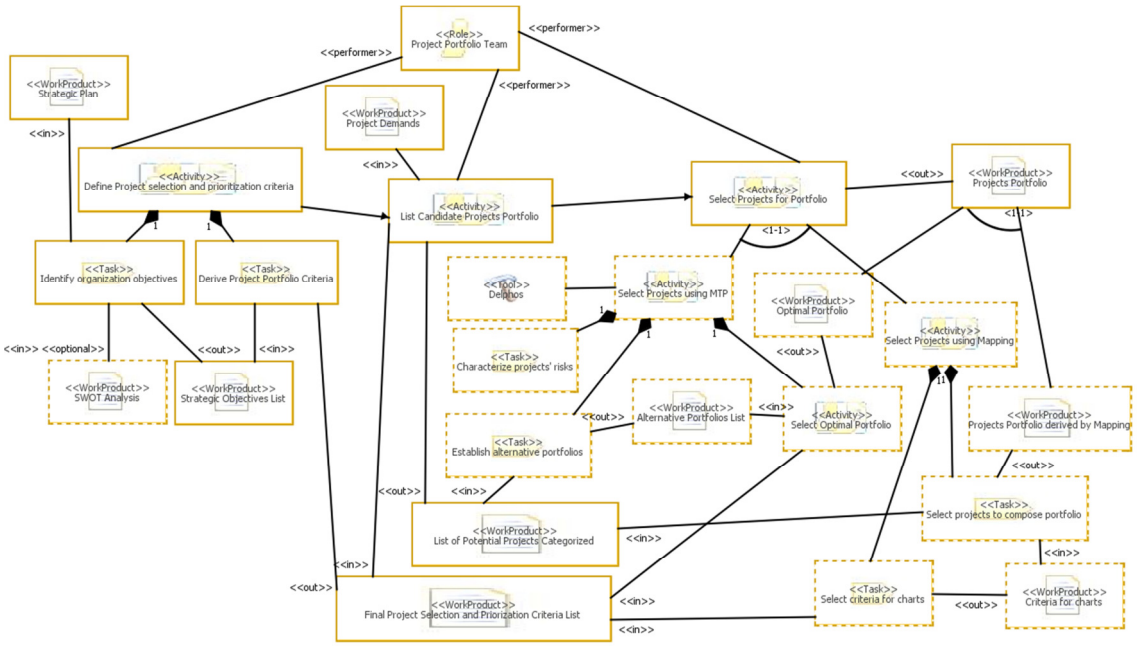


Figura 3.9 – Exemplo de Visão Estrutural de um Modelo de Características de LPrS de Gerência de Portfólio

A *visão comportamental* representa o fluxo de execução das unidades de trabalho representadas por atividades que foram especificadas na visão estrutural e tratamento de possíveis opcionalidades nesses fluxos.

A construção dessa modelagem deve ser realizada de forma gradual. A modelagem da visão estrutural e da visão comportamental podem ser realizadas de forma paralela. Desta forma, após a identificação das unidades de trabalho, é possível conduzir a especificação de ambas as visões paralelamente (Figura 3.10).

O resultado final desta atividade consiste em um modelo de características representado segundo o metamodelo e a notação *OdysseyProcess-FEX*, inicialmente definidos em TEIXEIRA (2011), ambos evoluídos neste trabalho para melhor atender os requisitos do nível de abstração que contemplam. O metamodelo e a notação *OdysseyProcess-FEX* encontram-se brevemente descritos no Capítulo 4. Informações detalhadas foram especificadas em TEIXEIRA *et al*. (2016).

Tal representação do domínio foi desenvolvida visando atender aos requisitos identificados para modelagem de variabilidades de processos de software e descritos no Capítulo 4. É importante ressaltar que a modelagem de um domínio de processos

de software contempla mais informações do que as relacionadas a modelagem de um único processo que são contempladas pelas linguagens de processos atualmente existentes. Nestas linguagens falta suporte à representação de aspectos de variabilidades e contextos de adaptação em um nível de abstração facilmente compreensível por engenheiros de processos e desenvolvedores de software. Mesmo analisando as propostas de modelagem inseridas nas abordagens de LPrS e de adaptação de processos de software, pode-se observar que ainda não há uma representação de processos de software que atenda a todos os requisitos listados de forma a explicitar os conceitos inerentes a abordagens de LPrS e que contemplem a especificação de variações tanto nos elementos quanto nas relações entre eles. Desta forma, a modelagem foi um tópico fortemente tratado neste tese com a especificação das representações envolvidas.

É importante ressaltar que o modelo de características deve ser inspecionado para verificar inconsistências semânticas que não foram detectadas ao longo da modelagem. Uma técnica de inspeção baseada em *checklist*, denominada *Process Variability Modeling Checking* (PVMCheck) (TEIXEIRA *et al*., 2015), foi proposta para apoiar esta atividade e encontra-se descrita no Capítulo 4.

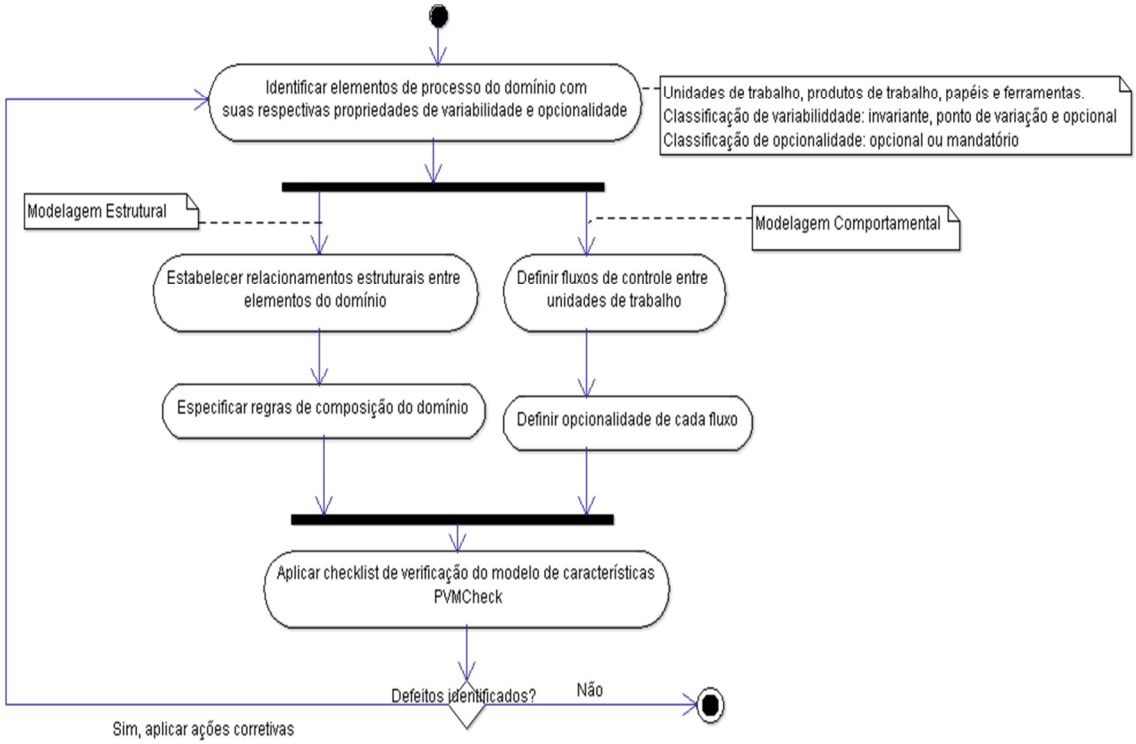


Figura 3.10 – Atividades da Modelagem de Características do Domínio (visão estrutural e comportamental)

A atividade de Modelagem do Contexto do Domínio inclui a representação dos fatores que podem influenciar a definição e composição de processos, indicando diretrizes para escolha de elementos de processos mais adequados a determinados cenários pré-definidos. Uma definição de **contexto** amplamente adotada corresponde a "qualquer informação que possa ser utilizada para caracterizar a situação de uma entidade, onde uma entidade pode ser uma pessoa, lugar, ação ou objeto que seja considerado relevante para a situação" (DEY *et al*., 2001). Para a área de processos, fatores como o tipo do produto; paradigma de desenvolvimento; domínio de aplicação do processo; projetos, estrutura e cultura organizacional; padrões e modelos de referências são possíveis fontes contendo informações que impactam a definição e a execução dos processos. Um modelo contextual visa organizar essas informações e apoiar o gerente de projeto na tomada de decisão durante a definição de um processo específico de um projeto que atenda a um contexto em particular. Um contexto pode então ser entendido como um cenário em que valores específicos são atribuídos a um conjunto das informações que descrevem as entidades definidas na modelagem. O modelo de contexto poderia ser entendido como um artefato de representação de requisitos no domínio de engenharia de processo de software (HURTADO ALEGRÍA, 2012).

A gestão de contexto exige a definição de modelos de informação que determinem o que vem a ser o contexto em um determinado domínio (NUNES *et al*., 2007). O gerenciamento de contexto em LPrS abrange diferentes atividades, desde a identificação das informações de contexto relevantes em um domínio e sua representação por meio de um modelo que represente como o contexto pode impactar o domínio de processos; assim como a captura e armazenamento das informações de contexto e suas aplicações ao provocar alterações e adaptações na derivação de processos, podendo essa informação ser reutilizada para apoiar novas definições baseadas na análise de casos similares.

Ainda existe pouco conhecimento sobre quais artefatos de processos devem ser selecionados para atender certas configurações de projetos (KUHRMANN, 2014). Do mesmo modo, falta uma capacidade de capturar a flexibilidade requerida por processos de software devido à falta de habilidade para expressar flexibilidade usando as linguagens de modelagem de processos de hoje (KUHRMANN, 2014).

Este trabalho usa o conceito de contexto aplicado como um meio de suportar a fase de EAPS. Para isso, durante a etapa de análise do domínio durante a fase de EDPS, a modelagem contextual deverá ser realizada utilizando-se da abordagem UbiFEX (FERNANDES *et al*., 2011) proposta como mecanismo para apoiar a modelagem de características de Linha de Produtos de Software Sensíveis ao Contexto (LPSSC). Esta abordagem foi escolhida por ser aplicada combinada à modelagem de características de linha de produtos de software, de forma similar a proposta nesta tese para LPrS. Além disso, os conceitos propostos pela UbiFEX são similares aos conceitos apresentados na proposta de modelagem contextual no domínio de processos de software da abordagem CASPER (HURTADO ALEGRÍA, 2012). Este fato reforça a aplicabilidade da UbiFEX no contexto desta tese. Ainda podemos mencionar que o ferramental de apoio de reutilização, adaptado nesta tese, já possuía suporte à modelagem de contexto de acordo com a abordagem UbiFEX, que foi adaptada para aplicação à LPrS.

A abordagem UbiFEX inclui os seguintes elementos: entidades, informações, definições e regras de contexto. Entidades de contexto representam as dimensões/fatores de contexto que devem ser monitorados e podem contribuir para adaptações no domínio em estudo. Cada entidade é caracteriza por informações de contexto, que são os dados que devem ser coletados para descrever a entidade e cujos valores definem as situações de contexto que podem desencadear ações a serem tomadas com relação ao domínio. As ações são descritas por meio de regras de contexto, que implicam na seleção ou exclusão de características de processos de software do domínio como apoio à definição de processos a partir da linha derivada. A Figura 3.11 apresenta as principais atividades da modelagem do contexto de uma LPrS.

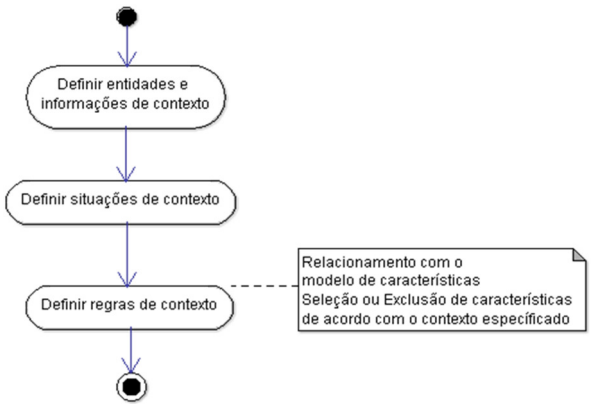


Figura 3.11 – Atividades da Modelagem Contextual do Domínio

Exemplo de aplicação desta modelagem consiste na modelagem do fator equipe como entidade de contexto. Algumas informações desta entidade a serem analisadas seriam: tamanho, experiência, estabilidade, proximidade, dentre outras. A representação gráfica deste modelo pode ser vista na Figura 3.12.

Para caracterizar uma entidade, as situações de contexto devem ser definidas atribuindo valores específicos às informações, o que representa cenários relevantes para o domínio. Por exemplo, uma equipe pode ser considerada pequena, média ou grande. Situações de contexto caracterizam as informações. Por exemplo, o tamanho da equipe pode ser caracterizado como pequena, média ou grande. No caso de uma situação que caracteriza uma Equipe Pequena, os valores da informação de contexto são definidos da seguinte forma Equipe.Tamanho < 10. Outro exemplo poderia ser determinar o tipo da equipe com valores da informação de proximidade. Quando uma equipe está reunida no mesmo local de trabalho, a situação de contexto caracterizada é de uma equipe reunida (Equipe.Proximidade = 1). Caso contrário, a equipe será caracterizada como distribuída (Equipe.Proximidade = 0).

Por último, regras de contexto são especificadas indicando impactos na configuração dos processos de uma LPrS como, por exemplo, decisões a respeito da seleção de variantes em um ponto de variação. Essas regras têm como propriedades: um identificador e uma expressão que relaciona o modelo contextual com o modelo de características. Vale ressaltar, que tais regras devem respeitar as regras de composição estabelecidas em um modelo de características, que estão descritas no Capítulo 4 desta tese. Desta forma, regras de contexto não devem indicar a seleção de características mutuamente excludentes entre si, ou ainda, ao indicar a seleção de uma característica que tenha relações de dependências com outras, estas também devem ser indicadas para seleção. Em um domínio que inclui uma atividade de Engenharia de Requisitos com diferentes alternativas para elicitar os requisitos como Realizar Entrevistas, Aplicar Questionário, Usar Cenários, Aplicar *Brainstorm*, e Construir Protótipo, as situações do tipo de proximidade da equipe podem indicar um tipo de elicitação mais adequado. Esta relação seria descrita pela seguinte regra de contexto: Equipe Distribuída *implies* Aplicar Questionário, que pode ser entendida como Equipe.Proximidade = 0 implica na seleção da característica Aplicar Questionário durante a derivação do processo.

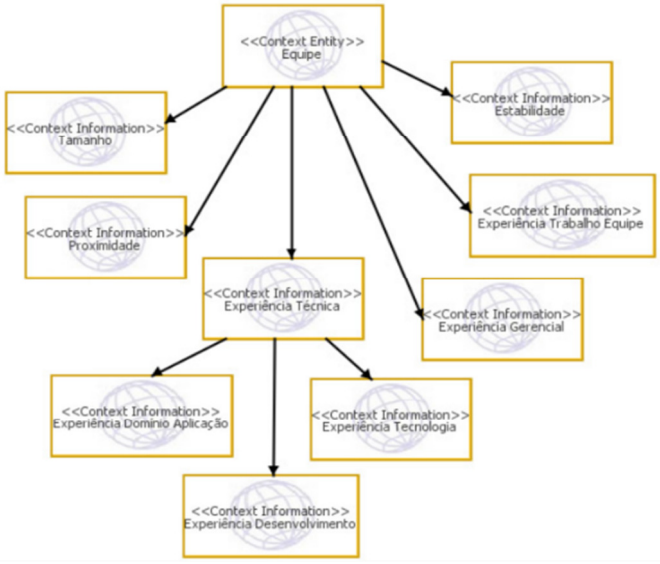


Figura 3.12 – Exemplo Modelo de Contexto utilizando notação UbiFEX

Uma proposta de entidades e informações de contexto para o cenário de processos de software foi apresentada em LEITE (2011). As entidades Organização, Projeto, Equipe, Cliente, Processo e Produto basearam-se na proposta de ARAUJO *et al.* (2004a), que fornece uma visão das principais dimensões de contexto envolvidas no desenvolvimento de software: indivíduo, papel, equipe, tarefa, projeto, organização, domínio da engenharia de software, produto de software, domínio do negócio e cliente/usuário e foram selecionadas de acordo com uma revisão bibliográfica para identificação de itens de informação que são comumente utilizados na adaptação de processos (LEITE, 2011), que também derivou possíveis informações de contexto relevantes.

Outros fatores podem ser levados em consideração para a definição do contexto associado a uma LPrS. BARRETO (2011) aponta as diretrizes de adaptação dos processos padrão de uma organização como uma das principais fontes de informação, pois nestas são estabelecidas as situações em que uma ou outra adaptação no processo pode ocorrer. Também aponta outras possíveis fontes como necessidades a serem atendidas pelos processos; necessidade de aderência a modelos de maturidade; características das organizações que utilizarão o projeto, tais como: tamanho, nível de maturidade, tipo de organização (governamental e privada) e os tipos de software desenvolvidos; métodos e técnicas que precisam ser usados, uma vez que diferentes organizações ou mesmo diferentes projetos em uma mesma organização podem usar

vários métodos e técnicas diferentes; e características específicas de cada processo considerado.

Outra revisão da literatura (KALUS e KUHRMANN, 2013) apontou um conjunto de critérios para adaptação de processos de software. Neste trabalho, quarenta e nove critérios foram identificados e associados a vinte ações de adaptação, com a especificação de um catálogo. Os critérios foram divididos em quatro grupos: time, ambiente interno, ambiente externo e objetivos. Cada item foi apresentado associado a uma descrição e breve explicação da implicação na adaptação de processos. Por exemplo, uma das informações associada a time é o tamanho da equipe, que é um indicador para o esforço necessário para estabelecer a coordenação. Desta forma, as ações a serem aplicadas estão relacionadas a forma de estabelecer a comunicação entre os membros da equipe, por exemplo, com o aumento ou redução da documentação.

Pode-se observar que as abordagens sugerem entidades e informações de contextos similares ou complementares. Desta forma, pela análise dos itens mais referenciados na literatura da área, é possível identificar um conjunto de itens sugestivos que podem auxiliar a caracterização de contextos pré-definidos para LPrSs.

Outra atividade da etapa de análise do domínio consiste em situar o domínio em relação a padrões e especializações dentro de uma organização. Desta forma, o escopo do domínio pode ser dividido em subescopos do domínio, distinguindo elementos de processos em tipos diferentes de aplicação dentro de um panorama geral no contexto organizacional. Dependendo da abrangência do domínio, a definição de subáreas dentro do escopo total pode ser importante para estabelecer nichos de elementos e categorizar os elementos segundo alguns critérios como elementos que pertencem a processos padrão da organização ou escopos especializados que representam diferentes domínios de aplicação de produtos de software desenvolvidos a partir dos processos derivados; paradigmas de desenvolvimento; etc.

Uma das formas de organizar internamente o escopo do domínio é seguindo os conceitos de uma abordagem em níveis de definição de processos, como representado na Figura 3.13. Inicialmente, um processo padrão da organização é especificado contendo apenas ativos de processo que devem ser incorporados a quaisquer processos da organização, pois estão de acordo com a cultura, conhecimento e

experiência organizacional. Tal processo padrão pode ser especializado de acordo com tipos de software produzidos, paradigmas de desenvolvimento e domínios de aplicação adotados. Para isso, outros elementos de processo e adaptações precisam ser implantadas para atender tais contextos. No último nível, em um processo específico para um projeto, devem-se considerar diferentes aspectos para realizar novas adaptações como tamanho e a complexidade do produto, características de qualidade desejadas, a expectativa de vida útil, as características da equipe, disponibilidade de recursos, dentre outros.

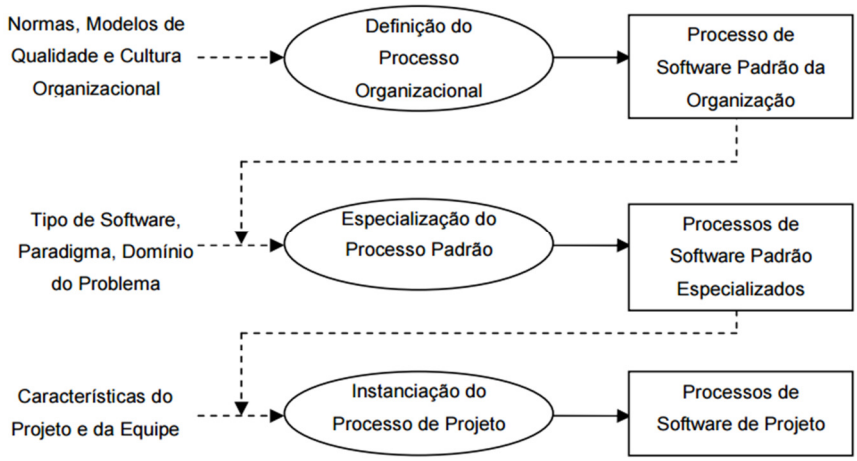


Figura 3.13 – Abordagem em Níveis para Definição de Processos (Fonte: BERTOLLO, 2006)

Seguindo esta abordagem de níveis, subescopos podem ser definidos representados por um *escopo padrão* e *escopos específicos* (Figura 3.14). Esses escopos devem ser especificados pela seleção de elementos do modelo de características e, possivelmente, recortes do modelo contextual. No escopo padrão, são organizadas as características que representam um processo padrão para a organização, que descreve os elementos de processo que devem estar presentes em qualquer processo definido. Esse escopo deve trabalhar todas as características consideradas mandatórias para o domínio. Escopos específicos devem organizar características de acordo com valores assumidos para cada item de um conjunto de determinados fatores pré-determinados durante a modelagem do domínio que especificam um nicho de processos que atendam aquele cenário. Fatores como o tipo de software a ser desenvolvido, o paradigma de desenvolvimento a ser adotado, o nível de maturidade a ser considerado pela organização, as características da equipe, da qualidade do produto a ser desenvolvido, dentre outros, podem ser considerados.

Situações de contexto podem ser usadas para caracterizar os valores assumidos por um conjunto de fatores para cada escopo específico definido. Desta forma, uma associação com o modelo de contexto do domínio pode ser realizada com a pré-seleção de elementos do modelo de características relacionados a tais situações de contexto e agrupadas por escopos. Outras características consideradas opcionais ou variáveis podem ainda ser agrupadas dentro do escopo específico. Esta organização do domínio em subescopos visa auxiliar na fase de EAPS, com a caracterização parcial ou total de tipos de projetos por subescopos específicos.

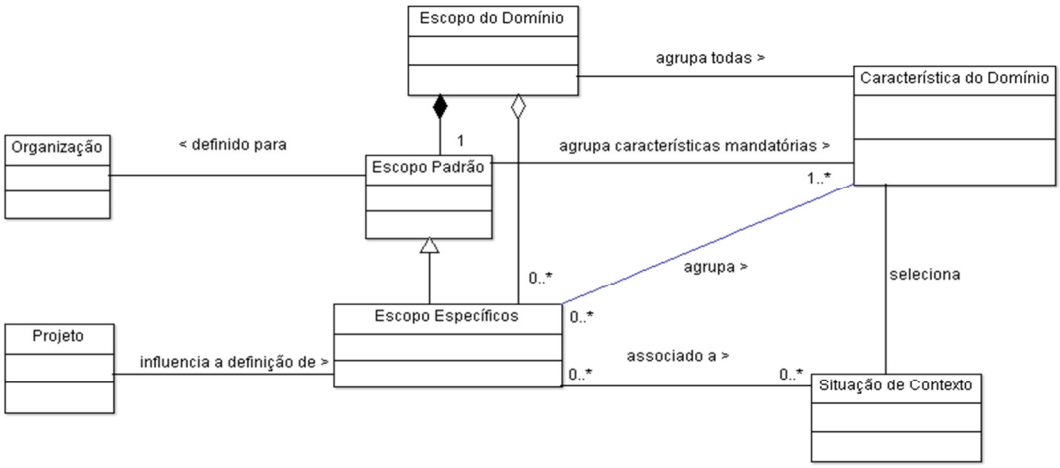


Figura 3.14 – Estrutura de Subescopos do Domínio

3.2.1.2. Projeto do Domínio de Processos de Software

O resultado final da etapa de PDPS consiste na Modelagem de Componentes da LPrS. Esta etapa está dividida em atividades de descrição e organização dos ativos do domínio por meio de componentes de processos de software. Um componente de processo pode ser entendido como uma abstração de um fragmento de processo baseada no princípio da "caixa-preta". Desta forma, um componente representa uma parte modular de um processo que encapsula seu conteúdo e se comunica com o ambiente via interfaces. Maiores detalhes sobre a definição do conceito de componente e interfaces encontram-se descritos no Capítulo 4. Informações detalhadas foram especificadas em TEIXEIRA *et al*. (2016).

As duas atividades principais desta etapa da fase de EDPS consistem da identificação e modelagem dos componentes de processos de software a partir do modelo de características criado e no agrupamento de componentes para redução de acoplamento e aumento de modularidade da LPrS, gerando uma Arquitetura de LPrS

(Figura 3.15). Desta forma, uma Arquitetura de LPrS consiste na organização da linha em componentes acoplados e relacionados via interfaces de dados e de controle com suas propriedades de variabilidade e opcionalidade especificadas.

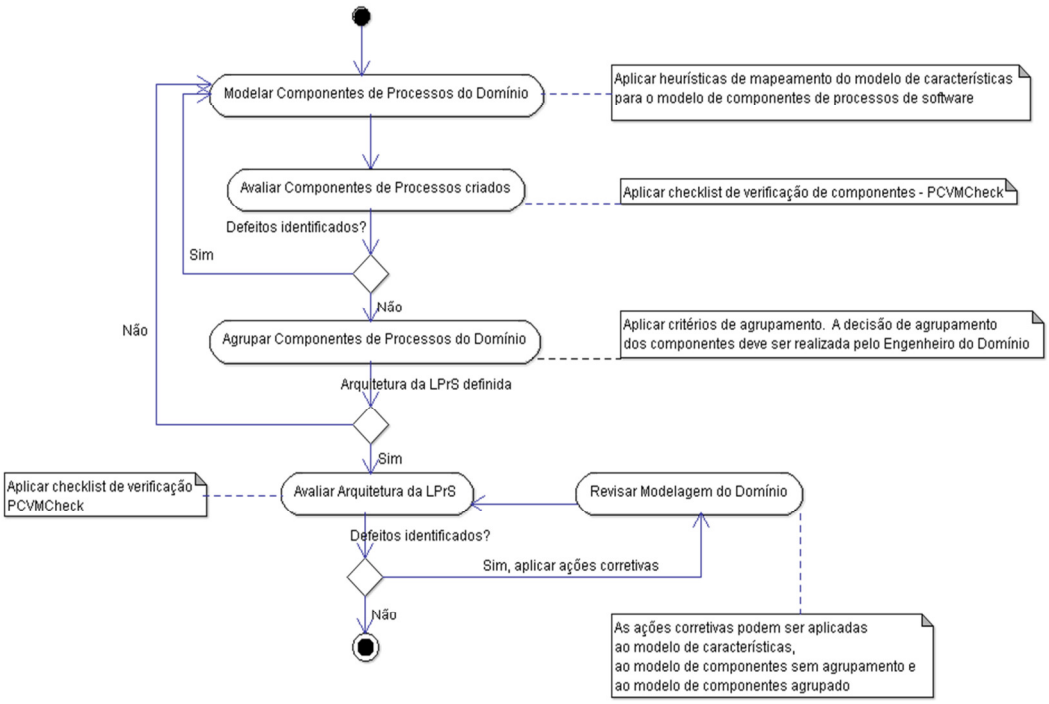


Figura 3.15 - Atividades da Etapa de Projeto do Domínio de Processos de Software

A modelagem de componentes (Figura 3.17) admite a definição de variabilidades por meio de componentes não vinculados a uma única forma de realização, desde que atendam a um conjunto de restrições estabelecidas. A modelagem de componentes, proposta neste trabalho como o segundo nível de representação da LPrS, deve ser realizada segundo o metamodelo e a notação *OdysseyProcess-FEX* nível de projeto (*Process Component View*), que se encontram detalhados na Seção 4.2.2. Como não existe um consenso sobre a definição de componentes de processos de software na literatura, esta tese propõe uma definição para o termo, assim como uma representação gráfica deste artefato. Além disso, a especificação interna de um componente é também definida e agrega as propriedades de variabilidade e opcionalidade. Cada componente é composto por um conjunto de elementos de processos relacionados (unidades de trabalho, produtos de trabalho, papéis ou ferramentas), e representa minimamente uma atividade. A comunicação entre componentes e entre componentes e o meio é realizada por meio de interfaces de dados e de controle. Interfaces de dados representam a troca de produtos de trabalho e interfaces de controle representam fluxos de execução entre componentes.

A Figura 3.16 apresenta um exemplo de modelo de componentes da mesma LPrS apresentada na Figura 3.9.

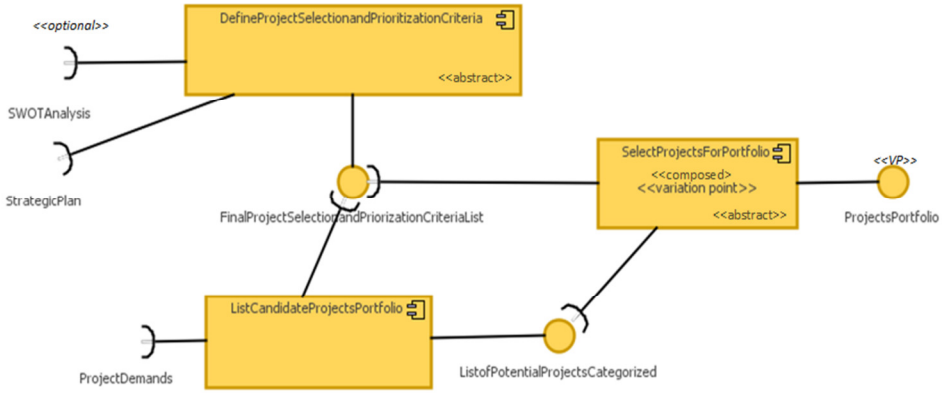


Figura 3.16 - Atividades da Modelagem de Componentes de Processos de Software

A atividade de mapeamento do modelo de características para um modelo de componentes é apoiada por heurísticas definidas com base no projeto arquitetural baseado em componentes no contexto de Engenharia de Domínio de Software definido em (BLOIS, 2006) e nas restrições especificadas no metamodelo definido nesta tese para os dois níveis de abstração. A descrição completa das heurísticas encontra-se na Seção 4.3 do Capítulo 4.

A atividade de agrupamento é apoiada por critérios de agrupamentos definidos com base nos trabalhos de técnicas de agrupamento de componentes no contexto de LPS descritos em (BLOIS, 2005) e técnicas de análise visual de modularidade de modelos de processos de software (BOLAÑOS e MARTÍNEZ, 2013). Os critérios são apresentados na Seção 4.4 do Capítulo 4.

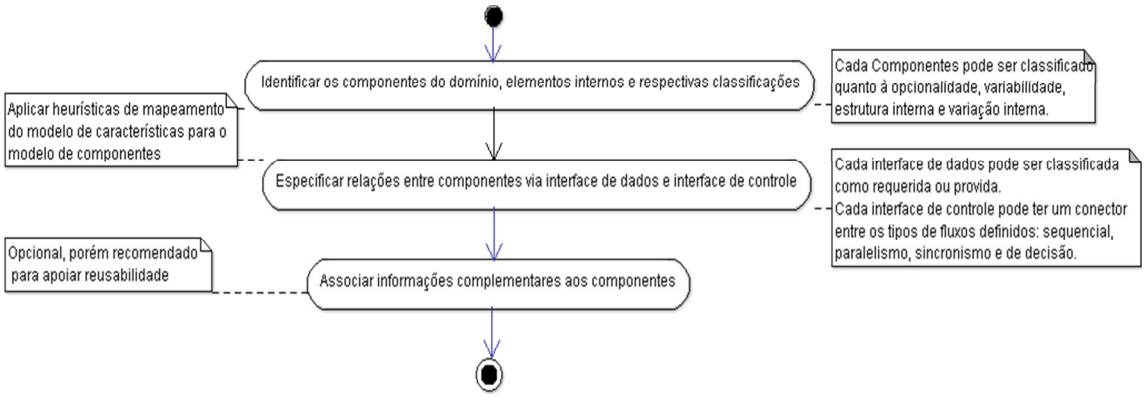


Figura 3.17 - Atividades da Modelagem de Componentes de Processos de Software

Para completar a etapa de PDPS, informações complementares de projeto caracterizando dados de execução ou apoio à instanciação e implementação ser associadas aos componentes, incluindo medidas de interesse, dados de utilização e

orientações de uso. Aos produtos de trabalho, associados aos componentes e suas interfaces, podem ser definidos *templates*. Aos papéis podem ser descritos dados de agentes ou atores específicos para auxiliar na fase de EAPS e instanciação de um processo derivado a partir da linha. *Scripts* podem ser associados a ferramentas dentro do contexto de um componente.

A atividade de associação de medidas aos componentes de processo será considerada opcional nesta metodologia. A aplicação do conceito é uma adaptação da metodologia aplicada em BARRETO (2011) (Figura 3.18). Os principais itens para trabalhar medidas incluem a possibilidade de associá-las aos componentes, assim como *baselines* de desempenho. *Baselines* de desempenho, quando associados a um componente, indicam a estabilidade deste com a determinação dos valores mínimos e máximos possíveis de serem atingidos pela medida em um determinado contexto de aplicação do componente. Desta forma, uma *baseline* de desempenho é descrita por valores de limites de controle (inferior e superior) e a descrição do contexto em que foi estabelecida.

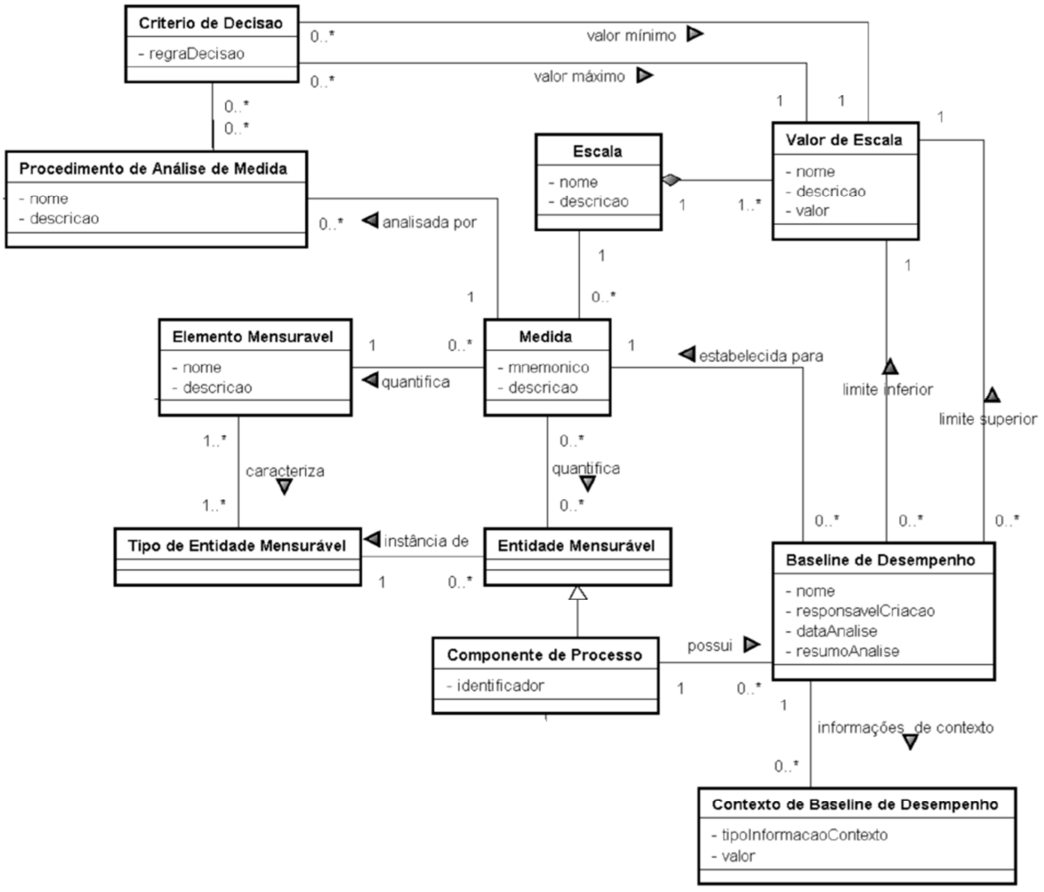


Figura 3.18 – Principais conceitos para tratar medidas associadas a componentes de processos (Adaptado de BARRETO, 2011)

É importante ressaltar que o modelo de componentes de processos, assim como o modelo de características, deve ser inspecionado para verificar inconsistências sintáticas e semânticas que não foram detectadas ao longo da modelagem. Uma técnica de inspeção baseada em *checklist*, denominada *Process Component Variability Modeling Checking* (PCVMCheck), foi proposta para apoiar esta atividade e encontra-se descrita na Seção 4.5.1.2. A técnica foi desenvolvida baseada em laudos de avaliação de componentes e LPrS desenvolvidos em BARRETO (2011). No entanto, foi adaptada para contemplar propriedades específicas do metamodelo de componentes desenvolvido nesta tese e consolidar questões já tratadas na técnica PVMCheck (TEIXEIRA *et al*., 2015).

Nesta técnica, uma primeira etapa da inspeção visa verificar o componente como um elemento individual passível de reutilização, como em uma metodologia de DPBC. Para isso, é analisada a consistência do componente como um elemento modular que deve apresentar coesão e granularidade adequada com propósito bem definido e elementos suficientes para entregar os resultados esperados de acordo com seus objetivos. No entanto, como componentes foram construídos dentro de uma metodologia que trata de opcionalidade e variabilidade, tais propriedades devem ser consideradas, mas o componente por si deve ter meios de resolver qualquer configuração descrita. A verificação do componente baseada no *checklist* deve também considerar essas questões.

Após a verificação dos componentes individualmente, uma avaliação da arquitetura como um todo deve ser realizada considerando o propósito e os objetivos da LPrS e verificando a completude dos elementos para atingir os possíveis resultados esperados e a corretude das interfaces e relações entre os elementos. Um *checklist* também foi desenvolvido para auxiliar esta atividade.

Caso defeitos sejam identificados ao longo da verificação, ações corretivas devem ser identificadas e aplicadas em todos os níveis de abstração da linha, para manter a consistência entre os elementos e garantir a confiabilidade dos produtos a serem gerados.

3.2.2. Engenharia de Aplicação de Processos de Software (EAPS)

A Engenharia de Aplicação de Processos de Software (EAPS) (Figura 3.19), também denominada de processo de desenvolvimento *com* reutilização, consiste no recorte e seleção de ativos de processos reutilizáveis definidos na fase de EDPS.

O gerente de projeto, de posse da caracterização de um projeto específico e dos requisitos a serem alcançados por um processo, possui o perfil para atuar como responsável pela EAPS. Este pode contar com o apoio da equipe de processos e desenvolvimento de software na execução das atividades de definição do processo. O nível de experiência deste profissional pode impactar fortemente nas decisões de definição de um processo, exigindo profissionais mais capacitados e com maior conhecimento adquirido. No entanto, durante a EDPS, o conhecimento a respeito de projetos anteriores como lições aprendidas, boas práticas, dados de utilização, cenários de sucesso e caracterização de desempenho podem ser explicitados por meio dos artefatos e informações complementares representadas por uma LPrS. Desta forma, o uso de LPrS visa reduzir a influência e dependência de profissionais especializados ou altamente experientes na definição de novos processos.

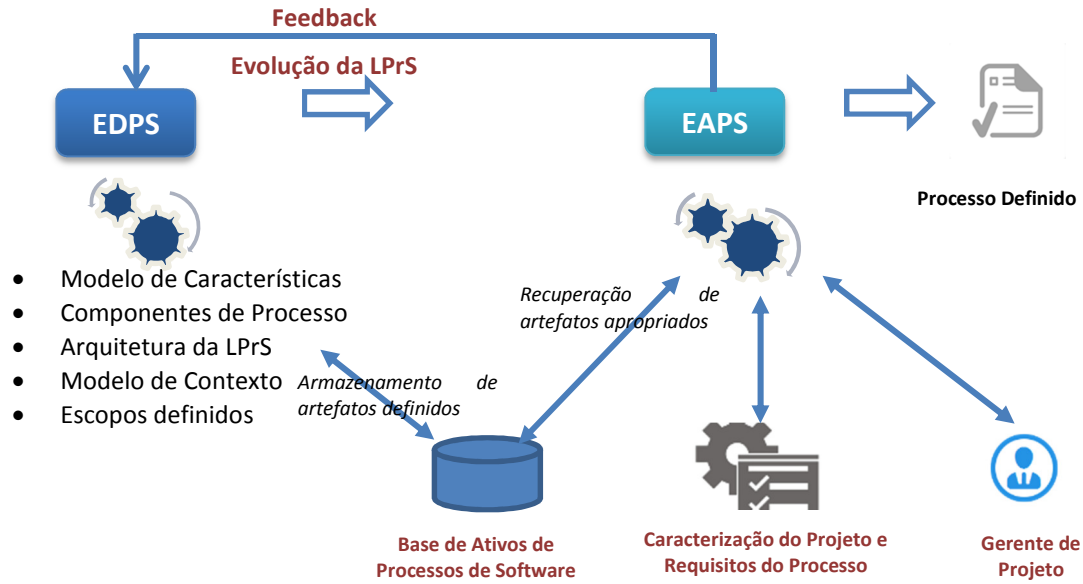


Figura 3.19 – Visão Geral EAPS

O processo da fase EAPS inicia com a definição de informação sobre o processo a ser derivado, contendo nome, descrição, propósito e requisitos a serem atendidos pelo processo. Após esta tarefa, a EAPS é composta por uma sequência de atividades de recorte e seleção dos elementos que deverão compor o processo a partir de duas

grandes etapas (Figura 3.20): (1) *Análise da Aplicação de um Processo de Software* (**AAPS**) e o (2) *Projeto da Aplicação de um Processo de Software* (**PAPS**).

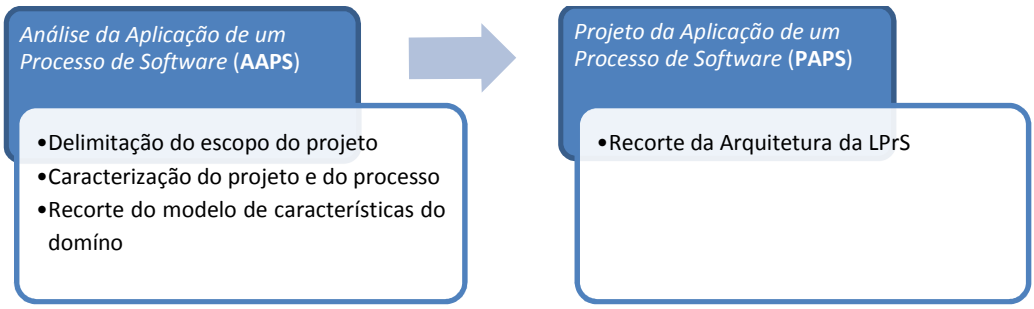


Figura 3.20 – Etapas da EAPS (AAPS e PAPS)

Tal processo deve prover a possibilidade de realizar revisões entre as atividades para manter a consistência e possibilitar refinamentos do domínio. Como proposta inicial, os artefatos são recortados em diferentes níveis de abstração, partindo do mais alto nível, um modelo de características e um modelo contextual, até o mais baixo nível com o recorte dos artefatos descritos por componentes organizados em uma arquitetura da linha. Como os artefatos vão sendo construídos com o estabelecimento de rastros entre níveis de abstração, o recorte em um nível mais alto é refletido no nível mais baixo, mantendo consistência das seleções.

A *Análise da Aplicação de um Processo de Software* define os requisitos e particularidades do processo a ser definido, consistindo em um primeiro nível de recorte ou derivação, baseado na seleção em um alto nível de abstração, que representa elementos de processos em um modelo de características do domínio de processos de software. Para isso, é necessário que esta etapa inicie com a delimitação do escopo do projeto e do processo a ser definido, atividade que pode ser apoiada pela seleção de um escopo padrão, e, possivelmente, escopos especializados (artefatos gerados opcionalmente na ADPS), pré-selecionando um conjunto de características possíveis para o processo a ser derivado a partir da LPrS. A seguir, o contexto de aplicação em que o processo será utilizado deverá ser recortado com a seleção das situações de contexto que o caracterizam. Essa caracterização do contexto recorta parte do modelo de características utilizando-se das regras de contexto definidas. A atividade final desta etapa consiste na análise do modelo com as características resultantes do pré-recorte para a tomada de decisão de quais variações serão pertinentes no processo a ser definido, selecionando características opcionais e

variantes. Os recortes devem levar em consideração as dependências e restrições descritas por regras de composição no modelo de características.

O *Projeto da Aplicação de um Processo de Software* determina a arquitetura do processo por um segundo recorte, que seleciona o conjunto de componentes que deve estar presentes no processo específico baseado no pré-recorte realizado no modelo de características e refletido por rastros estabelecidos entre os níveis de abstração. Desta forma, decisões finais de variabilidades em nível de projeto devem ser conduzidas, levando em consideração dados de utilização e medidas associadas aos componentes, além de pré-condições e restrições derivadas do modelo de características. A adaptação dos componentes selecionados a particularidades não previstas deve ser realizada ainda nesta etapa.

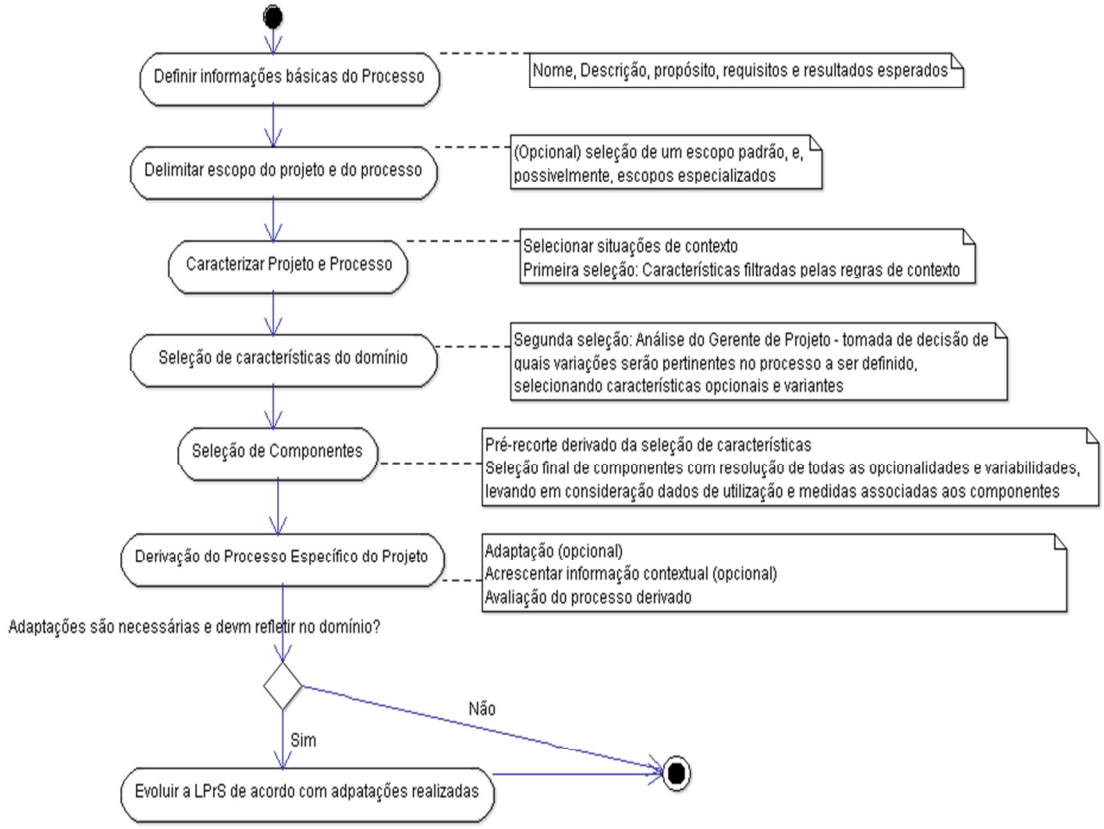


Figura 3.21 - Atividades da EAPS

Alguns filtros podem ser aplicados para auxiliar na análise dos componentes e focar em informações que podem ser relevantes para determinar a seleção, apresentando ao gerente de projeto apenas os componentes que atendem determinados critérios especificados. Tipos de filtros passíveis de aplicação dentro da metodologia:

1. Podem ser filtrados os componentes que possuem dados de utilização associados que permitam identificar possíveis desvios ou casos de sucesso;
2. Podem ser filtrados os componentes que possuem medidas com informações coletadas de execuções anteriores que permita analisar o comportamento do componente;
3. Podem ser filtrados apenas componentes considerados estáveis no domínio, ou seja, o componente possui uma *baseline* de desempenho associada;
4. Podem ser filtrados os componentes considerados capazes, ou seja, com *baseline* de desempenho estabelecida e faixa de valores dentro do objetivo esperado para o componente;
5. Podem ser filtrados os componentes com maior taxa de reutilização no domínio, ou seja, foram selecionados mais vezes em projetos anteriores similares.

O processo derivado a partir da LPrS deve ser avaliado tanto sintática quanto semanticamente. Uma das verificações consiste em analisar se o processo derivado é válido de acordo com consistências das trocas de produtos produzidos e requeridos pelos componentes de processo e a ordenação do fluxo de execução do processo. Outra avaliação consiste em verificar a adequação do processo a seu propósito e resultados esperados. Essa avaliação pode ser conduzida pelos responsáveis pela garantia da qualidade do processo e do produto e pode ser realizada por meio de uma técnica de revisão por pares. A identificação de não conformidades durante as atividades de verificação devem gerar ações corretivas que devem ser aplicadas no processo e, possivelmente, refletidas no domínio.

No processo derivado que está sendo avaliado, todos os componentes de processo têm as suas interfaces requeridas satisfeitas pelos artefatos gerados pelos demais componentes de processo ou pelas entradas disponíveis externamente. Essa verificação garante que o fluxo de dados está correto e não haverá impedimento da execução de um componente por falta de informação requerida. Além disso, deve ser avaliado se todos os resultados esperados pelo processo são satisfeitos por interfaces providas de um ou mais componentes de processos selecionados.

A verificação da ordenação consiste em analisar se o fluxo de execução respeita a precedência imposta por critérios de entrada e saída os componentes e que os produtos requeridos estão sendo providos por componentes anteriores no fluxo. Um possível algoritmo para verificação consiste em (MAGDALENO, 2013):

1. Todos os componentes que precisam ser avaliados são colocados em uma lista de pendências (LAP) que é percorrida até que esteja vazia (o que representa que um processo válido foi composto) ou que se verifique que a combinação de componentes não é válida. Além disso, todos os artefatos disponíveis são armazenados em uma lista LAC, que vai mantendo a lista dos artefatos produzidos pelo processo.
2. Os componentes são removidos da lista LAP em rodadas sucessivas que verificam se as interfaces requeridas daquele componente são satisfeitas com os artefatos presentes em LAC. Cada componente satisfeito é guardado na lista de resolvidos (LAR) e adiciona as suas interfaces providas também em LAC para que outros componentes consecutivos possam ser satisfeitos.
3. Ao final, caso todas as pendências de LAP tenham sido resolvidas, o processo ordenado está indicado em LAR e a lista LAC indica todos os artefatos que foram produzidos por esta opção do processo.

O resultado da EAPS é um processo definido com todos os elementos que devem compor sua execução. No entanto, após o processo definido e verificado, informações relacionadas à fase de instanciação devem ser incluídas. Além disso, tal processo pode ser modelado segundo uma linguagem de modelagem que possua algum suporte ferramental envolvido na execução de um processo instanciado. Desta forma, essa atividade pode ser conduzida, mas deve levar em consideração a máquina de processos selecionada e sua forma de representação do modelo de processo na linguagem executável. O apoio a estas etapas encontra-se fora do escopo desta tese.

3.3. CONSIDERAÇÕES FINAIS

O método proposto para a ELPrS foi apresentado neste capítulo, com foco nas atividades envolvidas na fase de EDPS. Uma descrição preliminar de uma EAPS para seleção dos elementos que irão definir um processo a ser derivado a partir da linha

também é disponibilizada na Seção 3.2.2. No entanto, esta fase precisa ser evoluída, sendo apontada como oportunidade de trabalho futuro.

Como resultado da fase de EDPS foram descritos, para a etapa de análise do domínio, o modelo de características, o modelo de contexto e, opcionalmente, o modelo de subescopos. Para a fase de projeto, um modelo de componentes deve ser especificado com rastros estabelecidos com o modelo de característica definido.

A fase de construção de uma LPrS, assim como na área de reutilização de produtos, apresenta um custo inicial para, posteriormente, contemplar os benefícios esperados. Por isso, o processo de EDPS proposto pode ser considerado robusto. No entanto, dependendo do cenário em que tal processo está sendo utilizado e do resultado da atividade de delimitação do escopo, o nível de detalhamento e seleção das atividades a serem seguidas podem ser calibrados pelo engenheiro de domínio. Desta forma, apesar da metodologia indicar a realização de todo o método proposto, alguns recortes podem ser realizados para aderência às necessidades de determinadas equipes, domínios e engenheiros de domínio. Por exemplo, o nível de detalhamento de um modelo de características até o nível de tarefas e passos pode ser omitido, apenas representando características de unidades de trabalho como atividades. Além disso, a modelagem contextual pode ser suprimida ou utilizado algum contexto pré-definido. Em outro cenário, o engenheiro de domínio pode optar por iniciar a modelagem pelo nível de componentes, suprimindo a modelagem proposta no nível de análise. Algumas diretrizes podem ser desenvolvidas para guiar a seleção de algumas das atividades e modelagens propostas, de forma a atender aderências a determinados objetivos. No entanto, esse recorte do método proposto pode ser considerado uma oportunidade de trabalho futuro e como melhoria da metodologia.

A proposta de metamodelo e notação para modelagem de variabilidades em dois níveis, características e componentes, é apresentada no Capítulo 4. As heurísticas para mapeamento entre artefatos dos dois níveis também é apresentada no Capítulo 4. Por último, os critérios de agrupamento de componentes em elementos arquiteturais mais coesos encontram-se especificados no Capítulo 4. A infraestrutura de reutilização provida como suporte ferramental a etapas da metodologia proposta é descrita no Capítulo 6 desta tese.

Capítulo 4 – Modelagem de Variabilidades de LPrS

4.1. INTRODUÇÃO

A metodologia proposta ***OdysseyProcessReuse*** foi apresentada no Capítulo 3 desta tese. Do conjunto de elementos principais que a compõem, o método foi detalhado com apresentação dos principais artefatos envolvidos. Neste Capítulo, serão apresentados os demais elementos incluindo a estrutura definida para modelagem de variabilidades de LPrS nos dois níveis de abstração definidos. Esta estrutura contempla um metamodelo e notação para modelagem de características e componentes de LPrS, heurísticas de mapeamento entre os artefatos dos dois níveis de abstração e critérios para apoiar o agrupamento de componentes de processos da LPrS.

O uso de estudos experimentais para avaliar o desenvolvimento de um método, processo, técnica, produto, modelo ou ferramenta pode apoiar a validação ou identificação de problemas presentes em novas tecnologias propostas. Esta tem sido uma das formas cientificamente aceitas para obter indícios sobre a viabilidade e eficiência, ou seja, os benefícios (ou não) esperados de alguma destas propostas. Neste Capítulo, são apresentados os estudos de viabilidade realizados para avaliar os elementos aqui apresentados. Esta técnica de experimentação foi escolhida para se obter indícios de que o objeto de estudo é adequado ao propósito para o qual foi desenvolvido. Estudos de viabilidade são utilizados com o objetivo de caracterização de uma tecnologia, para a verificação da viabilidade de sua aplicação e continuidade de dispêndio de recursos para desenvolvê-la. Este tipo de estudo se enquadra na categoria de estudos de caso, estratégia de estudo experimental selecionada devido à escassez de tempo e recursos como, por exemplo, a amostra de participantes disponível para o planejamento e execução dos estudos; ao tipo da avaliação que consiste na verificação da viabilidade de uso e eficácia da aplicação da abordagem proposta; e ao nível de controle baixo a moderado sobre as variáveis dos estudos (ex: perfil dos participantes). A proposta dos estudos é avaliar as atividades que compreendem o processo de EDPS. Para isso, foram identificadas atividades estratégicas do processo e estudos foram sendo realizados com focos nas diferentes contribuições propostas, que serão apresentados associados ao elementos analisado, ao longo do Capítulo.

4.2. MODELAGEM DE VARIABILIDADES DE LINHA DE PROCESSOS DE SOFTWARE

Diante da análise da literatura, pode-se enumerar um conjunto de observações para guiar a representação de uma LPrS e os conceitos que a contemplam. Dentre os elementos de processos de software mais referenciados pelos diferentes trabalhos de modelagem de processos de software podemos destacar como os principais: atividade; tarefa; papel; produto de trabalho (artefato); e ferramenta. A variação desses elementos pode se dar quanto à variabilidade e à opcionalidade dentro do domínio. Quanto à variabilidade, todos os elementos citados podem ser considerados passíveis de variação (configuração) diante de aspectos específicos de projeto. No entanto, o estabelecimento de alternativa de variação a um ponto de configuração deve respeitar os tipos dos elementos de processo. Já quanto à opcionalidade, todos os elementos citados acima podem ser classificados como *mandatórios* ou *opcionais*, dentro de um domínio de processos analisado. Outra forma de variação em uma família de processo consiste nas variações que podem ocorrer nos fluxos de controle entre as unidades de trabalho ou opcionalidades nas relações entre certos tipos de elementos.

Para tratar as diferenças que distinguem cada processo de uma LPrS, é necessário criar mecanismos em que essas variações no processo possam ser definidas (MARTÍNEZ-RUÍZ *et al*., 2008). A reutilização de processos de software baseada em variabilidades necessita de uma modelagem apropriada. Tal modelagem pode estar focada em uma fase inicial de análise, como uma representação em um nível alto de abstração, ou focada em fases de projeto e implementação com um detalhamento da família de processos e suas variabilidades em níveis mais baixos de abstração. A notação para modelagem de variabilidade pode ser textual, gráfica ou um misto das duas formas. No entanto, a expressividade de uma representação visual permite a identificação mais clara dos pontos de configuração dos processos a serem derivados. Em (TEIXEIRA, 2011), uma lista de requisitos desejáveis a uma notação que tenha por objetivo expressar variabilidade em LPrS foi definida, a partir do conhecimento existente na área de modelagem de variabilidades em LPS, e estendida durante o desenvolvimento desta pesquisa:

(1) A representação deve ser gráfica, permitindo um reconhecimento claro dos pontos onde os processos são similares e dos pontos onde diferem, necessitando de tratamento específico;

(2) A representação deve explicitar os elementos de processos fixos (invariantes), os elementos de processos configuráveis (pontos de variação) e suas alternativas de configuração (variantes). Os tipos de elementos do ponto de variação e suas variantes devem ser compatíveis e apropriadamente relacionados (atividades devem possuir outras atividades como variantes; tarefas devem possuir outras tarefas como variantes; produtos de trabalho devem possuir produtos de trabalho como variantes; papéis devem possuir outros papéis como variantes e ferramentais devem possuir outras ferramentas como variantes).

(3) A representação deve explicitar o grau de obrigatoriedade e opcionalidade na seleção de diferentes tipos de elementos ao instanciar um processo específico de projeto (classificação de opcionalidade). Além disso, algumas relações entre unidades de trabalho (atividades e tarefas) e seus elementos de suporte (produtos de trabalho, papéis e ferramentas) podem ser classificadas como opcionais se a participação do elemento não é essencial para garantir a execução da unidade de trabalho associada.

(4) A representação deve explicitar os relacionamentos entre elementos;

(5) A variabilidade comportamental deve ser modelada com alternativas de execução nos fluxos de controle e dados. Fluxos de controle definem o sequenciamento em que unidades de trabalho devem ser executadas. ***Fluxos alternativos*** e ***opcionais*** devem ser devidamente representados de forma a manter a consistência com a representação estrutural do processo;

(6) A representação deve explicitar relações de dependência e mútua exclusividade entre elementos, descrevendo a composição de processos que expressam a necessidade da seleção conjunta de elementos de processos e quando não é permitida a combinação de determinados elementos;

(7) A classificação de variabilidade de elementos de processos é ortogonal a classificação de opcionalidade. Desta forma, uma característica pode receber uma classificação quanto à opcionalidade e outra classificação complementar quanto à variabilidade.

A proposta desse trabalho está inserida no contexto de uma metodologia de LPrS que utiliza princípios da definição de processo baseada em componentes. A proposta de representação visa atender aos requisitos listados por meio de um modelo em diferentes níveis de abstração complementares, visando permitir uma análise gradual dos diferentes aspectos que constituem uma LPrS. Cada nível permite uma visão do conjunto de elementos de processos que o compõem e como estes se relacionam. O primeiro nível de abstração representa uma visão conceitual da LPrS, um alto nível de abstração de fácil entendimento representado por um modelo de domínio que agrega os elementos básicos de processos de software reutilizáveis (unidades de trabalho, papéis, produtos de trabalho e ferramentas) e analisa suas propriedades de opcionalidade e variabilidade no domínio. Uma das formas de representação deste nível consiste em um modelo de características, um modelo amplamente utilizado em LPS e que visa representar o conhecimento do domínio sem a necessidade de explicitar os detalhes de mais baixo nível de abstração. Além disso, os possíveis comportamentos esperados para processos a serem derivados são especificados por meio de fluxos e sequenciamentos de execução.

De forma adicional, um modelo de contexto pode ser especificado para caracterizar entidades e situações de contexto que apoiem a seleção de artefatos de processos reutilizáveis segundo regras de contexto estabelecidas.

O segundo nível complementa o modelo do domínio representando a LPrS com uma visão modular organizada por componentes e tendo como representação final uma Arquitetura de Componentes da LPrS. Tal arquitetura representa um arcabouço da LPrS pelo estabelecimentos das relações entre os componentes de processos de software identificados, que direciona a definição de processos específicos de projetos na EAPS, garantindo a consistência de composição dos elementos de processos reutilizáveis selecionados.

Tal modelagem, proposta neste trabalho para representação da LPrS em dois níveis, deve ser realizada segundo o metamodelo e a notação *OdysseyProcess-FEX*, inicialmente definidos em TEIXEIRA (2011) para representar LPrS usando um Modelo de Domínio de alto nível de abstração, um modelo de característica. A identificação inicial dos elementos a serem especificados pelo metamodelo foi realizada pela análise

de diferentes modelos de processos existentes como o *OpenUP*[14] e o RUP (*Rational Unified Process)* (IBM, 2009); metamodelos como o SPEM v.2 (OMG, 2008) e ontologias como a ontologia de processos (FALBO E BERTOLLO, 2005); um conjunto de representações de variabilidades de LPrSs, e o metamodelo de características do domínio de software *Odyssey-FEX* (OLIVEIRA, 2006). Esta última fonte foi a base para trabalhar aspectos específicos de uma modelagem de variabilidades com características, pois se trata de uma notação mais abrangente para representação das variabilidades referente às características de um domínio de software (TEIXEIRA *et al*., 2009).

O metamodelo visa explicitar os conceitos e as propriedades dos elementos e a notação especifica a representação gráfica dos conceitos formalizados pelo metamodelo aqui definido, com o intuito de apoiar o usuário na compreensão e bom uso da notação. O metamodelo possui um conjunto de restrições e propriedades, regras de boa formação que direcionam a construção e a verificação de consistência de um modelo do domínio de processos de software. Ambos foram evoluídos neste trabalho para melhor atender os requisitos do nível de características e especificar o nível de componentes.

A modelagem proposta trata variabilidades tanto de granularidade grossa quanto fina. A variabilidade de granularidade grossa, que se caracteriza por estar associada a pacotes (conjuntos) de elementos de processos (ALEIXO, 2011), pode ser trabalhada utilizando o conceito de componentes de processos. A variabilidade de granularidade fina, que se caracteriza pela especialização do conteúdo dos elementos de processos já existentes no núcleo da linha (ALEIXO, 2011), pode ser trabalhada pelo detalhamento dos elementos no modelo de características. Por exemplo, unidades de trabalho podem ser especificadas desde atividades até tarefas, que são compostas por passos, que podem ser classificados quanto à opcionalidade no domínio.

Variabilidades localizadas (pontuais), com interesse específico (ALEIXO, 2011), são diretamente modeladas nos modelos de análise e projeto. Variabilidades transversais, que agregam vários interesses (ALEIXO, 2011), podem ser trabalhadas combinando a modelagem do domínio com a modelagem contextual.

[14] Site: http://www.eclipse.org/epf/openup_component/openup_vision.php

4.2.1. Modelagem de Características

O nível de análise do metamodelo *OysseyProcess-FEX* (*Process Features View*) está descrito por três pacotes: a) Principal, que representa a parte da definição da taxonomia dos elementos, b) Relacionamento, que especifica propriedades dos relacionamentos existentes no modelo de características; e c) Regras de Composição, que especifica as regras de dependência e exclusividade entre as características de um modelo.

O metamodelo e a notação encontram-se detalhados em um Relatório Técnico (TEIXEIRA *et al*., 2016). A estrutura de especificação de cada pacote é realizada com uma explicação detalhada de cada elemento pertencente ao pacote associada a descrição dividida em sete campos:

• *Descrição*: representa uma descrição do elemento em alto nível. Especifica o significado principal do elemento;

• *Hierarquia*: especifica quais são as classes existentes em uma possível hierarquia, ou seja, determina as subclasses ou superclasses, do elemento descrito;

• *Atributos*: especifica o conjunto de atributos associados ao elemento. Para cada atributo, é definido o seu tipo básico, a sua cardinalidade e, caso exista, o seu valor *default*;

•*Associações*: estabelece a semântica das associações que podem ser estabelecidas entre elementos do metamodelo. Item de descrição incluído no Pacote de Relacionamentos e no Pacote Regras de Composição. Descreve o papel dos elementos envolvidos nos relacionamentos estabelecidos;

• *Restrições*: especifica a descrição de restrições que direcionam a construção e a verificação de consistência do modelo de características. O conjunto de todas as restrições especificadas para cada elemento constitui as regras de boa formação do metamodelo;

• *Notação*: especifica a descrição gráfica de representação visual do elemento definido; e

• *Exemplo*: especifica a descrição de um exemplo aplicando o elemento definido.

O pacote principal do metamodelo *OdysseyProcess-FEX* no nível de análise (*Process Features View*) (Figura 4.1) descreve a taxonomia dos elementos que o

constituem, definindo os elementos de processos de software reutilizáveis descritos como características, seus atributos e propriedades.

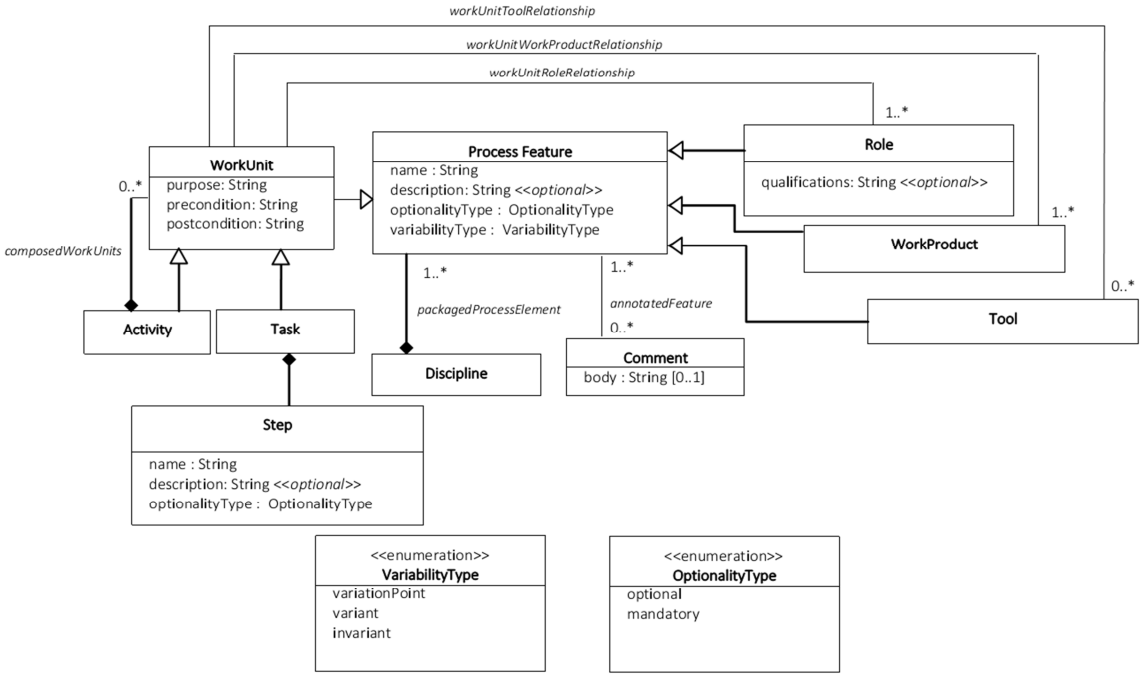


Figura 4.1- Metamodelo *OdysseyProcess-FEX*: Pacote Principal (*Process Features View*) (TEIXEIRA *et al*., 2016)

Este nível une a representação de conceitos de características, variabilidades (invariante, ponto de variação ou variante) e opcionalidades (mandatória ou opcional) à representação dos elementos que constituem a definição de processos de software, como atividades, tarefas, papéis, produtos de trabalho e ferramentas (Tabela 4.1), modelados como um grafo acíclico, sem necessariamente representar uma raiz para o modelo. Uma variante classificada como mandatória representa uma variante padrão dentro do ponto de configuração classificado como mandatório.

As classificações de elementos com relação à variabilidade e opcionalidade são mutuamente excludentes entre si. Desta forma, um elemento não pode receber simultaneamente dois tipos diferentes de classificações de uma mesma propriedade.

O relacionamento Alternativo denota a pertinência de uma variante (destino da relação) a um determinado ponto de variação (origem da relação) e permite a identificação gráfica do conceito de variabilidade. A opcionalidade é graficamente representada por um retângulo formado por uma linha tracejada para as características classificadas como opcionais.

Tabela 4.1 - Categorias de características *OdysseyProcess-FEX*

Categoria		Ícone	Estereótipo
Atividade	Elemento de processo que representa o agrupamento de unidades de trabalhos menores, representadas por outras atividades ou por tarefas, e que produzem um resultado esperado para o processo.		***<<activity>>***
Tarefa	Elemento de processo que representa uma unidade de trabalho descrita por de passos. Sua granularidade é definida como o menor agrupamento possível de ações.		***<<task >>***
Papel	Elemento de processo que representa um conjunto de habilidades, competências e responsabilidades de um indivíduo ou um conjunto de indivíduos. Não especifica um indivíduo ou recurso em particular.		***<<role>>***
Produto de Trabalho	Elemento de processo que representa um artefato consumido, modificado ou produzido por uma unidade de trabalho.		***<<work product>>***
Ferramenta	Elemento de processo que descreve capacidades de uma unidade de automação para apoiar a exceção da unidade de trabalho associada.		***<<tool>>***

Os relacionamentos entre as características incorporam relacionamentos da UML, relações que tornam explícitas a representação de variabilidades e relações específicas da área de processos de software incluindo relações de fluxos de controle (Tabela 4.2). Para cada relacionamento, um conjunto de restrições foi determinado de forma a especificar as possíveis combinações de categorias de características pertencentes ao relacionamento e algumas propriedades específicas.

Tabela 4.2 - Relacionamento do metamodelo *OdysseyProcess-FEX* Nível Análise Modelo de Características

Relacionamento		Representação
Alternativo	Relacionamento existente entre um ponto de variação e suas variantes. É representado por linhas simples entre Ponto de Variação e Variantes, interligadas por uma linha curva.	
Agregação	Uma associação que representa uma agregação (isto é, um relacionamento de todo/parte). Possui a representação de associações binárias, mas diferencia-se por adicionar um diamante não-preenchido na extremidade agregada da linha de associação.	
Composição	Representa um relacionamento de todo/parte mais forte do que agregação. Neste relacionamento, as partes não existem independentes do todo. Possui a representação de associações binárias, mas diferencia-se por adicionar um diamante preenchido na extremidade composta da linha de associação.	
Ligação PapelUnidadeDeTrabalho	Estabelece um relacionamento de associação entre uma característica da categoria Unidade de Trabalho (Atividade ou Tarefa) e uma ou várias características da categoria Papel participantes na sua execução. Tal relacionamento pode apresentar diferentes estereótipos que indicam o tipo de participação que determinado papel possui na execução da unidade de trabalho associada. Um conjunto inicial de possíveis responsabilidades foi definido: *<<performer>>; <<responsible>>; <<additional>>; <<assistant>>; <<supervisor>>;<<inspector>>; <<informed>>;<<consulted>>*. Tal relação possui o estereótipo *<<optional>>* para indicar que a relação é opcional no domínio.	*<<workUnitRoleRelationship>>*
Ligação ProdutoDeTrabalho UnidadeDeTrabalho	Estabelece um relacionamento de associação entre uma característica da categoria Unidade de Trabalho (Atividade ou Tarefa) e uma ou várias características da categoria ProdutoDeTrabalho que representam artefatos a serem consumidos, produzidos ou modificados pela execução da unidade de trabalho. Tal relacionamento pode apresentar diferentes estereótipos que indicam o tipo de participação do produto de trabalho na execução da unidade de trabalho associada: • Produto de Trabalho a ser consumido: ***<<in>>*** • Produto de Trabalho a ser produzido: ***<<out>>*** • Produto de Trabalho a ser modificado: ***<<inout>>*** Tal relação possui o estereótipo *<<optional>>* para indicar que a relação é opcional no domínio.	 *<<workUnitWorkProduct Relationship>>*

Relacionamento		Representação
Ligação Papel ProdutoDeTrabalho	Estabelece um relacionamento de associação entre uma característica ProdutoDeTrabalho e uma ou várias características Papel com algum tipo de responsabilidade sobre o produto de trabalho. Tal relacionamento pode apresentar diferentes estereótipos que indicam o tipo de participação que determinado papel possui na execução da unidade de trabalho associada. Um conjunto inicial de possíveis responsabilidades foi definido: *<<performer>>; <<responsible>>; <<additional>>; <<assistant>>; <<supervisor>>;<<inspector>>; <<informed>>;<<consulted>>*. Tal relação possui o estereótipo <<*optional*>> para indicar que a relação é opcional no domínio.	*<<workProductRole Relationship>>*
Ligação UnidadeDeTrabalho Ferramenta	Estabelece um relacionamento de associação entre um elemento de processo da categoria Unidade de Trabalho (Atividade ou Tarefa) e um ou vários elementos de processo da categoria Ferramenta que podem prover algum tipo de apoio ferramental na execução da unidade de trabalho. Tal relação possui o estereótipo <<*optional*>> para indicar que a relação é opcional no domínio.	*<<workUnitToolRelationship>>*
Fluxo de Controle – Fluxo de Sequência	Descreve uma relação de fluxo ordenado de execução entre duas unidades de trabalho (origem e destino), indicando que a execução da atividade destino só pode ser iniciada com o fim da execução da atividade anterior. É representada por uma linha com uma seta indicando a direção da sequência da origem para o destino. Tal relacionamento pode ser classificado como opcional quando uma ou as duas unidades de trabalho envolvidas são classificadas como opcionais.	Sequência mandatória Sequência opcional
Fluxo de Controle – Fluxo de Decisão	Descreve um ponto em um fluxo de execução onde uma escolha entre múltiplos caminhos deve ser realizada. A relação é representada por apenas um fluxo de entrada e múltiplos caminhos de execução que representam fluxos alternativos. Tal tipo de relação introduz condições em unidades de trabalhos e seus fluxos de execução. Tal relacionamento pode ser classificado como opcional quando uma ou as duas unidades de trabalho envolvidas são classificadas como opcionais.	Nó de decisão: Mandatório Opcional
Fluxo de Controle – Fluxo Paralelo	Descreve uma relação entre três ou mais unidades de trabalho, indicando que o fluxo de execução é dividido em múltiplos fluxos concorrentes (execução paralela) com uma unidade de trabalho como início do fluxo e múltiplas unidades de trabalho como sequências de execução interligadas por linhas simples e com um nó de divisão entre a origem e os destinos.	Nó de Divisão (*fork node*): Mandatório Opcional
Fluxo de Controle – Sincronismo	Descreve uma relação entre três ou mais unidades de trabalho, indicando que múltiplos fluxos de execução concorrentes devem ser sincronizados ao final e unidos em um único fluxo (execução síncrono) com múltiplas unidades de trabalho como inícios dos fluxos e uma unidade de trabalho como sequência final de execução interligadas por linhas simples e com um nó de junção entre as origens e o destino.	Nó de Junção (*Join node*): Mandatório Opcional

As Regras de Composição especificam a semântica de regras de dependência e de regras de mútua exclusividade entre características. Tais regras são especificadas pela estrutura: Antecedente + palavra-chave + Consequente. O antecedente e o consequente são expressões, que podem ser literais, expressão elementar, que designa uma única característica, ou booleanas, que denotam uma combinação entre características pelo uso de operadores booleanos: AND, OR, XOR, NOT. A palavra-chave apresentada na estrutura define o tipo de Regra de Composição: Regras Inclusivas são denotadas pela palavra-chave “requer”, enquanto que Regras Exclusivas são denotadas pela palavra-chave “exclui”. As Regras de Composição são graficamente

representadas por uma marcação, em todas as características pertencentes à regra, no canto inferior do retângulo que representa a característica envolvida na regra estabelecida. A marcação também apresenta uma numeração (n), que representa a ordem sequencial de criação das regras. Para Regras de Composição Inclusivas, a marcação usada com a numeração corresponde a "R_n" e é apresentada no canto inferior direito. Já para Regras de Composição Exclusivas, a marcação usada com a numeração corresponde a "X_n" e é apresentada no canto esquerdo.

Um exemplo de modelagem de características utilizando a *OdysseyProcess-FEX* é apresentado na Figura 20. Esta LPrS descreve três atividades mandatórias iniciais do planejamento de projetos de software: *Desenvolver o Termo de Abertura do Projeto*; *Determinar tamanho do Projeto* e *Determinar esforço do projeto*. O tamanho pode ser determinado por duas técnicas diferentes: contagem indicativa PF ou base histórica (variantes opcionais e mutuamente excludentes do ponto de variação *Determine Project Size*). O esforço pode ser dimensionado usando duas técnicas diferentes: usando PF ou base histórica (variantes opcionais e mutuamente excludentes do ponto de variação *Determine Project Effort*). Cada uma das variantes do cálculo do esforço depende do resultado de cada uma das técnicas do cálculo do tamanho. Essa dependência é expressa por meio de regras de composição exclusivas, graficamente representadas pelas marcações R e R_1. O principal papel responsável por essas atividades consiste no Gerente de Projeto, considerado mandatório no domínio. Uma ferramenta para análise dos dados históricos pode ser usada para apoiar a execução da atividade de Determinar tamanho do Projeto por meio de base histórica. A atividade Desenvolver o Termo de Abertura do Projeto produz como resultado o produto de trabalho Termo de Abertura (*Project Charter*) que é insumo para as duas atividades subsequentes. As regras de composição visualmente representadas pelas marcações R, R_1 e X correspondem a:

- R: Determine effort using size estimated by FP *requires* (Determine by Indicative Function Points (FP) AND FP Spreadsheet filled with Project Size)
- R_1: Determine effort using size estimated by historical base *requires* (Determine by projects historical base AND Project Size estimated by Historical Analysis)

- X: (Determine by Indicative Function Points (FP) AND FP Spreadsheet filled with Project Size) *excludes* (Determine by projects historical base AND Project Size estimated by Historical Analysis)

A visão apresentada na Figura 4.2 pode ser classificada como estrutural, identificando os elementos de processos que compõem o domínio sem especificar o fluxo de controle entre as unidades de trabalho.

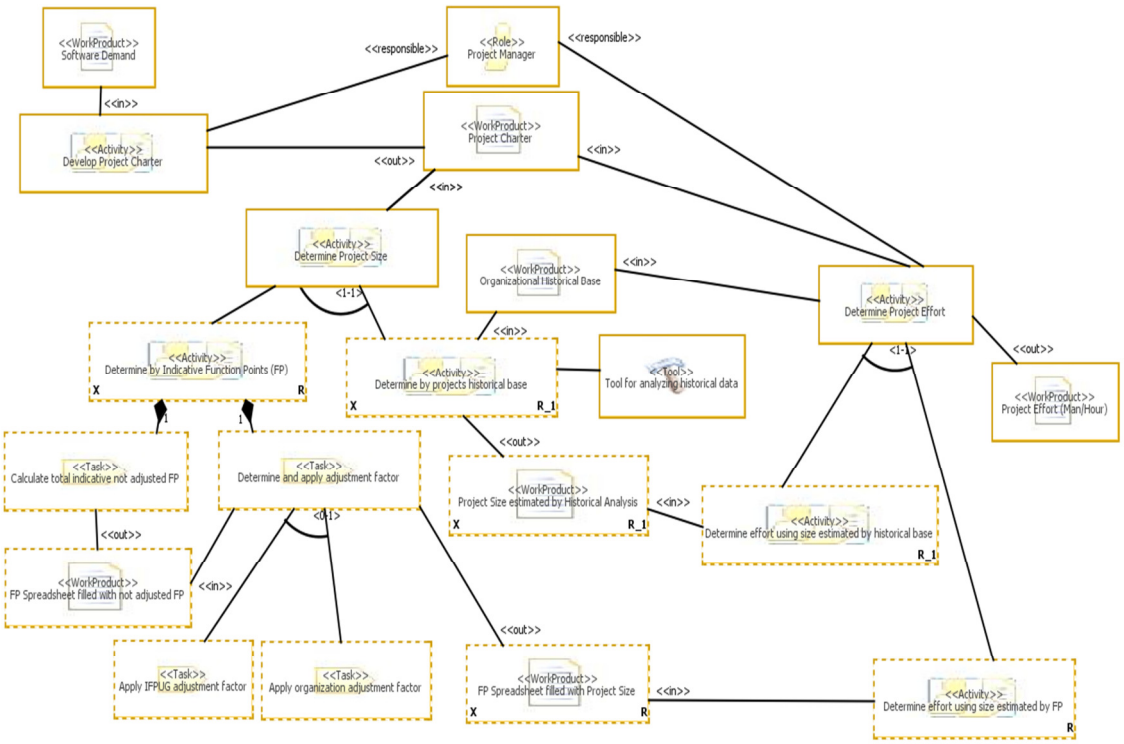


Figura 4.2- LPrS de Planejamento de Projetos de Software – Modelo de Características Visão Estrutural

Neste exemplo, apenas fluxos sequenciais foram especificados entre as unidades de trabalho. Desta forma, uma visão comportamental pode ser vista na Figura 4.3, que representa esses fluxos sequenciais do domínio modelado.

Figura 4.3 - LPrS de Planejamento de Projetos de Software – Modelo de Características Visão Comportamental

A primeira versão do metamodelo foi avaliada no trabalho de TEIXEIRA (2011). Diante dos resultados desta avaliação e avanço dos estudos de modelagem de variabilidades realizados durante o decorrer desta tese, adaptações foram propostas no metamodelo e na notação. Dentre as alterações realizadas no nível de análise,

podemos enumerar a retirada de duas categorias de características que não contribuem para o entendimento do domínio (conceito e prática). Essas categorias podem ser colocadas como comentários associados ao modelo e, para isso, o elemento Comentário (<<*Comment*>>) foi incluído. O elemento Comentário pode ser definido como uma anotação textual associada a um elemento com o objetivo de acrescentar alguma informação adicional para a organização de conhecimento sobre o domínio. Outra alteração foi a reorganização hierárquica dos elementos classificados nas categorias Atividade e Tarefa como tipos de Unidades de Trabalho. O elemento Passo que era um detalhamento da característica da categoria Tarefa foi eliminado por se tratar de uma descrição da forma de execução, mais apropriado para a visão estrutural. E a característica da categoria Disciplina foi reestruturada, passando a ter a definição do conceito de pacote para organização do domínio em escopos menores ou subdomínios. Elemento que representa uma categorização de trabalho baseado em uma similaridade de interesses e propósito de resultados. O relacionamento Herança foi removido por não contemplar nenhuma especificidade de representação dentro do domínio de processos. Além disso, algumas restrições descritas no metamodelo inicial foram reavaliadas (por exemplo, o tipo de alternativas associado a cada tipo de elemento só pode ser o mesmo do elemento classificado como ponto de variação, e o elemento ferramenta deixou de ser apenas invariante, podendo assumir outros tipos de variabilidade). O fluxo de controle entre unidades de trabalho também passou a melhor trabalhado com o tratamento da variação desses fluxos de controle por meio do conceito de opcionalidade. No trabalho atual, o nível de projeto do modelo de componentes foi especificado. As subseções abaixo descrevem de forma resumida os principais elementos do metamodelo e da notação nos níveis de análise e componentes. A descrição completa do metamodelo e da notação com exemplos encontra-se em um Relatório Técnico (TEIXEIRA *et al*., 2016).

4.2.2. Modelagem de Componentes

O nível de projeto (*Process Components View*) descreve os elementos do metamodelo de componentes em um único pacote constituído por componentes, seus atributos específicos e relações via o estabelecimento de interfaces (Figura 4.4). O metamodelo e a notação encontram-se detalhados em (TEIXEIRA *et al*., 2016). A

estrutura de descrição de cada elemento é composta pelos sete campos descritos na Seção 4.2.1.

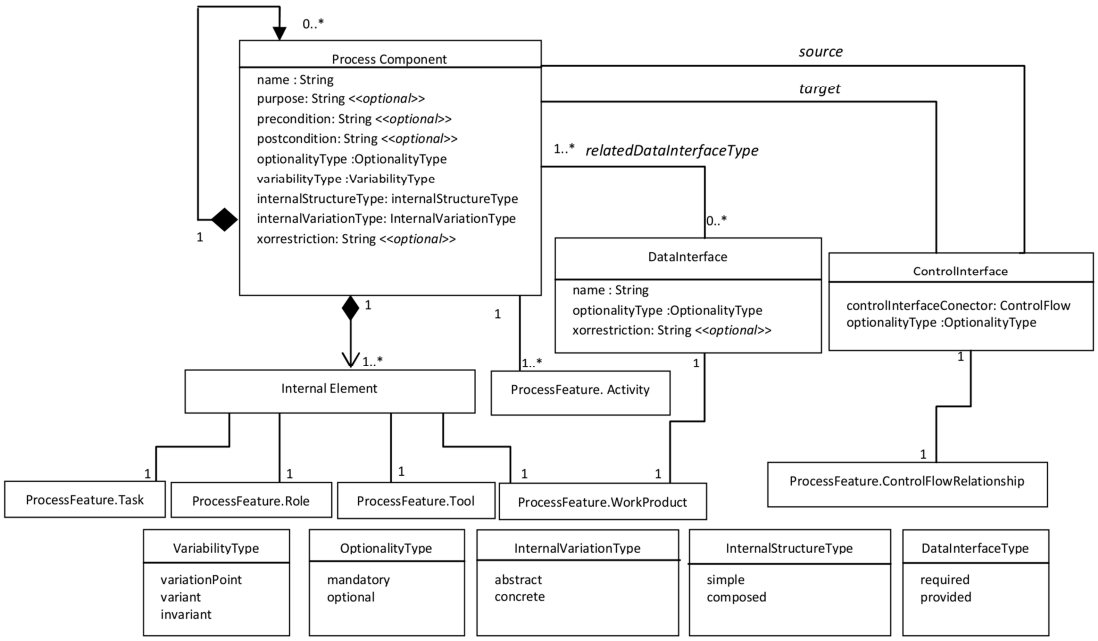


Figura 4.4 - Metamodelo do Nível de Componentes do *OdysseyProcess-FEX (Process Components View)*: Pacote Principal

Um componente de processo pode ser entendido como uma abstração de um fragmento de processo baseada no princípio da "caixa-preta". Desta forma, um componente representa uma parte modular de um processo que encapsula seu conteúdo e se comunica com o ambiente via interfaces. Cada componente é composto por unidades de trabalho (atividades e tarefas) que representam exatamente suas possibilidades de realização, ou seja, unidades de trabalho que quando executadas apresentam o comportamento do componente e produzem os resultados esperados (produtos de trabalho) que agregam valor ao projeto. Um componente de processo deve ser definido minimamente por uma atividade. Uma unidade de reutilização menor poderia introduzir uma complexidade desnecessária e tornar atividades de reutilização mais difíceis (BARRETO, 2011). A granularidade do componente pode ser minimamente uma atividade ou podendo ser composto por outro(s) componente(s), podendo assumir os valores simples e composto, respectivamente. Desta forma, um componente pode ser classificado quanto à opcionalidade (opcional e mandatório), variabilidade (ponto de variação, variante e invariante) e estrutura interna (simples ou composto). Tais classificações são ortogonais entre si. Além disso, cada componente possui uma classificação que indica se existe algum tipo de variação interna no

componente, ou seja, se existe alguma necessidade de configuração dos elementos que o compõem (unidades de trabalho, produtos de trabalho, papéis ou ferramentas). Pode assumir os valores concreto e abstrato. Um componente que não possui variação interna a ser resolvida (todos os elementos de processo que compõem o componente estão definidos da maneira que serão instanciados, sem nenhuma decisão adicional de definição) é classificado como concreto. Caso contrário, será classificado como abstrato.

O meio de comunicação entre componentes de processos e dos componentes com o meio é realizado via interfaces, descrevendo suas especificações. Dois tipos de interfaces podem ser definidos, interface de dados e interface de controle, representado fluxos de dados e fluxos de controle entre componentes, respectivamente.

A interface de dados representa o meio de troca de dados entre componentes de processos e entre componentes e o meio. Corresponde à definição da relação de um produto de trabalho com unidade(s) de trabalho envolvida(s) na execução do componente que extrapola a fronteira do componente, ou seja, troca entre componentes ou do componente com o meio. Cada interface deve estar associada a um produto de trabalho que pode ser requerido (entrada do componente) ou provido (saída do componente).

A execução do fluxo entre componentes é representada pelas interfaces de controle. Uma interface de controle compreende um componente como origem e outro componente destino de entrada do fluxo. Entre eles um conector é utilizado para estabelecer uma regra de associação entre os componentes. Conectores são linhas entre componentes usados para expressar o tipo de conexão estabelecido. Nessa metodologia, a regra de associação é representada pelos tipos de fluxos de controle especificados no nível de análise: sequencial, paralelo, síncrono e de decisão.

Um exemplo de modelagem de componentes utilizando a *OdysseyProcess-FEX* é apresentado na Figura 4.5. Este modelo de componentes corresponde a uma representação modular da LPrS apresentada pelo modelo de características da Figura 4.2. Para manter a consistência entre as diferentes representações envolvidas, é necessário criar mecanismos de rastreamento entre os diferentes níveis estabelecidos. Um conjunto de heurísticas para apoiar tal mapeamento e manutenção da

consistência de representações é proposto, estabelecendo relações entre os elementos dos diferentes níveis de abstração que compõem a linha.

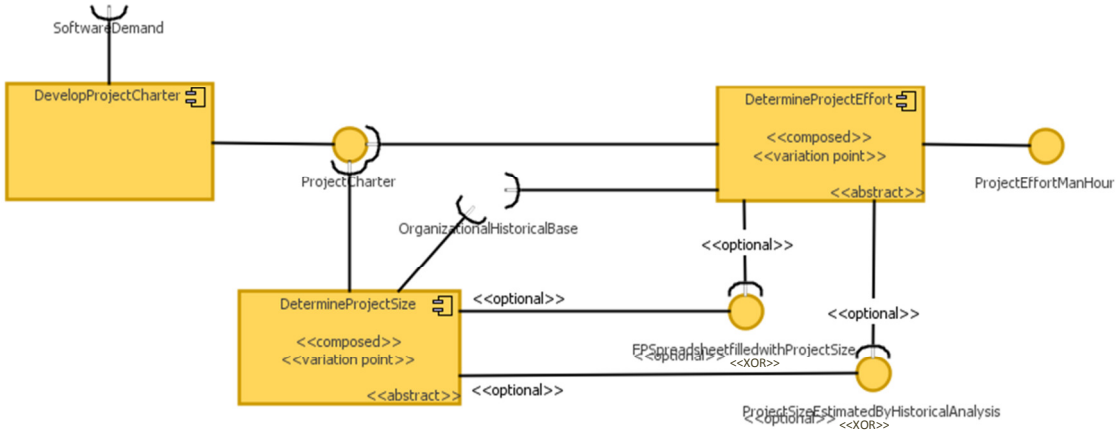


Figura 4.5 – Modelo de Componentes da LPrS de Planejamento de Projetos de Software

4.2.3. Estudo Comparativo de Modelagem de Variabilidade de LPrS

Apesar de diferentes representações serem propostas para a modelagem de variabilidades de processos de software, cada uma referente a uma notação específica, alguns conceitos possuem a mesma semântica, independente de representações gráficas ou nomenclaturas. Essas similaridades entre as notações representam propriedades comuns, formando um núcleo ou base de conceitos, a partir do qual cada notação pode ser estendida com novos conceitos. No entanto, apresentam particularidades que influenciam na modelagem do domínio e as difere entre si. Cada representação pode apresentar particularidades que buscam aumentar a capacidade semântica e sintática de representação e maior expressividade na modelagem dos artefatos de processos reutilizáveis.

Assim como a área de LPrS ainda pode ser considerada recente na literatura, a modelagem de processos de software reutilizáveis também não apresenta um consenso, não existindo uma notação padrão para a modelagem de variabilidades em LPrS. A escolha entre as propostas de notações de uma modelagem mais apropriada pode estar relacionada a uma série de fatores, como: maior conhecimento e familiaridade da equipe, popularização de uma determinada notação, disponibilidade de ambiente de desenvolvimento que suporte a notação, ou maior adequação aos requisitos que atendam à modelagem, dentre outros fatores.

Dentro desse contexto e levando em consideração a importância da modelagem de variabilidade de LPrS como suporte para a reutilização, um estudo

comparativo foi realizado entre a notação *OdysseyProcess-FEX* proposta neste trabalho e a modelagem proposta em BARRETO (2011). A escolha da modelagem de BARRETO (2011) para comparação se deve ao fato da abordagem proposta por este autor trabalhar os conceitos de características e componentes para representação de LPrS, os mesmos modelos aplicados na abordagem *OdysseyProcessReuse,* ainda que os conceitos tenham sido trabalhados com semânticas diferentes. Outras modelagens de variabilidades identificadas ao longo do Capítulo 2, Seção 2.4, em geral apresentam apenas um nível de abstração e não utilizam o conceito de componentes, o que poderia gerar uma análise incompleta dos conceitos propostos.

O estudo comparativo realizado entre as notações teve como objetivo observar as diferenças de propriedades de uma notação para outra e foi dividido em três classes de conceitos, visando facilitar a compreensão das notações e um maior detalhamento e formalização dos resultados dessa análise. A primeira classe de conceitos trata da taxonomia, fazendo um paralelo entre as diversas classificações e a presença de certas propriedades relativas a um elemento de processo, como, por exemplo, classificações de opcionalidade e variabilidade, dentre outras. A segunda classe trata do conceito de dependência e mútua exclusividade entre artefatos do domínio. A terceira, e última, trata do conceito referente aos relacionamentos presentes entre os elementos do domínio. A seguir, essas classes são detalhadas.

A análise entre as notações será realizada pela modelagem de uma LPrS nas duas notações observadas: LPrS para Medição no Contexto Organizacional (Figura 4.6). Esta LPrS foi inicialmente modelada em CARDOSO (2012), utilizando a abordagem proposta por BARRETO (2011). Nesta tese, a mesma LPrS será modelada segundo a abordagem proposta nesta tese e a notação *OdysseyProcess-FEX*.

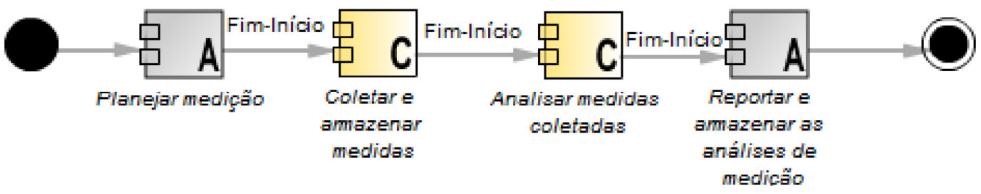


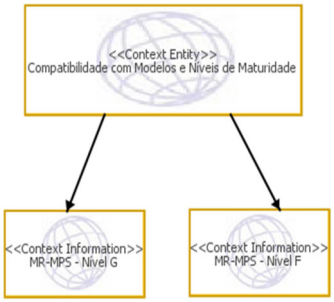
Figura 4.6 – LPrS para Medição no Contexto Organizacional (CARDOSO, 2012)

Na abordagem proposta por BARRETO (2011), a primeira atividade define características de processo que guiam a definição ou seleção (nos casos em que já existam alguns disponíveis na base de componentes) dos demais elementos

reutilizáveis. Uma característica pode ser vista como um aspecto, qualidade ou caracterização com a qual o processo precisa ser compatível e restringe a utilização de componentes. Características podem ser consideradas um conjunto de regras aplicadas a componentes de processo que guiam a definição de processos com base nas necessidades do processo (mapeadas como características de processo).

Na abordagem *OdysseyProcessReuse*, a atividade de modelagem também começa pela definição de um modelo de características. No entanto, o conceito de característica é utilizado de forma diferenciada. Tal modelo representa uma visão detalhada da LPrS representada por um modelo de domínio que agrega os *elementos básicos de processos de software reutilizáveis* (unidades de trabalho, papéis, produtos de trabalho e ferramentas) e analisa suas propriedades de opcionalidade e variabilidade no domínio sem especificar detalhes de instanciação ou implementação de tais elementos. De forma complementar, um modelo de contexto pode ser também especificado para caracterizar entidades e situações de contexto que apoiem a seleção de artefatos de processos reutilizáveis, segundo regras de contexto estabelecidas. Este modelo de contexto parece ter uma descrição mais aproximada da intenção de definição de características da abordagem de BARRETO (2011). A Tabela 4.3 apresenta um mapeamento dos conceitos da modelagem desta informação nas duas abordagens.

Tabela 4.3 – Modelagem de Informação Contextual do Domínio (informações que guiam a seleção de artefatos reutilizáveis)

<table>
<tr><th>OdysseyProcessReuse</th><th>BARRETO (2011)</th></tr>
<tr><td>Modelagem Contextual do Domínio: a definição do contexto é representada gráfica por meio de um modelo de características de contexto aplicando a notação UbiFEX.</td><td>Definição de Características: a abordagem não propõe uma representação gráfica para características, sendo usada descrição tabular.</td></tr>
<tr><td>Entidade de Contexto e Informações de Contexto
Entidade de Contexto_Ex: Compatibilidade com Modelos e Níveis de Maturidade
Informação de Contexto_Ex: MR-MPS - Nível G; MR-MPS - Nível F
 
</td><td>Tipo de Característica e Característica de Processo
Tipo de Característica_Ex: Compatibilidade com Modelos e Níveis de Maturidade
Característica de Processo_Ex: MR-MPS - Nível G; MR-MPS - Nível F
<table><tr><th>Tipo da Característica</th><th>Característica de Processo</th></tr><tr><td rowspan="2">Compatibilidade com Modelos e Níveis de Maturidade</td><td>MR-MPS - Nível G</td></tr><tr><td>MR-MPS - Nível F</td></tr></table></td></tr>
</table>

OdysseyProcessReuse	**BARRETO (2011)**
Situação de Contexto e Regras de Contexto Ex: • Situação de Contexto: Compatibilidade com Modelos e Níveis de Maturidade. MR-MPS - Nível F Atendido (contexto selecionado necessita do Nível F) • Regra de Contexto: **CR**: Compatibilidade com Modelos e Níveis de Maturidade. MR-MPS - Nível F Atendido *implies* Características do Domínio (Atividade_Planejar Medição) 	**Relação entre característica e componente**: regras aplicadas a componentes de processo que guiam a definição de processos apresentadas na descrição tabular do componente, campo: Características Atendidas <*Características atendidas pelo Componente de Processo*> Ex: Características Atendidas: MR-MPS - Nível F

Desta forma, para o componente Planejar Medição (Figura 4.6), por exemplo, a característica atendida é MR-MPS - Nível F, regra aplicada ao componente pela descrição tabular, campo: **Características Atendidas: MPS - Nível F** (abordagem BARRETO, 2011). Utilizando a abordagem *OdysseyProcessReuse* e a notação *OdysseyProcess-FEX* e *UbiFEX* para modelagem contextual, as situações de contexto são modeladas e regras de contexto são definidas, indicando quais características devem ser selecionadas quando situações de contexto ficam ativas: CR: Compatibilidade com Modelos e Níveis de Maturidade. MR-MPS - Nível F Atendido *implies* Características do Domínio (Atividade_Planejar Medição). Graficamente, a marcação **CR** que representa tal regra de contexto é apresentada no canto superior da característica requerida para este contexto, conforme apresentado na Tabela 4.3.

As maiores diferenças estão na representação gráfica disponibilizada pela abordagem *OdysseyProcessReuse* e a possibilidade da seleção de diferentes tipos de elementos de processos combinados de diferentes formas, já que a modelagem contextual implica na seleção ou não de características do domínio, diferentemente da abordagem de BARRETO (2011) que aplica tal seleção a componentes.

A abordagem *OdysseyProcessReuse* apresenta um nível de abstração de modelagem a mais que a abordagem de BARRETO (2011), onde o modelo de características da *OdysseyProcessReuse* contempla elementos descritos no modelo de componentes da abordagem de BARRETO (2011). Desta forma, as próximas análises

são realizadas comparando a modelagem de característica e de componentes pela notação *OdysseyProcess-FEX* e a modelagem de componentes proposta em BARRETO (2011).

4.2.3.1. Taxonomia

Essa classe de análise consiste nos elementos de processos que podem ser representados e suas propriedades. O primeiro elemento de processo observado consiste nas unidades de trabalho que são ações que executam alguma transformação em artefatos do processo (Tabela 4.4).

Tabela 4.4 – Modelagem Elementos de Processo do Domínio (Unidade de Trabalho)

OdysseyProcessReuse	**BARRETO (2011)**
Unidade de Trabalho: ações que executam alguma transformação em artefatos do processo	
Modelo de Características: Atividade e Tarefa Modelo de Componentes: Componente de Processo	Atividade e Componente de Processo
Propriedade Opcionalidade dos Elementos Unidades de Trabalho	
Todos os elementos (atividade, tarefa, componente de processo) podem ser classificados como ***opcional*** ou ***mandatório*** tanto no modelo de características como no de componente	Todos os elementos (atividade e componente de processo) podem ser classificados como ***opcional*** ou ***mandatório***
Propriedade Variabilidade dos Elementos Unidades de Trabalho	
Todos os elementos (atividade, tarefa, componente de processo) podem ser classificados como ***ponto de variação***, ***variante*** ou ***invariante*** tanto no modelo de características como no de componente	As variantes que compõem um componente abstrato apresentadas na descrição tabular do componente, campo: Variantes
Classificação do Componente de Processo como abstrato ou concreto	Classificação do Componente de Processo como abstrato ou concreto
Propriedade de Composição da Estrutura Interna de um Componente de Processo	
Classificação do Componente de Processo como simples ou composto	Sem correspondência

Um exemplo foi construído para um componente da LPrS Medição apresentada. O componente *Planejar Medição* foi modelado utilizando as duas notações e encontra-se descrito a seguir. O exemplo também abrange as variantes relacionadas a este componente, classificado como um ponto de variação do domínio.

Exemplo

- **Abordagem BARRETO (2011) –** LPrS modelada em CARDOSO (2012) - Componente de Processo: Planejar Medição

Planejar medição	
Identificador:	COP.MED.ORG.ABS.0058
Nome:	Planejar medição
Tipo:	Abstrato
Descrição:	Estabelecer plano de medição organizacional através da definição dos objetivos de medição oriundos das necessidades de informação organizacional, das medidas a serem coletadas, dos procedimentos de coleta e armazenamento, bem como dos procedimentos de análise e divulgação dos resultados. Para a identificação dos objetivos e necessidades de informação, podem ser utilizadas diversas técnicas, tais como: GQM (*Goal Question Metric*) e GQ(I)M (*Goal Question (Indicator) Measure*).
Definido por:	COPPE/UFRJ
Critérios de Entrada:	-
Critérios de Saída:	-
Responsável:	-
Participantes:	-
Ferramentas de Apoio:	-
Artefatos Requeridos:	-
Artefatos Produzidos:	-
Características Atendidas:	MR-MPS - Nível F
Medidas:	-
Variantes deste componente:	Planejar medição utilizando o método GQM; Planejar medição utilizando o método GQ(I)M.
Arquitetura Interna:	-

- **Abordagem *OdysseyProcess-FEX***

  - **Modelo de Características**

A granularidade de unidade de trabalho é diferente de uma abordagem para outra. Na modelagem usando *OdysseyProcess-FEX* no nível de características, é possível representar atividades e tarefas (Figura 4.7). Um exemplo desta diferença pode ser observado ao analisar o componente *Planejar Medição* e uma das variantes, *Planejar medição utilizando o método GQM* (CARDOSO, 2012). Na *OdysseyProcess-FEX* - nível de características, componentes são modelados como atividades. As atividades que compõem a arquitetura interna do componente variante *Planejar medição utilizando o método GQM,* descritas em (CARDOSO, 2012), são modeladas como tarefas.

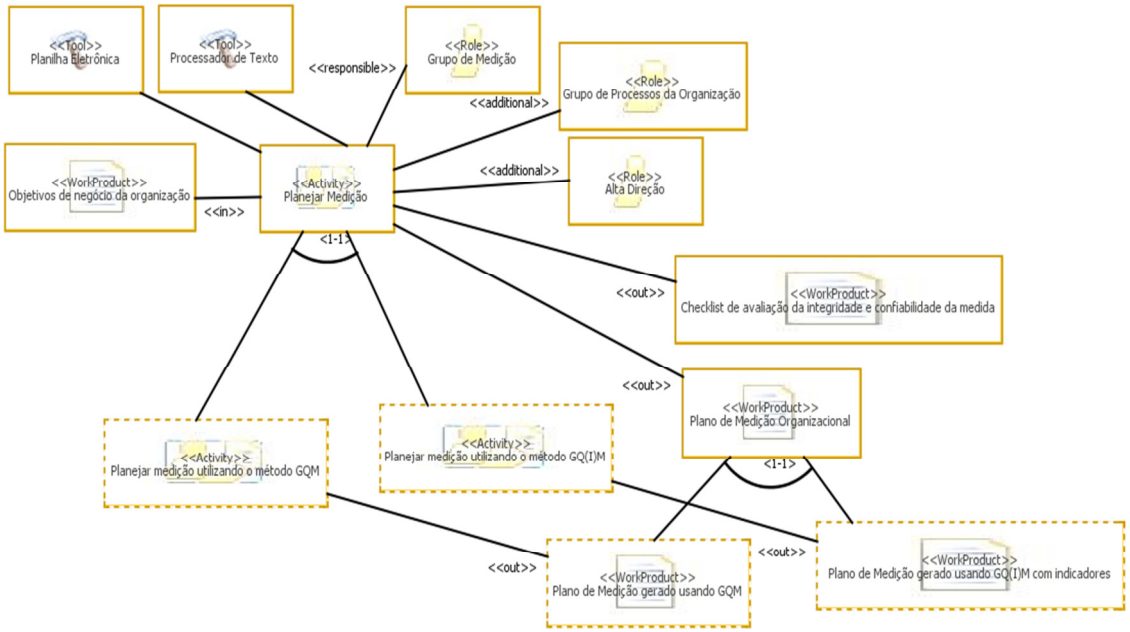


Figura 4.7 – Visão Estrutural da Atividade *Planejar Medição*

A classificação de opcionalidade também apresenta diferenças entre as duas modelagens. A atividade *Planejar medição utilizando o método GQM* é classificada como opcional na modelagem de características usando a *OdysseyProcess-FEX*, considerando o domínio como um todo. Essa classificação não é atribuída ao componente *Planejar medição utilizando o método GQM,* na modelagem apresentada em CARDOSO (2012).

Ao observar o componente *Planejar medição utilizando o método GQM* na modelagem apresentada em CARDOSO (2012), as atividades da arquitetura interna são modeladas como mandatórias e opcionais. Uma possível interpretação é que a classificação da opcionalidade é realizada observando o contexto do componente em si. Na modelagem usando a *OdysseyProcess-FEX* (Figura 4.8), essas atividades, que estão modeladas como tarefas, são classificadas como opcionais, pois a classificação de opcionalidade é observada com relação ao domínio como um todo. Apenas a tarefa *Especificar procedimentos de coleta e armazenamento dos dados de medição* é classificada como mandatória, pois é executada da mesma forma nas duas variantes associadas a *Planejar Medição*. Desta forma, a mandatoriedade ou opcionalidade com relação à atividade *Planejar medição utilizando o método GQM* é modelada pelos relacionamentos de composição e agregação, respectivamente. Como todas as tarefas são mandatórias na execução desta atividade, todas foram relacionadas utilizando o

relacionamento Composição. Tais relacionamentos não possuem correspondência direta na abordagem de BARRETO (2011), conforme descrito na Subseção 4.2.3.2.

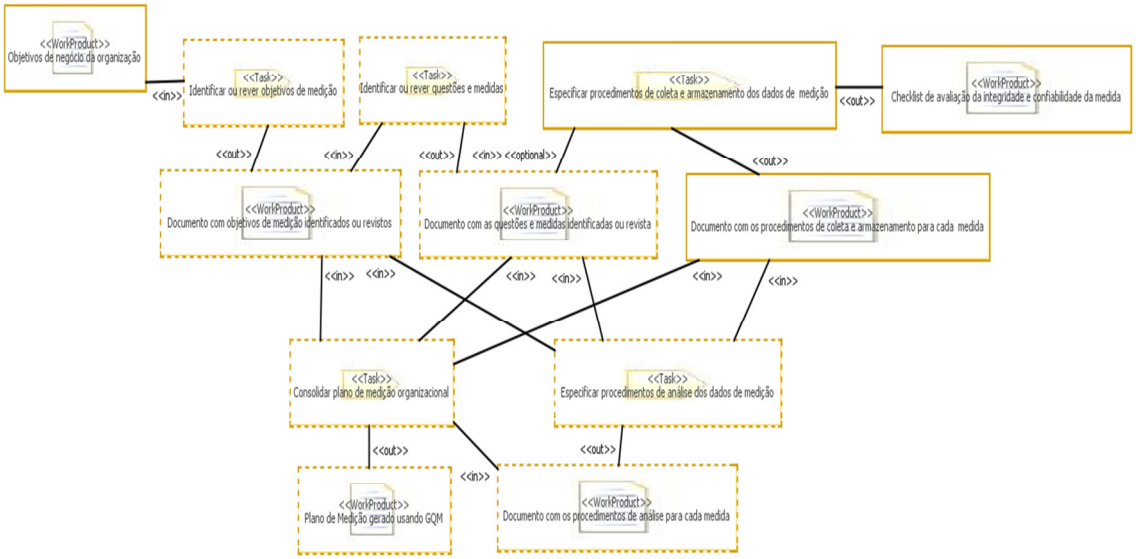


Figura 4.8 – Visão Estrutural da Atividade *Planejar Medição usando o método GQM*

A classificação de variabilidade também apresenta diferenças entre as duas modelagens. A notação *OdysseyProcess-FEX* atribui a classificação dos elementos como ponto de variação, variante e invariante que não está presente explicitamente na abordagem de BARRETO (2011). Apenas componentes que são classificados como abstratos podem possuir variantes em sua composição e as variantes são mutuamente excludentes entre si. No entanto, sem analisar a arquitetura interna não é possível identificar se o componente abstrato possui variantes ou elementos opcionais que justifiquem a classificação como abstrato. Na notação *OdysseyProcess-FEX,* um componente de processo possui quatro classificações ortogonais: opcionalidade (opcional ou mandatório), variabilidade (ponto de variação, variante ou invariante), variação interna (concreto ou abstrato) e estrutura interna (composto ou simples) (Tabela 4.4). A classificação de variação interna (concreto ou abstrato) tem a função similar a da proposta em BARRETO (2011). No entanto, as outras classificações auxiliam a identificar que tipo de configuração está associada ao componente sem necessariamente analisar sua arquitetura interna, já que estereótipos são usados para representar componentes que são pontos de variação (que possuem variantes) em sua estrutura interna. Além disso, é possível analisar se um componente é composto por outros pela classificação da estrutura interna, classificação que não tem correspondência na abordagem de BARRETO (2011).

Outra diferença é que na abordagem de BARRETO (2011), as variantes são sempre mutuamente excludentes entre si. Na abordagem *OdysseyProcess-FEX*, variantes podem ser alternativas complementares e um ponto de variação pode ser configurado por mais de uma variante em uma mesma definição. Assim como neste contexto, uma variante pode ser classificada como mandatória, classificação não viável na abordagem de BARRETO (2011).

No modelo de componentes usando a notação *OdysseyProcess-FEX*, a atividade *Planejar Medição* (modelo de características) é mapeada como um componente *Planejar Medição*, classificado com mandatório, ponto de variação composto e abstrato, com dois componentes internos classificados como opcionais, variantes, simples e concretos (Figura 4.9).

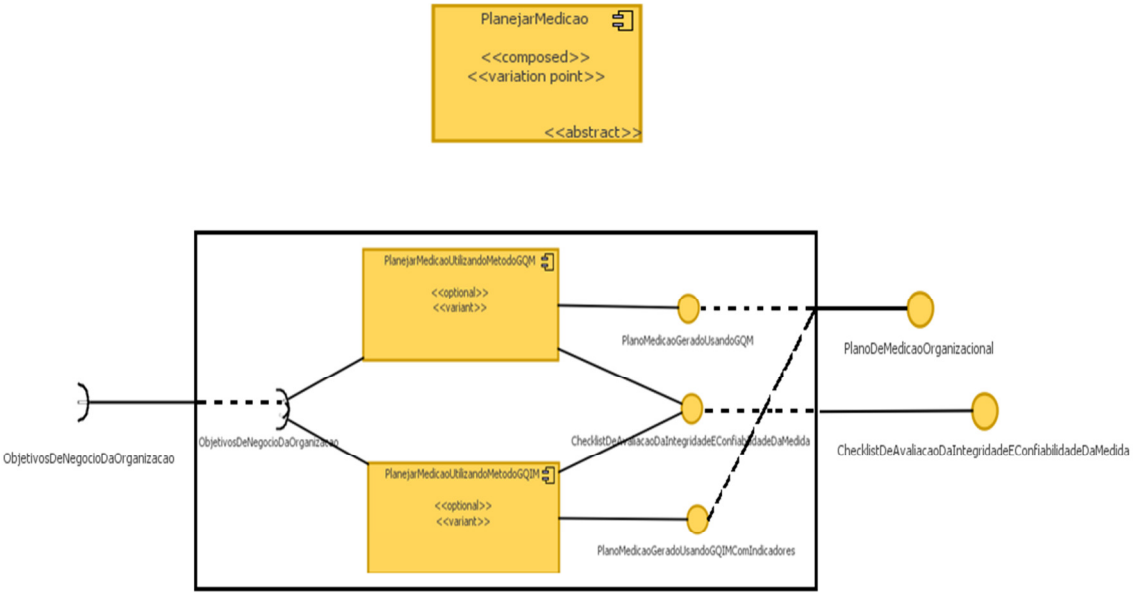


Figura 4.9 – Modelo de Componentes - Atividade *Planejar Medição* e variantes

Os demais elementos que constituem a descrição de um processo são papel, produto de trabalho e ferramenta. Tais elementos só apresentam classificação de opcionalidade e variabilidade na notação *OdysseyProcess-FEX*. Além disso, não são modelados graficamente na abordagem de BARRETO (2011), sendo apresentados apenas na descrição tabular do componente. O mapeamento destes elementos nas duas notações pode ser visto na Tabela 4.5.

Tabela 4.5 – Modelagem Elementos de Processo do Domínio (Papel, produto de trabalho, ferramenta)

OdysseyProcessReuse	**BARRETO (2011)**
Papel: representa um conjunto de habilidades, competências e responsabilidades de um indivíduo ou um conjunto de indivíduos. Não especifica um indivíduo ou recurso em particular	
Modelo de Características: Papel Modelo de Componentes: Papel como elemento interno de Componente de Processo A descrição do tipo de responsabilidade do papel com relação à participação na execução de uma unidade de trabalho é estabelecida pela relação entre os elementos: *<<performer>>, <<responsible>>, <<additional>>, <<assistant>>, <<supervisor>>, <<inspector>>, <<informed>>, <<consulted>>*	Papel na descrição tabular do componente, campo: Responsável e Participantes
Propriedade Opcionalidade do Elemento Papel	
Todo papel pode ser classificado como ***opcional*** ou ***mandatório*** tanto no modelo de características como no de componentes	Sem correspondência
Propriedade Variabilidade do Elemento Papel	
Todo papel pode ser classificado como ***ponto de variação***, ***variante*** ou ***invariante*** tanto no modelo de características como no de componentes	Sem correspondência
OdysseyProcessReuse	**BARRETO (2011)**
Produto de Trabalho: representa um artefato consumido, modificado ou produzido por uma unidade de trabalho	
Modelo de Características: Produto de Trabalho Modelo de Componentes: Produto de Trabalho como elemento interno ou associado a uma interface de Componente de Processo A descrição do tipo de participação do produto de trabalho na execução da unidade de trabalho associada é estabelecida pela relação entre os elementos: • Produto de Trabalho a ser consumido: ***<<in>>*** • Produto de Trabalho a ser produzido: ***<<out>>*** • Produto de Trabalho a ser modificado: ***<<inout>>***	Produto de Trabalho na descrição tabular do componente, campo: Artefatos requeridos e artefatos produzidos
Propriedade Opcionalidade do Elemento Produto de Trabalho	
Todo Produto de Trabalho pode ser classificado como ***opcional*** ou ***mandatório*** tanto no modelo de características como no de componentes	Sem correspondência

OdysseyProcessReuse	**BARRETO (2011)**
Propriedade Variabilidade do Elemento Produto de Trabalho	
Todo Produto de Trabalho pode ser classificado como ***ponto de variação***, ***variante*** ou ***invariante*** tanto no modelo de características como no de componentes	Sem correspondência
OdysseyProcessReuse	**BARRETO (2011)**
Ferramenta: descreve capacidades de uma unidade de automação para apoiar a exceção da unidade de trabalho associada.	
Modelo de Características: Ferramenta Modelo de Componentes: Ferramenta como elemento interno de Componente de Processo	Ferramenta na descrição tabular do componente, campo: Ferramentas de Apoio
Propriedade Opcionalidade do Elemento Ferramenta	
Toda Ferramenta pode ser classificada como ***opcional*** ou ***mandatório*** tanto no modelo de características como no de componentes	Sem correspondência
Propriedade Variabilidade do Elemento Ferramenta	
Toda Ferramenta pode ser classificada como ***ponto de variação***, ***variante*** ou ***invariante*** tanto no modelo de características como no de componentes	Sem correspondência

A classificação de opcionalidade e variabilidade associada a esses elementos consiste em mais um tipo de possibilidade de configuração do domínio, permitindo um maior grau de flexibilidade para a modelagem dos demais elementos de processo que também são passíveis de variação de uma definição de processo para outra.

Outro ponto é a visualização gráfica desses elementos com as unidades de trabalho correspondentes. Na abordagem proposta por BARRETO (2011), apenas componentes e atividades são modelados graficamente, todo o restante da descrição dos demais elementos associados a essas unidades de trabalho é feito de forma tabular. Pode ser interessante ter o recurso visual para identificar tais elementos e analisar as possibilidades de definição de novos processos.

Um exemplo para tratar esses elementos foi construído para o componente *Planejar medição utilizando o* método GQM, modelado utilizando as duas notações e encontra-se descrito a seguir (Figura 4.10).

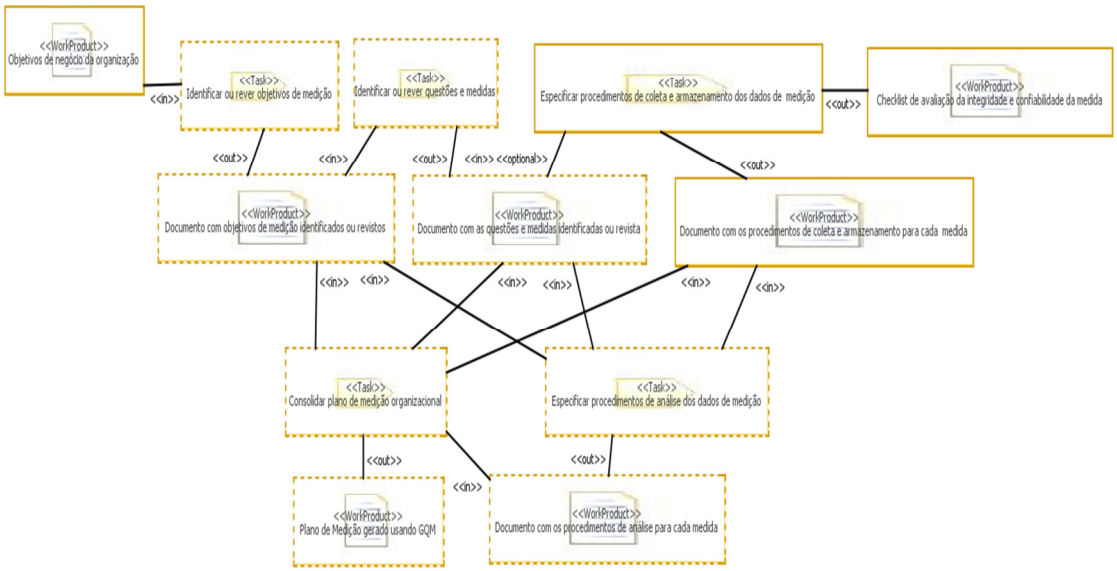
Exemplo

- **Abordagem BARRETO (2011)** – LPrS modelada em CARDOSO (2012)
  - Descrição tabular dos elementos constituintes do componente *Planejar medição utilizando o método GQM*

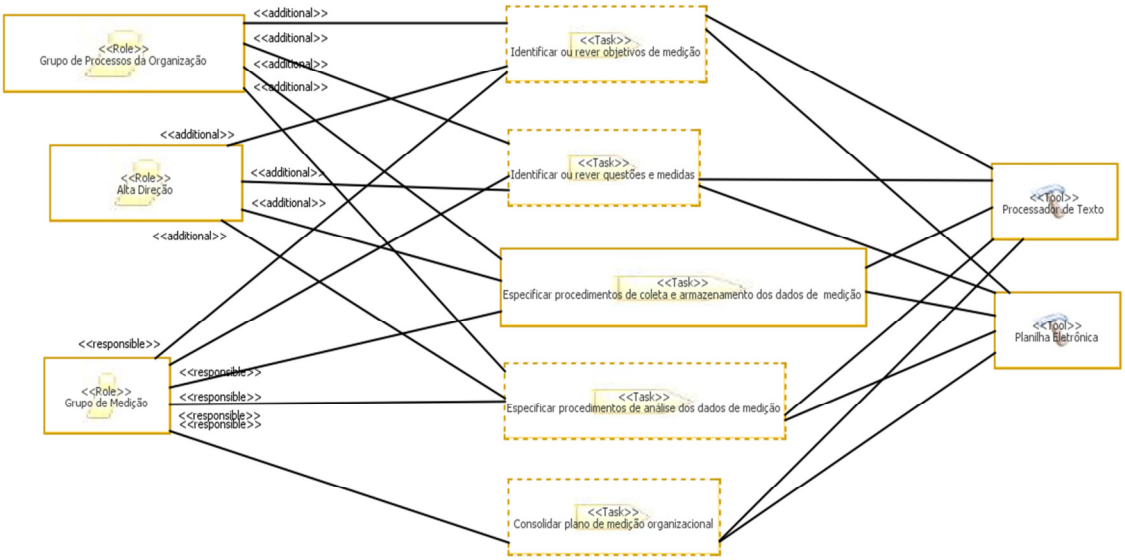
Planejar medição utilizando o método GQM	
Identificador:	COP.MED.ORG.CON.0059
Nome:	Planejar medição utilizando o método GQM
Tipo:	Concreto
Descrição:	Estabelecer plano de medição organizacional através da definição dos objetivos de medição oriundos das necessidades de informação organizacional utilizando o método GQM (*Goal Question Metric*). Esse plano deve conter quais medidas deverão ser coletadas, quais os procedimentos de coleta e armazenamento deverão ser adotados para cada medida, bem como quais os procedimentos de análise e divulgação dos resultados de medição.
Definido por:	COPPE/UFRJ
Critérios de Entrada:	Ter-se iniciado a implantação do processo de medição ou haver a necessidade de rever o plano de medição organizacional
Critérios de Saída:	Ter-se o plano de medição organizacional definido ou revisado.
Responsável:	Grupo de Medição
Participantes:	Alta Direção; Grupo de Processos da Organização.
Ferramentas de Apoio:	Processador de Textos; Planilha Eletrônica.
Artefatos Requeridos:	Objetivos de negócio da organização (necessidades de informação)
Artefatos Produzidos:	Documento com objetivos de medição identificados ou revistos; Documento com as questões e medidas identificadas ou revistas;
	Documento com os procedimentos de coleta e armazenamento para cada medida; Checklist de avaliação da integridade e confiabilidade da medida; Documento com os procedimentos de análise para cada medida; Plano de medição organizacional.
Características Atendidas:	MR-MPS - Nível F; Definir medidas utilizando o método GQM.
Medidas:	ESF_PLA_MED - Esforço para realizar o planejamento do programa de medição da organização
Variantes deste componente:	-

- **Abordagem *OdysseyProcess-FEX***

Visão Estrutural
Unidades de Trabalho e Produtos de Trabalho



Unidades de Trabalho, Papéis e Ferramentas



Visão Comportamental

Atividade *Planejar medição utilizando o método GQM* (Fluxo de Execução da Atividade) – Corresponde à Arquitetura Interna do Componente de Processo: *Planejar medição utilizando o método GQM*

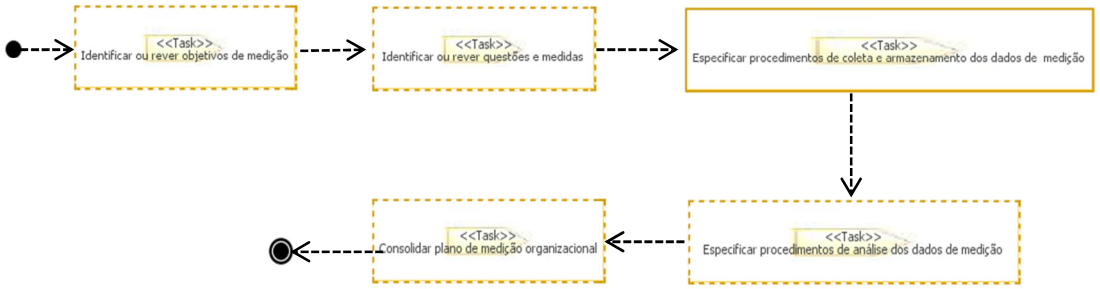


Figura 4.10 – Modelos da Atividade *Planejar medição utilizando o método GQM*

É importante notar que pela modelagem de características da *OdysseyProcess-FEX* é possível observar quais elementos de processos associados a uma atividade (unidade de trabalho) são opcionais ou variações no domínio. Por exemplo, os papéis associados à atividade *Planejar medição utilizando o método GQM* são mandatórios no domínio, pois aparecem relacionados a outras atividades do domínio classificadas como mandatórias. O produto de trabalho *Objetivos de negócio da organização* consumido como entrada, mesmo relacionado com esta atividade opcional, é mandatório no domínio, pois é requerido em outras atividades classificadas como mandatórias. No entanto, todos os produtos de trabalho produzidos nesta atividade são opcionais no domínio, pois dependem da seleção desta atividade na definição de

um processo específico para serem gerados. Este tipo de informação não é modelada seguindo a abordagem de BARRETO (2011).

4.2.3.2. Relacionamentos

Esta classe de análise envolve os relacionamentos presentes nas abordagens (Tabela 4.6). Os relacionamentos da notação *OdysseyProcess-FEX* não são apenas hierárquicos, permitindo a expansão do modelo em diversas direções. Inclui relacionamentos da UML, outros que possibilitam uma representação explícita da variabilidade no modelo, como por exemplo, o relacionamento Alternativo, que expressa a relação entre um ponto de variação e suas variantes, e outros que possibilitam uma representação explícita de relacionamentos próprios de processos, como ligações entre papéis e unidade de trabalho na modelagem estrutural ou fluxos de controle como sequências na modelagem comportamental. No nível de componentes ainda é possível estabelecer relações entre componentes e entre esses elementos e o meio pelas interfaces. Alguns dos relacionamentos apresentam o conceito de opcionalidade associado, indicando que a relação entre dois elementos do domínio não é necessariamente obrigatória, mesmo que ambos os elementos sejam obrigatórios no domínio. Por exemplo, o relacionamento entre o produto de trabalho de entrada (*Documento com as questões e medidas identificadas ou revistas*) da tarefa *Especificar procedimentos de coleta e armazenamento dos dados de medição* é relacionada de forma opcional a esta, pois tal tarefa também aparece na atividade Planejar Medição utilizando o método GQ(I)M, recebendo outro elemento como entrada (*Documento com as questões, indicadores e medidas identificadas ou revistas*), também associado de forma opcional a mesma tarefa.

Já na abordagem de BARRETO (2011), as relações são estabelecidas entre elementos como atividades e componentes por conexões de elementos de processo, que representam relacionamentos entre dois itens de uma arquitetura de processos. Cada conexão possui um item origem e um item destino, além de um tipo de conexão de elementos de processo (Simples; Fim-Início; Início-Início; Fim-Fim; Início-Fim). Uma conexão de elementos de processo também pode ser opcional, indicando que, no momento da definição de um processo baseado na arquitetura de processos, deverá ser tomada uma decisão sobre incluir ou não a conexão no processo resultante.

Tabela 4.6 – Modelagem Relacionamentos entre Elementos de Processo do Domínio

OdysseyProcessReuse	BARRETO (2011)
Alternativo Aplicados a todos os tipos de elementos de processo, respeitando restrições estabelecidas nas regras de boa formação do metamodelo. Por exemplo, variantes devem ser do mesmo tipo (categoria de elemento) do ponto de variação	Sem correspondência direta As variantes que compõem um componente abstrato apresentadas na descrição tabular do componente, campo: Variantes
Agregação Aplicados a todos os tipos de elementos de processo, respeitando restrições estabelecidas nas regras de boa formação do metamodelo.	Sem correspondência direta Relações estabelecidas na descrição tabular do componente
Composição Aplicados a todos os tipos de elementos de processo, respeitando restrições estabelecidas nas regras de boa formação do metamodelo.	Sem correspondência direta Relações estabelecidas na descrição tabular do componente
LigaçãoUnidadeDeTrabalhoPapel Propriedade que define a classificação do relacionamento quanto a sua opcionalidade	Relações estabelecidas na descrição tabular do componente sem caracterização de opcionalidade
LigaçãoUnidadeDeTrabalhoProdutoDeTrabalho Propriedade que define a classificação do relacionamento quanto a sua opcionalidade	Relações estabelecidas na descrição tabular do componente sem caracterização de opcionalidade
LigaçãoPapelProdutoDeTrabalho Propriedade que define a classificação do relacionamento quanto a sua opcionalidade	Sem correspondência direta
LigaçãoUnidadeDeTrabalhoFerramenta Propriedade que define a classificação do relacionamento quanto a sua opcionalidade	Relações estabelecidas na descrição tabular do componente sem caracterização de opcionalidade
Fluxo de Controle - Fluxo de Sequência Propriedade que define a classificação do relacionamento quanto a sua opcionalidade	Conexão de elementos de processo do tipo Fim-Início com caracterização de opcionalidade
Fluxo de Controle - Fluxo de Decisão Propriedade que define a classificação do relacionamento quanto a sua opcionalidade	Sem correspondência direta
Fluxo de Controle – Fluxo Paralelo Propriedade que define a classificação do relacionamento quanto a sua opcionalidade	Sem correspondência direta Conexão de elementos de processo do tipo Início-Início com caracterização de opcionalidade
Fluxo de Controle - Fluxo Síncrono Propriedade que define a classificação do relacionamento quanto a sua opcionalidade	Sem correspondência direta Conexão de elementos de processo do tipo Fim-Fim com caracterização de opcionalidade

OdysseyProcessReuse	**BARRETO (2011)**
Sem correspondência direta	Início-Fim – quando se iniciar a execução do elemento de processo origem, a execução do elemento de processo destino deve se encerrar.
Interface de Dados Corresponde à definição da relação de um produto de trabalho com unidade(s) de trabalho envolvida(s) na execução do componente que extrapola a fronteira do componente, ou seja, troca entre componentes ou do componente com o meio. Cada interface deve estar associada a um produto de trabalho que pode ser requerido (entrada do componente) ou provido (saída do componente)	Sem correspondência
Interface de Controle Fluxos de controle tendo um componente como origem e outro componente destino do fluxo. Entre eles um conector é utilizado para estabelecer uma regra de associação representada pelos tipos de fluxos de controle especificados no nível de análise: sequencial, paralelo, síncrono e de decisão.	Conexão de elementos de processo Não apresenta correspondência direta para o tipo "*decisão"*

A modelagem do fluxo de execução entre as tarefas da atividade *Planejar medição utilizando o método GQM* foi realizada por fluxos de controle sequenciais, classificados como opcionais dentro do domínio, pois conectam tarefas opcionais dentro do domínio (Figura 4.10). Esta modelagem difere da arquitetura interna do Componente de Processo Planejar medição utilizando o método GQM apresentado em CARDOSO (2012), onde os fluxos fim-início são modelados como mandatórios, tendo a visão do contexto do componente em si (Figura 4.11). O mesmo poderia ter sido feito na modelagem usando a notação *OdysseyProcess-FEX*. No entanto, seguindo a proposta da abordagem *OdysseyProcessReuse* que trabalha o conceito de opcionalidade com relação do domínio como um todo, preferiu-se modelar os fluxos como opcionais. Caso, existisse a necessidade de representação e variação de possibilidades de fluxos dentro da atividade, uma opção de modelagem sugerida pela abordagem *OdysseyProcessReuse* seria associar diferentes visões comportamentais à atividade em questão, cada uma representando um possibilidade de execução.

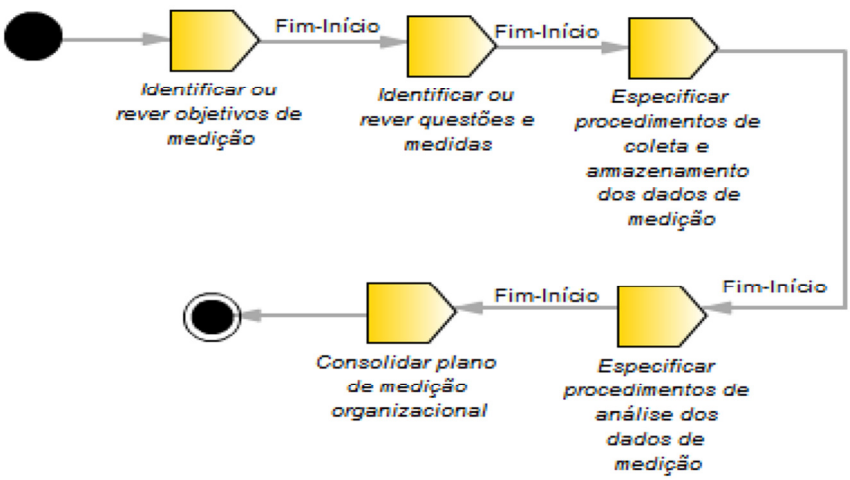


Figura 4.11 – Arquitetura Interna do Componente de Processo: *Planejar medição utilizando o método GQM* (CARDOSO, 2012)

4.2.3.3. Restrições de Dependência e Mútua Exclusividade

Outra classe sob análise é a de dependência e mútua exclusividade entre elementos de processo, que pode ser expressa por uma regra de composição inclusiva ou uma regra de composição exclusiva, respectivamente, na notação *OdysseyProcess-FEX*. Tais regras são descritas no modelo de características e mapeadas para o modelo de componentes, mantendo a consistência entre os níveis de abstração. A definição dessas restrições consiste em elementos do modelo combinados por expressões formadas por duas ou mais características conectadas por operadores booleanos (AND, OR, NOT, XOR), permitindo uma representação mais expressiva das relações que definem dependência e mútua exclusividade. A dependência também pode ser expressa pelo fluxo de execução das unidades de trabalho, indicando quais são predecessoras e sucessoras a unidade analisada.

Na abordagem proposta por BARRETO (2011), modelo proposto prevê dois tipos de relações: relação de dependência e de conflito, que são aplicados a características. Essa relação é estabelecida apenas entre duas características e não prevê a análise envolvendo todos os tipos de elementos de processos. A dependência também pode ser expressa pelo fluxo de execução das unidades de trabalho e da descrição dos critérios de entrada e saída de um componente.

A Tabela 4.7 apresenta algumas formas de trabalhar com regras de composição e as correlações entre as notações.

Tabela 4.7 – Modelagem Relações de Dependência e Mútua Exclusividade do Domínio

OdysseyProcessReuse	BARRETO (2011)
Relações de Dependência – Regras de Composição Inclusiva entre duas ou mais características. Podem ser estabelecidas relações entre diferentes tipos de elementos de processos (atividade, tarefa, papel, produto de trabalho e ferramenta). Mapeamento para o modelo de componentes como pré-condições.	Sem correspondência completa Relação de dependência entre duas características de alto nível não atingindo a granularidade dos diferentes tipos de elementos.
Relações de Mútua Exclusividade – Regras de Composição Exclusiva entre duas ou mais características. Podem ser estabelecidas relações entre diferentes tipos de elementos de processos (atividade, tarefa, papel, produto de trabalho e ferramenta). Mapeamento para o modelo de componentes como restrições.	Sem correspondência completa Relação de conflito entre duas características de alto nível não atingindo a granularidade dos diferentes tipos de elementos.
Dependência por critérios de entrada e saída de componentes de processo	Dependência por critérios de entrada e saída de componentes de processo
Dependência por fluxo de execução (unidades de trabalho predecessoras e sucessoras)	Dependência por fluxo de execução (unidades de trabalho predecessoras e sucessoras)

Um exemplo de aplicação de regras de composição utilizando a notação *OdysseyProcess-FEX* pode ser vista na Figura 4.12. Sendo as variantes mutuamente excludentes entre si, o que pode ser observado pela cardinalidade máxima do ponto de variação com o valor igual a 1, os produtos de trabalho também são excludentes entre si no domínio. Desta forma, uma regra de composição exclusiva foi definida e é representada graficamente pela marcação X nas características envolvidas (**X**: E-mail com resultado das análises de medição *excludes* (Apresentação com o resultado das análises de medição AND (Agendamento de reunião com membros de cada perfil de necessidade de informação AND Feedback dos participantes sobre os resultados apresentados registrado em ata de reunião)). Ao mesmo tempo, alguns elementos de processo são dependentes entre si, pois a medida que uma variante é escolhida todos os produtos de trabalho devem ser gerados conjuntamente. Tal dependência é descrita pela regra de composição inclusiva representada pela marcação R (**R:** Apresentação com o resultado das análises de medição *requires* (Agendamento de reunião com membros de cada perfil de necessidade de informação AND Feedback dos participantes sobre os resultados apresentados registrado em ata de reunião)).

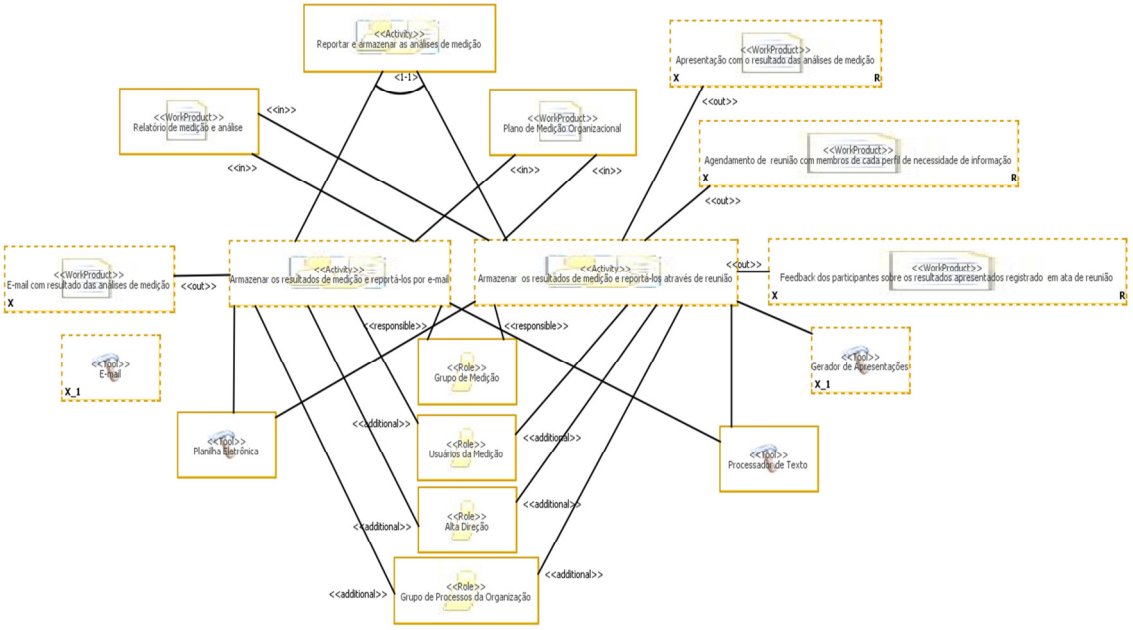


Figura 4.12 – Atividade *Reportar e armazenar as análises de medição* modelada usando a notação *OdysseyProcess-FEX*

Para finalizar a análise da modelagem entre as notações, o modelo de componente é gerado na metodologia *OdysseyProcessReuse* por um conjunto de heurísticas e analisado segundo um conjunto de critérios para apoiar possíveis agrupamentos (Figura 4.13).

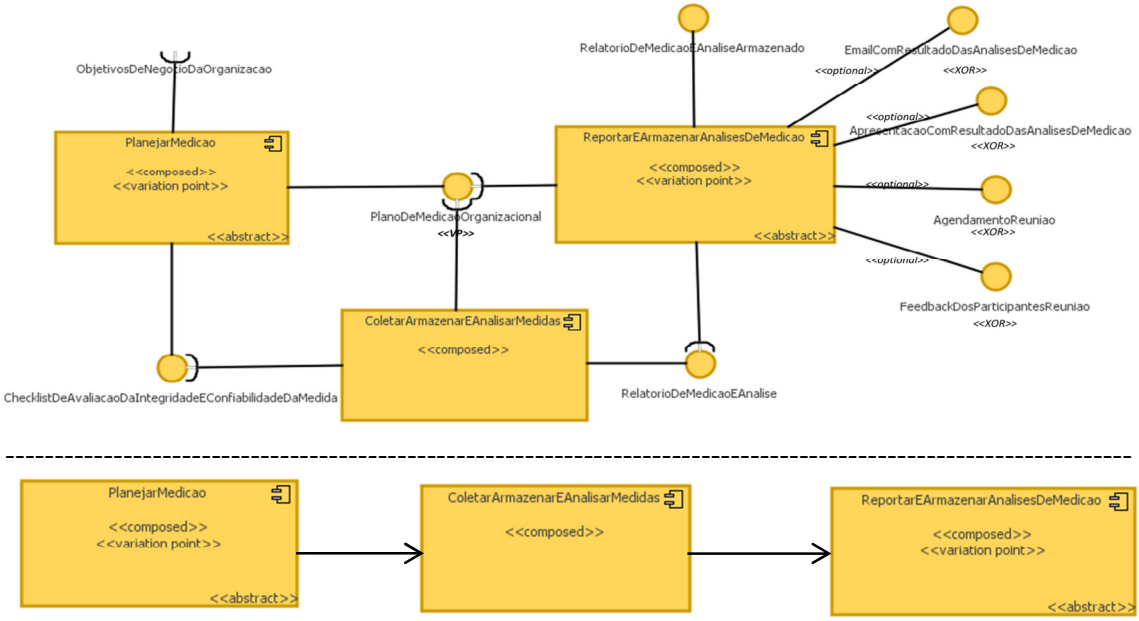


Figura 4.13 – Atividade *Reportar e armazenar as análises de medição* modelada usando a notação *OdysseyProcess-FEX*

Inicialmente, com as heurísticas de mapeamento, são obtidos os quatro componentes como na Figura 4.6. No entanto, após aplicar os critérios de

agrupamento, o agrupamento dos componentes de coleta e análise foi indicado, resultando em três componentes finais. A modelagem completa da LPrS Medição utilizando a notação *OdysseyProcess-FEX* pode ser vista no Apêndice IV.

Vale ressaltar que o objetivo deste estudo foi realizar uma análise comparativa apenas das representações de variabilidades propostas nas duas abordagens e não dos objetivos e resultados esperados da aplicação das abordagens no desenvolvimento de LPrSs.

Com relação à modelagem, podemos observar que a *OdysseyProcess-FEX* apresenta uma maior expressividade de representação dos diferentes artefatos que compõem uma LPrS. Por outro lado, essa expressividade e maior número de elementos de modelagem levam a uma maior complexidade de entendimento da notação, o que requer maior tempo para absorção do conhecimento, ainda que seja possível realizar a modelagem com um conjunto mínimo de elementos escolhidos pelo engenheiro do domínio de acordo com a proposta e objetivo do cenário de aplicação.

Outro ponto a ser analisado, é que uma estrutura de descrição tabular pode ser considerada como complementar à representação visual, com a definição de atributos e propriedades textuais com descrições mais longas e de difícil reprodução gráfica. Além disso, muitas organizações apresentam seus processos utilizando uma representação tabular. Isto implica em um esforço dos profissionais para repensar essa informação na forma gráfica. No entanto, o resultado pode gerar ganhos de identificação visual de aspectos antes incorporados em descrições textuais, como a replicação de informação e os pontos de configuração e adaptação.

4.3. HEURÍSTICAS DE MAPEAMENTO

O mapeamento entre os dois níveis de abstração, ainda que considerado como transformações de elementos 1:1, fornecem visões modelares do domínio e precisam observar as diferentes propriedades e classificações para a manutenção da consistência entre os modelos. Um conjunto de heurísticas foi proposto para auxiliar nesse mapeamento de artefatos, suas relações, regras de composição e classificações de opcionalidades e variabilidades do nível de características para o nível de componentes de uma LPrS, seguindo as propriedades do metamodelo proposto e garantindo a consistência entre as etapas de análise e projeto do domínio. Tais

heurísticas foram derivadas baseadas em BLOIS (2006), aplicando analogias entre software e processo e realizando adaptações nos conceitos da metodologia de Engenharia de Domínio baseada em componentes de software. A taxonomia e relacionamentos, propriedades e regras de boa formação do metamodelo *OdysseyProcess-FEX* para a modelagem de características e para modelagem de componentes foram considerados para definição das heurísticas entre os dois níveis de abstração. As heurísticas são apresentadas na Tabela 4.8, descrevendo o respectivo mapeamento que apoia as transformações: (a) características de unidades de trabalhos em componentes ou elementos internos de um componente (AH, AH1 e AH2 – *Activity heuristics* e TH – *Tasks heuristic*); (b) relações entre produtos de trabalho e unidades de trabalho em interfaces de dados ou relações entre elementos internos de um componente (WPH, WPH1, WPH2 – Work Product *heuristics*); (c) papéis em elementos internos de um ou mais componentes (RH – *Role heuristic*); (d) ferramentas em elementos internos de um ou mais componentes (TlH- *Tool heuristic*); (e) fluxos de controle em interfaces de controle entre componentes (CFH – *Control Flow heuristic*); e (f) regras de composição em restrições ou pré-condições (IRH – *Inclusive rule heuristic* e ERH – *Exclusive rule heuristic*).

Tabela 4.8- Heurísticas de Mapeamento de Modelo de Características para Modelo de Componentes de uma LPrS

ID	Elemento do Modelo de Características	Elemento do Modelo de Componentes	Heurísticas
Atividade			
AH	Atividade	Componente de Processo de Software	Cada atividade deve ser mapeada para um componente de processo de software com a mesma classificação de opcionalidade e variabilidade do elemento correspondente no nível de características da LPrS.
AH1	Atividade	Componente de Processo de Software	Cada componente de processo derivado será classificado como **composto** quando for internamente estruturado por outros componentes. Caso contrário, será classificado como **simples**. A seguir, são descritos dois casos de componentes de processo classificado como compostos após o mapeamento de atividades: • AH1.1. Atividades relacionadas por relacionamento alternativo devem ser agrupadas em um componente composto, que representa a atividade classificada como ponto de variação, com sua configuração interna definida por componentes que representam cada uma das atividades que são as variantes relacionadas.
	Classificação da estrutura interna do componente de processo de software		

ID	Elemento do Modelo de Características	Elemento do Modelo de Componentes	Heurísticas
			• AH1.2. Agregação e Composição são tipos de associação que especificam uma relação todo/parte. Atividades relacionadas por composição ou agregação devem ser agrupadas em um componente composto, que representa a atividade classificada como o todo em uma associação, com sua configuração interna definida por componentes que representam cada uma das atividades que são as partes da associação.
AH2	Atividade	Componente de Processo de Software	Cada componente de processo derivado será classificado como **concreto** quando for não existir nenhuma variação interna a ser resolvida (todos os elementos de processo que compõem o componente estão definidos da maneira que serão instanciados, sem nenhuma decisão adicional de definição). Caso contrário será classificado como **abstrato.** A seguir, são descritos casos de componentes de processo classificado como abstratos após o mapeamento de atividades: • AH2.1. Componente de processo classificado como ponto de variação. • AH2.2. Componente com um dos seus elementos internos é classificado como ponto de variação, variante, ou opcional. • AH2.3. Relações classificadas como opcionais entre elementos de processos, como relações entre papéis/ferramentas e produtos de trabalho/unidades de trabalho; ou unidades de trabalho relacionadas por agregação.
	Classificação de variação interna do componente de processo de software		
Tarefa			
TH	Tarefa	Elemento interno de um componente de processo de software	Tarefas representam partes de uma atividade e devem ser mapeadas como elementos internos de um componente que apresenta a atividade. Cada tarefa deve manter a mesma classificação de opcionalidade e variabilidade do elemento correspondente no nível de características da LPrS. • TH1.1. Tarefas relacionadas por relacionamento alternativo devem ser agrupadas em um mesmo componente.
Relações entre Produto de Trabalho e Unidades de Trabalho – produtos consumidos ou produzidos (<<in>> /<<out>>)			
WPH	Relação entre Produto de Trabalho e Unidade de Trabalho	Interface de dados de um componente de processo de software ou relação entre elementos internos de um componente de processo de software.	**WPH1**. Quando um produto de trabalho é trocado apenas entre unidades de trabalho internas a um componente, não ultrapassando sua fronteira, este deve ser agrupado ao componente como seu elemento interno mantendo a classificação de opcionalidade e variabilidade do elemento correspondente no nível de características da LPrS. Assim, como as relações envolvendo o determinado produto e as respectivas

ID	Elemento do Modelo de Características	Elemento do Modelo de Componentes	Heurísticas
			unidades de trabalho devem manter a classificação de opcionalidade dos relacionamentos do modelo de característica. **WPH2**. Quando um produto de trabalho é trocado além da fronteira de um componente, ou seja, entre componentes ou do componente com o meio, tal relação deve ser representada com uma interface de dados. Cada interface deve possuir o produto de trabalho associado com a mesma classificação de opcionalidade e variabilidade do elemento correspondente no nível de características da LPrS. A interface deve manter a classificação de opcionalidade do relacionamento entre o produto de trabalho e a unidade de trabalho correspondente no modelo de característica, a não ser que represente uma delegação entre interfaces de componentes compostos (olhar heurística WHP2.3). • WHP2.1. Cada relação em que o produto de trabalho representa uma entrada ao componente deve ser mapeada como uma **interface de dados requerida** do componente de processo. • WHP2.2. Cada relação em que o produto de trabalho representa uma saída do componente deve ser mapeada como uma **interface de dados provida** do componente de processo. • WHP2.3. Cada interface de dados de um componente interno a um componente composto deve ser mapeada como uma interface de dados do componente composto e tais interfaces devem estar conectadas por uma relação de delegação. Neste caso, a classificação de opcionalidade da interface deve ser de acordo com sua obrigatoriedade ou não com relação ao domínio.
Papel			
RH	Papel	Elemento interno de um componente de processo de software	Cada papel deve ser definido como elemento interno de um ou mais componentes de processo, seguindo a mesma classificação de opcionalidade e variabilidade do elemento correspondente no nível de características da LPrS. Este papel vai estar inserido em um componente que contém ou representa outro elemento de processo ao qual o papel se relaciona (unidade de trabalho ou produto de trabalho). As relações envolvendo um determinado papel devem ser descritas dentro do(s) componente(s) relacionado(s), mantendo a classificação de opcionalidade do(s) relacionamento(s) no modelo de característica.

ID	Elemento do Modelo de Características	Elemento do Modelo de Componentes	Heurísticas
Ferramenta			
TIH	Ferramenta	Elemento interno de um componente de processo de software	Cada ferramenta deve ser definida como um elemento interno de um ou mais componentes de processo, seguindo a mesma classificação de opcionalidade e variabilidade do elemento correspondente no nível de características da LPrS. Esta ferramenta vai estar inserida em um componente que contém ou representa outro elemento de processo ao qual a ferramenta se relaciona (unidade de trabalho). As relações envolvendo uma determinada ferramenta devem ser descritas dentro do(s) componente(s) relacionado(s), mantendo a classificação de opcionalidade do(s) relacionamento(s) no modelo de característica.
Fluxo de Controle			
CFH	Fluxo de Controle	Interface de controle de um componente de processo de software	Cada fluxo de controle estabelecido entre unidades de trabalho mapeadas em diferentes componentes devem ser descritos como interfaces de controle. Uma porta deverá ser definida como entrada para fluxo de controle e outra porta deverá ser definida como saída para o fluxo de controle. Um conector deve ser estabelecido para descrever o tipo do fluxo entre os componentes: sequencial; paralelismo; síncrono; junção e decisão.
Regras de Composição			
IRH	Regra de Composição Inclusiva	Pré-condições de um componente de processo	Cada componente envolvido em uma regra de composição inclusiva como antecedente deve possuir uma pré-condição especificada indicando a dependência com os componentes que compõem o consequente da regra .
ERH	Regra de Composição Exclusiva	Restrições de um componente de processo	Para cada componente envolvido em uma regra de composição exclusiva, uma restrição deve ser definida indicando a relação de mútua exclusividade estabelecida na regra. Cada componente deve ser marcado com um estereótipo <<*XOR*>> no diagrama.

4.3.1. Exemplo de Aplicação das Heurísticas de Mapeamento do Modelo de Características para o Modelo de Componentes de Processo de Software

Um exemplo de aplicação das heurísticas foi realizado no domínio de Planejamento de Projeto de Software. O mapeamento completo deste exemplo encontra-se descrito no Apêndice II. Partindo do modelo de características da Figura 4.2, um modelo de componentes da Figura 4.5 foi alcançado com sete componentes sendo derivados aplicando as heurísticas AH (Tabela 4.9). Dentre os sete, quatro componentes (C3, C4, C6 e C7) são elementos internos de outros componentes compostos (C2 e C5) (Tabela 4.9). Aplicando a heurística TH, todas as tarefas foram

mapeadas como elementos internos dos componentes, mantendo a mesma classificação de variabilidade e opcionalidade do modelo de característica. Por exemplo, a tarefa "*Determine and apply adjustment factor*" (ponto de variação e opcional) e suas variantes (opcionais) foram mapeadas como elementos internos do componente C3 (variante e opcional). O gerente de projeto, de posse da caracterização de um projeto específico e dos requisitos a serem alcançados por um processo, possui o perfil para atuar como responsável pela EAPS. Este pode contar com o apoio da equipe de processos e desenvolvimento de software na execução das atividades de definição do processo. O nível de experiência deste profissional pode impactar fortemente nas decisões de definição de um processo, exigindo profissionais mais capacitados e com maior conhecimento adquirido. No entanto, durante a EDPS, o conhecimento a respeito de projetos anteriores como lições aprendidas, boas práticas, dados de utilização, cenários de sucesso e caracterização de desempenho podem ser explicitados por meio dos artefatos e informações complementares representadas por uma LPrS. Desta forma, o uso de LPrS visa reduzir a influência e dependência de profissionais especializados ou altamente experientes na definição de novos processos.

Tabela 4.9 – Heurísticas aplicadas na LPrS de Planejamento de Projetos de Software

Modelo de características	**Heurísticas**	**Modelo de Componentes**	
Elemento		**Elementos e classificação**	
Develop Project Charter	AH	**C1**	Invariante / Mandatório / Simples / Concreto
Determine Project size	AH / AH 1.1 / AH 2.1	**C2**	Ponto de variação / Mandatório / Composto / Abstrato
Determine Project Effort		**C5**	Ponto de variação / Mandatório / Composto / Abstrato
Determine by Indicative Function Points (FP)	AH e AH 2.2	**C3**	Variante / Opcional / Simples / Abstrato
Determine by projects' historical base	AH	**C4**	Variante / Opcional / Simples / Concreto
Determine effort using size estimated by FP	AH	**C6**	Variante / Opcional / Simples / Concreto
Determine effort using size estimated by historical base	AH	**C7**	Variante / Opcional / Simples / Concreto

Cada relação de produto de trabalho e unidade de trabalho que extrapola as fronteiras do componente de processo deve ser mapeada como uma interface de dados (WPH2). Como exemplo, o produto de trabalho "*Project Charter*" possui duas

relações no domínio: (1) relação de saída da atividade "*Develop Project Charter*", mapeada como interface provida do componente C1 classificada como mandatória (WHP2.2); e (2) relações de entrada das atividades "*Determine Project Size*" e "*Determine Project Effort*", mapeadas como interface requeridas dos componentes C2, C3, C4, C5, C6 e C7, todas classificadas como mandatórias (WHP2.1).

A parte comportamental dos processos que corresponde aos fluxos de controle que deve ser mapeada usando a heurística CFH. Tal heurística estabelece que cada fluxo de controle estabelecido entre unidades de trabalho mapeadas em diferentes componentes devem ser descritos como interfaces de controle. Um conector deve ser estabelecido descrevendo cada fluxo sequencial entre os componentes. O resultado deste mapeamento pode ser visto na Figura 4.14.

Figura 4.14 – Visão Comportamental do Modelo de Componentes da LPrS de Planejamento de Projetos de Software

Da forma como o mapeamento foi especificado é possível analisá-lo de forma bidirecional. Assim, apesar do mapeamento descrever a transformação de um modelo de característica par um modelo de componentes, é possível realizar a transformação inversa com uma aplicação similar das heurísticas. Por exemplo, ao se criar um componente de processo no nível de projeto, uma atividade deve ser incluída no modelo de características, com as mesmas classificações de variabilidade e opcionalidade do componente. Os elementos internos de um componente são características relacionadas à atividade correspondente ao componente, respeitando o tipo do elemento e suas classificações de variabilidade e opcionalidade. Caso a participação do elemento interno seja opcional na derivação do componente, a ligação deste com a atividade deve ser classificada como opcional. Neste caso, tarefas são associadas com o relacionamento agregação. As ligações de papéis, produtos de trabalho ou ferramentas com a atividade passam a ter a propriedade opcional especificada como verdadeira. Interfaces de dado serão mapeadas como ligações entre a atividade e produtos de trabalho correspondentes, seguindo a classificação de

opcionalidade atribuída. Interfaces de controle representam fluxos de controle entre atividades.

Os mapeamentos definidos podem ser automatizados como forma de auxiliar o engenheiro de domínio na modelagem usando os dois níveis de abstração. Todas as heurísticas podem ser automatizadas para a transformação entre modelos. Algumas apresentam uma complexidade computacional um pouco mais trabalhosa como a verificação de produtos de trabalho que são apenas trocados internamente em um componente e aqueles que são considerados como interfaces. Outro exemplo é a classificação quanto à variação interna do componente a ser gerado, que envolve a análise de todos os elementos e relações dentro do componente e com suas interfaces. No entanto, ainda que custosas, todas as heurísticas podem ser providas com um suporte automatizado. Informações complementares do nível de projeto , como medidas, *baselines*, *templates*, agentes e *scripts*, devem ser especificadas pelo engenheiro do domínio após o mapeamento.

4.3.2. Estudo de Viabilidade das Heurísticas de Mapeamento

Neste estudo de caso, a viabilidade de aplicação de um conjunto de heurísticas para apoiar o mapeamento dos elementos de processo no nível de modelo de características para componentes de processos de software, considerando a manutenção das propriedades de variabilidade, opcionalidade e restrições inerentes ao modelo de características de LPrS, foi observada pela utilização das heurísticas no mapeamento de um domínio de processos de software em ambiente acadêmico. Os estudos iniciais foram realizados em ambiente acadêmico pelo fato de que a inserção de uma tecnologia em um ambiente industrial envolve altos custos e riscos (ex: disponibilidade e horas de trabalho dos profissionais, impacto nos projetos em andamento etc.) que devem ser minimizados para que ela possa ser aplicada com sucesso (BARCELOS, 2006). Assim, com um primeiro estudo de viabilidade, espera-se avaliar a aplicabilidade do conjunto de heurísticas em ambiente acadêmico antes de inseri-lo em um contexto industrial.

4.3.2.1. Planejamento do Estudo

- **Objetivo Geral**: Este estudo visa compreender a viabilidade de aplicação de heurísticas para apoiar o mapeamento dos elementos de processo do nível de

modelo de características para componentes de processos de software em modelos de representação de variabilidades de processos de uma LPrS.

Baseado em (BASILI *et al*., 1994), o objetivo do estudo foi definido como:

Analisar um conjunto de heurísticas para apoiar a geração de um modelo de componentes de uma LPrS a partir de um modelo de características

Com o propósito de caracterizar

Com respeito à efetividade e eficiência em gerar modelos de componentes de LPrS

Do ponto de vista de pesquisadores de Engenharia de Software

No contexto de desenvolvedores de software, aplicando as heurísticas em um domínio de processos de software

- **Questões:**
  - **Questão (1):** *Qual a efetividade de cada participante?*
  - **Métrica:**
    - **Efetividade =** número de elementos corretos do modelo de componentes derivado/ total de elementos corretos no modelo original
  - **Questão (2):** *Qual a eficiência de cada participante?*
  - **Métrica:**
    - **Eficiência =** número de elementos corretos do modelo de componentes derivado/ tempo dedicado à atividade de mapeamento.

A corretude do modelo derivado por cada participante é analisada de acordo com um modelo de componentes destino, que foi especificado antes da aplicação do estudo pelo pesquisador desta tese (modelo de componentes apresentado na Seção 4.2). Um total de 50 elementos foi identificado neste modelo original incluindo elementos, suas relações e propriedades.

- **Hipótese:**
  - **Hipótese Nula (H0):** Não existe equivalência entre a efetividade na derivação do modelo de componentes realizada pelos participantes mais experientes do que os menos experientes.

- **Hipótese Alternativa (H1):** Participantes mais experientes possuem maior efetividade do que os menos experientes.

- **Seleção do contexto:** O estudo é executado em ambiente acadêmico por estudantes de pós-graduação de um curso de reutilização de software**.** Para a execução do estudo, parte do domínio de Planejamento de Projetos de Desenvolvimento de Software, com foco nas atividades iniciais deste processo e suas variações, foi modelado utilizando a notação *OdysseyProcess-FEX* nível de análise (modelo de características apresentado na Seção 4.2).
- **Seleção dos participantes:** Foram selecionados cinco participantes cujo perfil foi analisado segundo alguns critérios: (A) experiência acadêmica; (B) experiência com processos de software (implantando, definindo, gerenciando ou revisando); (C) experiência com reutilização de software (aplicando de maneira *ad-hoc*, usando ou definindo componentes, LPS, ou outros artefatos reutilizáveis, aplicando modelagem de características, ou sendo responsável pela elaboração de ativos reutilizáveis ou pela definição da forma de se reutilizar software em alguns projetos); (D) experiência com reutilização de processos de software, aplicando de maneira *ad-hoc*, usando ou definindo componentes e/ou LPrS, ou sendo responsável pela elaboração de ativos de processo reutilizáveis ou pela definição da forma de se reutilizar processos em alguns projetos); (E) experiência com o domínio. Os participantes foram selecionados em função da sua disponibilidade para participação no estudo.
- **Variáveis:**
  - **Variáveis Independentes:** domínio de processo de software representado pelo modelo de características usando a notação *OdysseyProcess-FEX*, experiência dos participantes em reutilização de processos de software, conhecimento prévio dos participantes do domínio usado no estudo, número de elementos definidos no modelo de componentes destino (componentes, elementos internos, interfaces).
  - **Variáveis Dependentes***:* número de elementos corretos definidos no modelo de componentes derivado por cada participante (componentes, elementos internos, interfaces), número de elementos derivados, mas

que não foram definidos no modelo de componentes destino (discrepâncias), tempo (medido em minutos), efetividade e eficiência.

- **Validade do estudo:** Algumas ameaças à validade do estudo foram identificadas como:
    - Número reduzido de participantes, sendo todos de um mesmo curso e com experiência limitada em reutilização de processos.
    - A utilização de apenas um domínio para aplicação das heurísticas pode impactar a validade externa. A aplicação em outros cenários e domínios deve ser considerada.
    - Da forma como o conhecimento foi transmitido, não se tem garantias do entendimento a contento das heurísticas observadas. Mesmo com a leitura introdutória e apresentação das heurísticas com exemplos de aplicação, esse fator pode ser ainda mais evidenciado pelos apontamentos feitos pelos participantes no formulário pós-estudo da necessidade de mais exemplos e informações adicionais.
    - A ausência de um método comparativo pode ser considerado outro fator, o ideal seria a separação dos participantes em dois grupos (um grupo utilizando um processo *ad-hoc* e outro utilizando as heurísticas disponíveis).
    - A aplicação apenas em ambiente acadêmico.
    - O tamanho da amostra impacta a validade de conclusão, onde o número reduzido de participantes não possibilita uma análise estatística, e limita os resultados que podem ser considerados apenas indícios e não generalizáveis.

4.3.2.2.Execução do Estudo

Preparação

O convite para participação do estudo foi enviado por e-mail com breve explicação do contexto e formulário de caracterização anexado. O preenchimento do formulário foi realizado anteriormente ao dia de aplicação do estudo para análise do perfil de cada participante. Após confirmação de participação, um documento contendo uma leitura introdutória sobre o metamodelo e notação *OdysseyProcess-FEX*

para os dois níveis de abstração (análise e projeto, modelo de características e modelo de componentes) foi enviado aos participantes, também por e-mail, com o intuito de fornecer um conhecimento prévio sobre a modelagem utilizada no estudo.

No dia do estudo, um treinamento foi fornecido, contendo:

- Uma apresentação sobre o contexto de realização do estudo e sobre a modelagem utilizada (metamodelo e notação *OdysseyProcess-FEX* para os dois níveis de abstração) e uma explicação sobre o domínio em questão.
- Uma apresentação das heurísticas propostas, a serem aplicadas no estudo.

Operação

A cada participante foi entregue um conjunto de documentos, contendo o modelo de características do domínio de Planejamento de Projetos de Software e material de apoio à aplicação do estudo: documento com as heurísticas a serem aplicadas e formulário para o preenchimento do mapeamento a ser realizado. O Apêndice V apresenta a documentação entregue aos participantes.

Os participantes analisaram o modelo de características entregue e realizaram o mapeamento para o modelo de componentes utilizando as heurísticas fornecidas. Ao final da atividade, os participantes preencheram um formulário de avaliação pós-estudo contendo itens objetivos (e.g., sim, não, parcialmente) e um item subjetivo (e.g., justificativa das respostas). O resultado do mapeamento e das respostas do formulário de avaliação é que efetivamente foram considerados para avaliar a viabilidade das heurísticas.

4.3.2.3. Análise dos resultados obtidos

Inicialmente, o perfil de cada participante foi analisado segundo os critérios estabelecidos. Os dois participantes com titulação de mestrado apresentaram os maiores níveis de experiência de forma geral, considerando todos os critérios (P1 e P2). Apenas três participantes (P1, P3 e P5) possuíam experiência na indústria: 11; 1 e 2.5 anos, respectivamente. Todos os participantes, ainda que não sejam especialistas, já aplicaram o processo de Gerenciamento de Projetos de Software. O nível final de experiência foi calculado como a soma dos valores de cada critério, ponderada pelo

número de critérios estabelecido, tendo todos os critérios o mesmo peso estabelecido (Tabela 4.10).

Tabela 4.10 – Perfil dos Participantes

Critério / Participante	Acadêm.	Exp. em Processos	Exp. em reutiliz. de software	Exp. em reutiliz. de processos de software	Exp. em Ger. de Projetos	Nível de Experiência
P1	0,50	0,88	0,64	0,33	0,50	**0,57**
P2	0,50	0,44	0,21	0,08	0,50	**0,35**
P3	0,00	0,38	0,18	0,21	0,50	**0,25**
P4	0,00	0,25	0,25	0,00	0,50	**0,20**
P5	0,25	0,06	0,14	0,00	0,50	**0,19**

Cada item reportado por participante foi analisado segundo sua categoria (componente, elemento interno ou interface), suas classificações e relações no modelo derivado. Desta forma, um componente foi avaliado quanto às suas quatro classificações (variabilidade, opcionalidade, estrutura e variação interna). Além disso, foi analisado se cada elemento possuía os elementos internos correspondentes com as respectivas classificações corretamente atribuídas. No caso das interfaces de dados, foi observado se a relação produto de trabalho e componente foi apropriadamente mapeada com classificação da interface como requerida ou provida, e se a classificação da opcionalidade estava correta (interface mandatória ou opcional). Foi observado se as regras de composição inclusivas foram mapeadas como pré-condições e se a regra de composição exclusiva foi colocada como restrição nos componentes correspondentes. Os fluxos foram analisados, se mapeados como fluxos sequenciais entre os componentes. Discrepâncias na modelagem impactam negativamente na avaliação do mapeamento realizado. Por exemplo, para a atividade *Develop Project Charter* do modelo de características (ver modelo na Seção 4.2), era esperado o mapeamento apresentado na Figura 4.15. Para cada discrepância em um modelo derivado, era descontado 0,25 no valor do item modelado.

	Tipo do Elemento	Classificação	Elemento Interno		Interfade de Dados				
	Componente	Invariante, Mandatório, Concreto, Simples	R1	el. Interno + mandatório + invariante+ associado a C1	W1 (Software Demand)	i.dados + requerida + mandat. + C1			
C1 = 5	1	0,25+0,25+0,25+0,25=1	0,25+0,25+0,25+0,25=1		0,25+0,25+0,25+0,25=1				
					W2 (Project Charter)	i.dados + provida + mandat. + C1			
					0,25+0,25+0,25+0,25=1				
					2				
							Erros	Total C1	%
P1	1	1	1		1	1	0	**5**	1,00
P2	1	1	1		1	0	-0,5	**3,5**	0,70
P3	1	1	1		0,75	0,5	-1,25	**3**	0,60
P4	1	1	1		0	1	-0,5	**3,5**	0,70
P5	1	1	1		0	0	-1	**2**	0,40

Figura 4.15 – Exemplo de Análise de Mapeamento por Componente Esperado

A Tabela 4.11 apresenta uma sumarização dos resultados obtidos pelas análises. Como pode ser observado, nenhum dos participantes atingiu o valor máximo de efetividade (igual a 1), indicando que o modelo de componentes destino não foi alcançado por completo. No entanto, apesar de não conseguirem mapear corretamente todos os elementos do modelo destino, todos os participantes derivaram os sete componentes esperados, apenas apresentando inversões na organização da estrutura interna e nas classificações dos tipos de variação interna dos componentes. Os maiores problemas encontrados referem-se às relações com os produtos de trabalho e o mapeamento do fluxo de controle. Apenas um participante mapeou corretamente os principais fluxos de controle. A separação da análise da LPrS em duas visões (estrutural e comportamental) no modelo de características pode ter sido um dos motivos que levou a não modelagem do fluxo no nível de componentes. Regras de composição também só foram completamente mapeadas de forma correta para pré-condições e restrições por um participante.

Tabela 4.11 – Sumarização dos resultados de análise

Participante	Nível de Experiência	Tempo	#Elementos Corretos	Efetividade	Eficiência
P1	0,57	128	34	0,68	0,271
P2	0,35	157	28,75	0,575	0,183
P3	0,25	81	23	0,46	0,284
P4	0,20	97	27,5	0,55	0,284
P5	0,19	90	21,5	0,43	0,239

Não foi observada nenhuma diferença significativa nos valores de efetividade e eficiência entre participantes com mais e menos experiência, indicando que não há como refutar a hipótese nula estabelecida, mas não temos evidência suficiente para confirmá-la. No entanto, podemos fazer algumas análises adicionais com relação a esses valores (Figura 4.16):

- Houve uma queda, ainda que não significativa, no valor da efetividade junto à diminuição do nível de experiência do participante.
- Os valores de eficiência são muito próximos, mas observando os participantes mais experientes houve uma maior discrepância entre eles.

O participante com o segundo maior nível de experiência teve um valor de efetividade próximo aos demais e deve o menor valor de eficiência entre todos os participantes, podendo ser considerado um *outlier*.

- O tempo utilizado pelos participantes para realização da atividade demonstra a complexidade do mapeamento entre os níveis de abstração definidos

O número de fatores envolvidos na modelagem e os novos conceitos especificados pela abordagem como variabilidade e opcionalidade, assim como, a proposta de uma nova modelagem para processos são fatores que podem influenciar nessa atividade e impactam na complexidade do mapeamento. Isso pode ser notado pelas respostas obtidas com o formulário pós-estudo. A maioria dos participantes concorda, ou parcialmente concorda, que tanto a notação como o domínio são de fácil entendimento. No entanto, mesmo com a etapa de treinamento e a leitura introdutória, alguns participantes indicaram a necessidade de alguns exemplos como informação adicional para dar suporte à aplicação das heurísticas.

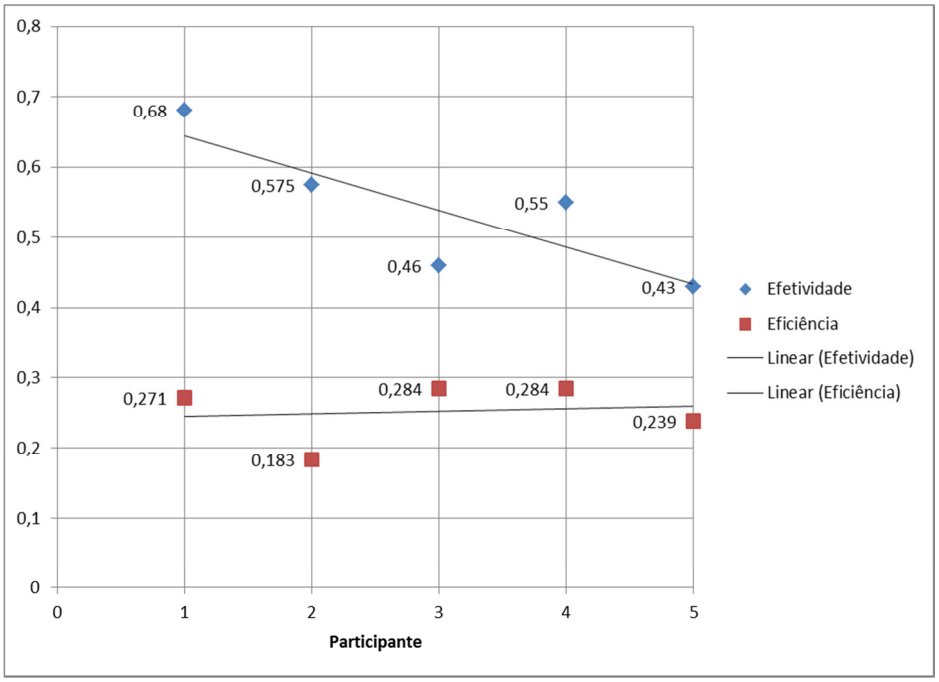


Figura 4.16 – Gráfico de Dispersão de Eficiência e Efetividade na aplicação das heurísticas de mapeamento

A Tabela 4.12 apresenta um resumo das respostas obtidas com relação à avaliação das heurísticas pelos participantes. Nenhuma sugestão foi indicada para

reorganizar a ordem de apresentação das heurísticas e todos os participantes concordam com o mapeamento de apoio à classificação de variabilidade e opcionalidade e ao agrupamento de elementos conectados pelos relacionamentos alternativo, agregação e composição no mesmo componente. Nenhuma redundância nas heurísticas foi apontada pelos participantes. Uma das maiores dificuldades apontadas pelos participantes consiste no mapeamento de produtos de trabalho e em suas relações com unidades de trabalho.

Tabela 4.12 – Sumarização dos resultados de avaliação das heurísticas pelos participantes

Avaliação Mapeamento							
Questões X Respostas	Questão 1	Questão 2	Questão 3	Questão 4	Questão 5	Questão 6	Questão 7
DT - Discordo Totalmente	0	0	4	0	3	0	0
DP - Discordo Parcialmente	0	0	0	0	1	0	0
I - Indiferente	0	1	0	1	0	0	0
CP - Concordo Parcialmente	3	2	1	3	1	0	1
CT - Concordo Totalmente	2	2	0	1	0	5	4
	Resposta esperada: CP ou CT = 5/5	Resposta esperada DT ou DP: = 0/5	Resposta esperada DT ou DP: = 4/5	Resposta esperada DT ou DP: = 0/5	Resposta esperada DT ou DP: = 4/5	Resposta esperada: CP ou CT = 5/5	Resposta esperada: CP ou CT = 5/5
	Resposta não esperada: DT ou DP = 0/5	Resposta não esperada: CP ou CT = 4/5	Resposta não esperada: CP ou CT = 1/5	Resposta não esperada: CP ou CT =4 /5	Resposta não esperada: CP ou CT = 1/5	Resposta não esperada: DT ou DP = 0/5	Resposta não esperada: DT ou DP = 0/5

Questão 1	1. A ordem de apresentação das heurísticas foi apropriada para aplicação. Ou seja, a ordem de aplicação das heurísticas de mapeamento foi a mesma da apresentação das heurísticas no documento. Em caso negativo, ou parcialmente negativo, em que ordem as heurísticas foram aplicadas e o que pode facilitar?
Questão 2	2. Alguma informação adicional poderia ser usada como suporte para aplicação das heurísticas. Em caso afirmativo ou parcialmente afirmativo, qual(is)?
Questão 3	3. Foram identificadas heurísticas redundantes ou similares. Em caso afirmativo, ou parcialmente afirmativo, indique quais heurísticas poderiam ser agrupadas ou quais heurísticas poderiam ser removidas
Questão 4	4. Em algum momento você sentiu dificuldade na aplicação de uma ou mais heurísticas. Em caso afirmativo ou parcialmente afirmativo, qual(is)? Justifique indicando a dificuldade enfrentada e uma possível sugestão de melhoria.
Questão 5	5. Em algum momento você desejou mapear uma característica para um determinado elemento no modelo de componentes e esta forma de mapeamento não era sugerida dentro das heurísticas disponibilizadas. Em caso afirmativo ou parcialmente afirmativo, qual(is)? Justifique indicando o mapeamento desejado.
Questão 6	6. As decisões sobre classificação de opcionalidade (mandatório e opcional) e classificação de variabilidade (invariante, ponto de variação e variante) sugeridas pelas heurísticas são adequadas. Em caso negativo, ou parcialmente negativo, aponte onde foi identificado o problema e qual heurística seria mais adequada.
Questão 7	7. As heurísticas de agrupamento de características de unidades de trabalho relacionadas por relacionamentos alternativos, composição ou agregação, em um componente de processo de software composto são adequadas. Em caso negativo, ou parcialmente negativo, aponte qual heurística seria mais adequada. Justifique.

A maioria dos participantes concorda que um suporte computacional poderia auxiliar na atividade de mapeamento, melhorando a aplicação das heurísticas, principalmente em casos de LPrS mais extensas e complexas. No entanto, não é descartada a necessidade de interação humana, mantendo o engenheiro de domínio como o responsável pela validação e aprovação da aplicação das heurísticas apropriadas. Pelas análises realizadas, o suporte ferramental pode auxiliar principalmente na completude de aplicação das heurísticas e garantia de mapeamento de todos os elementos do nível de análise para o modelo de componentes.

Com a realização do estudo foi possível ter evidências, ainda que não generalizáveis, sobre a viabilidade de aplicação das heurísticas como apoio ao mapeamento de um modelo de características para um modelo de componentes de LPrS. As observações e análises permitiram identificar pontos de melhoria nas heurísticas, que derivaram alterações na formulação original das heurísticas. Os

principais pontos alterados estão relacionados com as categorias de unidades de trabalho e suas classificações sendo agrupadas nas heurísticas AH (AH1 e AH2) e TH. Além disso, maiores detalhes sobre a classificação dos componentes em relação a sua estrutura e variação interna foram providos e colocados juntos às heurísticas de mapeamento das unidades de trabalho. As heurísticas sobre produtos de trabalho foram reescritas com explicações mais detalhadas sobre as diferenças entre os produtos de trabalho mapeados como elementos internos de um componente ou como interfaces, indicando o mapeamento da relação desses produtos com as unidades de trabalho e não o produto de trabalho em si. Essa nova versão das heurísticas está descrita na Seção 4.3.

Uma forma de melhoria dos resultados para alcançar maiores graus de efetividade e eficiência poderia ser obtida com apoio automatizado na aplicação das heurísticas. Tal apoio, que visa evitar a ocorrência de erros, garantindo a correta aplicação dos mapeamentos propostos, foi implementado e encontra-se descrito na Seção 6.4 do Capítulo 6.

4.4. CRITÉRIOS DE AGRUPAMENTO

É importante notar que, em geral, em um modelo de componentes gerado após o mapeamento do modelo de características temos um alto número de componentes resultantes. No entanto, quanto maior o número de componentes envolvidos e a troca de informações entre eles, mais complexa pode se tornar a arquitetura da LPrS, comprometendo, consequentemente, a sua compreensão, manutenibilidade e, sobretudo, a sua reutilização. Neste cenário, é possível detectar componentes mutuamente dependentes, ou seja, provendo e requerendo interfaces entre si. Além disso, existem componentes requerendo muitas interfaces de um número restrito de componentes ou componentes associados a um mesmo escopo do domínio. Desta forma, tais componentes podem ser ainda organizados em módulos que reflitam o entendimento da interação intensiva entre grupos de componentes e melhore a qualidade da arquitetura resultante. Tais agrupamentos devem levar em consideração componentes altamente relacionados, visando aumentar o entendimento do domínio, por meio do tratamento de aspectos de acoplamento, análise de escopos aos quais os

componentes pertencem dentro do domínio e de relações entre as fronteiras de um componente avaliando seu grau de responsabilidade e dependência.

As abordagens de DPBC e LPrS, em geral, não apresentam mecanismos para estruturar componentes nem tampouco propõem algum critério de agrupamento de componentes em particular. Desta forma, um conjunto inicial de seis critérios de agrupamento foi proposto para apoiar a identificação dessa possibilidade de organização de componentes em elementos arquiteturais de maior granularidade, reduzindo o número de componentes e o alto número de relações entre eles, tratando a complexidade da arquitetura da LPrS e melhorando a capacidade de reutilização dos elementos. Ao longo da apresentação de cada critério, um exemplo de aplicação será mostrado utilizando o modelo de componentes da LPrS de Planejamento de Projetos de Software (Figura 4.5)., derivado pós aplicação das heurísticas de mapeamento na Seção 4.3.

Cada critério analisa componentes sob pontos de vista diferentes, e podem ser combinados para apoiar a tomada de decisão de um agrupamento mais adequado de acordo com aspectos do domínio. Representações visuais dos critérios foram definidas para melhor representar visualmente os valores dos critérios e apoiar a análise das relações entre componentes via interfaces de dados baseados em três perspectivas: (1) Visão de Responsabilidades; (2) Visão de Dependências e (3) Visão de Instabilidades. A perspectiva de responsabilidades analisa as interações por provimento de interfaces e o número de componentes que são atendidos por tais interfaces. A perspectiva de dependências analisa as interações por interfaces requeridas e o número de componentes dependentes de tais interfaces. A perspectiva de instabilidades apresenta os valores do critério de instabilidade e nenhuma relação é representada, já que o critério representa um aspecto de combinação das perspectivas anteriores.

Em cada perspectiva, os componentes serão representados por caixas e os relacionamentos entre os componentes por linhas com setas indicando a direção dos fluxos de produtos. A espessura da linha varia de acordo com o número de relações estabelecidas entre os dois componentes conectados, indicando o grau da relação (dependência ou responsabilidade). Desta forma, linhas mais grossas indicam casos em que os critérios analisados na determinada perspectiva assumem valores mais altos. Já

as caixas devem ser preenchidas por cores em um intervalo do branco ao preto com variações de cinza. Assim, componentes com valores de critérios mais baixos, possuem caixas preenchidas com cores mais claras e, no caso de assumirem os valores mínimos no domínio, as caixas serão preenchidas com a cor branca. Componentes com valores de critérios mais altos possuem caixas preenchidas com cores mais escuras e, no caso de assumirem os valores máximos no domínio, as caixas serão preenchidas com a cor preta.

Filtros também podem ser aplicados para selecionar critérios a serem aplicados, tipos de componentes a serem avaliados (de acordo com as classificações de opcionalidade, variabilidade, estrutura interna e variação interna); número máximo de componentes a serem agrupados em um único componente; mostrar rede de dependências dentro de todo o domínio de um único componente a ser analisado, dentre outros.

Os critérios aqui propostos foram baseados em duas abordagens, BLOIS *et al*., (2005), BLOIS (2006) e BOLAÑOS e MARTÍNEZ (2013). A primeira metodologia trata de um Processo de Engenharia de Domínio baseado em Componentes de Software que propõe critérios para agrupar componentes gerados a partir de um modelo de características de um domínio de software. Aplicando analogias entre software e processo, foram adaptados alguns critérios para uso na análise do modelo de componentes de processos de software, no contexto de uma arquitetura, analisando o acoplamento em função da troca de artefatos de processos por meio de suas interfaces providas e requeridas. A segunda metodologia busca analisar visualmente modularidade em processos de software aplicando métricas de software para processos analisando por tipo de elemento de processo (tarefa, papel ou artefato). O critério de instabilidade descrito a seguir foi fortemente baseado nesta abordagem.

A aplicação dos critérios visa prover informações ao engenheiro do domínio para auxiliar na tomada de decisão de agrupamento ou não de componentes tendo como resultado esperado uma arquitetura de LPrS refinada que leva em consideração aspectos de acoplamento e coesão e aspectos inerentes à LPrS, como classificações de variabilidade, opcionalidade e rastros existentes entre artefatos e escopos do domínio. E ainda que proposto num contexto de LPrS, tais critérios podem ser utilizados em outras abordagens que trabalhem com componentes de processos de software.

4.4.1. Critério de Agrupamento por meio da Análise de Escopos do Domínio Definidos

Durante a análise do domínio (ADPS), características do domínio podem ser organizadas em diferentes escopos. Um escopo representa um conjunto de conceitos comuns dentro do domínio caracterizando subdomínios que podem auxiliar na fase de derivação de um processo a partir da LPrS na EAPS, como escopos caracterizando processos padrões ou específicos. Uma das possibilidades de agrupamento consiste em analisar as relações entre componentes que foram mapeados de características pertencentes a um mesmo escopo.

C1 - Agrupamento de componentes relacionados a um mesmo escopo: O agrupamento de componentes mapeados de características que pertencem a um mesmo escopo dentro do domínio é sugerido ou reforçado, baseado na suposição que esses componentes trocam mais informação entre si e possuem um maior número de relações entre eles, pois tratam de conceitos comuns e tópicos relacionados. É importante notar que a combinação com outros critérios pode melhorar a análise de acoplamento entre componentes organizados por escopos.

- **Exemplo**: Analisando a LPrS de Planejamento de Projetos de Software, todos os componentes pertencem ao mesmo escopo e são candidatos a agrupamentos. Neste caso, a aplicação de outros critérios podem suportar melhor a tomada de decisão de agrupar ou não os componentes.

4.4.2. Critérios de Agrupamento por meio da Análise de Componentes com relações entre si

Esse grupo de cinco critérios visa analisar o grau de responsabilidade e dependência de cada componente, sugerindo agrupamentos com o objetivo de minimizar o acoplamento e aumentar a coesão de componentes relacionados. A aplicação desses critérios deve considerar apenas interfaces que trocam informações entre componentes, denominadas interfaces internas. Estas interfaces produzem impactos no aspecto de acoplamento entre componentes. Interfaces de interação direta com o ambiente (denominadas interfaces externas), requerendo ou provendo artefatos diretamente do meio, não devem ser consideradas para cálculo dos valores de cada critério abaixo.

4.4.2.1. Análise pelo grau de responsabilidade dos componentes

Dois critérios (CPI e CRnC) podem ser utilizados para analisar o grau de responsabilidade de cada componente no domínio.

1. Uma das formas de responsabilidade de um componente no domínio consiste no provimento de artefatos que são insumos para atividades subsequentes no domínio. Este grau de responsabilidade pode ser avaliado pelo valor do critério que calcula o número de interfaces providas (CPI).

2. O grau de responsabilidade de um componente pode ser medido também por análise direta do número de componentes relacionados e não somente pelo número de interfaces trocadas. Desta forma, poderia ser importante observar que um componente pode possuir um CPI alto pelo número de interfaces providas, mas todas representam dependência de apenas um componente. Além disso, o critério que calcula o número de componentes que requerem serviços do componente analisado (CRnC) avalia tanto a dependência pelo provimento dos artefatos (dependência no fluxo de dados) como no caso de dependências entre componentes (unidades de trabalho) que representam a necessidade da execução de ações que podem não gerar nenhum recurso a ser consumido (dependência no fluxo de controle) como, por exemplo, atividades de verificações/validações que permitem a continuidade de execução do processo.

C2 - Análise de componentes relacionados por interfaces providas - *Component Provided Interfaces* (CPI): Análise do grau de responsabilidade de cada componente dentro do domínio pela análise de interfaces providas.

***Component Provided Interfaces* (CPI)** - Cálculo do número de interfaces providas por cada componente.

É importante lembrar que se devem considerar apenas interfaces trocadas entre componentes e descartar interfaces que produzem produtos de trabalho apenas para o ambiente (interfaces externas), já que as interfaces trocadas entre componentes são as que produzem impactos no aspecto de acoplamento.

Para guiar a análise deste critério, o valor médio de CPI no domínio pode ser considerado como um parâmetro. Desta forma, quando o valor CPI do componente for maior que o CPI médio, a possibilidade de agrupamento deste componente com todos

ou alguns dos componentes para os quais as interfaces são providas deve ser considerada.

Considerações para aplicar o agrupamento:

1. Todas as interfaces são providas para um único componente. Neste caso, o agrupamento é sugerido.
2. No caso do provimento de interfaces para mais de um componente:
    a. Considerar os componentes que também fornecem alguma interface para o componente analisado, indicando que há uma troca de serviços entre os componentes. Esses componentes devem ser priorizados para a realização de agrupamento.
    b. Considerar componentes que requerem um maior número de interfaces dentre o total das interfaces providas. Esses componentes devem ser priorizados para a realização de agrupamento, pois indicam um grau maior de dependência com o componente analisado.
    c. As interfaces são providas para um número de componentes que permite o agrupamento de todos os componentes envolvidos sem um aumento significativo na complexidade do componente resultante ou aumento do valor de algum critério de acoplamento. Essa análise depende de outros fatores como objetivos dos componentes; número de serviços providos por todos os componentes envolvidos; número de dependências com outros componentes do domínio; tipo de opcionalidade ou variabilidade no domínio; dentre outros.
    d. É necessário analisar se o componente possui um alto grau de responsabilidade que justifique o alto número de interfaces providas, fornecendo artefatos para um número razoável* de diferentes componentes no domínio. Neste caso, o agrupamento não é aconselhável.

*Este número pode ser avaliado de acordo com o critério que calcula o número de componentes que requerem um componente (CRnC). Se o valor for maior do que o CRnC médio do domínio pode indicar que o componente possui uma responsabilidade dispersa e o agrupamento não é aconselhável.

- **Exemplo**: Analisando a LPrS de Planejamento de Projetos de Software, o componente Determinar Tamanho do Projeto ("*Determine Project Size*") apresenta um valor de CPI igual a dois. Tal valor é superior ao valor médio de CPI, que é igual a um.

$$CPI_{DetermineProjectSizeComponent} = 2 > CPI_{Médio} = 1$$

Segundo o critério, este cenário sugere o agrupamento com outros componentes, possivelmente com o componente para Determinar o Esforço do Projeto ("*Determine Project Effort*").

C3 - Análise de componentes que requerem um componente - *Components Requiring a Component* (CRnC): Análise do grau de responsabilidade, indicando o número de componentes que requerem serviços do componente analisado, representando a responsabilidade deste componente na execução de determinadas ações com a produção ou não de artefatos.

***Components Requiring a Component* (CRnC)** - Cálculo do número de componentes requisitantes de um determinado componente.

Para guiar a análise deste critério o valor médio de CRnC no domínio pode ser considerado como um parâmetro. Desta forma, quando o valor CRnC do componente for maior que o CRnC médio a possibilidade de agrupamento deste componente com aqueles que o requerem.

Algumas considerações similares as do critério CPI podem ser utilizadas para aplicar ou não o agrupamento:

1. Apenas um componente requer serviços do componente analisado (CRnC = 1). Neste caso, o agrupamento é sugerido.
2. No caso dos serviços serem requeridos por mais de um componente (CRnC >1):
    a. Considerar os componentes que também fornecem algum serviço ao componente analisado, indicando que há uma troca entre eles. Esses componentes devem ser priorizados para a realização de agrupamento.
    b. Considerar componentes que requerem um maior número de serviços dentre o total provido. Esses componentes devem ser priorizados para a realização de agrupamento, pois indicam um grau maior de dependência com o componente analisado.

c. O número de componentes requerendo os serviços do componente analisado permite o agrupamento de todos os componentes envolvidos sem um aumento significativo na complexidade do componente resultante ou aumento do valor de algum dos critérios de acoplamento. Essa análise depende de outros fatores como objetivos dos componentes; número de serviços providos por todos os componentes envolvidos; número de dependências com outros componentes do domínio; tipo de opcionalidade ou variabilidade no domínio; dentre outros.

d. É necessário analisar se o componente possui um alto grau de responsabilidade que justifique o alto número de dependências, fornecendo serviços para um número alto de diferentes componentes no domínio (mais próximo do valor CRnC máximo no domínio). Neste caso, o agrupamento não é aconselhável.

- **Exemplo**: Analisando a LPrS de Planejamento de Projetos de Software, o componente Desenvolver Termo de Abertura ("*Develop Project Charter*") apresenta um valor de CRnC igual a dois. Tal valor é superior ao valor médio de CRnC, que é igual a um.

$$CRnC_{DevelopProjectCharterComponent} = 2 > CRnC_{Médio} = 1$$

No entanto, pode-se observar que a responsabilidade do componente é balanceada, estando igualmente distribuída entre os outros dois componentes no modelo. Neste caso, mais informações poderiam auxiliar na tomada de decisão de agrupamento, indicando outros aspectos de relação entre os componentes envolvidos.

Os dois critérios citados acima podem ser usados individualmente. No entanto, a análise individual, ainda que apresente alguma possibilidade de reestruturação do modelo, pode não prover informação suficiente para guiar o agrupamento mais adequado, que seria melhorar os valores dos critérios, minimizando o acoplamento. Para gerar melhores resultados o ideal é combinar os critérios e avalia-los em conjunto. Um alto número de interfaces providas por um componente (CPI > CPI médio) para um grupo reduzido de componentes (CRnC <= CRnC médio) pode indicar

uma reestruturação no domínio com o agrupamento de todos ou parte dos componentes envolvidos nesta relação de serviços providos, conforme descrito acima.

- **Exemplo:** Analisando a LPrS de Planejamento de Projetos de Software segundo a visão de responsabilidades na Figura 4.17 abaixo, a sugestão de reestruturação indica considerar:
  1. O agrupamento do componente Determinar Tamanho do Projeto ("*Determine Project Size*"), que está preenchido pela cor preta por apresentar um valor de CPI máximo no domínio ($CPI_{DetermineProjectSizeComponent} = 2$).
  2. Uma relação mais forte entre os componentes Determinar Tamanho do Projeto ("*Determine Project Size*") e Determinar o Esforço do Projeto ("*Determine Project Effort*"), onde o primeiro provê duas interfaces ao segundo. Isto é evidenciado pela linha que representa essas conexões sendo a mais grossa da figura, por indicar um número maior de interfaces trocadas.
  3. O valor CRnC do componente Determinar o Esforço do Projeto ("*Determine Project Effort*") ser igual a zero, indicando que não há fornecimento de serviços deste componente para os outros no domínio, apenas provê resultados para o meio.
  4. A responsabilidade balanceada do componente Desenvolver Termo de Abertura ("*Develop Project Charter*") que apesar de possuir o valor de CRnC igual a dois e superior ao valor do CRnC médio do domínio, provê interfaces aos outros dois componentes, como pode ser observado pelas linhas que saem deste componente na figura.

  Este cenário reforça o agrupamento do componente Determinar o Esforço do Projeto ("*Determine Project Effort*") com o componente Determinar Tamanho do Projeto ("*Determine Project Size*"), já que este irá minimizar os valores de CPI no domínio.

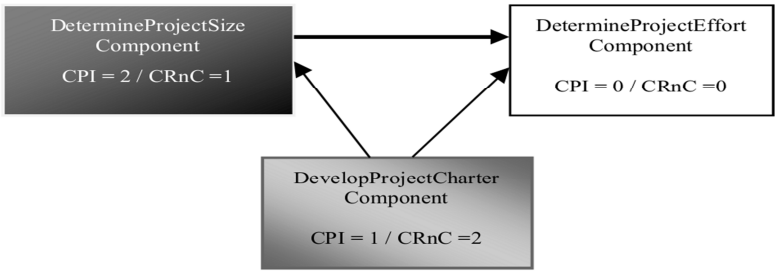


Figura 4.17 – Visão de Responsabilidades dos componentes da LPrS de Planejamento de Projetos de Software

4.4.2.2. Análise pelo grau de dependência dos componentes

Dois critérios (CRI e CRdC) podem ser utilizados para analisar o grau de dependência de cada componente no domínio.

1. Uma das formas de dependência de um componente no domínio consiste em requerer artefatos que são produzidos por atividades precedentes no domínio. Este grau de dependência pode ser avaliado com o valor do critério que calcula o número de interfaces requeridas (CRI).

2. O grau de dependência de um componente pode ser medido também por análise direta do número de componentes relacionados e não somente pelo número de interfaces trocadas. Desta forma, poderia ser importante observar que um componente pode possuir um CRI alto pelo número de interfaces requeridas, mas todas representaram dependência de apenas um componente. Esse tipo de informação pode auxiliar na tomada de decisão de agrupamento e é tratado pelo critério CRdC. Além disso, o critério que calcula o número de componentes requeridos pelo componente analisado (CRdC) avalia tanto a dependência pelo requerimento dos artefatos (dependência no fluxo de dados) como no caso de dependências entre componentes (unidades de trabalho) que representam a necessidade da execução de ações que podem não gerar nenhum recurso a ser consumido (dependência no fluxo de controle) como, por exemplo, atividades de verificações/validações que permitem a continuidade de execução do processo.

C4 - Análise de componentes relacionados por interfaces requeridas - *Component Required Interfaces* (CRI): Análise do grau de dependência de cada componente dentro do domínio pela análise de interfaces requeridas.

***Component Required Interfaces* (CRI)** - Cálculo do número de interfaces requeridas de cada componente.

É importante lembrar que se devem considerar apenas interfaces trocadas entre componentes e descartar interfaces que requerem produtos de trabalho apenas do ambiente (interfaces externas), já que as interfaces trocadas entre componentes são as que produzem impactos no aspecto de acoplamento.

Para guiar a análise deste critério, o valor médio de CRI no domínio pode ser considerado como um parâmetro. Desta forma, quando o valor CRI do componente for maior que o CRI médio a possibilidade de agrupamento deste componente com todos ou alguns dos componentes requeridos deve ser considerada.

Considerações para aplicar o agrupamento:

1. O agrupamento é fortemente sugerido nos casos em que uma interface é requerida por apenas outro componente.
2. No caso de interfaces requeridas de mais de um componente:
    a. Considerar os componentes para os quais o componente analisado fornece alguma interface, indicando que há uma troca de serviços entre os componentes. Esses componentes devem ser priorizados para a realização de agrupamento.
    b. Considerar componentes que proveem um maior número de interfaces dentre o total das interfaces requeridas. Esses componentes devem ser priorizados para a realização de agrupamento, pois indicam um grau maior de responsabilidade com o componente analisado.

As interfaces são requeridas de um número de componentes que permite o agrupamento de todos os componentes envolvidos sem um aumento significativo na complexidade do componente resultante ou aumento do valor de algum critério de acoplamento. Essa análise depende de outros fatores como objetivos dos componentes; número de serviços providos por todos os componentes envolvidos; número de dependências com outros componentes do domínio; tipo de opcionalidade ou variabilidade no domínio; dentre outros.

- **Exemplo**: Analisando a LPrS de Planejamento de Projetos de Software, o componente Determinar o Esforço do Projeto ("*Determine Project Effort*") apresenta um valor de CRI igual a três. Tal valor é superior ao valor médio de CRI, que é igual a 1.33.

$$CRI_{DetermineProjectEffortComponent} = 3 > CRI_{Médio} = 1.33$$

Segundo o critério, este cenário sugere o agrupamento com outros componentes do domínio. Analisando apenas este critério, não há informação de qual dos outros dois componentes seria o mais adequado.

C5 - Análise de componentes requeridos - *Components Required by a Component* (CRdC): Análise do grau de dependência, indicando o número de componentes requeridos pelo componente analisado, representando a dependência deste componente por outro(s) responsável(is) pela execução de determinadas ações com a produção ou não de artefatos.

Components Required by a Component (CRdC) - Cálculo do número de componentes requeridos por um determinado componente.

Para guiar a análise deste critério o valor médio de CRdC no domínio pode ser considerado como um parâmetro. Desta forma, quando o valor CRdC do componente for maior que o CRdC médio, a possibilidade de agrupamento deste componente com aqueles que o requerem.

Algumas considerações similares as do critério CRI podem ser utilizadas para aplicar ou não o agrupamento:

1. O agrupamento é fortemente sugerido nos casos em que há dependência de apenas outro componente (CRdC = 1).
2. No caso dos serviços serem requeridos de mais de um componente (CRdC >1):
    a. Considerar os componentes para os quais o componente analisado fornece alguma interface, indicando que há uma troca de serviços entre os componentes. Esses componentes devem ser priorizados para a realização de agrupamento.
    b. Considerar componentes que proveem um maior número de serviços dentre o total dos serviços requeridos. Esses componentes devem ser priorizados para a realização de agrupamento, pois indicam um grau maior de responsabilidade com o componente analisado.
    c. O número de componentes requeridos pelo permite o agrupamento de todos os componentes envolvidos sem um aumento significativo na complexidade do componente resultante ou aumento do valor de algum dos critérios de acoplamento. Essa análise depende de outros fatores como objetivos dos componentes; número de serviços providos por

todos os componentes envolvidos; número de dependências com outros componentes do domínio; tipo de opcionalidade ou variabilidade no domínio; dentre outros.

Obs.: É importante observar que em modelos em que os componentes com poucas interfaces providas, os valores de CRI e CRdC serão similares.

- **Exemplo**: Analisando a LPrS de Planejamento de Projetos de Software, o componente Determinar o Esforço do Projeto ("*Determine Project Effort*") apresenta um valor de CRdC igual a dois. Tal valor é superior ao valor médio de CRdC, que é igual a 1.

$$CRdC_{DetermineProjectEffortComponent} = 2 > CRdC_{Médio} = 1$$

Segundo o critério, este cenário sugere o agrupamento com outros componentes do domínio. No entanto, como tal componente possui relações de dependências com outros dois componentes, analisando apenas este critério não há informação de qual dos outros dois componentes seria o mais adequado.

Os dois critérios citados acima podem ser usados individualmente. No entanto, a análise individual, ainda que apresente alguma possibilidade de reestruturação do modelo, pode não prover informação suficiente para guiar o agrupamento mais adequado, que seria melhorar os valores dos critérios, minimizando o acoplamento. Para gerar melhores resultados, o ideal é combinar os critérios e avalia-los em conjunto. Um alto número de interfaces requeridas por um componente (CRI > CRI médio) de um grupo reduzido de componentes (CRdC <= CRdC médio) pode indicar uma reestruturação no domínio com o agrupamento de todos ou parte dos componentes envolvidos nesta relação de serviços requeridos, conforme descrito anteriormente.

- **Exemplo**: Analisando a LPrS de Planejamento de Projetos de Software segundo a visão de dependências na Figura 4.18, a sugestão de reestruturação indica considerar:

O componente Determinar o Esforço do Projeto ("*Determine Project Effort*") possui um alto grau de dependência, requisitando três interfaces ($CRI_{DetermineProjectEffortComponent} = 3$) dos outros dois componentes ($CRdC_{DetermineProjectEffortComponent} = 2$).

O componente Determinar o Esforço do Projeto (“*Determine Project Effort*”) possui uma relação mais forte com o componente Determinar Tamanho do Projeto (“*Determine Project Size*”) que fornece duas das três interfaces requeridas, como pode ser observado pela linha mais grossa entre esses componentes.

Este cenário reforça o agrupamento do componente Determinar o Esforço do Projeto (“*Determine Project Effort*”) com o componente Determinar Tamanho do Projeto (“*Determine Project Size*”).

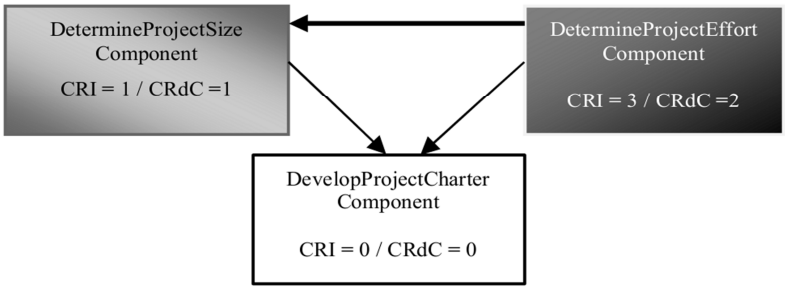


Figura 4.18 – Visão de Dependências dos componentes da LPrS de Planejamento de Projetos de Software

4.4.2.3. Análise pelo grau de instabilidade de um componente

Este critério obtém a relação entre o grau de dependência e responsabilidade de cada componente, combinando os valores dos critérios CRI e CPI, conforme a seguir.

C5 – Instabilidade (I): Uma forma de combinar a relação entre o grau de responsabilidade (CPI) e o grau de dependência de cada componente (CRI) consiste no critério de instabilidade.

- Altos valores de CPI indicam a relevância de um componente dentro do domínio. Alterações neste componente tendem a afetar os componentes a ele relacionados.
- Valores altos de CRI afetam negativamente um componente, pois indicam que são componentes que podem ser afetados por outros.

O critério de instabilidade visa analisar um equilíbrio entre essas relações com maior peso para o grau de dependência:

$$I = CRI / (CRI + CPI)$$

1. Quando o valor de I ficar próximo de zero (I -> 0), o componente é considerado mais estável e com um grau de dependência mais baixo.

2. Quando o valor de I ficar próximo de um (I -> 1), indica instabilidade e alto grau de dependência de um componente.
3. Quando o valor de I ficar próximo de meio (I -> 0,5), pode ser considerada a existência de um equilíbrio entre responsabilidade e dependência de um componente. Desta forma, este é considerado o parâmetro para análise do componente do domínio.

Quando um componente possuir o valor de I acima do parâmetro de equilíbrio (I > 0,5), uma reorganização do modelo de componentes é aconselhável para reduzir a dependência do componente. Nos casos em que o componente for considerado instável (I próximo de 1), a reestruturação do modelo com o agrupamento do componente com os relacionados é reforçada.

Este critério deve ser aplicado em cenários com mais de um componente que apresentam troca de informações via interfaces, onde valores de CPI e CRI sejam maiores que zero.

- **Exemplo**: Analisando a LPrS de Planejamento de Projetos de Software, o componente Determinar o Esforço do Projeto ("*Determine Project Effort*") pode ser considerado o mais instável, pois apresenta um valor de instabilidade igual a 1. Na visão gráfica (Figura 4.19), este componente é apresentado pela cor preta indicando essa instabilidade e focando a atenção para que alguma reestruturação seja realizada com a participação deste componente.

  Por outro lado, o componente Desenvolver Termo de Abertura ("*Develop Project Charter*") apresenta um valor de instabilidade igual a zero, indicando que o componente não depende de nenhum outro no domínio.

  O componente Determinar Tamanho do Projeto ("*Determine Project Size*") apresenta um valor de instabilidade igual a 0.33 mais próximo de 0.5 que representa um balanceamento entre dependência e responsabilidade.

Figura 4.19 – Visão de Instabilidades dos componentes da LPrS de Planejamento de Projetos de Software

Os cinco critérios listados anteriormente (CPI, CRnC, CRI, CRdC e I) trabalham com a análise das relações entre componentes considerando diferentes pontos de vistas (fluxo de dados e de controle, número de interfaces e número de componentes). Isso enfatiza que a combinação dos critérios é recomendada para analisar a possibilidade de reestruturação do modelo de componentes.

- **Exemplo:** Analisando a LPrS de Planejamento de Projetos de Software e combinando os cinco critérios (CPI, CRI, CRnC, CRdC e I) com a análise visual das três perspectivas, pode-se analisar:
  1. O componente Determinar o Esforço do Projeto ("*Determine Project Effort*") apresenta alto grau de dependência no domínio dependendo dos outros dois componentes ($CRnC_{DetermineProjectEffortComponent} = 2$) e baixa responsabilidade, pois não produz serviços para nenhum outro componente ($CRnC_{DetermineProjectEffortComponent} = 0$).
  2. Considerando os dois componentes dos quais o componente Determinar o Esforço do Projeto ("*Determine Project Effort*") depende, apenas um deles depende do outro: o componente Determinar Tamanho do Projeto ("*Determine Project Size*") depende do componente Desenvolver Termo de Abertura ("*Develop Project Charter*").
  3. Caso o componente Determinar o Esforço do Projeto ("*Determine Project Effort*") seja agrupado com o componente Desenvolver Termo de Abertura ("*Develop Project Charter*"), os dois componentes resultantes no modelo ainda continuaram possuindo dependências um do outro.
  4. Caso o componente Determinar Tamanho do Projeto ("*Determine Project Size*") seja agrupado com o componente Desenvolver Termo de Abertura ("*Develop Project Charter*"), o componente Determinar o Esforço do Projeto ("*Determine Project Effort*") irá manter os valores dos critérios CRI = 3 e I = 1, mantendo o cenário de alta dependência e instabilidade.

  Este cenário reforça o agrupamento do componente Determinar o Esforço do Projeto ("*Determine Project Effort*") com o componente Determinar Tamanho do Projeto ("*Determine Project Size*"). Tal agrupamento minimiza os valores dos critérios, i.e., os valores máximos de

CRI e CPI passam a ser um, indicando diminuição da probabilidade de impactos entre os componentes.

Existe a possibilidade de um segundo agrupamento, restando apenas um componente agrupando todos os outros. Mas esta situação não é indicada porque a complexidade de tal componente seria elevada e os propósitos dos componentes não estariam mais claramente separados. Desta forma, o modelo final é apresentado na Figura 4.20.

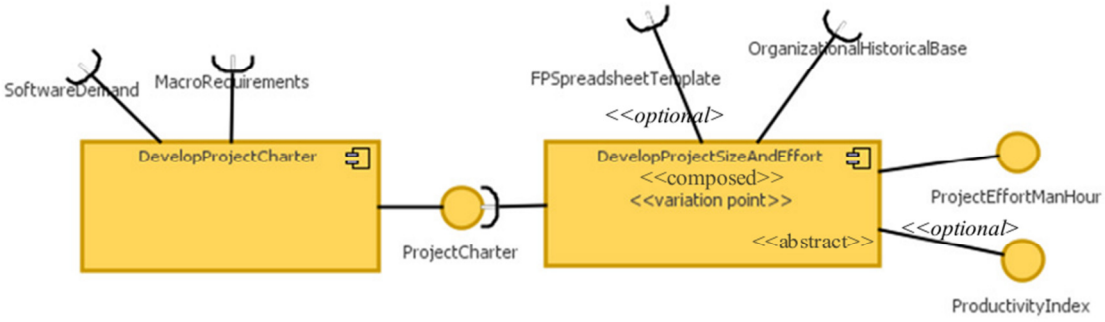


Figura 4.20 – Modelo Final de Componentes Agrupados da LPrS de Planejamento de Projetos de Software

4.4.3. Outras considerações

O agrupamento de componentes ainda que de extrema importância para organização da arquitetura em módulos mais compreensíveis e com maior coesão pode ser considerado uma atividade opcional que deve ser conduzida de acordo com a validação do engenheiro de domínio. Para apoiá-lo nesta tomada de decisão de diagnóstico da complexidade do domínio e análise das diferentes possibilidades de agrupamento, uma lista de observações deve ser considerada como informação complementar aos critérios propostos:

- Os agrupamentos aplicados devem minimizar os valores dos critérios CPI, CRI, CRnC e CRdC, visando reduzir o acoplamento entre os componentes e as interdependências. Desta forma, reduzindo as dependências pode-se diminuir a probabilidade de impactos entre componentes;
- Devem ser evitados os agrupamentos de componentes que pertencem a escopos diferentes;
- O agrupamento deve considerar a complexidade envolvida na execução do componente resultante. Desta forma, deve-se evitar o agrupamento de

componentes compostos por ações com alto grau de dificuldade para ser executado;

- Os propósitos dos componentes que forem agrupados devem ser analisados para não haver conflitos de objetivos ou alterações nos resultados a serem esperados pelo componente resultante. Possivelmente, devem ser evitados agrupamentos de componentes com propósitos altamente controlados e resultados relevantes a múltiplos componentes no domínio;
- Os valores dos critérios representam relações de dependências e responsabilidades diretas entre dois componentes. É importante observar o grau dessas relações em redes de componentes relacionados em outras relações secundárias;
- A classificação de opcionalidade e variabilidade de componentes e seus elementos internos pode afetar a atividade de agrupamento. É importante considerar que classificações de um componente podem ser alteradas, por exemplo, um componente concreto agrupado com um componente opcional passa a ser classificado como abstrato;
- Agrupamentos que contém pelo menos um componente classificado como mandatório deve manter essa classificação para todo o componente agrupado resultante;
- Agrupamentos que contém pelo menos um componente classificado como ponto de variação deve manter essa classificação para todo o componente agrupado resultante;
- O agrupamento de componentes que apresentam restrições relacionadas a regras de composição exclusivas deve ser evitado;
- O agrupamento de componentes que apresentam pré-condições relacionadas a regras de composição inclusivas devem ser estimuladas;
- As interfaces de controle que representam os possíveis fluxos de execução podem auxiliar na análise de dependências de componentes que devem ser executados anteriormente a outros, principalmente, se não há relações de troca de produtos de trabalhos entre tais componentes;

- O agrupamento deve manter ou aumentar a coesão da estrutura interna do componente resultante. A coesão indica que os elementos que compõem um componente trabalham em conjunto para atingir um propósito em comum. Este aspecto pode ser analisado segundo duas perspectivas:
    1. Todos os elementos internos estão relacionados, sem a presença de conjuntos isolados de elementos dentro de um componente. A exceção para esta análise são componente classificados como pontos de variação, onde variantes caracterizadas como componentes internos podem apresentar-se como conjuntos isolados de alternativas para resolução do componente em tempo de definição de processo a partir da linha;
    2. Uma alta coesão também pode ser observada quando um componente possui mais relações internas do que externas.
        a. Coesão = número de elementos internos de um componente com mais relações internas/ número total de elementos internos. O valor de referência seria um, indicando que todos os elementos internos estão altamente relacionados entre si. Esse cálculo deve ser aplicado por tipo de elemento separadamente: unidades de trabalho, papéis, produtos de trabalho e ferramentas.

O cálculo dos valores dos critérios pode ser automatizado e a representação gráfica das perspectivas de dependências, responsabilidades e instabilidades podem ser derivada automaticamente. No entanto, independentes dos agrupamentos sugeridos pelos valores dos critérios, a percepção e os aspectos de interesse do engenheiro de domínio são prevalentes e guiam a decisão final de quais componentes podem ou não serem agrupados em elementos arquiteturais. Por isso, o agrupamento é uma atividade semi-automatizada que precisa ser completada pelo usuário da informação.

Os critérios propostos são alguns aspectos que podem ser observados relacionados à organização dos elementos do domínio e o acoplamento dos componentes relacionados. No entanto, outros aspectos podem ser considerados para proposição de agrupamento como aspectos de colaboração entre equipes de

desenvolvimento; localização dos papéis envolvidos na execução de determinados componentes; componentes relacionados a determinados tipos de metodologia de desenvolvimento de software ou a determinados tipos de produtos; dentre outros aspectos. Essas diferentes possibilidades de visões para observação e análise devem ser consideradas para extensões dos critérios em oportunidades de trabalhos futuros. Outro ponto a ser considerado é que agrupamentos podem ser sugeridos para reorganizar o domínio, como proposto nesta tese, mas podem ser aplicados considerando melhorias nos processos a serem derivados. Diferentes propósitos para a aplicação dos critérios devem ser pesquisados como oportunidades de trabalhos futuros.

4.4.4. Estudo de Viabilidade de Critérios de Agrupamento

Neste estudo de caso, a viabilidade de aplicação de um conjunto de critérios para apoiar o agrupamento de componentes de processo, considerando aspectos como acoplamento e granularidade, foi observada pela utilização dos critérios em um domínio de processo de software em ambiente acadêmico. Esse estudo, apesar de realizado de forma separada, foi complementar ao estudo apresentado na Seção 5.4, contanto com o mesmo domínio e o mesmo conjunto de participantes.

Apesar de importante para organização do conhecimento do domínio, a atividade de agrupamento de componentes de processos de software pode ser considerada opcional e deve ser conduzida e validada pelo engenheiro de domínio. Os critérios propostos podem ser entendidos como recomendações para suportar a decisão de agrupamento de componentes, mas são apenas sugestões e indicações que precisam ser interpretadas e combinadas com a experiência do engenheiro de domínio, que é o responsável pela decisão dos agrupamentos mais adequados à modelagem do domínio. Desta forma, a intenção deste estudo foi analisar se os critérios propostos seriam utilizados de maneira apropriada e se de fato auxiliariam a tomada de decisão para o agrupamento de componentes de processos. A análise foi subjetiva e inclui a observação:

- Se algum agrupamento era proposto pelo participante;
- Se o agrupamento proposto era o mesmo do agrupamento feito pelo observador; e

- Formulário pós-estudo.

4.4.4.1. Planejamento do Estudo

- **Objetivo Geral**: Este estudo visa compreender a viabilidade de aplicação de um conjunto de critérios de agrupamento de componentes de processos de software considerando aspectos de acoplamento e granularidade.

  Baseado em (BASILI *et al*., 1994), o objetivo do estudo foi definido como:

  Analisar um conjunto de critérios de agrupamento de componentes de processos de software

  Com o propósito de caracterizar

  Com respeito ao uso dos critérios propostos

  Do ponto de vista de pesquisadores de Engenharia de Software

  No contexto de desenvolvedores de software aplicando os critérios em um domínio de processos de software

- **Questões:**
  - **Questão (1):** *Os critérios propostos geraram algum agrupamento pertinente de componentes no domínio?*
  - **Métrica:**
    - **Número de agrupamentos de componentes que não afeta negativamente a organização do domínio.**

Uma proposta de organização do domínio em componentes agrupados seguindo a aplicação dos critérios foi apresentada na Seção 4.4 e foi derivada antes da aplicação deste estudo. Como o agrupamento de componentes pode ser considerado opcional, o estudo se propõe a observar se algum agrupamento será sugerido pelo participante. Caso algum agrupamento seja sugerido, se ele é o mesmo que o esperado (agrupamento sugerido na Seção 4.4). Não sendo o agrupamento esperado, será observado se o agrupamento não impacta negativamente a modularidade do conhecimento representando, como, por exemplo, um agrupamento excessivo colocando todos os componentes em apenas um único, aumentando a complexidade e não distinguindo componentes por propósitos claramente distintos. Além disso, deve observar se os valores dos critérios foram minimizados, objetivo da proposta do agrupamento como forma de alcançar melhorias no acoplamento e na granularidade do domínio.

- **Hipótese:**
    - **Hipótese Nula (H0):** É possível aplicar os critérios para apoiar a atividade de agrupamento de componentes, gerando componentes com diminuição do acoplamento no domínio.
        - **Hipótese Alternativa (H1):** Não é possível aplicar os critérios para apoiar a atividade de agrupamento de componentes, gerando componentes com diminuição do acoplamento no domínio.
- **Seleção do contexto:** O estudo é executado em ambiente acadêmico por estudantes de pós-graduação de um curso de reutilização de software**.** Para a execução do estudo, parte do domínio de Planejamento de Projetos de Desenvolvimento de Software, com foco nas atividades iniciais deste processo e suas variações, foi modelado utilizando a notação *OdysseyProcess-FEX* nível de projeto (modelo de componentes apresentado na Seção 4.2). Domínios maiores não foram aplicados pelo cenário de aplicação manual do estudo, sem recursos de visualização e filtros de informação, e viabilidade de aplicação dentro do tempo disponível.
- **Seleção dos participantes:** Foram selecionados os mesmos cinco participantes do estudo de viabilidade de aplicação das heurísticas, pela familiaridade deles com a modelagem proposta e com o domínio a ser utilizado.
- **Variáveis:**
    - **Variáveis Independentes:** domínio de processo de software representado pelo modelo de componentes de processos de software usando a notação *OdysseyProcess-FEX*, experiência dos participantes em reutilização de processos de software, conhecimento prévio dos participantes do domínio usado no estudo, número de elementos definidos no modelo de componentes.
    - **Variáveis Dependentes***:* número de agrupamentos sugeridos por cada participante, o agrupamento sugerido pelo participante e o tempo para realização da atividade (medido em minutos).
- **Validade do estudo:** Algumas ameaças à validade do estudo foram identificadas como:

- o número reduzido de participantes, sendo todos de um mesmo curso e com experiência limitada em reutilização de processos.
- A utilização de apenas um domínio para aplicação dos critérios pode impactar a validade externa. A aplicação em outros cenários e domínios deve ser considerada.
- Da forma como o conhecimento foi transmitido não se tem garantias do entendimento a contento dos critérios observados. Mesmo com a leitura introdutória e apresentação dos critérios com exemplos de aplicação, esse fator pode ser ainda mais evidenciado pelos apontamentos feitos pelos participantes no formulário pós-estudo da necessidade de mais exemplos e informações adicionais.
- O fato dos participantes serem os mesmos do estudo de viabilidade das heurísticas de mapeamento pode gerar algum impacto (positivo ou negativo) na execução deste segundo estudo.
- A ausência de um método comparativo pode ser considerado outro fator, o ideal seria a separação dos participantes em dois grupos (um grupo utilizando um processo *ad-hoc* e outro utilizando os critérios disponíveis).
- A aplicação apenas em ambiente acadêmico.
- O tamanho da amostra impacta a validade de conclusão, onde o número reduzido de participantes não possibilita uma análise estatística, e limita os resultados que podem ser considerados apenas indícios e não generalizáveis.

4.4.4.2. Execução do Estudo

Preparação

O convite para participação do estudo foi enviado por e-mail com breve explicação do contexto.

No dia do estudo, um treinamento foi fornecido contendo:

- Uma apresentação sobre o contexto de realização do estudo e apresentação sobre o modelo de componentes de processos de

software utilizado segundo o metamodelo e notação *OdysseyProcess-FEX*.

- Uma apresentação dos critérios de agrupamento sugeridos a serem aplicados no estudo.

Operação

A cada participante foi entregue um conjunto de documentos contendo o modelo de componentes de processos de software do domínio de Planejamento de Projetos de Software e material de apoio à aplicação do estudo: documento com os critérios de agrupamento a serem aplicados e formulário para preenchimento de possíveis agrupamentos a serem realizados. O Apêndice VI apresenta a documentação entregue aos participantes.

Os participantes analisaram o modelo de componentes entregue e realizaram a atividade de agrupamento de componentes utilizando os critérios sugeridos. Foi explicado que a realização de agrupamento era opcional a cargo de avaliação do participante.

Ao final da atividade, os participantes preencheram um formulário de avaliação pós-estudo contendo itens objetivos (e.g., sim, não, parcialmente) e um item subjetivo (e.g., justificativa das respostas). O resultado da atividade e das respostas do formulário de avaliação é que efetivamente foram considerados para avaliar a viabilidade de aplicação dos critérios.

4.4.4.3. Análise dos resultados obtidos

O agrupamento de componentes esperado está descrito na Seção 4.4 do Capítulo 4, assim como várias considerações sob a aplicação de cada critério no modelo de componentes utilizado no estudo. Desta forma, criou-se uma base para interpretar os possíveis mapeamentos a serem sugeridos por cada participante.

Todos os participantes indicaram pelo menos um agrupamento a ser realizado no modelo de componentes (Tabela 4.13). Três participantes indicaram o agrupamento de componentes esperado, agrupando os componentes que tratam das atividades para determinar o tamanho e o esforço do projeto. Tal agrupamento minimiza os valores dos critérios, i.e., os valores máximos de CRI e CPI passam a ser um, indicando diminuição da probabilidade de impactos entre os componentes. Um

participante indicou dois agrupamentos consecutivos, primeiro o agrupamento esperado e depois um segundo agrupamento com todos os componentes reunidos em um único componente final. Mas esta situação não é indicada porque a complexidade de tal componente seria elevada e os propósitos dos componentes não estariam mais claramente separados. Outro participante propôs outro tipo de agrupamento, reunindo o componente *Develop Project Charter* e o componente *Determine Project Effort*. No entanto, neste caso, os dois componentes resultantes no modelo ainda continuaram possuindo as mesmas dependências um do outro, mantendo altos os valores dos critérios analisados. O tempo gasto para a realização da atividade foi em média de 39,6 minutos.

Tabela 4.13 – Sumarização dos resultados de aplicação dos critérios de agrupamento de componentes pelos participantes

Participante	Nível de Experiência	Tempo (minutos)	#Agrupamentos de Componentes
P1	0,57	27	1 (esperado)
P2	0,35	51	1 (esperado)
P3	0,25	27	1
P4	0,20	29	1 (esperado)
P5	0,19	64	2

Os maiores problemas identificados foram classificações incorretas de interfaces, omissão de interfaces, uso de mais de um tipo de conector associado a uma interface de controle e alteração no fluxo de execução (inversão de predecessores e sucessores).

Os critérios foram combinados de diferentes formas por cada participante para apoiar o agrupamento proposto. Este cenário enfatiza que mesmo sendo possível a aplicação de cada critério separadamente, na maioria dos casos, o uso combinado é indicado para melhorar a tomada de decisão sobre o agrupamento de componentes.

Todos os cinco critérios de análise de componentes com relações entre si (critérios de dependência, responsabilidade e instabilidade) foram aplicados e nenhum dos participantes indicou o uso de critério de análise de escopos do domínio definidos, conforme o esperado.

De acordo com o preenchimento do formulário de avaliação pós-estudo, pode-se observar que os participantes concordam ou parcialmente concordam que a

notação, o domínio e os critérios são de fácil entendimento. No entanto, mesmo com a sessão de treinamento, algum tempo foi requerido para absorver o conhecimento envolvido e alguns participantes indicaram a necessidade de alguns exemplos como informação adicional útil para entendimento da aplicação dos critérios. A Tabela 4.14 apresenta um resumo das respostas obtidas com relação à avaliação das heurísticas pelos participantes.

Tabela 4.14 – Sumarização dos resultados de avaliação dos critérios de agrupamento pelos participantes

	Avaliação Mapeamento						
	Questão 1	Questão 2	Questão 3	Questão 4	Questão 5	Questão 6	Questão 7
	CP	DT	CP	CT	CT	CT	DT
	CP	DT	CP	CP	CP	CT	DT
	CT	DT	DT	CP	CP	CT	DT
	I	DT	CP	CT	CT	I	DT
	CT	DT	DT	CT	CP	I	DT
	Resposta esperada: CP ou CT = 4/5	Resposta esperada: DT ou DP= 5/5	Resposta esperada: DT ou DP= 2/5	Resposta esperada: CP ou CT = 5/5	Resposta esperada: CP ou CT = 5/5	Resposta esperada: CP ou CT = 3/5	Resposta esperada: CP ou CT = 0/5
	Resposta não esperada: DT ou DP = 0/5	Resposta não esperada: CT ou CP = 0/5	Resposta não esperada: CT ou CP = 3/5	Resposta não esperada: DT ou DP = 0/5	Resposta não esperada: DT ou DP = 0/5	Resposta não esperada: DT ou DP = 0/5	Resposta não esperada: DT ou DP = 5/5
Questão 1	1. Os critérios disponibilizados para apoiar o agrupamento de componentes de processos de software, visando reduzir o acoplamento, são adequados. Em caso negativo, ou parcialmente negativo, indique os critérios que na sua opinião não são adequados, justificando.						
Questão 2	2. Em algum momento você desejou agrupar determinados componentes e esta forma de agrupamento não foi apoiada pelas análises realizadas usando os critérios disponibilizados. Em caso afirmativo, ou parcialmente afirmativo, aponte qual(is) critério(s) poderiam ser modificados, justificando						
Questão 3	3. Em algum momento você sentiu dificuldade na aplicação de um ou mais critérios. Em caso afirmativo ou parcialmente afirmativo, qual(is)? Justifique indicando a dificuldade enfrentada e uma possível sugestão de melhoria						
Questão 4	4. A forma de representação visual dos critérios e a separação por visões (responsabilidade, dependência e instabilidade) são adequadas e auxiliam na análise da organização dos componentes. Em caso negativo, ou parcialmente negativo, o que poderia ser melhorado? Justifique						
Questão 5	5. Um modelo de componentes agrupados é mais compreensível quando comparado a um modelo original com mais componentes dispostos separadamente. Em caso negativo, ou parcialmente negativo, justifique						
Questão 6	6. A utilização de filtros para aplicação/visualização dos critérios em etapas seria desejável. Em caso afirmativo, ou parcialmente afirmativo, justifique						
Questão 7	7. Eu já utilizei outras abordagens de LPrS envolvendo agrupamentos de componentes de processo. Em caso afirmativo, ou parcialmente afirmativo, aponte qual mecanismo e se possui algum apoio ferramental						

Algumas sugestões foram indicadas para melhorias dos critérios. Uma das sugestões consiste em prover informação adicional para decidir o agrupamento mais apropriado como considerações sobre graus de dependência e responsabilidades de uma rede de componentes e não apenas entre dois componentes diretamente relacionados. O critério de instabilidade deveria de alguma forma variar mais em função do CPI. O critério CRI apresenta um peso maior, já que quando este assume o valor igual a zero, o critério CPI não apresenta nenhum impacto na análise da instabilidade. Além disso, se o valor de CPI passa a ser zero, a instabilidade sempre assume um valor igual a um.

Participantes não identificaram nenhum critério redundante e a maioria (4 entre os 5) concordam que um suporte computacional poderia auxiliar na atividade de agrupamento, pela possibilidade de aplicação de filtros para análise de diferentes pontos de vista dos critérios, garantindo que todos os componentes serão considerados na análise e provendo sugestões de agrupamento. No entanto, a

importância de interação do engenheiro de domínio foi apontada, reforçando sua responsabilidade em tomar a decisão do agrupamento mais apropriado para organizar o domínio, tendo os critérios apenas como apoio e direcionamento.

Os resultados obtidos com a análise resultaram em melhorias nas recomendações e considerações adicionais que podem ser aplicadas junto aos critérios propostos. A descrição de cada critério foi estendida e alguns direcionamentos foram adicionados de como aplicá-los de forma combinada nas três perspectivas definidas: dependência, responsabilidade e instabilidade. As melhorias foram descritas na Seção 4.4 do Capítulo 4.

Com a realização do estudo foi possível ter evidências, ainda que não generalizáveis, sobre a viabilidade de aplicação de critérios para apoiar o agrupamento de componentes de processos de software. Além disso, todos os participantes concordam, ou parcialmente concordam, que um modelo com os componentes agrupados é mais compreensível quando comparado a um modelo original com mais componentes dispostos separadamente, reforçando a evidência da importância do agrupamento de componentes.

4.4.4.4. Considerações combinadas dos estudos de viabilidade das heurísticas e dos critérios de agrupamento

Analisando os resultados alcançados com os estudos de viabilidade das heurísticas e dos critérios podemos enumerar alguns pontos (Figura 4.21):

- Tempo de realização dos estudos: O estudo de aplicação das heurísticas de mapeamento exigiu um tempo maior dos participantes para realização da atividade. A média de tempo gasto neste estudo foi de 110.6 minutos. Esse fator pode demonstrar alta complexidade para realização da atividade de mapeamento entre os níveis de abstração. Já no estudo de aplicação dos critérios o tempo médio foi de apenas 39.6 minutos. Este fator pode estar ligado a uma menor complexidade de realização da atividade, pode ter sido impactado pela absorção de conhecimento proveniente da aplicação do primeiro estudo, já que os participantes foram os mesmos, e ainda pode indicar que os critérios de fato apoiaram a tomada de decisão de qual agrupamento poderia ser aplicado.

Outro ponto a se analisar com relação ao tempo é que no estudo das heurísticas, os participantes mais experientes foram os que tiveram os maiores tempos registrados. Este fator pode ter sido influenciado pelo maior detalhamento aplicado por tais participantes, o que pode ser notado pelo preenchimento do formulário de mapeamento resultante.

Já para a aplicação dos critérios, o participante mais experiente teve um dos menores tempos e o participante menos experiente apresentou o maior tempo para realização da atividade. Este fator pode indicar uma maior necessidade de conhecimento externo à abordagem para tomada de decisão do agrupamento mais adequado ao domínio.

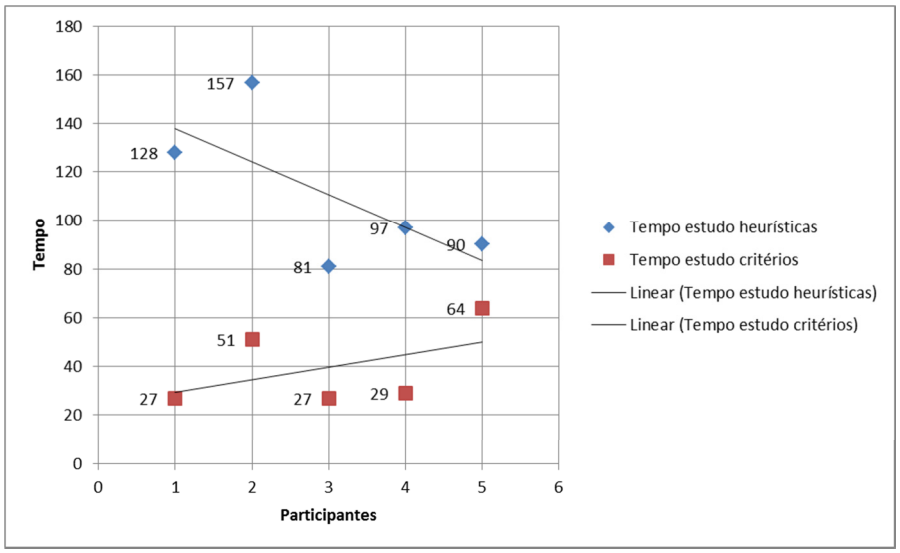


Figura 4.21 – Análise tempo nos estudos

Observando os resultados obtidos pelos participantes *versus* os resultados esperados podemos notar que o participante com maior nível de experiência alcançou os melhores resultados esperados nos dois estudos. O participante com o segundo maior nível de experiência teve um desempenho bom no estudo de aplicação dos critérios de agrupamento, gerando o agrupamento esperado, mas não teve rendimento diferenciado na aplicação das heurísticas de mapeamento, mantendo uma quantidade de acertos próxima a dos demais participantes com nível de experiência inferior.

A maioria dos participantes alcançou o agrupamento esperado para os componentes do domínio apresentado, com apenas um dos participantes indicando outro agrupamento não apropriado para o domínio.

Já com relação ao mapeamento entre os níveis de abstração, os participantes tiveram um rendimento menor com apenas três participantes alcançando pelo menos 50% da modelagem esperada e apenas o participante com maior nível de experiência teve uma taxa de 68% de acerto, ainda distante do número total de acertos esperados.

Tabela 4.15 – Sumarização dos resultados de avaliação das heurísticas e critérios de agrupamento

Participante	#Elementos Corretos (Heurísticas)	#Agrupamentos de Componentes (Critérios)
P1	34	1 (esperado)
P2	28,75	1 (esperado)
P3	23	1
P4	27,5	1 (esperado)
P5	21,5	2 (esperado + segundo agrupamento)

4.5. INSPEÇÃO DA MODELAGEM DO DOMÍNIO

A atividade de definição de processos de software é uma atividade propensa a erros que podem ser custosos durante a execução do processo, acarretando a atrasos, emprego de esforço desnecessário e retrabalho. Neste cenário, a detecção de defeitos no momento de definição é altamente indicada para evitar a propagação destes para momentos posteriores do desenvolvimento onde os custos de correção de desvios são maiores.

Atividades de validação e verificação representam uma importante abordagem para garantia da qualidade dos modelos utilizados antes da implementação, simulação ou execução, apoiando a entrega de software de qualidade (COLLOFELLO, 1988). A Inspeção de Software pode ser definida como um exame visual do artefato de software com o intuito de detectar anomalias (IEEE, 2008), especialmente, prevenindo a introdução de defeitos semânticos nos primeiros estágios de projetos de software, sendo considerada uma das técnicas mais eficientes para garantia da qualidade (DENGER, 2006). Esta técnica é executada individualmente e suportada por instrumentos como *checklists* (DE MELLO *et al*., 2010) (DE MELLO *et al*., 2014).

Se no cenário de desenvolvimento de software convencional, a verificação busca atuar nos artefatos iniciais por estes formarem a base para construção de um software, no cenário de reutilização, existe ainda o agravante de que um único artefato de domínio mal especificado pode levar à propagação de defeitos nos

diversos produtos resultantes de uma linha. Entretanto, no contexto de reutilização de processos de software, os resultados de uma revisão da literatura indicaram a carência de tecnologia específica para a verificação de artefatos de processos de software desenvolvidos especialmente para a reutilização. Tal revisão foi conduzida em 2014 dentro do contexto de uma disciplina de pós-graduação por uma aluna de mestrado. Como resultado, poucas abordagens foram identificadas para detecção de anomalias em processos, focando em uma verificação sintática dos modelos de processos, sem suportar uma verificação semântica da adequação do modelo frente a um conjunto de requisitos.

Desta forma, técnicas de inspeção baseadas em *checklists* foram desenvolvidas no contexto deste trabalho. Em uma primeira etapa, a técnica denominada PVMCheck (*Process Variability Modeling Checklist*) (TEIXEIRA *et al*., 2015) foi proposta como forma de suportar a detecção semântica da anomalias nos modelos de variabilidades de LPrS, baseada no metamodelo e na notação da *OdysseyProcess-FEX* e na técnica de inspeção de modelos de características de LPS denominada FMCheck (DE MELLO *et al*., 2014). Esta técnica desenvolvida dentro do contexto da disciplina teve como foco a etapa de análise de domínio e modelos de características de LPrS. Em uma segunda etapa de análise, com foco em um modelo de componentes da etapa de projeto do domínio, um segundo *checklist* foi desenvolvido com base no conhecimento adquirido com a técnica PVMCheck (TEIXEIRA *et al*., 2015) e em laudos de verificação propostos no trabalho de BARRETO (2011).

4.5.1. Base de conhecimento para desenvolvimento das técnicas de verificação de processos de software

A reutilização tende a potencializar os atributos de desenvolvimento de software, sendo que para uma melhor garantia da qualidade é preferível que os defeitos sejam detectados nos primeiros artefatos, impedindo a propagação destes nas instâncias derivadas. No contexto de melhoria de processos em geral, o foco está na fase de projeto, pois se encontra na primeira etapa do ciclo de vida, facilitando a correção e alteração caso defeitos sejam identificados (SÁNCHEZ-GONZÁLEZ *et al*., 2013). Como forma de apoiar a detecção de defeitos em artefatos de processos de software desenvolvidos especialmente para a reutilização, técnicas de verificação desses foram buscadas por uma revisão da literatura. A questão de pesquisa analisada

foi: "*Quais são as técnicas existentes para inspecionar artefatos de processo de software desenvolvidos para a reutilização*?". Devido a sua abrangência na realização de revisões sistemática, a busca foi conduzida na biblioteca digital SCOPUS[15]. Tal biblioteca agrupa publicações de diferentes e relevantes fontes da Engenharia de Software, como ACM, IEEE e outras. A derivação da *string* de busca foi realizada com base no conhecimento prévio aplicado em revisões da literatura de inspeção para LPS (DE MELLO *et al*., 2014) e em termos envolvidos em revisões da literatura de reutilização de processos de software conduzida durante o desenvolvimento desta tese. Como resultado desta busca, 340 publicações foram retornadas. Deste conjunto, 22 artigos foram selecionados para avaliação, segundo os seguintes critérios de inclusão previamente estabelecidos:

- Apenas abordagens baseadas no exame visual dos artefatos podem ser consideradas como técnicas de inspeção (IEEE, 2008), por isso, foram excluídas técnicas que especificam regras ou heurísticas sem a sua formalização e/ou execução ou técnicas de suporte a detecção por meio de ferramentas automatizadas como verificadores de modelos;
- O tipo de artefato inspecionado deveria ser relacionado a processos, projetado para reutilização e, preferencialmente, desenvolvido no contexto de LPrS;
- É desejável que a abordagem de inspeção tenha sido avaliada.

Após a leitura integral das publicações selecionadas, observou-se que nenhuma atendia aos três critérios estabelecidos, indicando uma limitação nas abordagens atuais, principalmente no que se refere à inspeção de LPrS. Um dos trabalhos retornados, ainda que não foque na verificação no contexto de processos de software, apresenta uma técnica de inspeção baseada em *checklist* para apoiar a detecção de defeitos semânticos em modelos de características de LPS denominada FMCheck (DE MELLO *et al*., 2014) e foi utilizada como base de conhecimento para desenvolvimento da técnica aqui proposta.

[15] https://www.elsevier.com/solutions/scopus

4.5.1.1. Process Variability Modeling Checking (PVMCheck)

A construção do *checklist* seguiu alguns passos: (1) identificação da categorização dos tipos de defeitos; (2) identificação de casos discrepantes; e (3) formulação das questões para verificação semântica de modelos de características de LPrS.

Para a realização de uma inspeção eficiente é importante o conhecimento sobre os tipos de defeitos e suas definições, para melhor identificação da causa e do seu ponto de origem. Diferentes taxonomias são propostas na literatura para uma clara categorização dos defeitos. Neste trabalho, foi adotada uma versão adaptada da taxonomia com cinco categorias comumente aplicada na classificação de defeitos detectados em modelos de software (TRAVASSOS *et al*., 1999) (Tabela 4.16).

Tabela 4.16 – Tipos de defeitos adaptados de (TRAVASSOS *et al*., 1999)

Categoria de Defeito	Descrição
Omissão	Informação necessária sobre o domínio que foi omitida no artefato.
Fato Incorreto	Alguma informação no artefato que contradiz informações da especificação de requisitos ou o conhecimento geral do domínio.
Inconsistência	A informação em certa parte do artefato está inconsistente com outra informação no artefato.
Ambiguidade	Informação não está clara, possibilitando múltiplas interpretações.
Informação estranha	A informação fornecida está fora do escopo, ou seja, não é necessária ou usada.

Com base no conjunto de requisitos para representação de variabilidade de processos de software identificada neste trabalho e trabalhada no metamodelo *OdysseyProcess-FEX*, casos discrepantes foram identificados com bases nos tipos de defeitos descritos anteriormente. Casos discrepantes são cenários que caracterizam uma discrepância, ou seja, representam uma situação genérica em que um defeito pode ser detectado. Os casos discrepantes identificados estão basicamente relacionados a: (i) consistência entre elementos no modelo; (ii) clareza relacionada à interpretação; (iii) corretude quando comparada a descrição textual do domínio; e (iv) completude quando comparado ao escopo domínio. Um total de 159 casos discrepantes foi identificado e organizado nos seguintes grupos:

- Elementos de Processos: 57 casos discrepantes relacionados à clareza e completude da descrição do elemento e a corretude com a categoria do

elemento. Correspondem a casos relacionados à análise individual de cada elemento e a consistência entre eles;

- Opcionalidade: 2 casos discrepantes relacionados a classificação correta de opcionalidade dos elementos considerando todo o domínio como descrito no documento base; e
- Variabilidade: 4 casos discrepantes relacionadas com anomalias nas alternativas representadas.
- Relacionamentos: 63 casos discrepantes referentes à representação apropriada dos relacionamentos entre os elementos do modelo, em comparação com o relacionamento especificado no documento base;
- Regras de Composição: 33 casos discrepantes relacionados à verificação da correta participação dos elementos em regras semânticas de dependência e mútua exclusividade entre elementos do domínio, representadas como descrições textuais lógicas na *OdysseyProcess-FEX*;

Alguns dos casos discrepantes podem ser considerados em um contexto mais amplo de reutilização de processos, tratando outras representações dos modelos de processos reutilizáveis. Outros casos são mais específicos do metamodelo *OdysseyProcess-FEX*. A descrição dos casos discrepantes encontra-se no Apêndice III. O conjunto de casos discrepantes não tem a intenção de cobrir todos os cenários possíveis de defeitos semânticos relacionados à inspeção de modelos de LPrS, mas os casos relativos aos requisitos de variabilidade de processos identificados nesta metodologia.

Com base nos casos discrepantes identificados e visando lidar com o desafio do número de questões envolvidas em um *checklist,* foram identificados 34 itens para apoiar a verificação semântica de modelos de características de LPrS. Um número excessivo de questões pode lidar com retrabalho. Por outro lado, um *checklist* deve conter um conjunto de itens representativos para apoiar de fato a atividade de inspeção. Desta forma, os casos discrepantes foram compilados em 34 itens, levando em consideração que alguns cenários identificados são similares e podem ser cobertos por uma mesma questão, mais genérica e aplicável a um grupo de situações (Tabela 4.17). Os itens foram distribuídos em três grupos de verificação: (1) 12 itens no grupo Verificação de Elementos de Processos; (2) 14 itens no grupo de Verificação de

Relacionamentos; e (3) 8 itens no grupo de Verificação de Regras de Composição. Os itens relacionados à variabilidade e opcionalidade são propriedades transversais e encontram-se presentes nas questões dos três grupos.

Tabela 4.17- *PVMCheck* - *Checklist* para Inspeção do Modelo de Caraterísticas de LPrS

PVMCheck – *Process Variability Modeling Checking*	
#	**Item de verificação: Taxonomia – Elementos de Processo**
1	Todas as características do modelo foram descritas com clareza?
2	Existe apenas informação necessária e suficiente descrevendo cada elemento de processo do modelo, ou seja, não tem informação desnecessária?
3	Existe algum elemento de processo, que embora correto, está fora do escopo do domínio?
4	Existe algum elemento de processo do escopo do domínio que foi omitido no modelo?
5	Existe algum atributo de descrição de um elemento de processo que foi omitido no modelo ou está incompleto?
6	Existem diferentes elementos de processo no modelo que representam o mesmo elemento do domínio?
7	Existe algum elemento de processo que foi modelado de forma isolada dos outros elementos?
8	O conjunto de habilidades e competências de cada elemento de processo classificado como papel foi claramente descrito e está em conformidade com a descrição do domínio?
9	A classificação de opcionalidade e variabilidade de cada elemento de processo está em conformidade com a descrição do domínio?
10	Existe algum comentário incorreto associado a um elemento do domínio?
11	Existe algum comentário que não provê nenhuma informação relevante no domínio?
12	Existe algum comentário que deveria ser representado como um elemento de processo ou um atributo?
#	**Item de verificação: Relacionamentos**
13	Existe algum relacionamento entre elementos de processos do domínio que não foi incluído no modelo?
14	Existe algum relacionamento entre elementos de processos do modelo que não está em conformidade com a descrição do domínio?
PVMCheck – *Process Variability Modeling Checking*	
#	**Item de verificação: Relacionamentos**
15	Todos os relacionamentos e seus atributos no modelo estão claramente descritos e em conformidade com a descrição do domínio?
16	Todos os relacionamentos de agregação e composição corretamente representados no modelo?
17	Todas as configurações do domínio que podem ser representadas por relacionamentos alternativos (relações entre ponto de variação e variantes) estão corretamente modeladas com os elementos corretamente conectados e a classificação de variabilidade corretamente definida?

18	De acordo com o domínio, todas as cardinalidades (valores mínimo e máximo) dos pontos de variação estão corretamente representadas no modelo?
19	A classificação de opcionalidade de um relacionamento está em conformidade com o tipo de participação da relação estabelecida no domínio?
20	Existe algum relacionamento alternativo em que a categoria do ponto de variação é diferente da categoria das suas variantes?
21	Existe algum relacionamento em que um a origem ou o destino não possui uma representação adequada no modelo?
22	Existe algum papel, produto de trabalho ou ferramenta não associada a uma unidade de trabalho correspondente?
23	A classificação de opcionalidade de cada relação de acordo com a descrição do domínio?
24	Existe algum papel em que sua responsabilidade não está em conformidade com a participação na execução da unidade de trabalho estabelecida no domínio?
25	Existe alguma unidade de trabalho sem produto de trabalho declarado como saída?
26	Os produtos de trabalho associados a uma unidade de trabalho estão corretamente classificados como entrada, saída ou modificados?
#	**Item de verificação: Regras de Composição**
27	Todas as regras de composição do domínio foram representadas no modelo?
28	Existe alguma regra de composição ou expressão no modelo que está fora do escopo do domínio?
29	Todas as regras de composição do domínio estão claramente descritas e em conformidade com a descrição do domínio?
30	Existe alguma regra de composição inclusiva que foi modelada de forma incorreta como uma regra de composição exclusiva (ou vice-versa)?
31	Existe alguma regra de composição inconsistente no modelo?
32	Existe alguma regra de composição no modelo cujo antecedente é representado como consequente (ou vice-versa)?
33	Todas as regras de composição inclusivas estão em conformidade com as classificações de mandatoriedade dos elementos envolvidos?
34	Existe alguma regra de composição exclusiva cujo consequente possui um elemento de processo classificado como mandatório no domínio?

4.5.1.2. Process Component Variability Modeling Checking (PCVMCheck)

O *checklist* para verificação da etapa de Projeto do Domínio de uma LPrS consiste em dois passos: (1) avaliação sintática e semântica de conteúdo e estrutura de cada componente de forma completa, numa perspectiva de uma abordagem DPBC (Tabela 4.18); e (2) avaliação da arquitetura da LPrS gerada (Tabela 4.19). Para cada item do modelo avaliado, segundo os itens de verificação descritos nas tabelas abaixo (Tabela 4.18 e Tabela 4.19), deve ser fornecido o resultado da avaliação como: Sim (S) – totalmente atendido, Não (N) – completamente discordante, Parcial (P) – parcialmente atendido, ou Não Aplicável (N.A.). A opção N.A. deve ser indicada se for

considerado que não é possível aplicar o item para avaliar o modelo em questão. Para cada opção Não ou Parcial marcada, justificar descrevendo o motivo da opção assinalada e indicando uma sugestão de ação corretiva. Algumas perguntas adicionais ao item descrito podem auxiliar na descrição do problema encontrado. De forma geral, os elementos do domínio que indicam o motivo devem ser citados na justificativa.

Tal *checklist* foi desenvolvido com base nos laudos de verificação da abordagem de BARRETO (2011) e com o corpo de conhecimento adquirido e trabalhado no *checklist* da visão de características da etapa de análise do domínio (PVMCheck) (TEIXEIRA *et al*., 2015).

Tabela 4.18- *PCVMCheck* Passo 1: *Checklist* para Inspeção de Componentes de Processos de Software

***Checklist* para Inspeção de Componentes de Processos de Software**			
Itens para verificação		**Resultado**	**Justificativa e Ação corretiva**
1	**Avaliação da relevância de um componente** O componente de processo definido, avaliando seu conteúdo e propósito, pode ser considerado relevante e adequado para ser reutilizado em diferentes definições de processos?		
2	**Avaliação da complexidade de um componente** O componente de processo definido, avaliando sua granularidade e número de ações envolvidas, pode ser considerado adequado em termos do grau de complexidade para ser reutilizado em diferentes definições de processos?		
3	**Avaliação da descrição de um componente** As informações descritas no componente de processo estão completas e corretas, onde cada componente de processo possui: 1. Nome, descrição e propósito definidos com informações suficientemente adequadas para o entendimento de seus objetivos e resultados esperados? 2. Critérios de entrada descrevendo claramente e adequadamente o que deve acontecer para que o componente possa começar a ser executado? 3. Critérios de saída descrevendo claramente e adequadamente as condições necessárias para que o componente termine sua execução? 4. Todas as atividades necessárias para completar seu propósito e entregar os resultados definidos? 5. Todos os produtos de entrada necessários definidos e representados como interfaces de dados requeridas? 6. Todos os resultados esperados definidos e representados como interfaces de dados providas?		
4	**Avaliação das Classificações de um Componente** O componente foi corretamente classificado quando a: 1. opcionalidade (opcional/mandatório)? 2. variabilidade (invariante/ponto de variação/variante)? 3. tipo de estrutura interna (**simples** – não possui outros componentes como elementos internos /**composto** - possui outros componentes como elementos internos)? 4. tipo de variação interna (**abstrato** - representa parte do processo que deve sofrer variação /**concreto** - representa parte do processo que não deve sofrer variação)?		
4.1	**Avaliação secundária para a Classificação de Opcionalidade de um**		

Checklist para Inspeção de Componentes de Processos de Software			
Itens para verificação		**Resultado**	**Justificativa e Ação corretiva**
	Componente 1. Caso o componente tenha sido classificado como **opcional**, os produtos gerados pelo componente são opcionais no domínio? 2. Caso o componente tenha sido classificado como **mandatório** e produza produtos de trabalho (excluídos casos de componentes que provocam transformações apenas), pelo menos um deles é mandatório no domínio?		
4.2	**Avaliação secundária para a Classificação de Variablidade de um Componente** Todas as possíveis formas de realização de um **ponto de variação** foram identificadas como variantes e não há nenhum componente variante sem estar agrupado como componente interno do ponto de variação correspondente?		
5	**Avaliação da Arquitetura Interna de um Componente - Papel** Todos os papéis participantes da execução do componente foram devidamente representados com suas responsabilidades atribuídas adequadamente, sendo um dos papéis o responsável pelo resultado e classificado como obrigatório dentro do componente?		
6	**Avaliação da Arquitetura Interna de um Componente - Ferramenta** As ferramentas de apoio à execução do componente, se existentes, foram corretamente definidas e são adequadas?		
7	**Avaliação da Arquitetura Interna de um Componente – Fluxo de Execução** O fluxo de execução das unidades de trabalho e componentes internos foi definido e possui ao menos um caminho de execução que leva do elemento inicial ao elemento final da arquitetura, com a indicação clara por onde a execução deve ser iniciada e terminada (fluxo de controle dentro do componente)?		
8	**Avaliação da Arquitetura Interna de um Componente – Fluxo de Dados** Todos os produtos de trabalho internos ao componente foram corretamente definidos e representam adequadamente insumos e produtos gerados e/ou modificados pelas unidades de trabalho/componentes internos do componente (o que é requerido por um é produzido por outro, em casos que a entrada não é requerida externamente ou que representa um elemento sem entradas requeridas)?		
9	**Avaliação de Restrições de Dependências e Mútua Exclusividade de um Componente** 1. Em cada componente de processo que possui relações de dependência, com outros componentes, descritas por regras de composição inclusivas no modelo de características apresenta a descrição de pré-condição com os componentes dos quais é dependente? 2. Em cada componente de processo que possui relações de mútua exclusividade, com outros componentes, descritas por regras de composição exclusivas no modelo de características apresenta a descrição de restrição com os componentes conflitantes e não estão agrupados em um mesmo componente?		
10. **Avaliação de Dados Complementares de um Componente**			
10.1	Foram selecionadas medidas pertinentes para o componente de processo? (*Opcional*) 1. Caso medidas tenham sido definidas, estas podem ser coletadas por meio da execução do componente? 2. Caso o componente de processo represente um subprocesso com potencial de ser controlado estatisticamente, foram definidas medidas adequadas, inclusive no nível de granularidade adequado, para possibilitar isso?		
10.2	Foram adicionados dados de utilização para o componente de processo? (*Opcional*)		

Checklist para Inspeção de Componentes de Processos de Software			
Itens para verificação		**Resultado**	**Justificativa e Ação corretiva**
	a. Caso dados de utilização tenham sido descritos, estes foram associados ao contexto em que foram coletados? b. A cada dado são associados os tipos que representam, por exemplo, lição aprendida, boas práticas, desvios, grau de reusabilidade, necessidade de adaptação?		
10.3	Foram adicionados orientação de uso para o componente de processo? (*Opcional*) a. Caso tenham sido descritas, os cenários a serem aplicadas foram definidos?		
10.4	Foram adicionados *templates* aos produtos de trabalho associados ao componente de processo? (*Opcional*) a. Caso tenham sido associados *templates*, arquivos demonstrativos foram adicionados?		
10.5	Foram adicionados agentes aos papéis definidos para o componente de processo? (*Opcional*) a. Caso tenham sido definidos, as informações associados ao agente foram descritas como nome, histórico, habilidades?		

Tabela 4.19- *PCVMCheck* Passo 2: *Checklist* para Inspeção da Arquitetura da LPrS

Checklist para Inspeção do Modelo de Componentes de Processos de Software ou Arquitetura da LPrS			
Itens para verificação		**Resultado**	**Justificativa e Ação corretiva**
1	**Avaliação da descrição de uma LPrS** 1. O nome da LPrS e sua descrição foram claramente descritos, fornecendo informações suficientemente adequadas para entendimento de seu propósito, objetivos e resultados esperados? 2. Tal descrição facilita sua recuperação para novas aplicações? 3. A LPrS definida tende a atender aos objetivos para os quais foi definida, com componentes de processo suficientes para atendê-los?		
2	**Avaliação da completude da arquitetura da LPrS com relação ao modelo de características** Todos os elementos de processos do modelo de características foram mapeados para o modelo de componentes, mantendo os relacionamentos já definidos?		
3	**Avaliação da arquitetura da LPrS - Fluxo de Controle** O sequenciamento definido na arquitetura da LPrS está adequado, ou seja, a cada dois componentes uma interface de controle é corretamente estabelecida (tipo de conexão entre componentes de processo é adequado)?		
4	**Avaliação da arquitetura da LPrS – Fluxo de Controle** As interfaces de controle entre componentes de processo opcionais estão classificadas como opcionais e na ausência desses componentes, o sequenciamento de execução da LPrS não é impactado com fluxos alternativos descritos?		
5	**Avaliação da arquitetura da LPrS – Fluxo de Dados** As interfaces de dados entre os componentes de processo da arquitetura da LPrS estão sendo respeitadas (todas as interfaces requeridas possuem produtos de trabalho gerados externamente ou por outro componente)?		

Ainda que a verificação sintática seja de extrema importância para a detecção de defeitos que podem gerar impactos na execução dos processos gerados, a verificação semântica complementa com a análise de aderência do domínio a seus

propósitos e objetivos estratégicos. O desenvolvimento de técnicas de inspeção baseadas em *checklists* foi proposto com o intuito de apoiar os dois tipos de verificação. Os itens que trabalham verificação sintática tem maior possibilidade de automatização, enquanto que a verificação semântica tem caráter de execução visual conduzida de forma manual por especialista no domínio. Aspectos de visualização podem ser trabalhados para apoiar essa análise e desenvolvidos em trabalhos futuros.

4.5.2. Estudo de Viabilidade de PVMCheck

O estudo de viabilidade da técnica de inspeção da modelagem de características de processos de software baseada em *checklist* (PVMCheck) proposta foi realizado em duas etapas: uma prova de conceito e um *quasi*-experimento (TEIXEIRA *et al*., 2015).

A prova de conceito foi conduzida com dois membros do grupo de Engenharia de Software Experimental (COPPE/UFRJ). Ambos os participantes apresentavam como perfil experiência relevante em desenvolver e aplicar processos de software, ainda que um dos participantes tenha mais tempo de experiência do que outro nesse contexto. Ambos apresentam conhecimento limitado sobre reutilização e modelagem de variabilidade. Os dois participantes foram convidados a realizar uma inspeção individual no mesmo modelo de uma LPrS pela aplicação da técnica proposta.

Como resultado das análises, observou-se que o participante com maior experiência com inspeções, tendo aplicado em vários projetos na indústria (P1, 5 defeitos, tempo de 120 min) detectou menos defeitos do que o outro participante, que só realizou inspeções em ambiente acadêmico por meio de atividades realizadas em disciplinas, (P2, 10 defeitos, tempo de 127 minutos), com tempos similares gastos para a realização da atividade de inspeção.

Uma das razões que podem explicar esse resultado inesperado pode ser encontrada nas respostas coletadas com o formulário pós-estudo, em que P2 considerou que a técnica contribui em seu desempenho e seu aprendizado para neste tipo de inspeção, ainda que tenha apontado a necessidade de melhorias em itens do *checklist* e no treinamento oferecido. Já P1 considerou que seu desempenho foi prejudicado pelo conteúdo insuficiente transmitido durante a preparação com relação às tecnologias envolvidas. Baseado nesses primeiros resultados, foram realizadas

melhorias na seção de treinamento; alterações no formulário de caracterização para melhor capturar o perfil dos participantes; ampliações na lista de defeitos no domínio usado, com acréscimo de defeitos encontrados pelos participantes que não tinham sido descritos pelos pesquisadores; e alterações na descrição de alguns itens do *checklist* foram realizadas, com o acréscimo de itens específicos para tratar variabilidade e opcionalidade, inclusão de explicações de apoio a análise de alguns questões, inclusão de avaliações de atributos específicos de características de unidades de trabalho, descrição de questões observando o conceito de opcionalidade aplicado aos relacionamentos, dentre outras alterações. A esta nova versão do *checklist,* que passou a conter mais oito questões com as alterações realizadas, foi então aplicada um *quasi*-experimento descrito nas subseções a seguir.

4.5.2.1. Planejamento do Estudo

- **Objetivo Geral**: Este estudo visa compreender a viabilidade de aplicação da técnica de inspeção baseada em *checklist* (PVMCheck) em modelos de variabilidades de processos de software, mais especificamente, modelos de características de LPrS.

  Baseado em (BASILI *et al*., 1994), o objetivo do estudo foi definido como:

  Analisar a inspeção de modelos de LPrS usando PVMCheck

  Com o propósito de caracterizar

  Com respeito à efetividade (defeitos identificados/total de defeitos esperados) e eficiência (defeitos identificados/ tempo) em detectar defeitos e à opinião dos inspetores

  Do ponto de vista de pesquisadores de Engenharia de Software

  No contexto de desenvolvedores de software (representados por graduandos e pós-graduandos de um curso de software do NES/IFluminense e do grupo de reutilização da COPPE/UFRJ) inspecionando modelos de LPrS em dois domínios diferentes. O primeiro domínio (D1) apresenta um domínio mais complexo com a combinação de Scrum e OpenUP. O segundo domínio (D2) consiste em um modelo mais simples, combinando OpenUP Basic e OpenUP.

- **Questões:**
  - **Questão (1):** *Qual a efetividade de cada participante em inspecionar modelos de LPrS usando PVMCheck?*

- **Métrica:**
  - **Efetividade =** número de defeitos identificados/número total de defeitos esperados
- **Questão (2):** *Qual a eficiência de cada participante em inspecionar modelos de LPrS usando PVMCheck?*
- **Métrica:**
  - **Eficiência =** número de defeitos identificados/ tempo dedicado à atividade de inspeção.

- **Hipótese:** Considerando a restrição de aplicar uma única tarefa a cada participante e a influência reduzida da experiência do inspetor na sua efetividade e eficiência ao usar um *checklist* (ou seja, espera-se analisar se a utilidade do *checklist* é de fato independente do nível de experiência do inspetor), as duas hipóteses nulas a seguir foram estabelecidas:
  - **Hipótese Nula (H01):** Não existe equivalência entre a efetividade de inspeções de LPrS realizadas pelo grupo composto por participantes com maior experiência do grupo composto por participantes com menor experiência.
  - **Hipótese Nula (H02):** Não existe equivalência entre a eficiência de inspeções de LPrS realizadas pelo grupo composto por participantes com maior experiência do grupo composto por participantes com menor experiência.
- **Seleção do contexto:** O estudo é executado em ambiente acadêmico por graduandos e pós-graduandos de um curso de software do NES/IFluminense e do grupo de reutilização da COPPE/UFRJ**.** Para a execução do estudo, foram projetados como instrumentos duas especificações de domínios com seus respectivos modelos de LPrS de Planejamento de Projetos de Software usando a notação *OdysseyProcess-FEX* (nível de análise – modelo de características). O primeiro domínio (D1) apresenta um domínio mais complexo com a combinação de *Scrum* e *OpenUP*. O segundo domínio (D2) consiste em um modelo mais simples, combinando *OpenUP Basic* e *OpenUP*.
- **Seleção dos participantes:** Foram selecionados dezoito participantes, sendo 17 estudantes do curso do NES/IFluminense e um estudante do grupo de

reutilização da COPPE/UFRJ. Todos os participantes preencheram formulários de consentimento e caracterização. No entanto, ao final do estudo, observou-se que apenas 13 participantes preencheram todas as etapas de maneira adequada para análise, como, por exemplo com a identificação correta nos formulários preenchidos. Desta forma, a amostra de participantes final contou com esses 13 participantes O nível de experiência foi definido pela análise das respostas do formulário de caracterização, que possibilitou analisar o perfil do participante segundo alguns atributos: (A) experiência acadêmica; (B) experiência com processos de software (implantando, definindo, gerenciando ou revisando); (C) experiência com reutilização de software (aplicando de maneira *ad-hoc*, usando ou definindo LPS, aplicando a modelagem de características); e (D) experiência com inspeção de software de forma geral. Esta análise dos perfis permitiu distribuir os participantes em dois grupos: (A) mais experientes e (B) menos experientes. Após a divisão dos participantes, os domínios foram distribuídos, aleatoriamente, dentre de cada grupo definido.

- **Variáveis:**

  Variáveis independentes são aquelas que podem ser controladas e manipuladas no estudo experimental e que têm impacto nos valores das variáveis dependentes, representando a causa que afeta o resultado do processo de experimentação (WOHLIN *et al*., 2000). O efeito do(s) tratamento(s) é medido pelas variáveis dependentes (WOHLIN *et al*., 2000), que analisam o resultado do estudo.

  - **Variáveis Independentes:** domínios de processo de software representado por modelos de características usando a notação *OdysseyProcess-FEX*, experiência dos participantes em reutilização de processos de software e conhecimento prévio dos participantes do domínio usado no estudo.
  - **Variáveis Dependentes***:* número de defeitos identificados, número de defeitos esperados, tempo (medido em minutos), efetividade e eficiência.

- **Validade do estudo:** Algumas ameaças à validade do estudo foram identificadas, como o número reduzido de participantes e com experiência

limitada em inspeção e em reutilização de processos de software. Além disso, o número limitado de domínios a serem inspecionados é outro aspecto que deve ser apontado. A lista de defeitos esperados construída pelos pesquisadores da proposta não conseguiu ser completa. Desta forma, a ausência dessa completude na lista de defeitos de forma pré-estabelecida à execução do estudo pode ser considerada uma ameaça à validade. Alguns defeitos foram considerados como defeitos não conhecidos, sendo detectados durante a aplicação do *quasi*-experimento, e impactando diretamente no cálculo da efetividade.

Este estudo é classificado como *in vitro*, executado de forma síncrona com todos os participantes utilizando os mesmos recursos (artefatos impressos) para realizar a inspeção. Como ameaça à validade interna, pode-se considerar o fato do mesmo grupo que desenvolveu a técnica ter aplicado o estudo, mesmo baseado em práticas experimentais. Como ameaça à validade externa pode-se considerar a definição da amostra de participantes, que foi escolhida por conveniência e disponibilidades destes, típico em *quasi*-experimentos (DE MELLO E TRAVASSOS, 2013).

4.5.2.2. Execução do Estudo

Preparação

O convite para participação do estudo foi enviado por e-mail com uma breve explicação do contexto e formulário de caracterização anexado. O preenchimento do formulário foi realizado anteriormente ao dia de aplicação do estudo para análise do perfil de cada participante. Após confirmação de participação, um documento contendo uma leitura introdutória sobre o metamodelo e a notação *OdysseyProcess-FEX* para o nível de análise, modelo de características, foi enviado aos participantes, também por e-mail, com o intuito de fornecer um conhecimento prévio sobre a modelagem utilizada no estudo.

Conforme planejado, baseado na análise da caracterização preenchida por cada participante, estes foram distribuídos em dois grupos: (A) mais experientes e (B) menos experientes. Foi observado que esses grupos foram significativamente diferentes (*Student t-test, p-value=0.99*). Essa composição é especialmente relevante

no contexto das hipóteses alternativas investigadas neste estudo (equivalência de efetividade e eficiência entre os grupos). Os domínios foram então distribuídos aleatoriamente dentro de cada grupo.

No dia do estudo, um treinamento foi fornecido, contendo:

- Uma apresentação sobre o contexto de realização do estudo, incluindo uma apresentação sobre inspeção de software e sobre a modelagem utilizada (metamodelo e notação *OdysseyProcess-FEX* – nível análise: modelo de características).
- Uma apresentação do *checklist* proposto a ser aplicado na inspeção.

Operação

A cada participante foi entregue um conjunto de documentos, contendo o modelo de características de um dos domínios (D1 ou D2) e material de apoio à aplicação do estudo: documento com o *checklist* a ser utilizado e formulário para preenchimento do resultado da inspeção com a descrição das discrepâncias identificadas. Os participantes analisaram o modelo de características entregue e realizaram a inspeção utilizando o *checklist* fornecido. Ao final da atividade, os participantes preencheram um formulário de avaliação pós-estudo contendo itens objetivos (e.g., sim, não, parcialmente) e um item subjetivo (e.g., justificativa das respostas).

As discrepâncias reportadas pelos 13 participantes finais foram revisadas por dois pesquisadores que classificaram cada item como defeito ou falso positivo. O tipo do defeito reportado foi revisado e reclassificado, quando necessário. O resultado da inspeção e das respostas do formulário de avaliação é que efetivamente foram considerados para avaliar a viabilidade da técnica proposta.

4.5.2.3. Análise dos resultados obtidos

A Tabela 4.20 apresenta uma sumarização dos resultados obtidos pelas análises. Como pode ser observado, nenhum dos participantes atingiu o valor máximo de efetividade (igual a 1) e três participantes não identificaram nenhum defeito. Nesta tabela, também, é possível notar a distribuição aleatória dos domínios entre os participantes de cada grupo. O domínio mais complexo (D1) foi inspecionado por quatro participantes (1 participante do grupo A e 3 participantes do grupo B).

Enquanto o domínio mais simples foi inspecionado por nove participantes (5 participantes do grupo A e 4 participantes do grupo B). A distribuição não foi mais equilibrada pelo não aproveitamento de cinco inspeções do grupo de 17 participantes iniciais, que não preencheram adequadamente o material do estudo.

Tabela 4.20 – Sumarização dos resultados de análise

Grupo	Participante	Domínio	Nível de Experiência	Tempo	#Defeitos	Eficiência	Efetividade
A	JCC	D2	0.604	90	0	0	0
	GCBC	D2	0.479	91	7	0.077	0.5
	RMS	D2	0.458	53	2	0.038	0.143
	RSM	D1	0.396	88	4	0.045	0.308
	RNG	D2	0.271	98	0	0	0
	ROM	D2	0.271	54	7	0.13	0.5
B	JGA	D1	0.208	68	0	0	0
	RVS	D1	0.188	50	2	0.04	0.154
	TCZ	D2	0.167	79	4	0.051	0.286
	TSC	D2	0.167	74	1	0.014	0.071
	LPD	D2	0.167	74	2	0.027	0.143
	LFS	D1	0.146	55	4	0.073	0.308
	PAS	D2	0.125	92	3	0.033	0.214

A distribuição da efetividade e da eficiência observada foi normal (*Saphiro-Wilk test*) e nenhum *outlier* foi detectado em nenhuma distribuição (Figura 4.22). No entanto, a variância do Grupo A para ambas as variáveis foi significativamente diferente do Grupo B. A complexidade do domínio também provou influência sobre os resultados. Com a aplicação do teste de Equivalência com um limite de não-similaridade de 10% (uso da ferramenta JMP[16]), não foi observado equivalência entre as distribuições de efetividade e eficiência nos grupos A e B. Desta forma, não foi possível rejeitar as hipóteses H01 e H02.

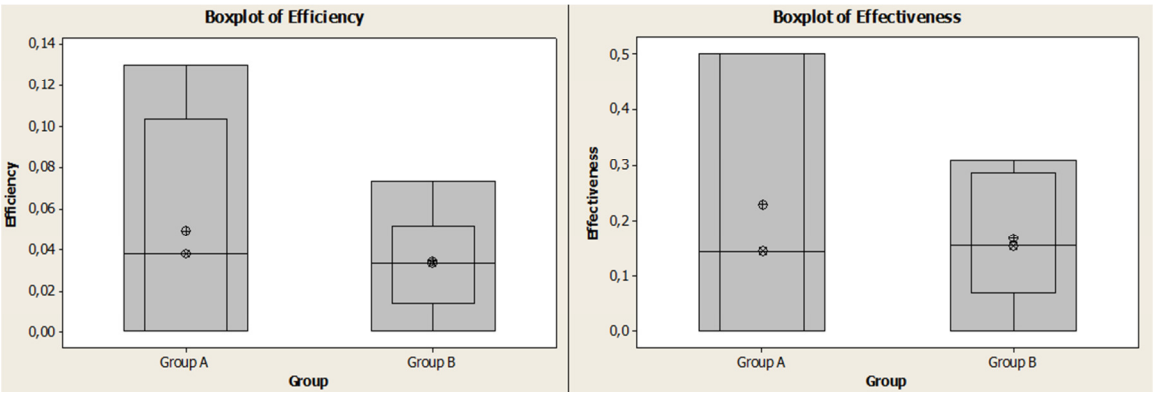


Figura 4.22 – Distribuição de Eficiência e Efetividade nos grupos A e B

[16] JPM - "http://www.jmp.com/"

Nenhum dos participantes foi capaz de identificar todos os defeitos conhecidos (previamente estabelecidos pelos pesquisadores desta técnica), em ambos os domínios (69% em D1 e 71% em D2). Apenas um dos defeitos entre os nove detectados em D1 foi reportado por mais de um participante. Dos dez defeitos detectados em D2, oito foram detectados por mais de um participante.

Os resultados coletados dos formulários pós-estudo demonstram que a maioria dos participantes concorda, ou parcialmente concorda, que os instrumentos utilizados durante o estudo auxiliaram a realização das atividades. Tanto a notação *OdysseyProcess-FEX* quanto o PVMCheck foram apontados como de fácil entendimento. A maioria dos participantes concorda, ou parcialmente concorda, em aplicar PVMCheck em inspeções futuras, já que a técnica de fato auxiliou a detecção de defeitos. Em geral, os participantes não concordam que o tempo de inspeção é melhor ao aplicar o *checklist*. Cinco dos treze participantes sugeriram melhorias na sessão de treinamento e no *checklist* em si.

O número de defeitos identificados pelos participantes não foi tão próximo ao resultado esperado. O máximo de defeitos identificados por participante foi de 50% dos defeitos previamente definidos. Alguns pontos podem ser ter contribuído para esse resultado: (1) baixo nível de experiência dos participantes com o tema de reutilização de processos e modelagem de variabilidades; (2) complexidade de absorção do conhecimento da notação e do domínio utilizados; (3) número de questões do *checklist*; (4) descrição de itens do *checklist* que pode levar a problemas de interpretação ou falta de direcionamento para análise de algum aspecto importante na inspeção. Analisando esses pontos, melhorias no checklist foram realizadas, gerando uma nova versão que foi apresentada na Seção 4.5. Dentre as melhorias podem-se destacar reorganizações do conteúdo e a redução da quantidade de questões a serem aplicadas, evitando redundâncias e enfatizando aspectos inerentes de modelagem de variabilidades.

4.6. CONSIDERAÇÕES FINAIS

Este Capítulo completa a apresentação de elementos de apoio à metodologia *OdysseyProcessReuse*, definida no Capítulo 3. A modelagem de LPrS é apresentada contemplando dois níveis de abstração. Além disso, o metamodelo e notação

OdysseyProcess-FEX são descritos de forma a especificar os elementos principais envolvidos na modelagem de variabilidades dos artefatos de processos de software reutilizáveis. A modelagem proposta visa atender os requisitos de modelagem definidos na Seção 4.2.

Uma contribuição importante, quando comparado aos demais processos de EDPS e DPBC, é o apoio ao mapeamento dos requisitos do domínio até os componentes, considerando suas relações e, sobretudo, as variabilidades e opcionalidades dos artefatos envolvidos. As heurísticas de mapeamentos de características para componentes de processos foram apresentadas. Com a análise do exemplo explorado ao longo do capítulo, foi possível observar como a notação e as heurísticas podem apoiar na manutenção das propriedades principais envolvidas no desenvolvimento de LPrS, sendo estes dados relevantes para as decisões sobre a geração de elementos que compõem a arquitetura de LPrS. Tais elementos são gerados a partir do agrupamento de componentes de processos realizado com o apoio dos critérios definidos e apresentados na Seção 4.4.

Técnicas de inspeção baseadas em *checklists* foram desenvolvidas no contexto deste trabalho como forma de detectar defeitos no momento da definição da linha e evitar a propagação destes para momentos posteriores do desenvolvimento onde os custos de correção de desvios são maiores. A técnica denominada PVMCheck (*Process Variability Modeling Checklist*) (TEIXEIRA *et al*., 2015) foi proposta como forma de suportar a detecção semântica da anomalias nos modelos de variabilidades de LPrS, com foco na etapa de análise de domínio e modelos de características de LPrS. Um segundo *checklist* foi desenvolvido com base no conhecimento adquirido com a técnica PVMCheck (TEIXEIRA et al., 2015) e em laudos de verificação propostos no trabalho de BARRETO (2011), com foco na verificação de modelos de componentes da etapa de projeto do domínio.

No próximo Capítulo, a infraestrutura desenvolvida é apresentada, dentro do contexto do ambiente Odyssey (ODYSSEY, 2016). A existência desse ferramental de apoio é também um diferencial desta proposta.

A Tabela 4.21 apresenta um resumo comparativo das principais abordagens identificadas com a metodologia *OdysseyProcessReuse* proposta.

Tabela 4.21- Comparativo do apoio oferecido pelas abordagens de LPrS e a *OdysseyProcessReuse*

Publicação	Apoio às etapas da ELPrS (EDPS e EAPS)	Apoio à modelagem de variabilidade de LPrS	LPrS em diferentes níveis de abstração	Diretrizes para mapea-mento entre os artefatos	Uso de princípios de DPBC	Ferramen-tal apropriado	Técnicas de verifica-ção dos artefatos criados
Durán *et al*. (2004)	Parcial	Parcial	Não	Não	Não	Não	Não
Washizaki (2006a)(2006b)	Parcial	Parcial	Sim	Parcial	Não	Não	Não
Simidchieva *et al*. (2007)	Parcial	Parcial	Não	Não	Não	Não	Não
Thränert e Werner (2007)	Parcial	Parcial	Não	Não	Não	Não	Não
Armbrust *et al*. (2009)	Parcial	Parcial	Não	Não	Não	Não	Não
Martínez-Ruíz *et al*. (2009)(2011a) (2011b)	Sim	Parcial	Não	Não	Não	Não	Não
Ternité (2009)	Sim	Parcial	Não	Não	Não	Não	Não
Aleixo *et al*. (2010a)(2010b)	Sim	Parcial	Sim	Sim	Não	Sim	Não
Barreto *et al*. (2010) (2011)	Sim	Parcial	Sim	Sim	Parcial	Sim	Sim
Hurtado *et al*. (2011)(2013) Simmonds *et al*. (2013)	Sim	Parcial	Sim	Sim	Não	Sim	**Não**
Jafarinezhad e Ramsin (2012)	Parcial	Parcial	Não	Não	Não	Não	Não
Magdaleno *et al*. (2012)	Sim	Parcial	Sim	Não	Parcial	Sim	Não
Rouillé *et al*. (2012)	Sim	Parcial	Sim	Parcial	Não	Sim	Não
OdysseyProcess Reuse	Parcial	Sim	Sim	Sim	Sim	Sim	Sim

Capítulo 5 – Avaliações da Abordagem *OdysseyProcessReuse*

5.1. INTRODUÇÃO

O uso de estudos experimentais para avaliar o desenvolvimento de um método, processo, técnica, produto, modelo ou ferramenta pode apoiar a validação ou identificação de problemas presentes em novas tecnologias propostas. Esta tem sido uma das formas cientificamente aceita para obter indícios sobre a viabilidade e eficiência, enfim, os benefícios (ou não) esperados de alguma destas propostas.

Um conjunto de etapas para um processo de experimentação é proposto por WOHLIN *et al.* (2000), que inclui (1) a definição do estudo; (2) o planejamento do estudo; (3) a execução do estudo; (4) a análise e interpretação dos resultados; e (5) a apresentação dos resultados obtidos.

Três estratégias de estudo experimental podem ser adotadas, de acordo com WOHLIN *et al*. (2000), dependendo do propósito da avaliação e das condições para o estudo: (1) Investigação (*survey*), realizada principalmente por meio de entrevistas e questionários; (2) Estudos de caso, utilizados para o monitoramento pela coleta de dados para um propósito específico de estudo e análises estatísticas; e (3) Experimentos, que requerem alto nível de controle sobre as variáveis estudadas, em geral conduzidos em laboratório para confirmar teorias, avaliar a predição dos modelos ou validar medidas.

Neste trabalho, foram conduzidos estudos de viabilidade para avaliar partes da metodologia proposta. A proposta dos estudos foi avaliar as atividades que compreendem o processo de EDPS. Para isso, foram identificadas atividades estratégicas do processo e estudos foram sendo realizados com focos nas diferentes contribuições propostas, conforme apresentado no Capítulo 4.

Diferentes representações foram propostas na literatura para a modelagem de variabilidades de processos de software com particularidades que influenciam na modelagem do domínio e as difere entre si. Desta forma, a primeira avaliação apresentada consiste em um estudo comparativo entre duas propostas de modelagem para LPrS - a modelagem utilizando o metamodelo e notação *OdysseyProcess-FEX* propostos neste trabalho (Capítulo 4) e a modelagem proposta em BARRETO (2011). O

objetivo deste estudo consistiu em observar as diferenças de propriedades de uma notação para outra com relação à taxonomia, relacionamentos e restrições propostas pela representação. O estudo visou analisar a capacidade semântica e sintática da modelagem e a expressividade de representação dos artefatos de processos reutilizáveis e suas propriedades de variabilidade e opcionalidade, conceitos relevantes no cenário de LPrS. Um estudo comparativo foi escolhido como forma de avaliação pela complexidade envolvida com as modelagens analisadas, que tratam diferentes conceitos abordados de forma semanticamente diferente. Esta questão poderia implicar em uma dificuldade de transferência de conhecimento para múltiplos participantes, gerando um estudo altamente cansativo e sem garantias da absorção requerida para a aplicação do conhecimento. Isto poderia impactar, negativamente, os resultados do estudo. Outro ponto, é que um estudo envolvendo outros participantes, por ser altamente custoso, não iria permitir a análise tão detalhadas dos itens descritos na Seção 4.2.3.

A construção progressiva de uma LPrS com seus diferentes níveis de abstração necessita de um conjunto de procedimentos para realizar mapeamentos entre os artefatos, mantendo os rastros das propriedades de variabilidade, opcionalidade, relacionamentos, regras e restrições existentes de um artefato do domínio para o outro. Heurísticas de mapeamento entre o modelo e característica e o modelo de componentes propostos foram estabelecidas (Capítulo 4, Seção 4.3) e avaliadas por um estudo de viabilidade de aplicação. A efetividade e eficiência em gerar modelos de componentes de LPrS, o contexto de desenvolvedores de software, aplicando as heurísticas em um domínio de processos de software foram observadas (Seção 4.3.2).

Componentes de processo podem ser agrupados de forma a reduzir o acoplamento, aumentando a modularidade dentro do domínio. Critérios de agrupamento foram propostos com esse objetivo de melhor estruturar os componentes de uma LPrS (Capítulo 4, Seção 4.4). Tais critérios foram avaliados por meio de um estudo de viabilidade. Os critérios propostos podem ser entendidos como recomendações que precisam ser interpretadas e combinadas com a experiência do engenheiro de domínio, que é o responsável pela decisão dos agrupamentos mais adequados à modelagem do domínio. Desta forma, a intenção deste estudo foi analisar se os critérios propostos seriam utilizados de maneira apropriada e se de fato

auxiliariam a tomada de decisão para o agrupamento de componentes de processos. A análise foi subjetiva e inclui a observação: (i) se algum agrupamento era proposto pelo participante; (ii) se o agrupamento proposto era o mesmo do agrupamento feito pelo observador; e (iii) dos relatos pós-estudo fornecidos pelos participantes (Seção 4.4.4).

A atividade de definição de processos de software é uma atividade propensa a erros que podem ser custosos durante a execução do processo. Se isto já é evidenciado no cenário de desenvolvimento de software convencional, no cenário de reutilização, existe ainda o agravante de que um único artefato de domínio mal especificado pode levar à propagação de defeitos nos diversos produtos resultantes de uma linha. Desta forma, a detecção de defeitos no momento de definição de artefatos reutilizáveis é altamente indicada para evitar a propagação destes para momentos posteriores do desenvolvimento onde os custos de correção de desvios são maiores. O segundo estudo apresenta uma avaliação da técnica de inspeção da modelagem de características de processos de software (PVMCheck) proposta (Capítulo 4, Seção 4.5). O estudo teve por objetivo compreender a viabilidade de aplicação da técnica de inspeção baseada em *checklist* (PVMCheck) em modelos de variabilidades de processos de software, mais especificamente, modelos de características de LPrS (TEIXEIRA *et al*., 2015). Foram observados a efetividade (defeitos identificados/total de defeitos esperados) e eficiência (defeitos identificados/ tempo) alcançadas pela técnica (Capítulo 4, Seção 4.5.2).

Por último, um estudo de viabilidade de aplicação da abordagem de desenvolvimento de LPrS para apoiar a identificação de conhecimento a ser reutilizável foi conduzido. A proposta inicial do estudo era avaliar todas as atividades que compreendem o processo de Engenharia de Domínio descrito na metodologia apresentada no Capítulo 3. No entanto, a avaliação do processo como um todo é altamente custosa e exigiria um tempo considerável que foi avaliado como inviável dentro do contexto desta tese. Desta forma, etapas estratégicas foram identificadas e avaliadas nos quatro estudos já apresentados nas seções anteriores. Este último estudo apresentado visou avaliar o resultado da aplicação do processo de EDPS, verificando o apoio da abordagem à organização e representação de conhecimento de domínio de processo de software de forma a ser reutilizável. Desta forma, o objetivo consiste em analisar a capacidade de identificação de informações necessárias para

reutilização em um domínio modelado usando uma abordagem de LPrS construída por meio do método proposto. Tal estudo é apresentado neste Capítulo.

5.2. Estudo de Caso - Avaliação do Resultado de Aplicação da Abordagem de Desenvolvimento de LPrS Proposta

Neste estudo de caso, é avaliada a viabilidade de aplicação da abordagem de desenvolvimento de LPrS para apoiar a identificação de conhecimento a ser reutilizável. A proposta inicial do estudo era avaliar todas as atividades que compreendem o processo de EDPS descrito nesta abordagem. No entanto, a avaliação do processo como um todo é altamente custosa e exigiria um tempo considerável que foi considerado inviável dentro do contexto desta tese. Desta forma, etapas estratégicas foram identificadas e avaliadas em estudos complementares. Três estudos já foram apresentados nas seções anteriores com o intuito de avaliar a viabilidade de atividades estratégicas presentes no desenvolvimento e construção de uma LPrS usando a abordagem. O estudo desta seção visou avaliar o resultado da aplicação do processo de EDPS, verificando o apoio da abordagem à organização e representação de conhecimento de domínio de processo de software de forma a ser reutilizável. Desta forma, o objetivo consiste em analisar a capacidade de identificação de informações necessárias para reutilização em um domínio modelado usando uma abordagem de LPrS construída por meio do método proposto.

5.2.1. Planejamento do Estudo

- **Objetivo Geral**: Este estudo visa compreender a viabilidade de utilização de uma LPrS desenvolvida segundo a abordagem *OdysseyProcessReuse* para identificação de alternativas de definição de novos processos em um contexto específico.

  Baseado em (BASILI *et al*., 1994), o objetivo do estudo foi definido como:

  Analisar a Engenharia de Domínio de Processos de Software (EDPS) da abordagem *OdysseyProcessReuse* proposta

  Com o propósito de caracterizar

  Com respeito à capacidade de identificar informações específicas a serem reutilizadas

  Do ponto de vista de pesquisadores de Engenharia de Software

No contexto de gerentes de projetos analisando um domínio de processos para identificação de alternativas para definição de novos processos em ambiente acadêmico

- **Questões:**
    - **Questão (1):** *É possível identificar corretamente informações para reutilização em uma LPrS?*
        - *A análise de uma LPrS desenvolvida conforme a abordagem proposta auxilia na identificação correta de alternativas para a definição de novos processos?*
    - **Métrica:**
        - Avaliação da precisão da informação retornada pelo participante: avalia o quanto das respostas retornadas estão corretas (corresponde a informações reutilizáveis esperadas). É um indicador de exatidão.
            - Precisão: = |Número de Respostas Corretas Dadas| / | Número de Respostas Dadas|
        - Avaliação da cobertura da informação retornada pelo participante: avalia o quanto da resposta corresponde a informações reutilizáveis esperadas. É um indicador de eficácia e completude.
            - Cobertura: = | Número de Respostas Corretas Dadas| / | Número de Respostas Corretas Esperadas|
    - **Questão (2)**: *Qual o esforço de cada participante para responder cada tarefa?*
        - O esforço será avaliado pelo tempo gasto durante a execução das tarefas do estudo.
        - **Métrica**:
            - Tempo dedicado à atividade (expresso em minutos).
- **Tarefas:** O estudo foi planejado para a realização de um conjunto de tarefas a serem realizadas pelos participantes de forma a observar a capacidade dos modelos em prover as informações reutilizáveis de forma compreensível a um gerente de projeto. As tarefas planejadas para o estudo buscam verificar se o

participante consegue identificar as alternativas para definição de novos processos. As tarefas foram divididas em três grupos com diferentes níveis de complexidade. Essa estrutura foi baseada no trabalho de OLIVEIRA (2011). Os participantes não foram comunicados da existência e do objetivo desta divisão de tarefas em grupos, de forma a não influenciar a execução das atividades.

- **Tarefas Básicas -** consistem em tarefas mais simples, de forma a analisar elementos mais básicos e de observação mais direta. Se algum participante apresentar erro ou a não completude na resposta, deve ser observado se houve padrões do erro que podem indicar possíveis problemas no entendimento da abordagem ou das tarefas do experimento:
  - O Grupo de Medição é responsável por quais atividades do domínio?
    - Resposta Esperada: Todas.
    - Modelo 2
  - Os Usuário da Medição participam da execução de qual(is) atividade(s) e tarefa(s) do domínio?
    - Resposta Esperada:

      Atividades: Reportar e armazenar as análises de medição e suas variantes.

      Tarefas: Enviar o relatório de medição e análise aos interessados por e-mail e Realizar reunião com os interessados e apresentar resultados
    - Modelo 2, Modelo 6 e Modelo 8.
  - Quais métodos podem ser aplicados para planejamento da medição?
    - Resposta Esperada: GQM e GQ(I)M
    - Modelo 1
  - Esses métodos podem ser usados juntos na mesma definição de processo derivada a partir da linha? Por que e como isso é identificado no modelo?

- Resposta Esperada: Não, porque fazem parte de um ponto de variação com variantes excludentes entre si. Cardinalidade máxima do ponto de variação igual a um.
- Modelo 1

▪ Qual variante para reportar o resultado da análise deve ser selecionada para a utilização da ferramenta E-mail?

- Resposta Esperada: Armazenar os resultados das análises e reportá-los por e-mail
- Modelo 2 e Modelo 5.

○ **Tarefas Intermediárias** - tarefas que podem ser realizadas por meio de fatos extraídos da análise dos modelos:

▪ Qual o produto de trabalho mais requerido por todas as atividades do domínio?

- Resposta Esperada: Plano de Medição Organizacional
- Modelo 1 e Modelo 10.

▪ Do grupo de ferramentas definidas quais devem estar presentes em todas as definições de processos derivadas a partir da linha e quais não podem estar presentes na mesma definição de processo derivada a partir da linha?

- Resposta Esperada: Devem estar presentes em todas as definições as ferramentas: Processador de Texto, Planilha Eletrônica e SGBD. Não podem estar presentes na mesma definição: E-mail e Gerador de Apresentação
- Modelo 2

▪ Qual(is) é(são) o(s) produto(s) final(is) de processos a serem derivados a partir da LPrS?

- Resposta Esperada: Relatório de Medição e Análise Armazenado, E-mail, Apresentação, Agendamento da reunião, Feedback dos participantes.
- Modelo 1 e Modelo 10.

- Todos os processos derivados a partir da linha apresentam este(s) produto(s)? Em caso negativo, justifique e identifique qual(is) produto(s) deve(m) estar presente(s) em todo processo derivado e qual(is) tarefa(s) é(são) responsável(is) pela produção deste(s) produto(s).
    - Resposta Esperada: Apenas o Relatório de Medição e Análise Armazenado está presente em todos os processos derivados. Nos demais devem aparecer E-mail ou (Apresentação, Agendamento da reunião e Feedback dos participantes) dependendo da variante escolhida para reportar os resultados de análise da medição.

      Tarefa que produz o Relatório de Medição e Análise Armazenado é Armazenar os resultados da análise.
    - Modelo 1, Modelo 10, Modelo 6 e Modelo 8.
- Qual(is) componente(s) de processo não apresenta(m) variações na forma de execução, ou seja, qual(is) componente(s) é (são) executado(s) da forma como foi(foram) definido(s) sem nenhuma configuração adicional de definição?
    - Resposta Esperada: Componente ColetarArmazenarEAnalisarMedida e os componentes internos: ColetarEArmazenar Medida e AnalisarMedidasColetadas.
    - Modelo 10, Modelo 13.

Tarefa de Assimilação – consiste de tarefas mais difíceis e complexas, exigindo que o participante utilize o conhecimento prévio e o raciocínio de forma a obter o entendimento e interpretar as informações contidas nos instrumentos da avaliação:

- Medidas definidas associadas aos componentes representam dados que são coletados durante a execução de processos derivados a partir da linha. Tais medidas visam registrar informação de execução associada ao contexto que a gerou, podendo ser utilizada para apoiar a definição de novos processos.

Para selecionar a alternativa mais viável para reportar resultados de análise, o tempo gasto na divulgação pode ser um fator importante. Desta forma, considerando as medidas associadas aos componentes definidos, qual medida deveria ser analisada para apoiar a escolha da forma de reportar o resultado de medição mais apropriado para um projeto específico?

- Resposta esperada: SF_RPT_MED – Número de horas gastas pela equipe para realizar as atividades relacionadas à divulgação das análises e resultados de medição.
- Modelo 13.

- Selecione as alternativas do domínio mais adequadas para o cenário descrito abaixo, descrevendo como a atividade de Reportar e armazenar as análises de medição deve ser definida para este contexto, descrevendo tarefas com fluxo de execução, produtos de trabalho, papéis e ferramentas.

Cenário: Equipe com 10 membros localizados no mesmo escritório.

• Resposta esperada: Variante Armazenar os resultados de medição e reportá-los através de reunião.

As tarefas Armazenar os resultados das análises e Realizar reunião com os interessados e apresentar os resultados devem ser executadas sequencialmente. A primeira tarefa tem com produtos de trabalho de entrada relatório de medição e análise e plano de medição organizacional e produz o relatório de medição e análise armazenado. A segunda tarefa utiliza o produto de trabalho produzido pela primeira e produz três produtos: apresentação, reunião e *feedback*. O Grupo de Medição é responsável pela execução das duas tarefas. Na segunda tarefa os papéis: Alta direção, Usuário da Medição e Grupo de Processos da organização podem participar da sua execução de forma adicional. As ferramentas Processador de texto e Planilha

Eletrônica são usadas como suporte nas duas tarefas. E o gerado de apresentação apoia a execução da segunda tarefa.

- Modelo 1, Modelo 4, Modelo 5, Modelo 8 e Modelo 9.

- **Hipótese:**
  - **Hipótese Nula (H0):** É possível identificar corretamente os pontos de configuração e os pontos invariantes de um domínio de processos de software modelado pelas técnicas propostas na abordagem *OdysseyProcessReuse*.
    - **Hipótese Alternativa (H1):** Não é possível identificar corretamente os pontos de configuração e os pontos invariantes de um domínio de processos de software modelado pelas técnicas propostas na abordagem *OdysseyProcessReuse*.
- **Seleção do contexto:** O estudo é executado em ambiente acadêmico por pesquisadores de Engenharia de Software. O estudo coloca cada participante no papel de engenheiro de processo contratado para o grupo de medição de uma organização que precisa entender quais as possibilidades de definições de novos processos dentro de um domínio específico modelado. Para a execução do estudo, o domínio de Medição no Contexto Organizacional foi desenvolvido, utilizando o processo de EDPS da abordagem *OdysseyProcessReuse* e modelado com a notação *OdysseyProcess-FEX* nível de análise e projeto (modelo de características e modelo de componentes).
- **Seleção dos participantes:** O perfil escolhido focou em participantes com algum conhecimento e experiência na aplicação de processos de software e reutilização. A seleção buscou identificar participantes com conhecimento sobre temas como variabilidades de processos e modelagem de LPrS. O perfil foi analisado segundo alguns critérios: (A) experiência acadêmica; (B) experiência com processos de software (implantando, definindo, gerenciando ou revisando); (C) experiência com reutilização de software (aplicando de maneira *ad-hoc*, usando ou definindo componentes, LPS, ou outros artefatos reutilizáveis, aplicando modelagem de características, ou sendo responsável pela elaboração de ativos reutilizáveis ou pela definição da forma de se reutilizar software em alguns projetos); (D) experiência com reutilização de

processos de software, aplicando de maneira ad-hoc, usando ou definindo componentes e/ou LPrS, ou sendo responsável pela elaboração de ativos de processo reutilizáveis ou pela definição da forma de se reutilizar processos em alguns projetos); (E) experiência com o domínio. Do total de cinco participantes, três pesquisadores já desenvolveram ou tiveram envolvidos em parcerias de pesquisas sobre LPrS no cenário brasileiro.

- **Variáveis:**
    - **Variáveis Independentes:** domínio de processo de software representado pelo modelo de características e modelo de componentes de processos de software usando a notação *OdysseyProcess-FEX*, experiência dos participantes em reutilização de processos de software, conhecimento prévio dos participantes do domínio usado no estudo.
    - **Variáveis Dependentes***:* número de respostas dadas, número de respostas corretas dadas e o tempo para realização da atividade (medido em minutos).
- **Validade do estudo:** Algumas ameaças à validade do estudo foram identificadas como:
    - A utilização de apenas um domínio para realização do estudo. A aplicação em outros cenários e domínios poderia ter sido considerada.
    - Da forma como o conhecimento foi transmitido não se tem garantias do entendimento a contento dos critérios observados. Mesmo com a leitura introdutória e dos exemplos fornecidos, esse fator pode ter influenciado no entendimento parcial da abordagem pelo participante.
    - A ausência de um método comparativo pode ser considerado outro fator, o ideal seria a separação dos participantes em dois grupos (um grupo utilizando esta abordagem e outro utilizando outra modelagem).
    - A aplicação apenas em ambiente acadêmico.
    - O tamanho da amostra impacta a validade de conclusão, onde o número reduzido de participantes não possibilita uma análise estatística, e limita os resultados que podem ser considerados apenas indícios e não generalizáveis.

5.2.2. Execução do Estudo

Preparação

O convite para participação do estudo foi enviado por e-mail com breve explicação do contexto. Anexados foram enviados formulários de caracterização do participante e documento com explicações dos principais conceitos da abordagem envolvidos no estudo que consistem em uma leitura introdutória que deve ser realizada no pré-estudo.

- Para realização do estudo uma breve explicação do que deve ser realizado foi enviado ao participante, com o formulário de apoio à execução do estudo com o questionário a ser respondido.
- Após a entrega do questionário respondido, um formulário pós-estudo foi enviado para coletar as impressões do participante e realizar uma análise subjetiva da abordagem.

Operação

A cada participante foi entregue um conjunto de documentos contendo a modelagem da LPrS do domínio de Medição para o Contexto Organizacional e material de apoio à aplicação do estudo: documento com as questões a serem respondidas. O Apêndice VII apresenta a documentação entregue aos participantes.

Os participantes analisaram a LPrS entregue e realizaram a atividade de coleta de informação reutilizáveis na definição de novos processo, respondendo às questões estabelecidas.

Ao final da atividade, os participantes preencheram um formulário de avaliação pós-estudo contendo itens objetivos e subjetivos. O resultado da atividade e das respostas do formulário de avaliação é que efetivamente foram considerados para avaliar a viabilidade de aplicação da abordagem para identificação de informações reutilizáveis adequadas.

5.2.3. Análise dos perfis dos participantes

Inicialmente, o perfil de cada participante foi analisado segundo os critérios estabelecidos. Quanto ao grau acadêmico, três participantes (P3, P4 e P5) apresentaram nível de mestrado completo e estão com o doutorado em andamento, um participante (P2) apresenta nível de doutorado completo e um participante (P1) apresenta pós-doutorado completo. O tempo de atuação na indústria variou de 4 (P3 e

P5) a 16 (P2) anos e na academia variou de 4 (P3) a 8 (P1 e P5) anos. Para analisar os critérios de experiência em processos de software, reutilização de software, reutilização de processos de software foram utilizados os seguintes graus: 1 = nenhum; 2 = conhecimento teórico (estudei em aula ou em livro ou apliquei em projeto em sala de aula); 3 = participei em um ou mais projetos na indústria, mas sem ser o responsável; 4 = participei em até 03 oportunidades na indústria como responsável; e 5 = participei em mais de 03 oportunidades na indústria como responsável incluindo diferentes organizações e/ou diferente tipos de processos. Para cada grau foi atribuído um valor em escala de 0 a 1 para combinação e análise ponderada dos resultados (Grau 1, valor 0 atribuído; Grau 2, valor 0,25 atribuído; Grau 3, valor 0,5 atribuído; Grau 4, valor 0,75 atribuído; e Grau 5, valor 1 atribuído).

ID	Processo			
	Implant.	Defin.	Revis.	Gerenc.
P1	**4**	**4**	**4**	**3**
Número de Anos	2	3	3	2
Número de Projetos	3	5	5	3
P2	**4**	**4**	**5**	**3**
Número de Anos	7	6	10	0
Número de Projetos	15	8	30	0
P3	**3**	**2**	**2**	**2**
Número de Anos	1	0	0	0
Número de Projetos	1	0	0	0
P4	**3**	**3**	**3**	**3**
Número de Anos	2	3	3	3
Número de Projetos	3	1	1	1
P5	**3**	**3**	**2**	**2**
Número de Anos	2	2	0,5	0,5
Número de Projetos	3	3	1	1

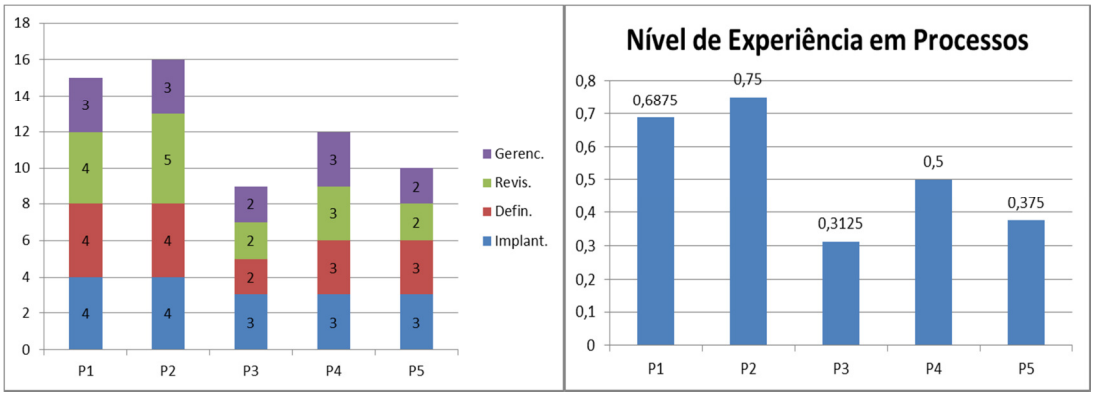


Figura 5.1 – Análise do perfil dos participantes com relação ao nível de experiência com processos de software

Com relação à atuação com processos de software na indústria, os participantes demonstraram algum grau de experiência com atividades de definição, implantação, revisão e gerenciamento de processos de software. O grau de experiência em cada atividade foi especificado segundo os graus definidos acima e

complementado com a descrição do tempo e número de processos em que a atuação foi exercida. Em geral, a atividade de gerenciamento apresentou menor tempo e número de projetos por parte dos participantes. O participante P2 apresenta maior nível de experiência trabalhando com processos de software, conforme pode ser visto na Figura 5.1.

O conhecimento sobre reutilização de processos de software foi adquirido com atuação na academia por desenvolvimento de trabalhos na área (Tabela 5.1).

Tabela 5.1 – Perfil dos Participantes quanto ao conhecimento em reutilização de processos de software

ID	Reutilização de processos de software						
	ad-hoc	LPrS	Componentes	Modelagem LPrS	Definição de componentes	Responsável pela elaboração de ativos ou da maneira de reutilizar	Nível de Exp. Total em Reutiliz. de Processos de Software
P1	**2**	**2**	**1**	**2**	**1**	**1**	-
	0,25	0,25	0	0,25	0	0	0,125
P2	**4**	**2**	**3**	**2**	**3**	**2**	-
	0,75	0,25	0,5	0,25	0,5	0,25	0,416666667
P3	**3**	**2**	**2**	**2**	**2**	**2**	-
	0,5	0,25	0,25	0,25	0,25	0,25	0,291666667
P4	**2**	**1**	**2**	**1**	**1**	**1**	-
	0,25	0	0,25	0	0	0	0,083333333
P5	**2**	**1**	**1**	**1**	**1**	**1**	-
	0,25	0	0	0	0	0	0,041666667

1 = nenhum
2 = conhecimento teórico (estudei em aula ou em livro ou apliquei em projeto em sala de aula)
3 = participei em um ou mais projetos na indústria, mas sem ser o responsável
4 = participei em até 03 oportunidades na indústria como responsável
5 = participei em mais de 03 oportunidades na indústria como responsável

Quando aplicado na indústria, a reutilização era realizada com aplicação de partes existentes de processos anteriores para definir novos processos, de maneira *ad-hoc*. De todos os participantes, o participante P2 apresentou o maior grau de experiência em reutilização de processos de software, conhecimento amadurecido com desenvolvimento de doutorado na área. O participante P5 apresentou menor grau de experiência em reutilização de processo de software, desconhecendo a técnica de LPrS. Três participantes têm experiência com a técnica de LPrS por conhecimento teórico, tanto da aplicação quanto modelagem. Alguns participantes aplicaram componentes (P2, P3 e P4), mas apenas o participante P4 afirmou ter definido componentes. Essa análise pode ser visualizada na Figura 5.2.

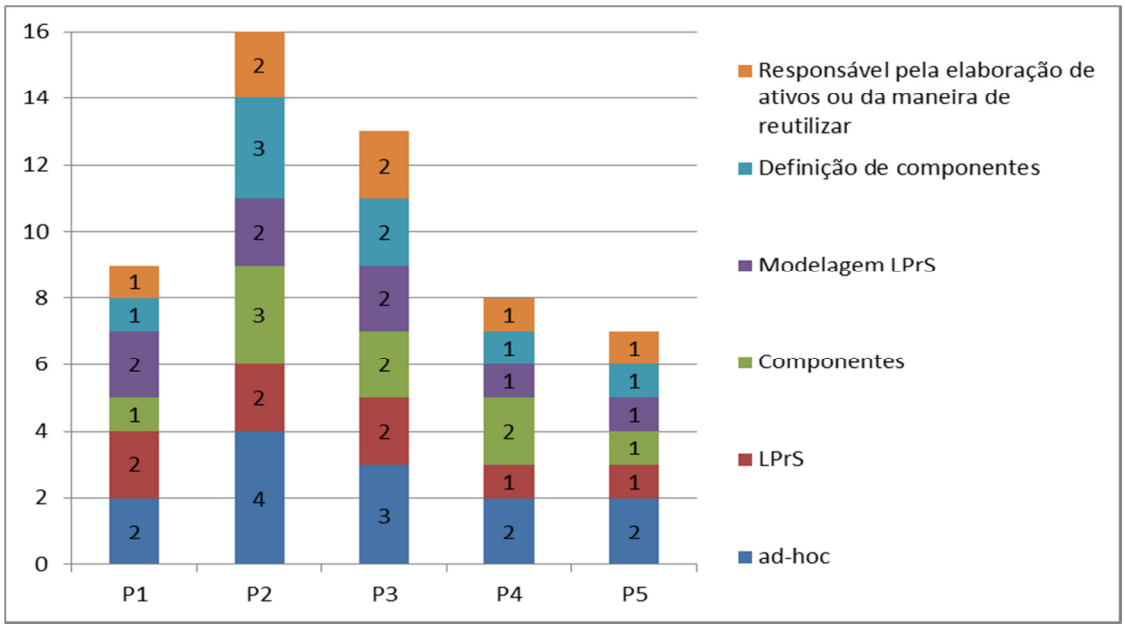


Figura 5.2 – Análise do perfil dos participantes com relação ao nível de experiência com reutilização de processos de software

Desta forma, pode-se observar que todos os participantes (P1, P2, P3, P4 e P5) possuíam experiência na indústria aplicando processos de software. Os participantes, ainda que não sejam especialistas, já aplicaram o processo de Medição, com exceção do participante P4 que indicou não ter aplicado o processo. Os três participantes com titulação de mestrado apresentaram os menores níveis de experiência de forma geral, considerando todos os critérios, na seguinte ordem P5 > P3 > P4. O nível final de experiência foi calculado como a soma dos valores de cada critério, ponderada pelo número de critérios estabelecidos, tendo todos os critérios o mesmo peso estabelecido (Tabela 5.2).

Tabela 5.2 – Perfil dos Participantes segundo todos os critérios analisados

ID	Acadêm.	Nível de Exp. em Reutiliz. de software	Nível de Exp. em Processos	Nível de Exp. em reutiliz. de processos de software	Nível de Exp. com processo de Medição	Nível de Experiência total em Processos	Nível de Experiência Geral
P1	1,00	0,43	0,69	0,13	0,50	0,44	0,55
P2	0,75	0,29	0,75	0,42	1,00	0,72	0,64
P3	0,50	0,32	0,31	0,29	0,50	0,37	0,39
P4	0,50	0,11	0,50	0,08	0,00	0,19	0,24
P5	0,50	0,21	0,38	0,04	1,00	0,47	0,43

O participante P2 apresentou o maior nível de experiência de forma geral. No entanto, tal participante apresentou o menor grau de experiência com reutilização de software. Dois participantes apresentaram maior familiaridade com o processo de Medição (P2 e P5). O participante P4 apresentou os menores valores de experiência,

principalmente nos tópicos de reutilização de software, reutilização de processos de software e nenhum conhecimento do processo de Medição. O participante P5 apresentou o menor grau de experiência com reutilização de processos de software e o segundo menor grau em experiência com processos de software.

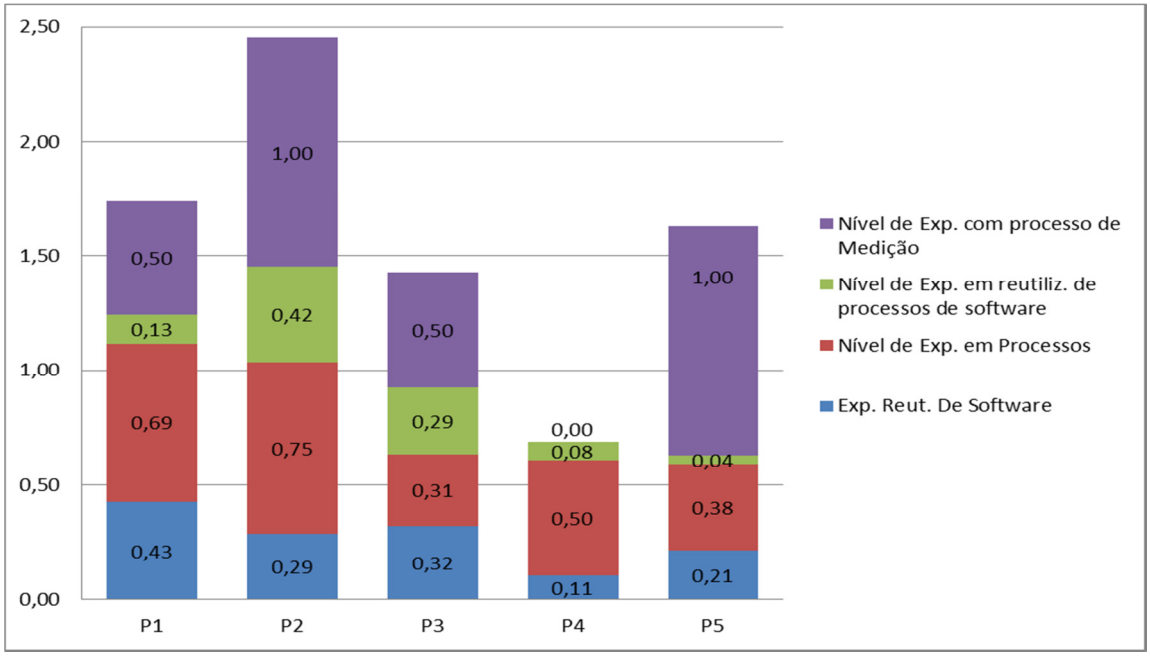


Figura 5.3 – Análise do perfil dos participantes pelos diferentes critérios

5.2.4. Análise dos resultados obtidos

Os resultados serão apresentados em grupos divididos pelos tipos de tarefas determinados: (1) básicas (tarefas 1 a 4); (2) intermediárias (tarefas 5 a 8); e (3) assimilação (tarefas 9 e 10). Para cada grupo serão observados: (i) o tempo utilizado pelo participante para responder cada tarefa; (ii) o número de respostas certas dadas por cada participante; (iii) a precisão das respostas dadas por cada participante para cada tarefa; e (iv) a cobertura das respostas dadas por cada participante para cada tarefa.

5.2.4.1. Análises dos resultados das tarefas básicas

Os primeiros resultados observados correspondem às quatro tarefas iniciais, denominadas de tarefas básicas. A Tabela 5.3 apresenta o número de respostas esperadas por tarefa.

Tabela 5.3 – Respostas Esperadas das Tarefas Básicas

Tarefas Básicas					
Tarefa	O Grupo de Medição é responsável por quais atividades do domínio?	Os Usuário da Medição participam da execução de qual(is) atividade(s) e tarefa(s) do domínio?	Quais métodos podem ser aplicados para planejamento da medição?	Esses métodos podem ser usados juntos na mesma definição de processo derivada a partir da linha? Por quê e como isso é identificado no modelo?	Qual variante para reportar o resultado da análise deve ser selecionada para a utilização da ferramenta E-mail?
Resposta Esperada	Todas: 1. Planejar Medição e suas duas variantes 2. Coletar e armazenar medidas 3. Analisar medidas coletadas 4. Reportar e armazenar as análises de medição e suas duas variantes	Atividades: Reportar e armazenar as análises de medição e suas variantes. Tarefas: Enviar o relatório de medição e análise aos interessados por e-mail e Realizar reunião com os interessados e apresentar resultados	GQM e GQ(I)M	Não, porque fazem parte de um ponto de variação com variantes excludentes entre si. Cardinalidade máxima do ponto de variação igual a um.	Armazenar os resultados das análises e reportá-los por e-mail
Número de respostas esperadas	8	5	2	2	1

Pode-se observar que os participantes, em sua maioria, forneceram respostas corretas às tarefas enumeradas. Desta forma, apesar da cobertura não ter sido completa, a precisão foi completa em todos os casos (Figura 5.4).

	Tarefa 1	Tarefa 2	Tarefa 3	Tarefa 4	Precisão
P1	**4/4**	**1/1**	**4/4**	**1/1**	10 /10
P2	**4/4**	**4/4**	**4/4**	**1/1**	13/13
P3	**4/4**	**5/5**	**4/4**	**1/1**	14/14
P4	**4/4**	**3/3**	**0**	**1/1**	8/8
P5	**4/4**	**5/5**	**4/4**	**1/1**	14/14

Tarefas Básicas - Cobertura

	Tarefa 1	Tarefa 2	Tarefa 3	Tarefa 4	Cobertura
P1	**4/8**	**1/5**	**4/4**	**1/1**	10/18
P2	**4/8**	**4/5**	**4/4**	**1/1**	13/18
P3	**4/8**	**5/5**	**4/4**	**1/1**	14/18
P4	**4/8**	**3/5**	**0**	**1/1**	8/18
P5	**4/8**	**5/5**	**4/4**	**1/1**	14/18

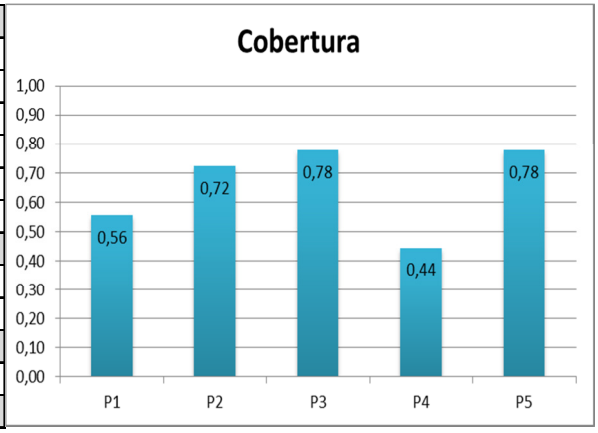


Figura 5.4 – Análise respostas Tarefas Básicas

No entanto, a Tarefa 1 apresentou um problema comum de interpretação entre os participantes. Nenhum indicou as subatividades em que determinado papel também aparece quando modelado associado à atividade pai de uma hierarquia. Esse problema pode ter sido causado por essa informação não ter ficado clara nem nos modelos apresentados para análise, nem no material introdutório. Baseado nessa falta apresentada pelos participantes, a informação que um papel, ferramenta ou produto de trabalho associados a uma atividade mais acima em uma hierarquia deve ser considerado nas demais atividades e tarefas abaixo na hierarquia, foi acrescentada ao metamodelo *OdysseyProcess-FEX*. Esse mesmo problema de interpretação pode ter ocorrido na Tarefa 2, já que um dos participantes indicou apenas a atividade pai da

hierarquia e não analisou a participação do papel nas subatividades e tarefas relacionadas.

Quanto à questão da mútua exclusividade entre variantes (Tarefa 3), esta foi corretamente respondida por quatro participantes, indicando que ficou claro a representação de pontos de configuração do domínio e suas alternativas. Um dos participantes indicou dificuldade em entender como o conceito "método" estava sendo tratado na modelagem, mas respondeu corretamente os dois itens da questão. Outro participante deixou de responder a questão alegando não entendimento. Acredita-se que o termo método usado na questão possa ter levado a dúvida na interpretação de como identificar o elemento na modelagem. Procedimentos descrevem métodos, técnicas e diretrizes que podem ser adotadas por projetos para atender determinados objetivos e resultados. Elementos de processos são descritos e relacionados de forma a atender determinado procedimento. Na modelagem proposta, é possível enumerar os procedimentos em um modelo contextual que permita indicar quais métodos, técnicas ou diretrizes podem ser trabalhadas e relacionar com o modelo e características para determinar quais os elementos devem ser selecionados para atender tal procedimento. No entanto, o próprio modelo de características pode fazer referência a uma técnica ou método específico, como é o caso do exemplo modelado, onde os nomes dos elementos e suas descrições indicam que os métodos GQM e GQ(I)M podem ser adotados para realizar a estimativa do tamanho de um projeto.

A seleção da alternativa de participação da ferramenta foi realizada corretamente por todos os participantes, que conseguiram identificar em qual situação a ferramenta seria escolhida (Tarefa 4).

A média do tempo total gasto pelos participantes para responder as quatro tarefas foi de 13,2 minutos, com o menor tempo sendo de 10 minutos e o máximo de 15 minutos (Figura 5.5). Pode-se observar que o participante P3 foi um dos participantes com maior valor de cobertura (78%) com o menor tempo total desse primeiro grupo de tarefas (10 min). Tal participante apresentou o segundo maior nível de experiência com reutilização processos de software. O participante P5 também apresentou o maior valor de cobertura (78%) ainda que tenha atingindo o maior tempo (15 min). Este participante apresentou o menor nível de experiência em

reutilização de processos de software. A única tarefa incompleta destes participantes foi a Tarefa 1.

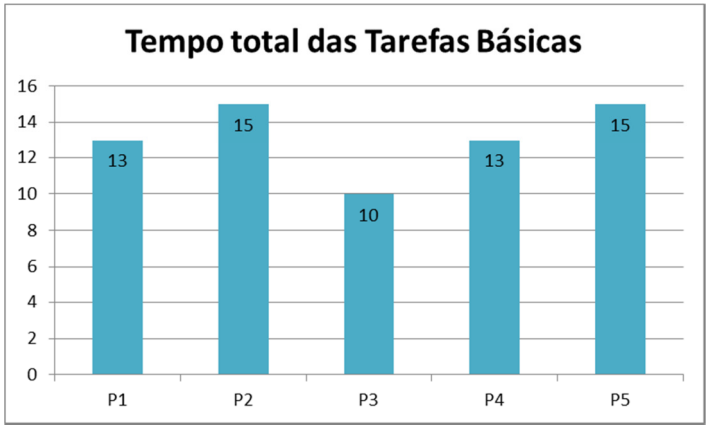


Figura 5.5 – Análise do Tempo Total das Tarefas Básicas

O tempo gasto em cada tarefa variou por participante (Figura 5.6). A Tarefa 3, que possui dois itens para resposta, apresentou maior tempo de execução, sendo este de 8 minutos. O menor tempo de resposta por tarefa foi de 1 minutos, com dois participantes apresentando este tempo para a Tarefa 4 e um participante apresentando este tempo para a Tarefa 1.

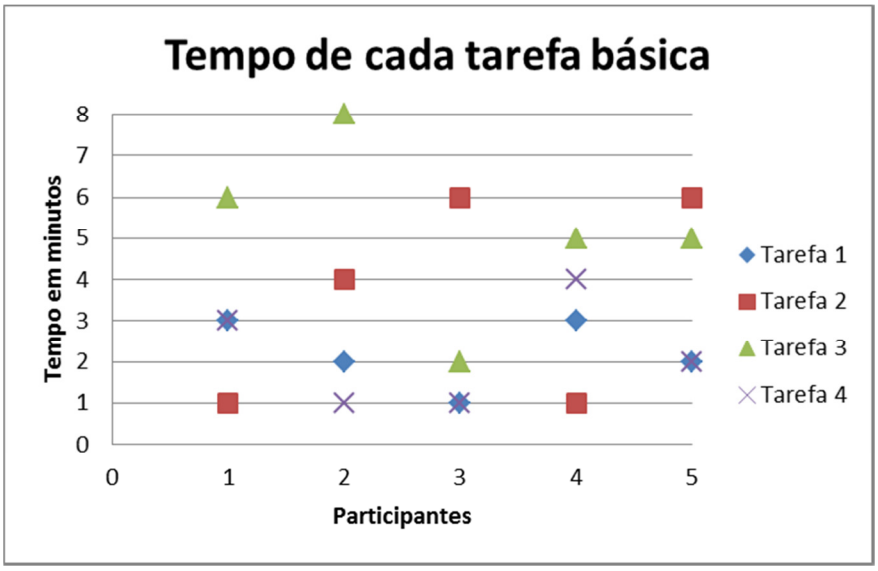


Figura 5.6 – Análise do Tempo por tarefa básica de cada participante

5.2.4.2. Análises dos resultados das tarefas intermediárias

Os resultados observados nesta etapa correspondem às quatro tarefas intermediárias do estudo. A Tabela 5.4 apresenta o número de respostas esperadas por tarefa.

Tabela 5.4 – Respostas Esperadas das Tarefas Intermediárias

Tarefas Básicas					
Tarefa	**Qual o produto de trabalho mais requerido por todas as atividades do domínio?**	**Do grupo de ferramentas definidas quais devem estar presentes em todas as definições de processos derivadas a partir da linha e quais não podem estar presentes na mesma definição de processo derivada a partir da linha?**	**Qual(is) é(são) o(s) produto(s) final(is) de processos a serem derivados a partir da LPrS?**	**Todos os processos derivados a partir da linha apresentam este(s) produto(s)? Em caso negativo, justifique e identifique qual(is) produto(s) deve(m) estar presente(s) em todo processo derivado e qual(is) tarefa(s) é(são) responsável(is) pela produção deste(s) produto(s).**	**Qual(is) componente(s) de processo não apresenta(m) variações na forma de execução, ou seja, qual(is) componente(s) é (são) executado(s) da forma como foi(foram) definido(s) sem nenhuma configuração adicional de definição?**
Resposta Esperada	Plano de Medição Organizacional	Devem estar presentes em todas as definições as ferramentas: Processador de Texto, Planilha Eletrônica e SGBD. Não podem estar presentes na mesma definição: E-mail e Gerador de Apresentação	Relatório de Medição e Análise Armazenado, E-mail, Apresentação, Agendamento da reunião, Feedback dos participantes	Não, apenas o **Relatório de Medição e Análise Armazenado** está presente em todos os processos derivados. Nos demais casos devem aparecer E-mail ou (Apresentação, Agendamento da reunião e Feedback dos participantes) dependendo da variante escolhida para reportar os resultados de análise da medição. A tarefa responsável pela produção do Relatório de Medição e Análise Armazenado é **Armazenar os resultados das análises**	Componente ColetarArmazenarEAnalisarMedida e os componentes internos: ColetarEArmazenar Medida e AnalisarMedidasColetadas
Número de respostas esperadas	1	5	5	2	1 ou 3, depende da interpretação do participante que pode indicar apenas o componente composto

Das quatro tarefas intermediárias, os participantes apresentaram um problema de interpretação com a Tarefa 7. Analisando a questão proposta e as respostas dadas, pode-se concluir que não está suficientemente claro qual a resposta esperada. No entanto, mesmo com a ambiguidade de interpretação, todos os participantes colocaram respostas similares e apresentaram todos os produtos de trabalho esperados como resposta e saída final de um processo a ser derivado pela linha. A ambiguidade está no fato de que produtos intermediários também foram citados pelos participantes. Tais respostas não podem ser consideradas erros diante da questão colocada, até porque todos os itens mencionados são de fato produtos dos possíveis processos a serem derivados. Além disso, foi observada corretamente a classificação de opcionalidade de cada um deles. O participante P2 foi o único que respondeu exatamente de acordo com o esperado. O participante P5 também se aproximou da resposta esperada, apenas deixando de indicar um dos produtos esperados como resposta. Os participantes P1 e P3 apresentaram respostas similares indicando os produtos finais, conforme apresentado na Tabela 5.4, e os produtos intermediários. O único participante que apresenta um desvio de resposta foi o participante P4 que não enumerou corretamente todos os produtos, nem respondeu o segundo item da tarefa.

As demais tarefas foram respondidas pelos participantes de acordo com o esperado, com apenas uma falta do participante P5 que omitiu umas das ferramentas na Tarefa 6.

Tarefas Básicas - Precisão					
	Tarefa 5	Tarefa 6	Tarefa 7	Tarefa 8	Precisão
P1	**1/1**	**5/5**	**6/19**	**1/1**	13/26
P2	**1/1**	**5/5**	**7/7**	**1/1**	14/14
P3	**1/1**	**5/5**	**6/16**	**1/1**	13/23
P4	**1/1**	**5/5**	**1/1**	**1/1**	8/8
P5	**1/1**	**4/4**	**6/6**	**1/1**	12/12
Tarefas Básicas - Cobertura					
	Tarefa 5	Tarefa 6	Tarefa 7	Tarefa 8	Cobertura
P1	**1/1**	**5/5**	**6/7**	**1/1**	13/14
P2	**1/1**	**5/5**	**7/7**	**1/1**	14/14
P3	**1/1**	**5/5**	**6/7**	**1/1**	13/14
P4	**1/1**	**5/5**	**1/7**	**1/1**	8/14
P5	**1/1**	**4/5**	**6/7**	**1/1**	12/14

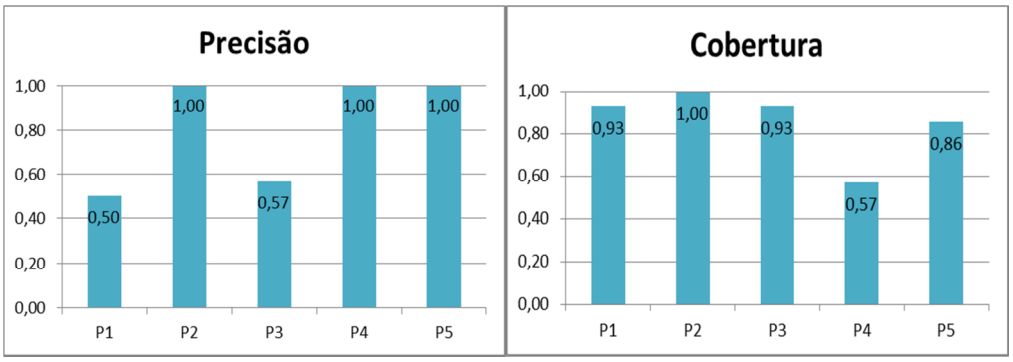


Figura 5.7 – Análise das respostas das Tarefas Intermediárias

A média do tempo total gasto pelos participantes para responder as quatro tarefas foi de 15,6 minutos, com o menor tempo sendo de 15 minutos e o máximo de 22 minutos.

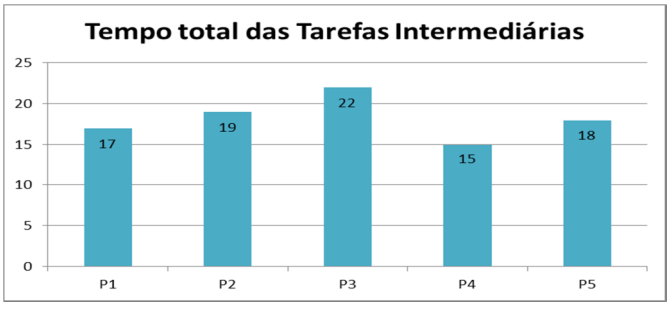


Figura 5.8 – Análise do Tempo Total das Tarefas Intermediárias

O tempo gasto em cada tarefa variou por participante. O participante P2, que respondeu corretamente todas as tarefas apresentadas, apresentou um tempo total de 19 minutos. Já o participante P4, que apresentou o menor tempo total desse grupo

de tarefas (15 minutos), teve o menor valor de respostas corretas, inclusive deixando de responder o segundo item da Tarefa 7.

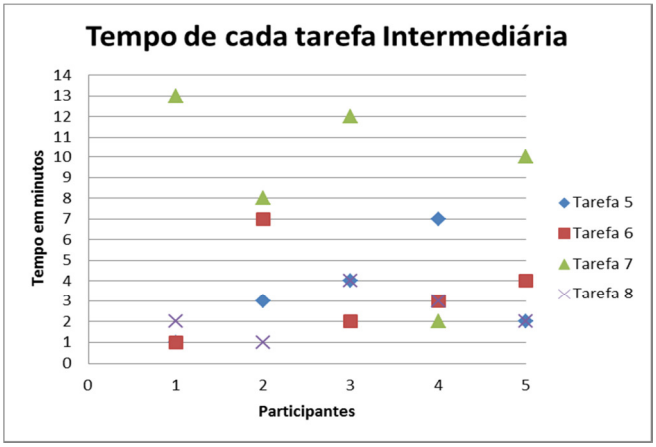


Figura 5.9 – Análise do Tempo por tarefa intermediária de cada participante

5.2.4.3. Análises dos resultados das tarefas de assimilação

Os resultados observados nesta etapa correspondem às duas últimas tarefas, tarefas de assimilação do estudo. A Tabela 5.5 apresenta o número de respostas esperadas por tarefa.

Tabela 5.5 – Respostas esperadas Tarefas de Assimilação

Tarefas de Assimilação		
Tarefa	**Medidas definidas associadas aos componentes representam dados que são coletados durante a execução de processos derivados a partir da linha. Tais medidas visam registrar informação de execução associada ao contexto que a gerou, podendo ser utilizada para apoiar a definição de novos processos. Para selecionar a alternativa mais viável para reportar resultados de análise, o tempo gasto na divulgação pode ser um fator importante. Desta forma, considerando as medidas associadas aos componentes definidos, qual medida deveria ser analisada para apoiar a escolha da forma de reportar o resultado de medição mais apropriado para um projeto específico?**	**Selecione as alternativas do domínio mais adequadas para o cenário descrito abaixo, descrevendo como a atividade de Reportar e armazenar as análises de medição C2deve ser definida para este contexto, descrevendo tarefas com fluxo de execução, produtos de trabalho, papéis e ferramentas. Cenário: Equipe com 10 membros localizados no mesmo escritório**
Resposta Esperada	· Resposta esperada: ESF_RPT_MED – Número de horas gastas pela equipe para realizar as atividades relacionadas à divulgação das análises e resultados de medição.	**Variante Armazenar os resultados de medição e reportá-los através de reunião**. As tarefas **Armazenar os resultados das análises e Realizar reunião com os interessados** e apresentar os resultados devem ser executadas **sequencialmente.** A primeira tarefa tem como produtos de trabalho de entrada **relatório de medição e análise e plano de medição organizacional e produz o relatório de medição e análise armazenado.** A segunda tarefa utiliza o produto de trabalho produzido pela primeira e produz três produtos: **apresentação, reunião e feedback**. O **Grupo de Medição** é responsável pela execução das duas tarefas. Na segunda tarefa os papéis: **Alta direção, Usuário da Medição e Grupo de Processos da organização** podem participar da sua execução de forma adicional. As ferramentas **Processador de texto e Planilha Eletrônica** são usadas como suporte nas duas tarefas. E o **gerador de apresentação** apoia a execução da segunda tarefa.
Número de respostas esperadas	1	17

Dois participantes não entenderam a Tarefa 9, relacionada às medidas associados aos componentes. Tal conceito não foi explicado no material introdutório. Como o estudo foi realizado à distância, sem treinamentos adicionais, apenas pelo material, a falta dessa informação pode ter contribuído para este problema. Os demais participantes responderam corretamente a medida esperada. Esse conceito foi discutido ao longo do Capítulo 3 desta tese e encontra-se descrito no metamodelo proposto.

Na Tarefa 10, a variante correta foi escolhida por todos os participantes. No entanto, apenas o participante P2 respondeu de forma completa todos os elementos que devem fazer parte da resposta esperada. O participante P1 apresentou o pior desempenho desse grupo de tarefas, não descrevendo todos os elementos que deveriam ser selecionados para completar a Tarefa 10. Os participantes P5, P3 e P4, aproximaram-se da resposta completa da Tarefa 10, indicando 15, 14 e 11 dos 17 elementos esperados, respectivamente (Figura 5.10). Acredita-se que um suporte ferramental auxiliaria na seleção dos elementos, evitando a incompletude das respostas.

Tarefas Básicas - Precisão			
	Tarefa 9	Tarefa 10	Precisão
P1	**0/1**	**4/4**	4/5
P2	**1/1**	**17/17**	18/18
P3	**1/1**	**14/14**	13/13
P4	**0/1**	**11/11**	11/12
P5	**1/1**	**15/15**	16/15
Tarefas Básicas - Cobertura			
	Tarefa 9	Tarefa 10	Cobertura
P1	**0/1**	**4/17**	4/18
P2	**1/1**	**17/17**	18/18
P3	**1/1**	**14/17**	15/18
P4	**0/1**	**11/17**	11/18
P5	**1/1**	**15/17**	16/18

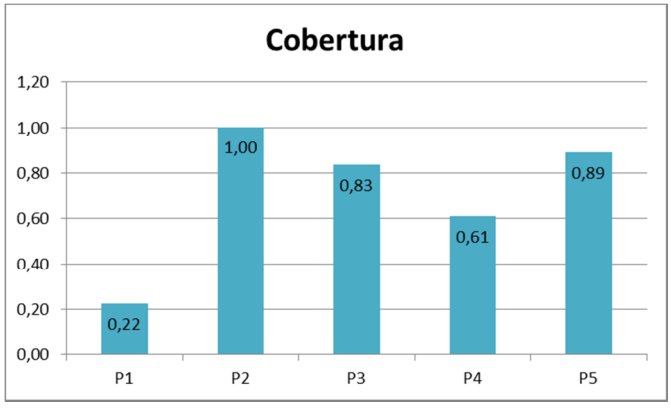


Figura 5.10 – Análise respostas das Tarefas de Assimilação

A média do tempo total gasto pelos participantes para responder as quatro tarefas foi de 16,4 minutos, sendo a média mais alta dos três grupos de tarefas. Esse tempo pode ser atribuído à necessidade de listar um por um dos 17 elementos que compõem a resposta da Tarefa 10. O menor tempo foi de 10 minutos do participante P2 que apresentou respostas completas e corretas às duas tarefas. O tempo máximo foi de 19 minutos. O tempo gasto em cada tarefa variou por participante, sendo maiores na execução da Tarefa 10.

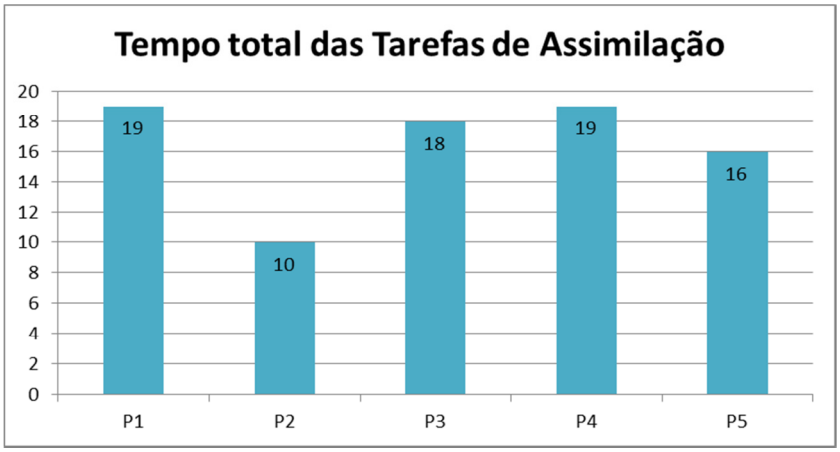


Figura 5.11 – Análise do Tempo Total das Tarefas de Assimilação

5.2.4.4. Sumarização dos resultados das tarefas

Três tarefas foram corretamente e completamente respondidas pelos cinco (Tabela 5.6). Quatro (Tarefa 1, 2, 7 e 10) das dez tarefas listadas tiveram um problema maior para serem completamente respondidas pelos cinco participantes. No entanto, em uma delas houve um problema de interpretação da questão apresentada (Tarefa 7), em que os participantes forneceram respostas corretas, só que consideraram produtos intermediários, além dos produtos finais esperados. Duas das tarefas apresentaram problemas similares (Tarefa 1 e 2), onde não foi analisada a relação dos elementos de papéis com subunidades que pertencem a uma composição de unidades de trabalho no modelo. Para este problema, foram indicadas observações sobre essas relações no relatório técnico do metamodelo (TEIXEIRA *et al*., 2016). A Tarefa 3 só não foi completamente respondida por um participante que deixou de responder a questão por não entender o significado no termo "*método*" usado. A Tarefa 6 só não foi completamente respondida por um participante que deixou de indicar um dos elementos esperados. A Tarefa 9 não foi corretamente respondida por um participante e esse problema pode ter sido gerado pela falta de informação do conceito de medida

no material introdutório. A Tarefa 10 foi respondida de forma incompleta, mas a variante correta foi escolhida. O uso de um ferramental pode auxiliar na completude dessa resposta.

Tabela 5.6 – Resumo dos resultados do estudo

	Tarefas Básicas				Tarefas Intermediárias				Tarefas de Assimilação	
	Tarefa 1	Tarefa 2	Tarefa 3	Tarefa 4	Tarefa 5	Tarefa 6	Tarefa 7	Tarefa 8	Tarefa 9	Tarefa 10
Número de partcipantes que acertou completamente a tarefa	0	2	4	5	5	4	1	5	3	1
Tipos de erros	Falta de análise de elementos relacionados em sudunidades de trabalho em Composição de		Falta de entendimento do termo *"método"*.	-	-	Um dos participantes não indicou um elemento dos cinco elementos esperados.	Problema de interpretação da tarefa colocada. Os participantes apresentaram produtos intermediários como finais dos processos.	-	O conceito de medida não foi apresentado no material introdutório do estudo.	Incompleto. Nem todos os elementos foram citados.
Tempo médio de realização	2,2 min	3,6 min	5,2 min	2,2 min	3,4 min	3,4 min	9 min	2,4 min	4 min	12,2 min

Esse resultado apresenta indícios que a metodologia auxilia na identificação dos pontos de configuração do domínio e permite a correta seleção de elementos de processos envolvidos. No entanto, observou-se que os conceitos da modelagem proposta precisam ser repassados aos usuários da metodologia, assim como a explicação do contexto de aplicação da abordagem.

5.2.5. Análise da avaliação pós-estudo

O formulário pós-estudo foi elaborado baseado em formulários propostos em experimentos realizados em outras propostas de reutilização de processo de software, como o trabalho de BARRETO (2001), que abordam questões pertinentes ao contexto de aplicação da proposta desta tese.

De forma geral, os participantes avaliaram a notação *OdysseyProcess-FEX* de fácil entendimento. Algumas observações sobre o material introdutório apontaram densidade na leitura e falta da informação sobre cardinalidade no ponto de variação e sobre medidas e *baseline* de desempenho associadas aos componentes. Não apresentaram dificuldades para entendimento do domínio utilizado.

Com relação aos benefícios esperados com o uso da abordagem, a Tabela 5.7 apresenta um resumo dos resultados apontados pelos participantes. De forma geral, os participantes avaliam que a abordagem tem algum potencial para aumentar a produtividade, diminuir o esforço (custo) e o tempo necessário para se definir um processo, quando comparado a não utilização da abordagem de reutilização. Apenas um participante não identificou esse potencial nos três aspectos listados.

Quanto à diminuição do retrabalho, quatro participantes indicaram que é de se esperar que a abordagem utilizada forneça o benefício na maioria das situações. Apenas um participante indicou que é de se esperar qualquer benefício a partir da utilização da abordagem utilizada.

Quanto ao aumento da qualidade (aumento na adequação dos processos definidos aos contextos para os quais foram estabelecidos), os participantes apresentaram opiniões bem diversificadas, variando desde nenhum até alto potencial.

A diversidade de respostas pode estar relacionado pelo estudo não apresentar de forma completa a metodologia proposta, sendo avaliado mais detalhadamente os resultados da aplicação do resultado de modelagem de uma LPrS.

Tabela 5.7 – Análise dos benefícios esperados com a adoção da abordagem

Avaliação dos Benefícios Esperados com o Uso da Abordagem Utilizada					
	Aumento de produtividade	**Diminuição de retrabalho**	**Aumento da qualidade**	**Diminuição de custos/esforço**	**Diminuição do tempo de desenvolvimento**
Alto	2	4	2	2	2
Médio	2		1	2	2
Baixo			1		1
Nenhum	1	1	1	1	

O valor Alto: é de se esperar que a abordagem utilizada forneça o benefício na maioria das situações.
O valor Médio: é de se esperar que a abordagem utilizada tenha algum potencial de fornecer o benefício esperado.
O valor Baixo: é de se esperar que a abordagem utilizada forneça o benefício apenas em poucos casos.
O valor Nenhum: é de se esperar qualquer benefício a partir da utilização da abordagem utilizada.

A avaliação do nível de dificuldades esperado decorrente da adoção da metodologia utilizada é apresentada na Tabela 5.8 abaixo.

Tabela 5.8 – Análise do nível de dificuldade esperado com a adoção da abordagem

Avaliação do Nível de Dificuldades esperado decorrente da adoção da abordagem utilizada				
	Dificuldades na Identificação e recuperação de elementos reutilizáveis		**Dificuldades no entendimento dos elementos de processos reutilizáveis**	**Falta de qualidade dos elementos reutilizáveis**
	Dificuldade para encontrar os elementos reutilizáveis necessários e identificar se estão ou não disponíveis	**Falta de informação para guiar a seleção dos elementos reutilizáveis.**		
Alto			1	
Médio	2	1	1	3
Baixo	3	2	2	1
Nenhum		2	1	1

O valor Alto: é de se esperar que a adoção da abordagem utilizada apresente a dificuldade na maioria das situações.
O valor Médio: é de se esperar que a adoção da abordagem utilizada tenha algum potencial de apresentar a dificuldade.
O valor Baixo: é de se esperar que a adoção da abordagem utilizada apresente a dificuldade apenas em poucos casos.
O valor Nenhum: é de se esperar que a adoção da abordagem utilizada nunca apresente dificuldade

Com relação à avaliação da metodologia proposta:

- Três participantes concordam que a metodologia auxilia na definição de processos, tornando a atividade mais fácil e com apoio à reutilização de conhecimento.

- Todos concordam que a abordagem utilizada facilita a seleção dos elementos de processos mais adequados para compor um processo em um dado contexto, considerando as situações do contexto e requisitos a serem atendidos pelo processo, e permite a definição de processos com base na escolha de componentes.
- Todos os participantes concordam que os artefatos ou alguns dos artefatos definidos são necessários. Um dos participantes acrescentou que tal representação tem por objetivo uma descrição "razoavelmente formal do processo" e eventualmente executável.
- Todos concordam que as opcionalidades e variabilidades ficaram explícitas na linha de processos criada. Um dos participantes apontou que a utilização de outros recursos de visualização (cor e formato) poderia contribuir ainda mais essa identificação.
- Todos concordam que linhas de processos criadas com a abordagem utilizada possam ser aplicadas na indústria. No entanto, um dos participantes indicou que falta o apoio à fase de desenvolvimento com os artefatos criados. Outro mencionou que ainda falta às organizações alcançar um nível de maturidade que permita implantar esse tipo de abordagem.

Foram listados alguns aspectos positivos da aplicação da abordagem para apoiar a reutilização de conhecimento de processos em novas definições:

- Reutilização de conhecimento, "com engenheiros de processo mais experientes definindo os itens reutilizáveis e 'garantindo' que itens obrigatórios sejam utilizados";
- "Definição explícita e clara das variabilidades na abordagem que torna fácil encontrá-las nos modelos";
- "Através de uma notação de fácil entendimento/utilização, a abordagem torna a atividade de definição de processos fácil, eliminando a necessidade de se criar novos processos de um mesmo domínio 'do zero', quando necessário";
- "Os ícones adotados, relacionamentos e estereótipos são conhecidos da comunidade de Engenharia de Software, isto torna mais fácil entender os elementos do processo";
- "O modelo para reutilização de processos é similar ao modelo para reutilização de produto de software, isto também facilita a aprendizagem da notação, no contexto da Engenharia de software";

- “Redução no tempo de definição de um processo que atenda a necessidades específicas de um projeto, quando a linha de processo já está criada”;

Foram listados alguns aspectos negativos da aplicação da abordagem para apoiar a reutilização de conhecimento de processos em novas definições:

1. “Esforço inicial grande; Necessidade de apoio ferramental; Eventualmente alguma barreira cultural”;
2. “Em muitos casos o custo/esforço de criação de uma Linha de Processo pode ser maior que fazer do zero”;
3. “Não há separação clara entre espaço do problema e da solução”;
4. “Necessidade de treinamento dos envolvidos no processo com relação ao funcionamento da abordagem/notação.”
5. Dificuldade de entendimento das regras de composição conforme descrito.
6. Falta de apoio computacional para instanciar e executar os processos reutilizados, além de coletar e explicitar no modelo, os resultados dos indicadores de processo que são usados ao selecionar um elemento de processo para reuso.
7. “A periodicidade da atualização da lista de variabilidade e opcionalidade, para o caso de existir novas situações”.

O custo inicial para reutilização, que envolve o desenvolvimento da LPrS, foi apontado por dois participantes. Este entrave já é enfrentado na área de reutilização de software e aspectos como tratamento de barreiras psicológicas, legais e econômicas; incentivos à reutilização; e análises de retorno de investimento devem ser estimulados para sobrepor esse entrave.

Com relação à segunda observação, pode-se relatar que a abordagem contextual, que especifica maiores detalhes do espaço do problema, não foi amplamente abordada no estudo em questão. Desta forma, maiores detalhes da representação desses dois espaços podem ser observados no Capítulo 3 desta tese.

Treinamentos podem aumentar o grau de compreensão de certos aspectos e conceitos envolvidos na metodologia proposta. Vale considerar que para este estudo, apenas um material introdutório foi enviado aos participantes. Este material é limitado e apresenta apenas partes dos elementos propostos no metamodelo e na notação *OdysseyProcess-FEX*. Isso pode ter gerado a dificuldade de entendimento de leitura e

interpretação das regras de composição utilizadas. Além disso, a forma de descrição das relações de dependência e mútua exclusividade entre as características foi uma escolha de modelagem que poderia ter sido descrita de outra forma.

Quanto ao apoio computacional para as fases de instanciação e execução, este se encontra fora do escopo desta tese. Ainda que alguns dados de instanciação e execução tenham sido previstos na fase de projeto da Engenharia de Domínio (agentes, *templates*, scripts, medidas e *baselines*), o foco está na definição de processos de software, atividade anterior à instanciação e execução.

A atualização da LPrS e seus pontos de variabilidade e opcionalidade estão fora do escopo desta tese. Esse item deve ser abordado em trabalhos específicos de análise da evolução da LPrS por adaptações ou atualizações dos contextos envolvidos.

Como melhorias para a metodologia proposta foram apontados:

(i) a extensão para apoio à fase de desenvolvimento *com* reutilização dos artefatos criados. Com relação a esta indicação, ainda que um processo inicial tenha sido proposto no Capítulo 3 desta tese, um maior apoio é apontado como trabalho futuro no Capítulo 7;

(ii) indicar a forma que a abordagem permite modelar variabilidades de granularidade fina. Para este tópico mencionado, a representação proposta permite a representação de variabilidade e opcionalidade em diferentes elementos do domínio e seus relacionamentos. Desta forma, é possível atribuir essas propriedades dos elementos de forma mais individualizada e pontual (modelo de características) e analisá-las de forma conjunta por meio de componentes. Além disso, buscou-se trabalhar com os principais elementos de processos existentes, onde uma atividade pode ser especificada por tarefas que chegam até o nível de detalhamento de passos. Além disso, tais unidades de trabalho podem possuir papel, produto de trabalho e ferramenta associados. Descrições mais detalhadas da possibilidade de representação encontram-se descritas no Capítulo 4 desta tese e no relatório técnico resultante como especificação do metamodelo e notação *OdysseyProcess-FEX* (TEIXEIRA *et al*., 2016); e

(iii) descrição de como a abordagem apoia outras etapas por meio de processos para definir *para* e *com* reutilização e apoio automatizado a partir dos

diagramas. Para este item, podemos citar os processos definidos no Capítulo 3 desta tese, que não foram abordados no estudo. Com relação ao apoio automatizado a partir dos diagramas, algumas implementações foram realizadas e descritas no Capítulo 6 desta tese. No entanto, o apoio à fase de execução de processos é apontado como trabalho futuro, indicado no Capítulo 7.

(iv) Melhorias no material de apoio e regras de composição mais intuitivas. O material de apoio foi desenvolvido apenas para aplicação do estudo e de fato, apresenta limitações na exposição de todos os conceitos envolvidos na metodologia proposta. O metamodelo e notação encontram-se melhor descritos no relatório técnico produzido (TEIXEIRA *et al.*, 2016). E maiores detalhes sobre a metodologia podem ser encontrados nos Capítulos 3 e 4 desta tese. Com relação às regras de composição, a proposta de uso de expressões visa possibilitar a combinação de duas ou mais características de processos. A complexidade das regras criadas fica a cargo do engenheiro de domínio. Mecanismos lógicos de interpretação das regras podem ser aplicados para auxiliar em verificações de consistência e interpretação em cenários mais complexos.

(v) Aplicação da metodologia em processos mais complexos. Estudos podem ser realizados em domínios mais extensos e complexos para analisar o comportamento da metodologia nesses cenários. Essas avaliações devem considerar a aplicação em casos reais e práticos na indústria, mas como citado por um dos participantes, muitas organizações ainda não atingiram determinado nível de maturidade para implantação de uma metodologia de reutilização de processos.

(vi) Levar em consideração os critérios de atualização da linha de processo, principalmente pela natureza evolutiva e dinâmica da execução de processos de software. Esse item pode ser analisado como trabalho futuro, que pode abranger o tópico de evolução de LPrS.

O suporte ferramental foi apontado como de grande importância no cenário de reutilização de processos de software. Apoio à modelagem, verificação de consistência e posterior apoio a derivação de processos a partir dos artefatos construídos foram

pontos listados pelos participantes. Além disso, foi citado o apoio à visão de perspectivas diferentes dos componentes ou níveis de detalhes para agilizar a busca de componentes. Agregação de conhecimento de execução e uso de sistemas de recomendação foram apontados como possíveis suportes à decisão. Uma infraestrutura foi implementada no ambiente Odyssey, conforme descrito no Capítulo 6 desta tese. Essa infraestrutura apoia a modelagem do domínio e implementa verificação de consistência por meio de mecanismos que acompanham a modelagem e notificam o usuário sobre inconsistências geradas. No entanto, esse apoio deve ser complementado com a aplicação dos *checklists* propostos. A infraestrutura já apresenta algum apoio à fase de derivação de processos a partir da linha, como verificação de inconsistências geradas à medida que seleções são realizadas no modelo de características. No entanto, como um apoio mais detalhado a esta etapa é considerado como trabalho futuro, a implementação de mais recursos de apoio também precisará ser desenvolvida.

5.3. CONSIDERAÇÕES FINAIS

Os estudos apresentados podem ser considerados avaliações iniciais da aplicabilidade da metodologia como um todo. No entanto, tais estudos apresentaram indícios da viabilidade de aplicação dos diferentes elementos que compõem a metodologia proposta.

Pode-se concluir que o desenvolvimento de domínios de processos de software para reutilização, envolvendo as etapas e artefatos definidos, pode ser altamente custoso sem a utilização de um suporte computacional. Além disso, o nível de experiência sobre conceitos de reutilização e a disponibilização de mecanismos de transmissão do conhecimento sobre a metodologia podem ser fatores de impacto para melhorias nos resultados esperados com a organização da informação do domínio e com o tratamento da complexidade de aplicação da metodologia.

Estudo em cenários mais complexos, especial com processos reais e na indústria, e com mais participantes devem ser aplicados para coleta de indícios mais sólidos. Além disso, as etapas descritas na metodologia devem ser aplicadas em conjunto por engenheiros de domínio para analisar o apoio a este papel na identificação e disponibilização da informação passível de reutilização.

Capítulo 6 – Implementação – Ambiente de Reutilização

6.1. INTRODUÇÃO

Uma vez que no desenvolvimento de uma LPrS, os modelos não são artefatos isolados, é interessante construí-los e utilizá-los em um ambiente de reutilização que forneça subsídios necessários a definição e integração dos artefatos envolvidos. Neste sentido, como apoio à reutilização utilizando a abordagem proposta é apresentado o ambiente Odyssey (ODYSSEY, 2016) adaptado para o desenvolvimento e utilização de LPrS.

O ambiente Odyssey (ODYSSEY, 2016) é uma infraestrutura de reutilização de software baseada em modelos de domínio, que oferece ferramentas de apoio à construção de aplicações de software seguindo abordagens sistemáticas como LPS. Contempla atividades do desenvolvimento de software *para* reutilização, processo de Engenharia de Domínio (ED), e atividades do desenvolvimento *com* reutilização, processo de Engenharia de Aplicação (EA). Essa infraestrutura foi implementada utilizando-se a linguagem Java (SUN, 2005) e vem sendo evoluída desde 1997 (WERNER *et al.*, 1999). Desde então, várias ferramentas foram introduzidas com o objetivo de apoiar os processos de ED e EA e tornar o ambiente mais completo. Por meio de uma análise focada na identificação das funcionalidades básicas de um ambiente de reutilização e as funcionalidades que só seriam utilizadas de acordo com a necessidade do usuário, uma versão reestruturada passou a ser disponibilizada como forma de tratar problemas de escalabilidade e desempenho (Figura 6.1). Essa versão denominada *Odyssey Light* (MURTA *et al.*, 2004) disponibiliza as funcionalidades básicas pelo *kernel* do ambiente e as funcionalidades secundárias pelos *plug-ins*, os quais podemos citar: ferramentas para a documentação de componentes (MURTA *et al.*, 2001), especificação e instanciação de arquiteturas específicas de domínios (XAVIER, 2001), apoio à engenharia reversa (VERONESE *et al.*, 2002), de notificação de críticas em modelos UML (DANTAS *et al.*, 2001), suporte a padrões de projeto (DANTAS *et al.*, 2002), de modelagem e execução de processos (MURTA, 2002), para controle de versões de modelos (OLIVEIRA *et al.*, 2004), apoio à edição concorrente de modelos (LOPES *et al.*, 2004), de transformação de modelos (MAIA *et al.*, 2005), entre outras.

O desenvolvimento deste trabalho visou estender o ambiente Odyssey para apoiar a abordagem de Engenharia de Linha de Processos de Software proposta. Desta forma, foram necessárias adaptações para flexibilizar o ambiente a trabalhar tanto com LPS quanto com LPrS, iniciadas no trabalho de TEIXERA (2011). O maior volume de trabalho se concentrou na disponibilização da notação *OdysseyProcess-FEX* para viabilizar a modelagem de características.

Nesta tese, essas adaptações foram estendidas buscando atender as atividades e os artefatos propostos pela metodologia *OdysseyProcessReuse*. O ambiente foi adaptado para tratar a atualização da notação para modelagem de características e suportar a modelagem de componentes do domínio de processos de software, artefatos que representam a informação a ser reutilizada. Além disso, como modelos de componentes podem ser derivados a partir do modelo de características, seguindo mapeamentos pré-definidos, mecanismos foram implementados para aplicação das heurísticas definidas, atendendo umas das atividades com alto custo se feita de forma manual. Essas adaptações foram realizadas no *kernel* do ambiente.

Como os agrupamentos de componentes e aplicação das técnicas de verificação apresentam mais fortemente a necessidade de interações com o engenheiro de domínio, e caracterizam-se como atividades semi-automatizadas, a implementação foi abordada como trabalho futuro. Essa decisão está relacionada com a necessidade de maiores análises para identificar qual a melhor forma de atender os usuários com redução da complexidade de execução das atividades e melhor apoio de análise dos artefatos.

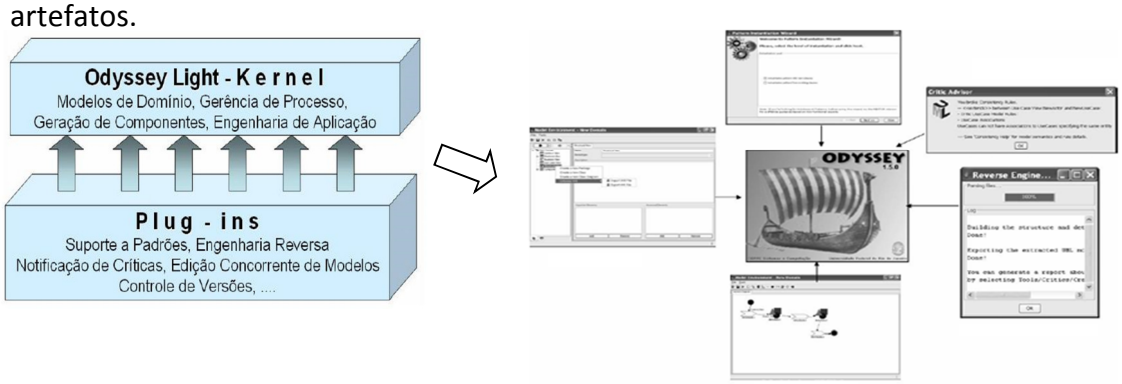


Figura 6.1 - Infraestrutura ambiente Odyssey

6.2. ESTRUTURA INICIAL DO AMBIENTE DE APOIO À REUTILIZAÇÃO

O ambiente antes estruturado apenas para viabilizar a construção de uma LPS passa a apoiar também a construção de uma Linha de Processos de Software. Desta

forma, são disponibilizadas as opções para a criação de domínios de processos de software e domínios de produtos de software, como pode ser visto na Figura 6.2, que representa a tela principal do ambiente.

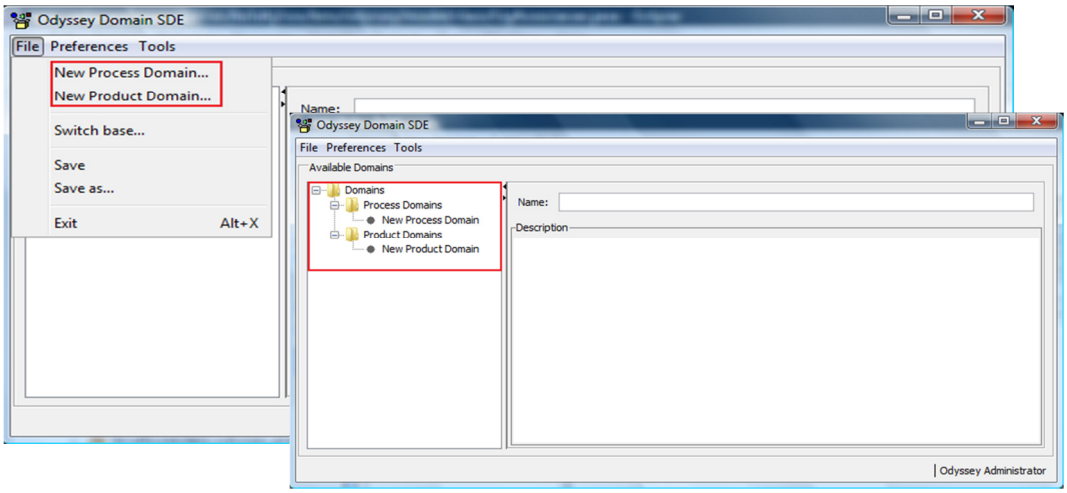


Figura 6.2 – Tela Principal do ambiente Odyssey adaptado

6.3. ESTRUTURA DO AMBIENTE DE APOIO AO DESENVOLVIMENTO DE LPRS (EDPS)

No ambiente Odyssey, a estrutura para modelagem se divide em três partes: Semântica, Léxica e de Apresentação. A estrutura Semântica é responsável pela representação conceitual das notações. As estruturas Léxica e de Apresentação ficam responsáveis pela representação gráfica, que consiste na visualização das notações.

A organização da árvore semântica é feita por meio de categorias de modelos. Cada modelo é composto por diferentes itens de modelagem que representam pacotes, diagramas, nós e ligações específicos à categoria do modelo. A partir de um destes elementos, é possível percorrer a árvore hierarquicamente por meio das relações entre os objetos, com o intuito de obter informações relativas ao domínio.

A parte Léxica refere-se à representação gráfica de cada elemento, formada por diagramas específicos a cada modelo representado. Esses modelos são construídos por meio de uma ferramenta de diagramação, denominada Editor de Diagramas. Para todo item do modelo semântico, existe zero ou mais itens léxicos correspondentes, nós e arestas que formam os desenhos dos diagramas. São os elementos léxicos que determinam a concordância com a representação de cada notação e linguagem adotada no ambiente. Para cada categoria de diagrama identificada no ambiente Odyssey para construção de uma LPrS (e.g., Diagrama de Escopos, Diagrama de Características e Diagrama de Componentes) existe uma representação por meio de

um painel específico denominado de diagramador. Cada painel apresenta os elementos léxicos pertencentes aquele diagrama. Neste painel, os elementos léxicos verificam se os desenhos estão de acordo com a notação determinada pela especificação associada ao painel.

A parte de Apresentação se destina às interfaces gráficas de edição dos elementos semânticos com a descrição de suas propriedades e atributos.

Esta seção se dedica a apresentar funcionalidades que foram adaptadas ou incluídas no ambiente Odyssey para apoiar a modelagem proposta na abordagem *OdysseyProcessReuse*. Desta forma, cada uma dessas partes descritas acima teve de ser adaptada para atender a modelagem proposta pela notação *OdysseyProcess-FEX*. Assim, são descritas os níveis de abstração envolvidas na modelagem de LPrS; as possibilidades de configurações dos artefatos, de acordo com os conceitos presentes na notação; a verificação da consistência dos modelos criados segundo o metamodelo *OdysseyProcess-FEX*; e a representação gráfica dos elementos e seus relacionamentos em diagramas.

A parte do ambiente de edição da LPrS estrutura o ambiente em três níveis de abstração: escopo, características e componentes (Figura 4.2). Esses elementos são apresentados pela árvore semântica disponibilizada na parte esquerda do diagramador do ambiente. Os artefatos dos diferentes níveis são relacionados por meio de rastros (ligações entre os elementos), que visam manter a consistência entre os modelos.

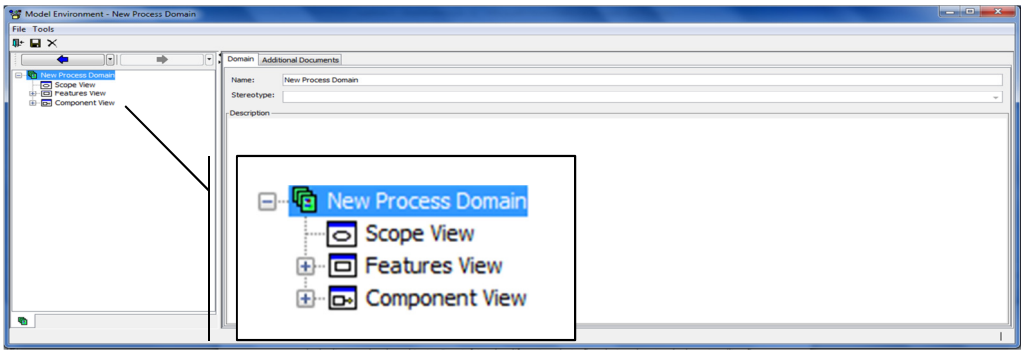


Figura 6.3 – Ambiente de Modelagem de LPrS e seus níveis de abstração

6.3.1. Nível de Escopo

O nível de abstração de escopos (*Scope View*) tem por objetivo situar o domínio em relação ao seu escopo, limites, relacionamentos com outros domínios de processos e principais elementos envolvidos. Desta forma, neste nível é possível especificar o escopo e relacionar as características de processos correspondentes (Figura 6.4a).

Além disso, é possível modelar um diagrama com os escopos definidos e estabelecer relações de dependência entre esses escopos (Figura 6.4b).

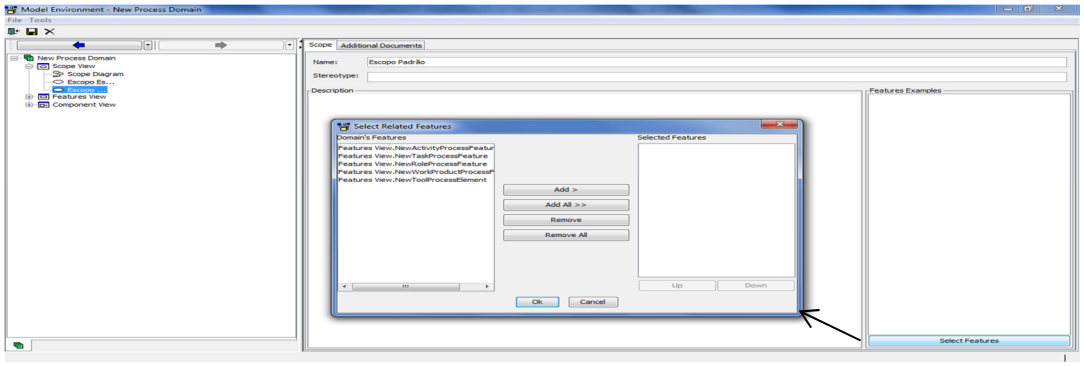


Figura 6.4a. Tela de Edição de Escopo

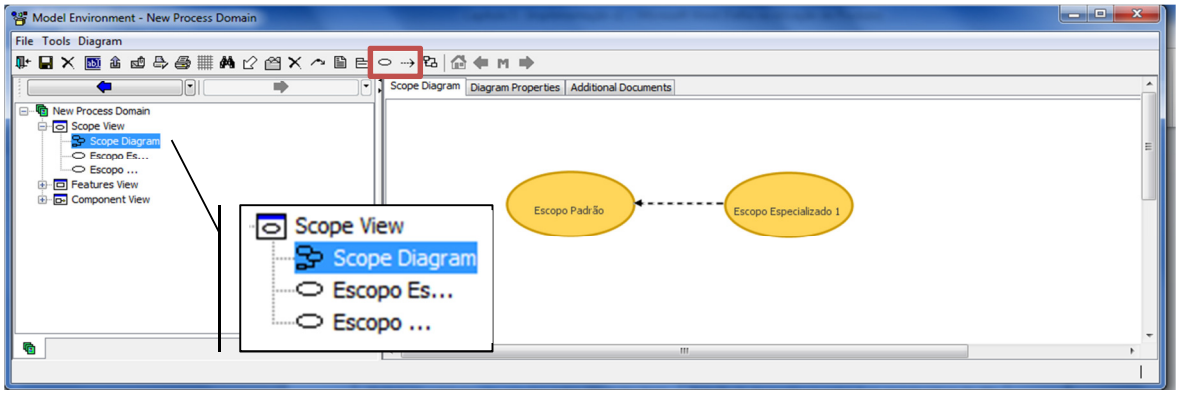


Figura 6.4b. Tela de Edição de Diagrama de Escopo

Figura 6.4 – Ambiente de Modelagem de LPrS – Nível de Escopo (*Scope View*)

6.3.2. Nível de Características

O nível de abstração de características, denominado no Odyssey de "*Features View*" (Figura 6.5) tem por objetivo modelar a LPrS no nível de análise, descrevendo a LPrS por meio de um modelo de características, incluindo a identificação dos elementos de processo que compõem o domínio com a especificação de suas propriedades de variabilidade e opcionalidade. Para isso, são providas cinco categorias de características (atividades e suas tarefas; produtos de trabalho; papéis e ferramentas) e dois tipos de diagramas (visão estrutural e visão comportamental do domínio), além de suporte a definição de regras de composição inclusivas e exclusivas. Neste nível, é também definido o modelo contextual do domínio e são estabelecidas as relações entre o contexto e o modelo de características.

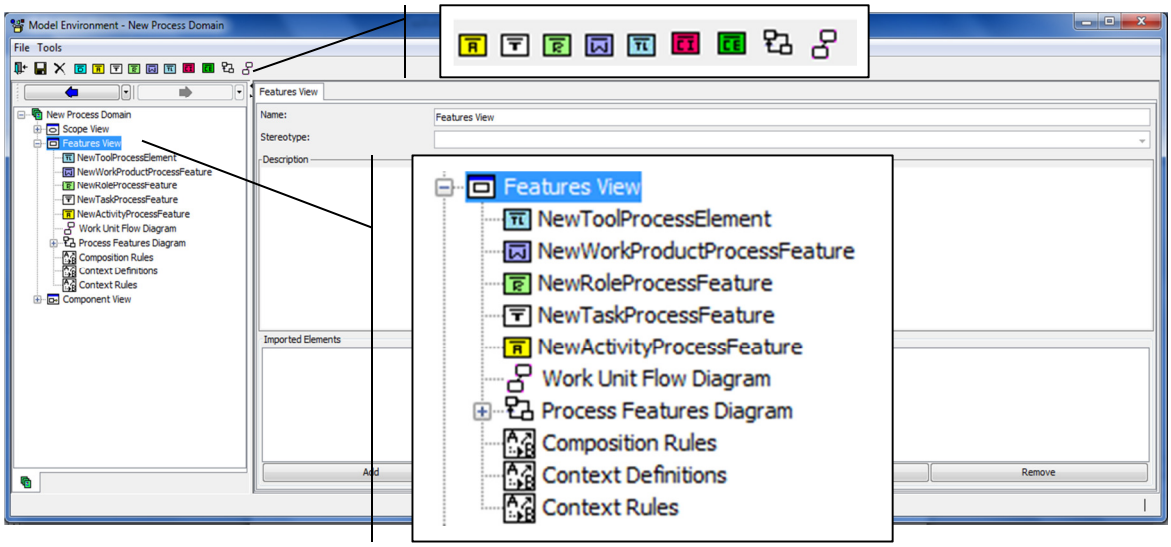


Figura 6.5 – Ambiente de Modelagem de LPrS – Nível de Características (*Features View*)

Cada categoria de característica apresenta um conjunto de dados que são configuráveis por um painel de edição associado. A Figura 6.6 apresenta um exemplo painel de edição de uma característica da categoria Atividade.

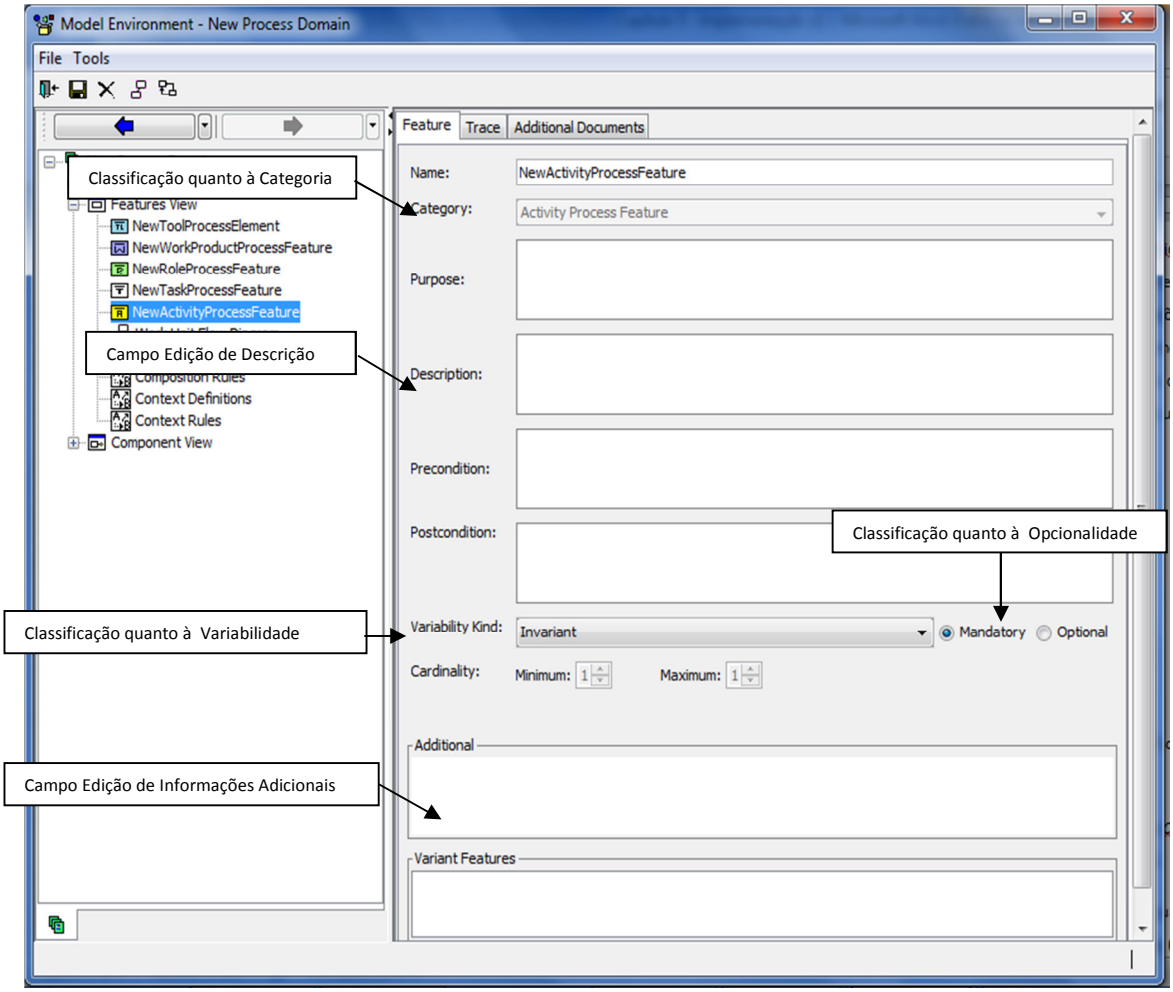


Figura 6.6 – Ambiente de Modelagem de LPrS – Painel de Edição de Característica - Nível de Características (*Features View*)

Os campos nome, categoria, descrição, tipo de variabilidade, tipo de opcionalidade e informações adicionais são comuns a todas as categorias de características. O campo de categoria não permite edição, estando de acordo com o tipo de característica previamente criada. Esse painel para edição de informações de determinada característica apresenta alguns campos específicos a certas categorias. Por exemplo, podemos citar os campos propósito, pré e pós-condição disponíveis para edição nas categorias Atividade e Tarefa, e o campo qualificações disponibilizado apenas para características da categoria Papel.

Caso a característica seja classificada com tipo de variabilidade correspondente a ponto de variação (Figura 6.7), também, é possível visualizar as variantes correspondentes no campo "*Variant Features*".

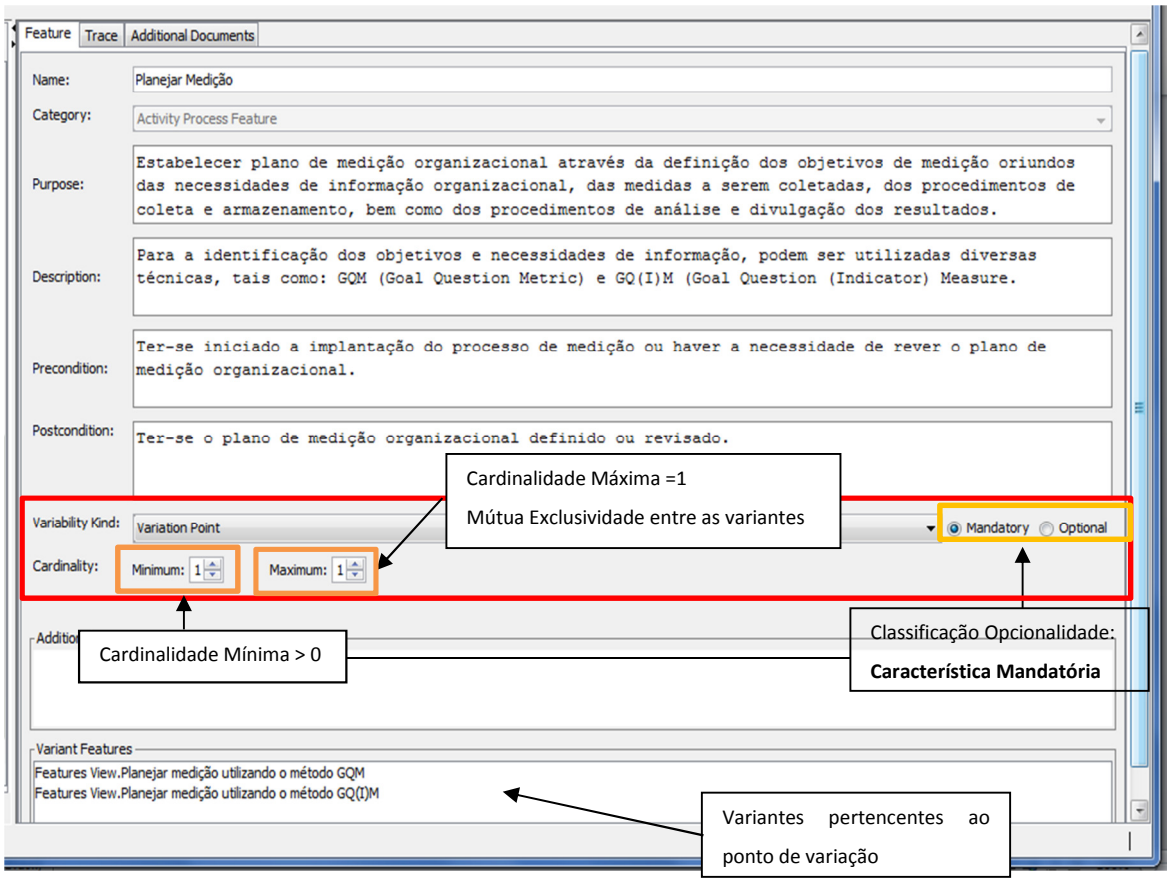


Figura 6.7 – Ambiente de Modelagem de LPrS – Painel de Edição de Característica Ponto de Variação

Esse campo é automaticamente preenchido e constantemente atualizado pela ferramenta, após o estabelecimento de relacionamentos Alternativo entre os elementos envolvidos. Além disso, o campo de cardinalidade também fica disponível para configuração. Neste caso, a classificação quanto à opcionalidade também será impactada pelo valor atribuído à cardinalidade. Desta forma, o tipo de opcionalidade

deve estar em conformidade com o valor da cardinalidade mínima, sendo automaticamente atualizado de acordo com a classificação quanto à opcionalidade, mantendo uma coerência entre as informações (cardinalidade mínima igual a 0 -> característica opcional, cardinalidade mínima maior ou igual a 1 característica mandatória). O valor da cardinalidade máxima igual a um determina a mútua exclusividade das variantes definidas para o ponto de variação.

O nível de características apresenta dois tipos de diagramadores, um para a modelagem da visão estrutural e outro para a modelagem da visão comportamental. Em cada diagrama inserem-se características (nós) e seus relacionamentos específicos (arestas). A exibição desses diagramas corresponde aos itens semânticos *Process Features Diagram* e *Work Unit Flow* Diagram (Figura 6.8).

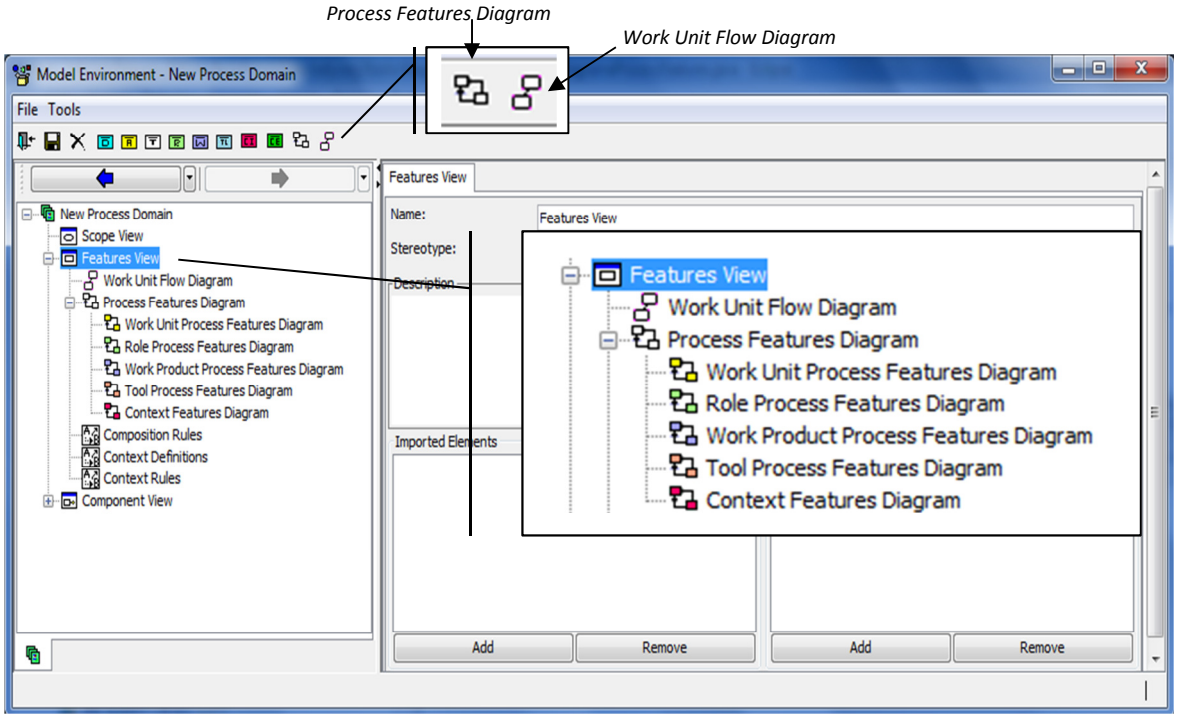


Figura 6.8 – Ambiente de Modelagem de LPrS – Diagramadores (Visão Estrutural e Visão Comportamental)

Na parte superior da janela do diagramador, uma barra de ferramentas auxilia o usuário no processo de modelagem. Nessa barra, encontram-se as ações necessárias à construção do modelo, tais como as ações relativas à inclusão e remoção de características e das ligações entre elas. Essas ligações, tanto semânticas quanto léxicas, só podem ser manipuladas pelo diagramador, já que não possuem representação na árvore semântica do Odyssey.

A fim de contemplar a notação *OdysseyProcess-FEX* nível de análise – modelo de características – visão estrutural, foram acrescentadas no diagramador as seguintes ações: criação das diferentes categorias de características contempladas pela notação e dos relacionamentos identificados (alternativo, composição, agregação, ligação entre elementos de processo – unidade de trabalho e produto de trabalho, unidade de trabalho e papel, unidade de trabalho e ferramenta, papel e produto de trabalho). Apesar de apenas dois ícones para relacionamentos (alternativo e associação), o tipo da ligação é instanciado de acordo com os elementos envolvidos na ligação estabelecida. Esses itens encontram-se destacados na Figura 6.9.

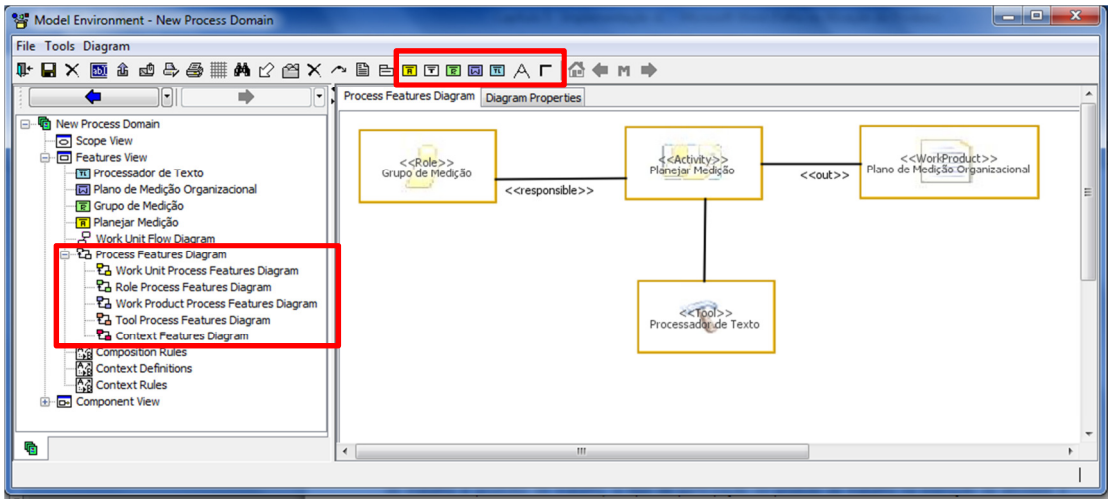


Figura 6.9 – Ambiente de Modelagem de LPrS – Diagramadores Visão Estrutural

Os tipos de elementos de processos relacionados pela relacionamento Alternativo são automaticamente atualizados pela ferramenta, que classifica origem como ponto de variação e destino como variante. A opcionalidade das ligações entre elementos de processo pode ser atribuída por meio de um painel de edição associada a cada relacionamento, acionado por um duplo clique sobre a ligação estabelecida.

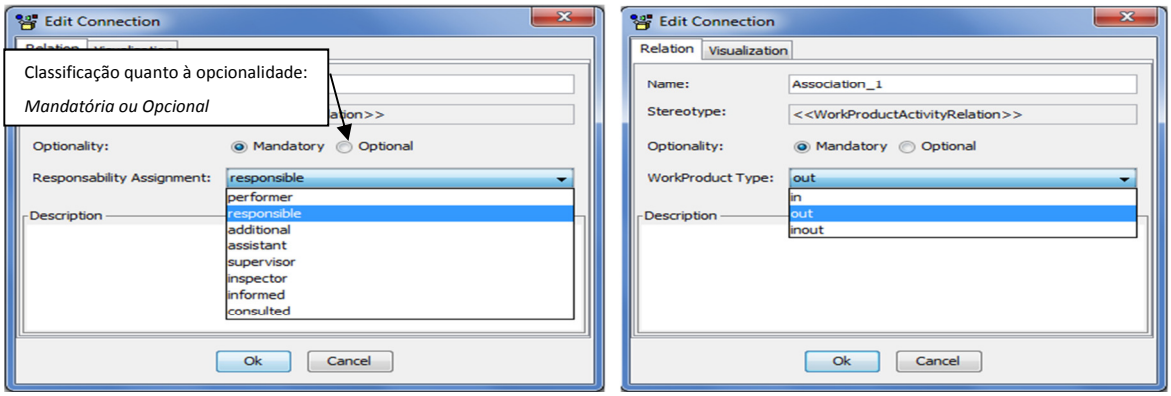


Figura 6.10 – Ambiente de Modelagem de LPrS – Diagramadores Visão Estrutural

No caso das ligações envolvendo unidades de trabalho e papéis, o tipo de participação do papel na execução da atividade é estabelecido pelo painel de edição (Figura 6.10a). No caso das ligações envolvendo unidades de trabalho e produtos de trabalho, o tipo de participação do produto de trabalho na execução da atividade é estabelecido pelo painel de edição (entrada: *in*, saída: *out*, modificado: *inout*) (Figura 6.10b).

Vale ainda ressaltar que as informações desse diagrama podem ser manipuladas por meio de cinco perspectivas diferentes, que correspondem aos sub-diagramas associados (i.e., *Work Unit, Role, Work Product* e *Tool Process Feature Diagram*). Para cada um deles, a barra é dinamicamente atualizada segundo o conjunto de elementos que fazem parte da perspectiva selecionada. A Figura 6.11 apresenta como exemplo a perspectiva *Role Process Feature Diagram*.

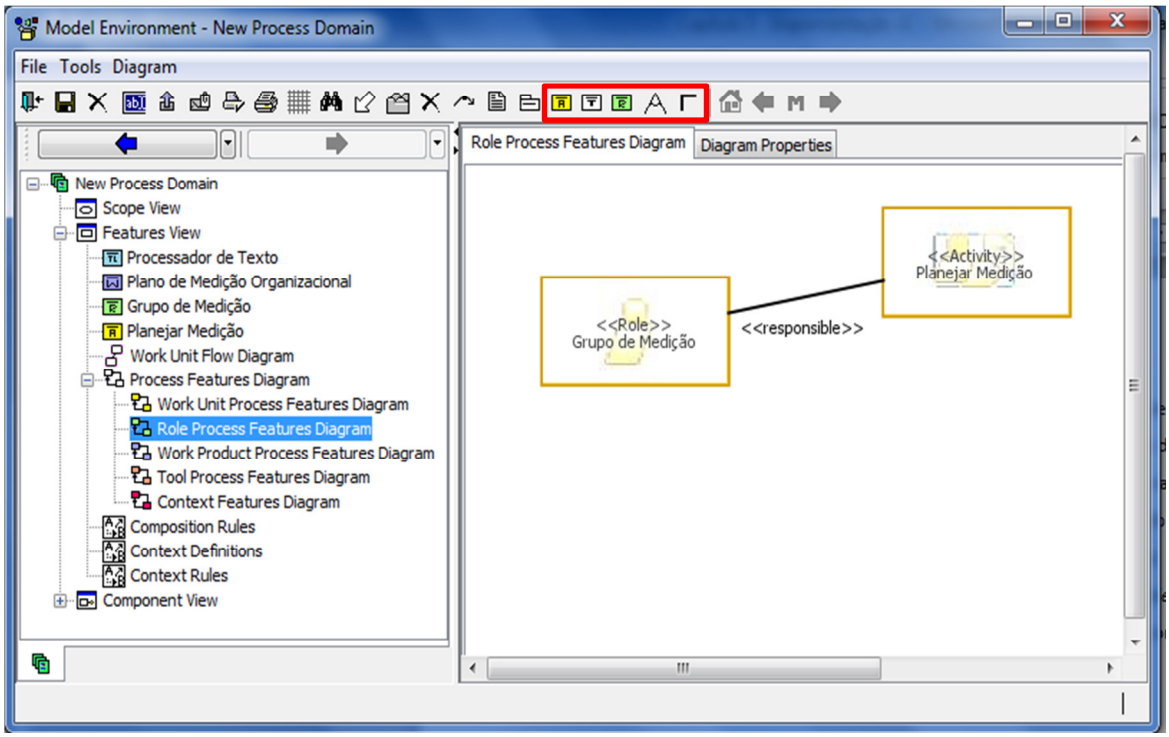


Figura 6.11 – Ambiente de Modelagem de LPrS – Diagramadores Visão Estrutural – Perspectiva *Role Process Feature Diagram*

A fim de contemplar a notação *OdysseyProcess-FEX* nível de análise – modelo de características – visão comportamental, foram acrescentadas no diagramador as seguintes ações: inclusão ou remoção de características das categorias atividade e tarefa e dos relacionamentos de fluxos de controle. Esses itens encontram-se destacados na Figura 6.12. A opcionalidade dos fluxos pode ser atribuída por meio de

um painel de edição associada a cada relacionamento, acionado por um duplo clique sobre a ligação estabelecida, semelhante aos painéis da Figura 6.10.

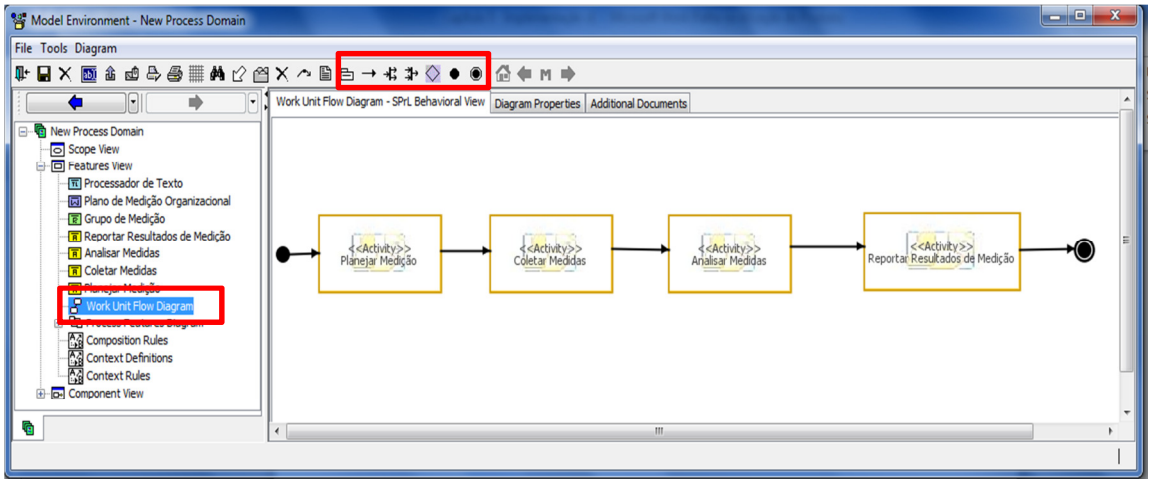
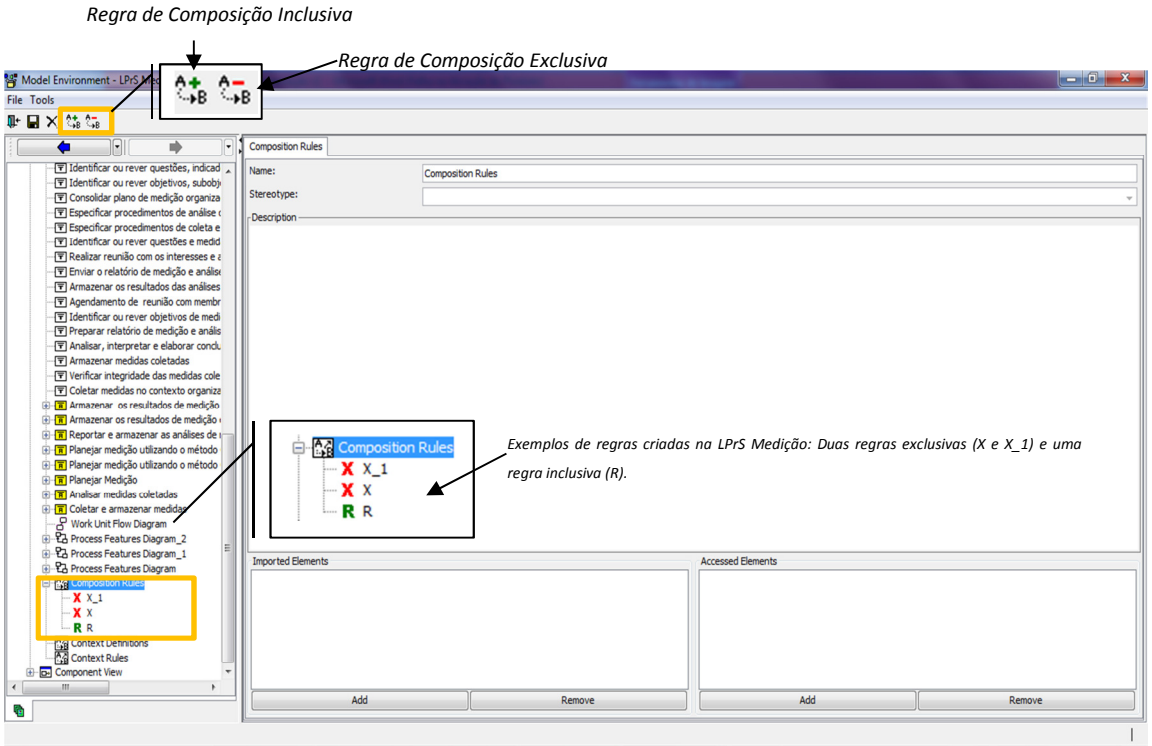


Figura 6.12 – Ambiente de Modelagem de LPrS – Diagramadores Visão Comportamental

Regras de composição são criadas a partir de painéis de edição que permitem a seleção das características e a combinação utilizando operadores booleanos que formam o antecedente e o consequente das regras. Dois tipos de regras podem ser criadas no ambiente: (1) regras de composição inclusiva e (2) regras de composição exclusiva (Figura 6.13).



Painel de Edição de Regra de Composição - *Exemplo de regra de Composição Inclusiva*

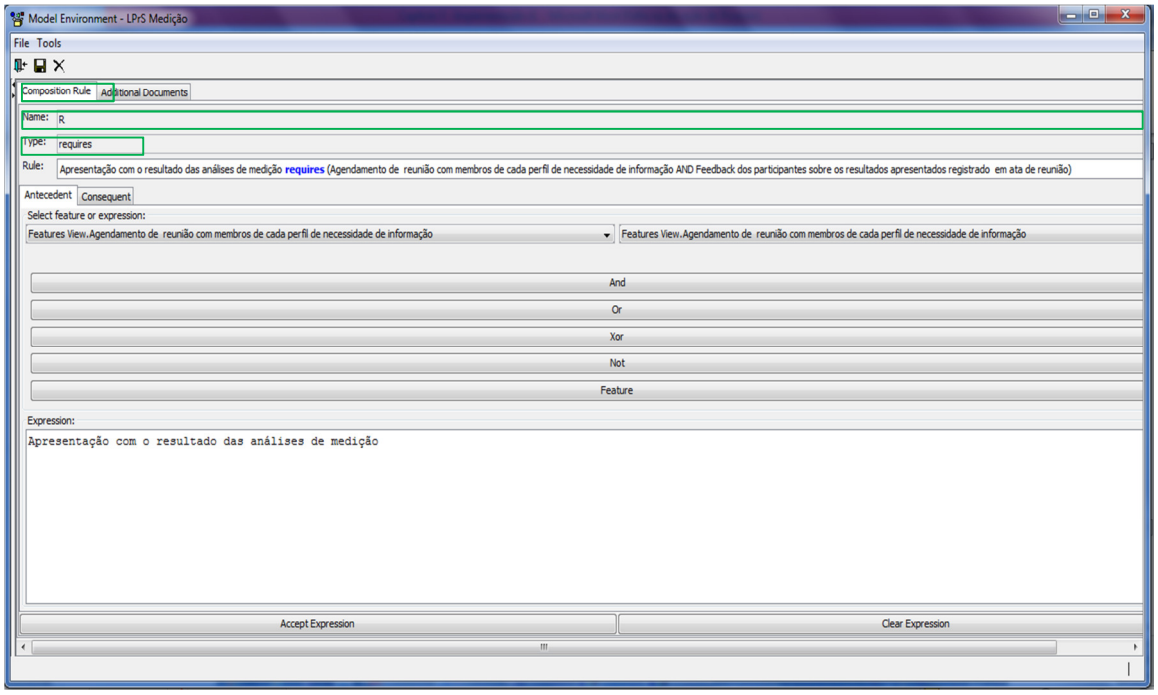
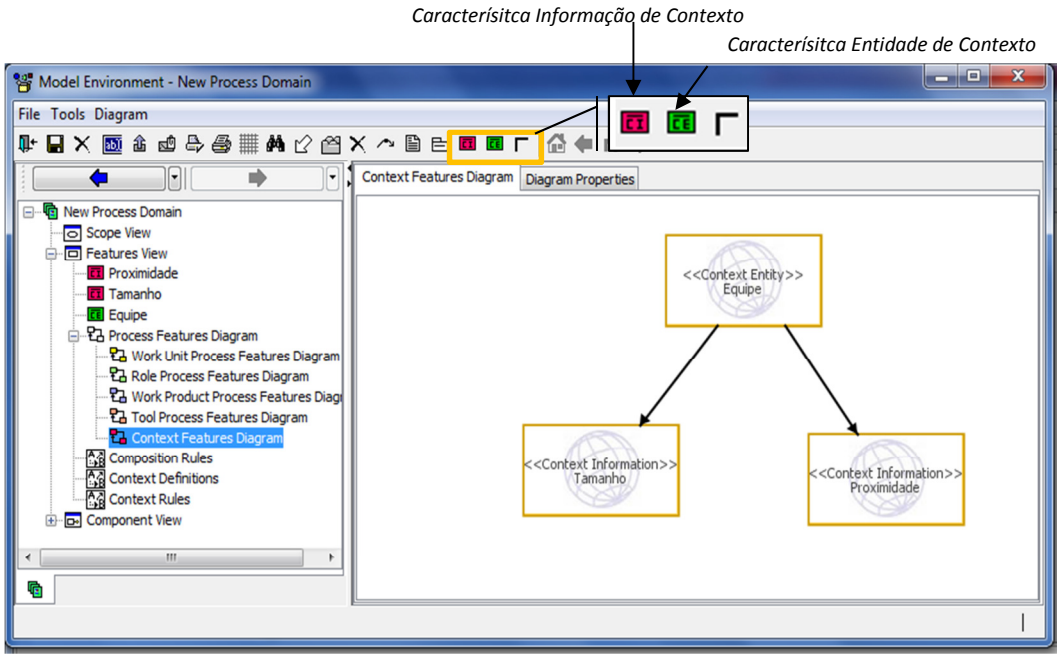


Figura 6.13 – Edição de Regras de Composição no ambiente

A modelagem do contexto é realizada por meio de um diagrama específico para este tipo de características (características de entidade e informação de contexto). Um exemplo deste tipo de modelagem pode ser visto Figura 6.14a. Com as características criadas e relacionadas, as definições de contexto são especificadas atribuindo valores às situações de contexto definidas (Figura 6.14b). Regras de contexto são criadas após a definição do contexto utilizando um painel de edição similar ao das regras de composição conforme pode ser visualizado na Figura 6.14c.

(a)



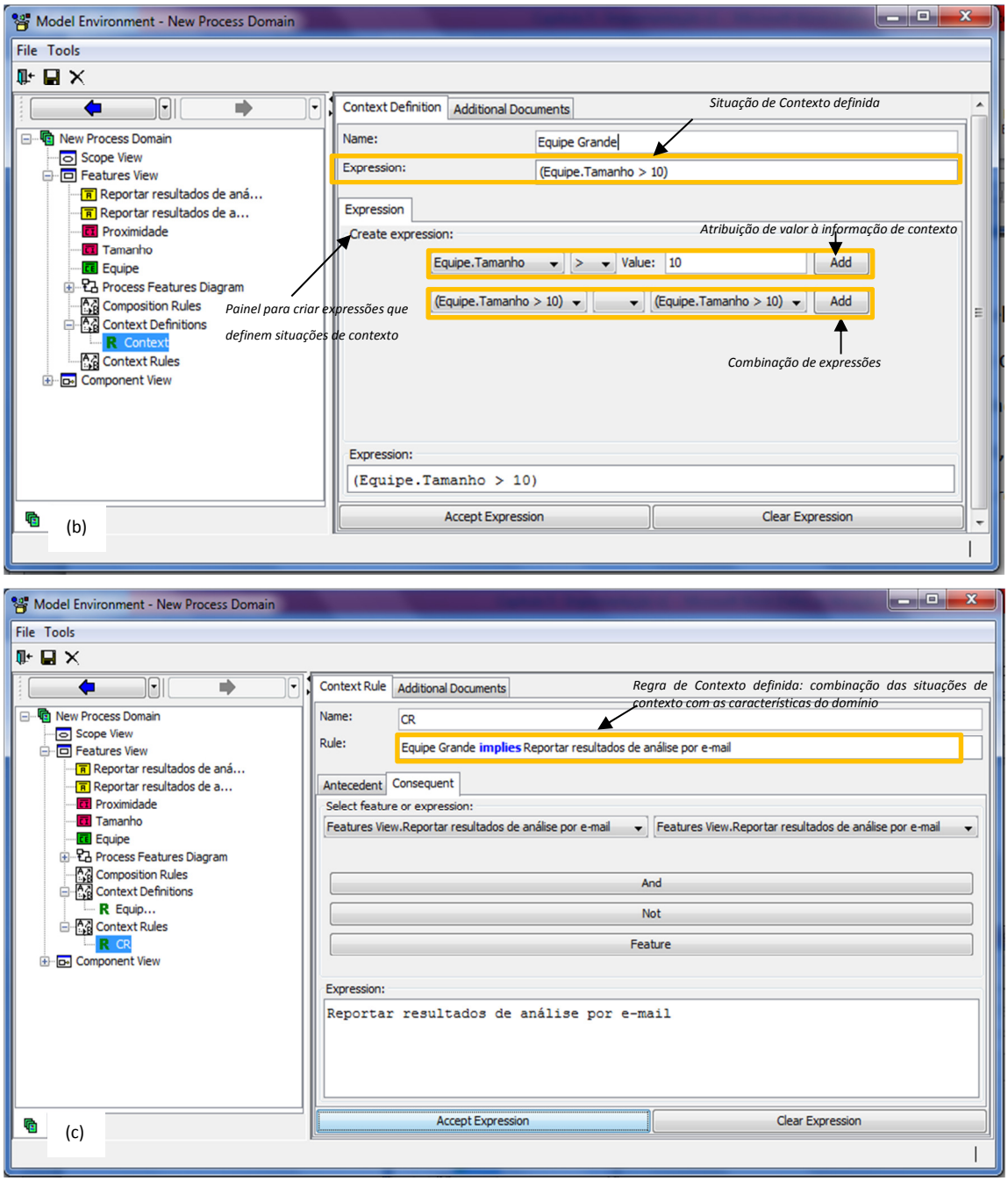


Figura 6.14 – Modelagem de Contexto no ambiente Odyssey

No intuito de viabilizar uma verificação da correta aplicação dos elementos definidos no metamodelo *OdysseyProcess-FEX,* para a construção de modelos de características dentro do domínio de processos de software, foi implementado neste trabalho um mecanismo que acompanha a atividade de modelagem e intervém na medida que inconsistências são introduzidas no modelo. O mecanismo monitora a inclusão, remoção e edição de elementos no ambiente Odyssey, enviando notificações aos desenvolvedores a cada vez que as regras de boa formação de um modelo são desrespeitadas.

A interação do mecanismo criado ocorre com a notificação ao usuário por mensagens enviadas no momento da ação que gerou a inconsistência e também de algumas restrições de modelagem diretamente implementadas no diagramador. Na Figura 6.15, podemos ver uma situação de atuação do mecanismo. Neste caso, o usuário é notificado da inconsistência gerada ao tentar associar uma variante mandatória a um ponto de variação opcional. A inconsistência é decorrente do fato que a ausência do ponto de variação em um processo instanciado deve implicar na ausência das variantes a eles associadas. Assim, nenhuma de suas variantes pode ser mandatória.

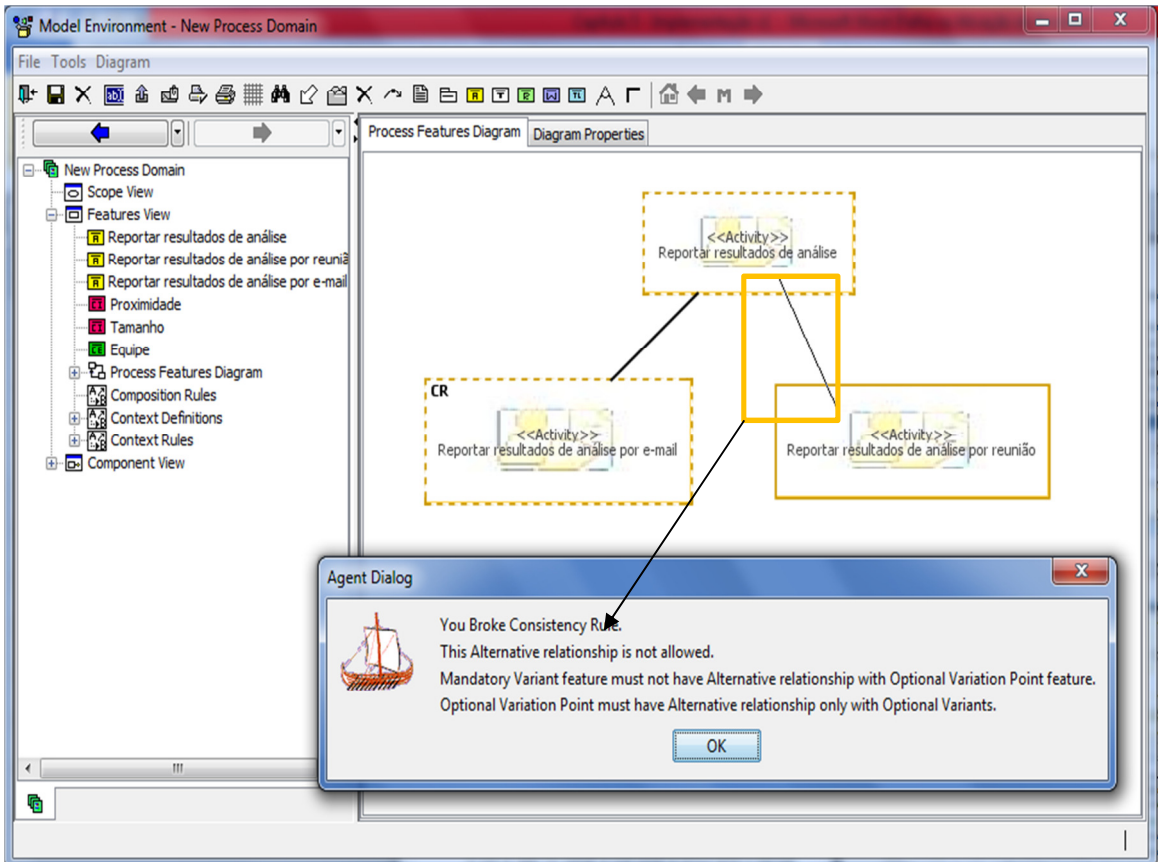


Figura 6.15 – Exemplo de notificação de inconsistência gerada pelo mecanismo de verificação

Neste trabalho, a definição das críticas e regras que são verificadas está amarrada ao código. Como trabalho futuro, a intenção é modificar o mecanismo para que seja incluído em um sistema de críticas que utiliza um arquivo descritor para armazenar as informações sobre as críticas e regras. O Oráculo (DANTAS *et al.,* 2001), mecanismo de críticas, foi desenvolvido dentro como *plug-in* do ambiente Odyssey e é uma ferramenta auxiliar à diagramação para realizar a verificação de consistências dos modelos implementados. Pode ser ativado de forma opcional, a cargo da necessidade

e escolha pelo usuário. O objetivo é realizar uma extensão nesta ferramenta para também contemplar as regras do metamodelo *OdysseyProcess-FEX*.

6.3.3. Nível de Componentes

O nível de abstração de componentes, denominado no Odyssey de "*Component View*" (Figura 5) tem por objetivo modelar a LPrS no nível de projeto, descrevendo a LPrS por meio de um modelo de componentes, incluindo a organização dos elementos de processo que compõem o domínio em módulos de construção de processos denominados componentes, mantendo a consistência com a especificação de suas propriedades de variabilidade e opcionalidade descritas no nível de análise.

Para isso, são providas telas para edição de componente de processo. A primeira tela apresenta um conjunto de campos para edição dos itens principias de um componente: nome, propósito, pré e pós-condições, restrições, tipo de variabilidade, tipo de opcionalidade, tipo de estrutura interna e tipo de variação interna (Figura 6.16). Além disso, informações adicionais sobre o componentes podem ser especificadas no campo Descrição.

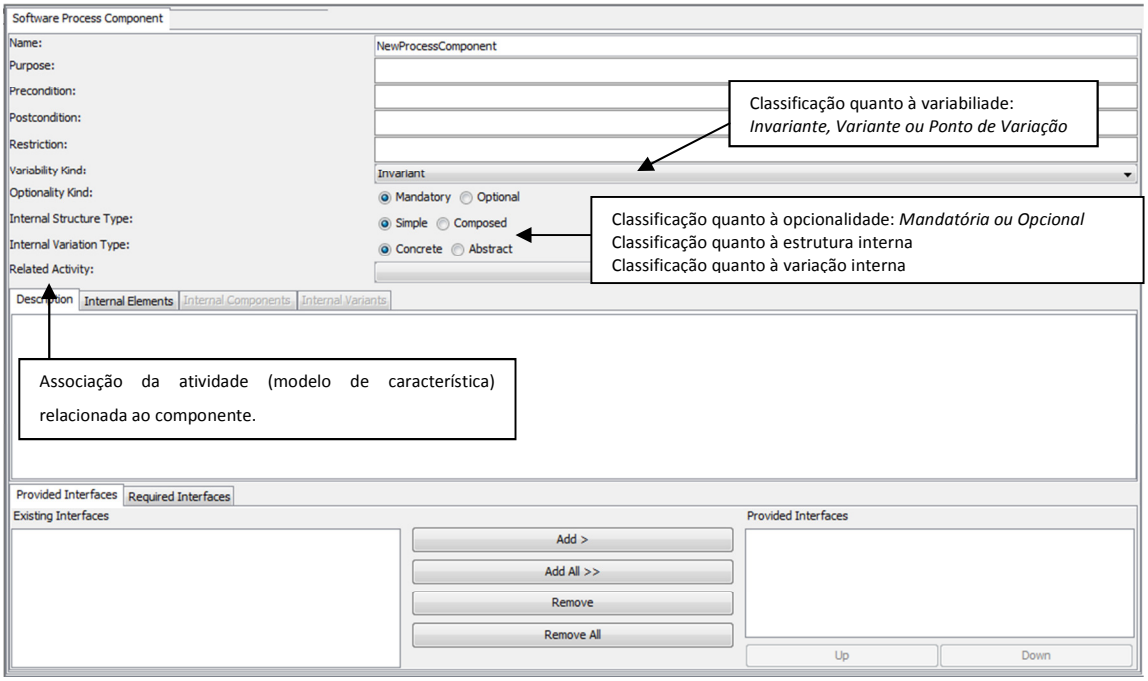


Figura 6.16 – Tela Principal de Edição de Componentes no ambiente Odyssey Nível Projeto

As abas de edição de componentes internos e variantes só são habilitadas de acordo com a classificação atribuída ao componente. Desta forma, ele precisa ser classificado como composto para habilitar a primeira aba, e classificado como ponto de

variação para habilitar a segunda. A Figura 6.17 apresenta um exemplo das abas ativas para um componente que é um ponto de variação composto.

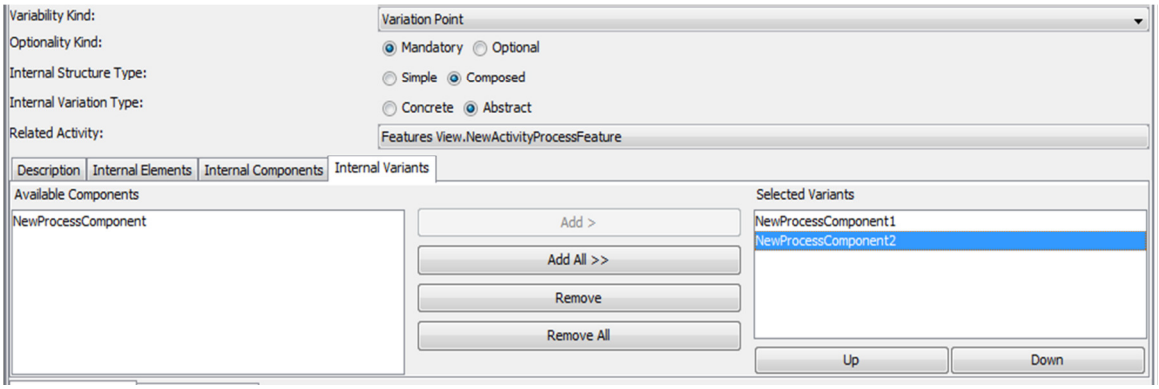


Figura 6.17 – Tela Principal de Edição de Componentes Abas Componentes internos e variantes internas

Ao selecionar os componentes que compõem o componente selecionado, automaticamente as interfaces desses componentes são processadas e um diagrama de componentes é adicionado ao componente que está sendo especificado. A Figura 6.18 apresenta um exemplo desta funcionalidade, após selecionar os componentes 1 e 2 para compor o componente editado, as interfaces deste são configuradas de acordo com as interfaces dos outros componentes. O componente 1 prove as interfaces 1 e 2, e requer a interface 3. Enquanto que o componente 2 prove a interface 4 e requer a interface 0 e 2. Desta forma, o componente editado passa a prover as interfaces 1, 2 e 4; e requer as interfaces 0 e 3.

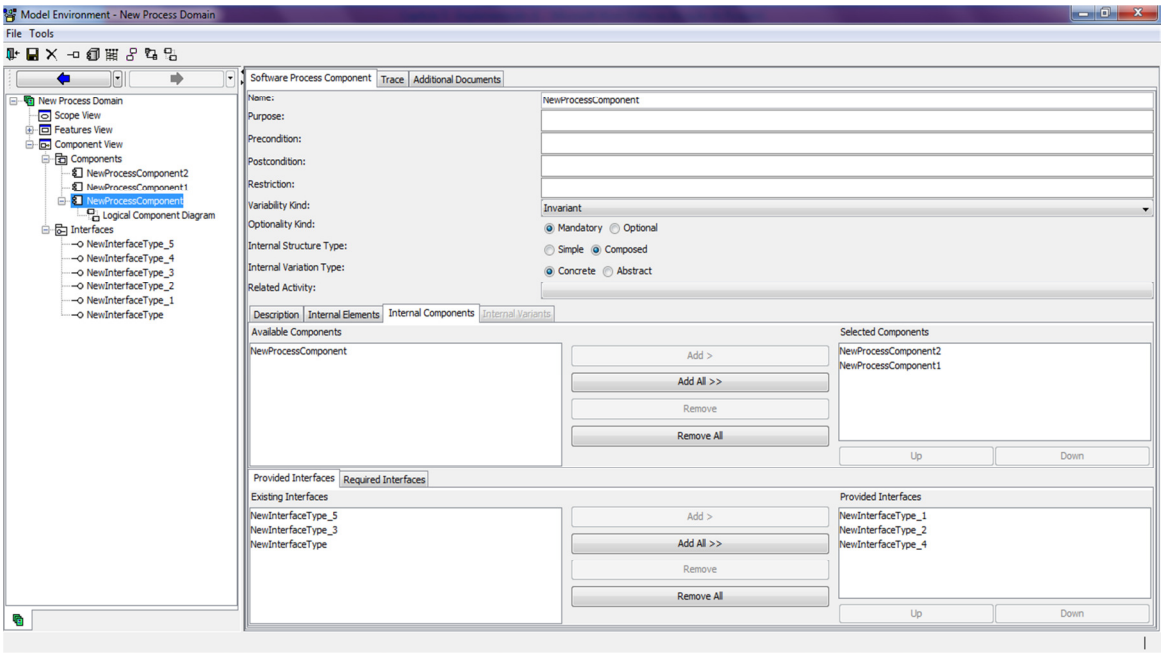


Figura 6.18 – Tela Principal de Edição de Componentes - Aba Componentes internos

O diagrama gerado como resultado da seleção interna descrita acima pode ser visualizado na Figura 6.19. Os dois componentes que compõem o elemento selecionado são apresentados com as suas interfaces correspondentes. Esse diagrama é gerado automaticamente pela ferramenta.

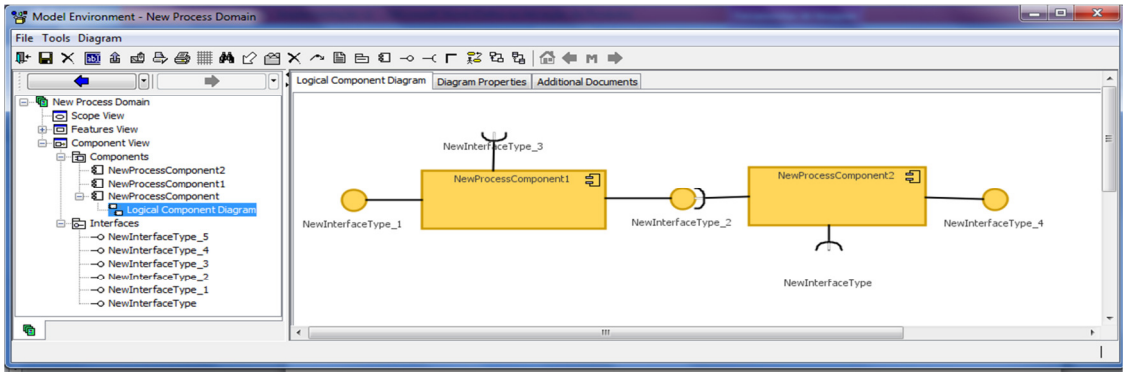


Figura 6.19 –Componentes internos – Diagrama criado após seleção de componentes

O mesmo funcionamento pode ser observado para o caso da edição dos componentes variantes de um componente classificado como ponto de variação. Neste caso, a classificação quanto à variabilidade é tratada em cada um dos componentes envolvidos, com os componentes selecionados sendo classificados como variantes automaticamente e o componente sendo editado como ponto de variação. Além disso, o componente ponto de variação passa a ser classificado como composto e um diagrama é criado com as variantes selecionadas.

Cada componente possui um campo para edição dos elementos internos que o compõem (Figura 6.20). Essa edição é realizada pela seleção dos itens listados a esquerda da tela. Esses itens são providos a partir do modelo de características especificado.

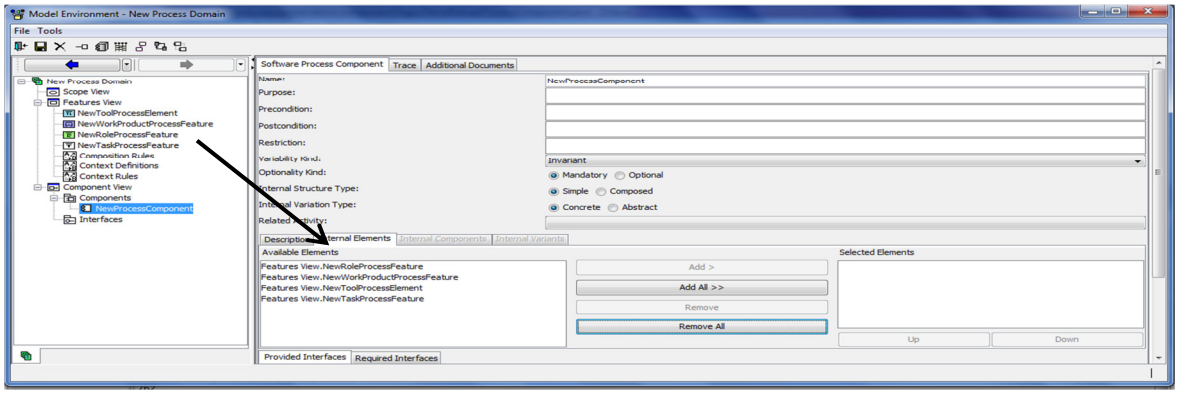


Figura 6.20 – Tela Principal de Edição de Componentes – Elementos Internos

Ao selecionar os itens relacionados, os rastros entre os dois níveis de abstração são estabelecidos e podem ser visualizados na aba Trace de cada elemento relacionado (Figura 6.21).

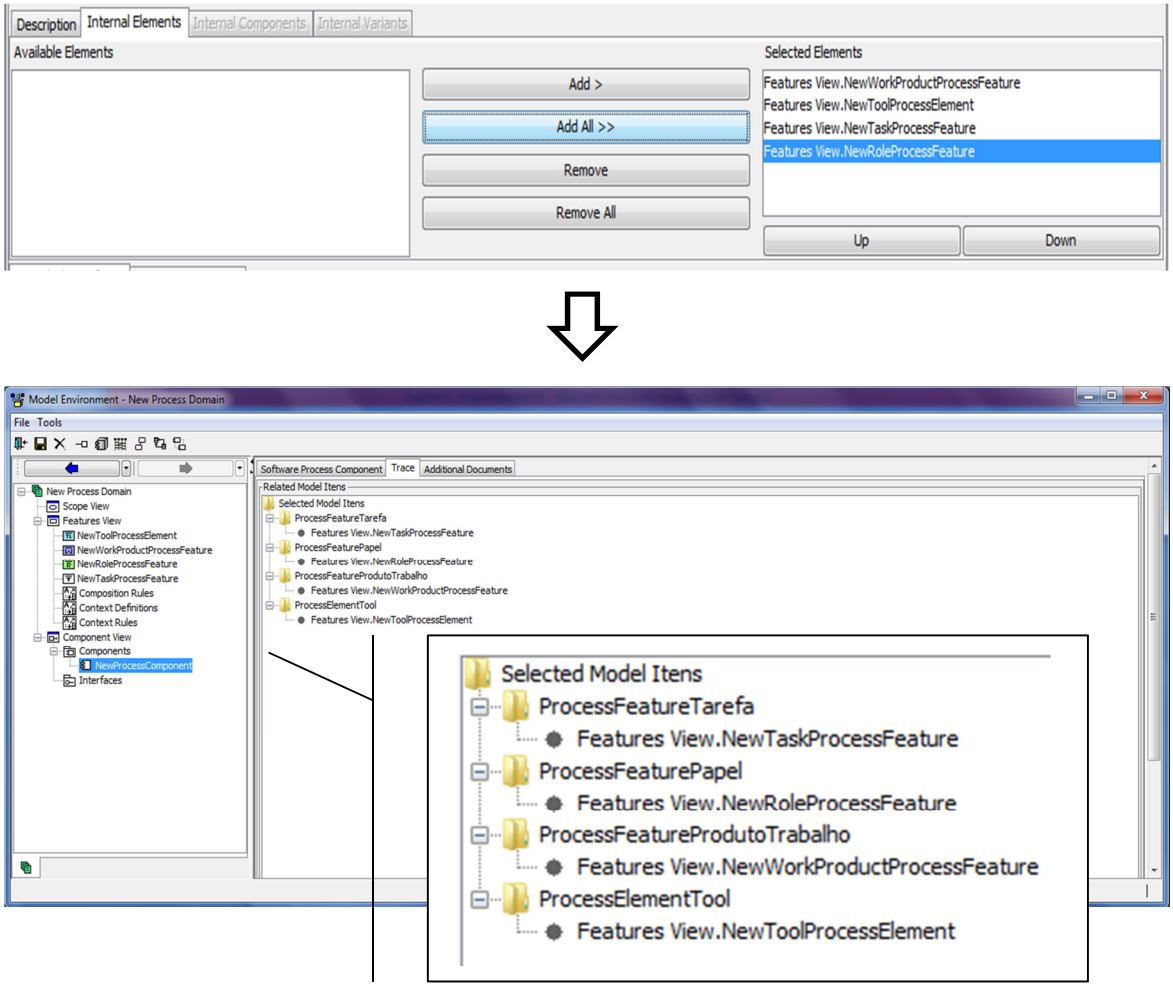


Figura 6.21 – Tela Principal de Edição de Componentes – Elementos Internos rastreabilidade com os elementos do modelo de características

A Aba *Measures* (Figura 6.22), associada a cada componente criado, disponibiliza um painel de edição das possíveis medidas a serem utilizadas para acompanhar o desempenho do componente e analisar o seu comportamento. Para cada medida, um conjunto de propriedades pode ser definido como nome, descrição, tipo da escala, valores que podem ser assumidos pela medida, fórmula para calcular a medida, procedimento de medição, quando e com periodicidade a medida deve ser coletada, tipo da distribuição da medida, e gráficos de controle que podem ser usados para análise da medida. Esse conjunto de propriedades é uma proposta inicial e pode ser estendido para atender objetivos de medição.

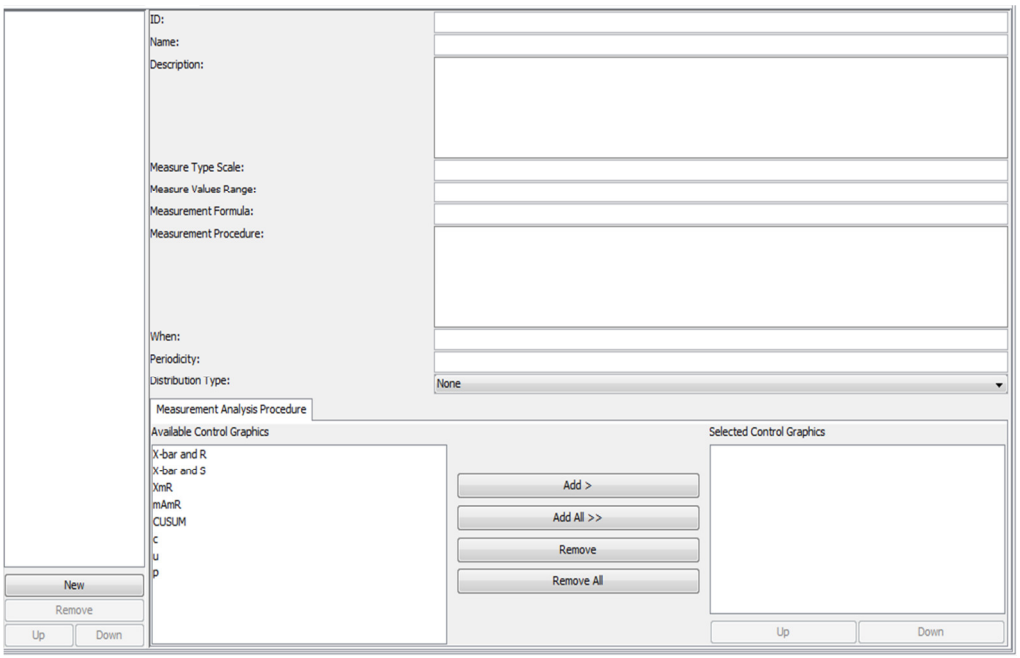


Figura 6.22 – Tela de Edição Medidas de Componentes

Baselines de desempenho podem ser associadas a componentes de processos de software e especificam valores mínimos e máximos possíveis de serem atingidos por um determinada medida em um determinado contexto de aplicação do componente. Os valores atribuídos aos limites de controle inferiores e superiores auxiliam a seleção de componentes na derivação de processos específicos pois especificam como o componente se comporta de acordo com o aspecto medido dentro de contexto coletado. Desta forma, a cada *baseline* de desempenho de um componente criada são especificados: (1) a medida associada; (2) os projetos em que os valores da medida foram coletados junto a sua caracterização de contexto; e (3) os valores dos limites de controle superior e inferior (Figura 6.23).

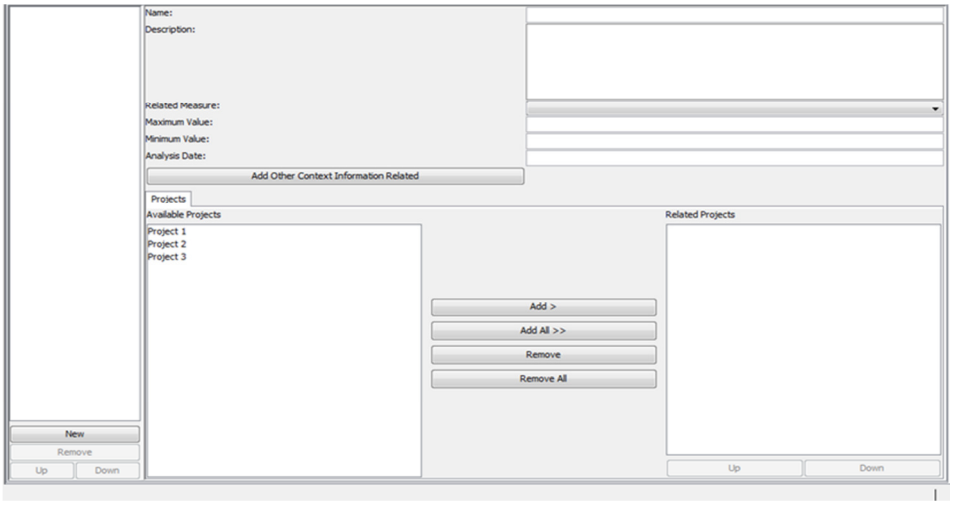


Figura 6.23 – Tela de Edição *Baselines* de Componentes

Interfaces de dados são editadas por meio de um painel que permite associar um produto de trabalho e especificar o tipo de opcionalidade relacionado à interface (Figura 6.24). A opcionalidade da ligação entre a interface e o componente é especificada por meio de um painel de edição da ligação, visível no painel de diagramação de componentes. O tipo da interface como provida ou requerida é especificado pela associação com o componente relacionado.

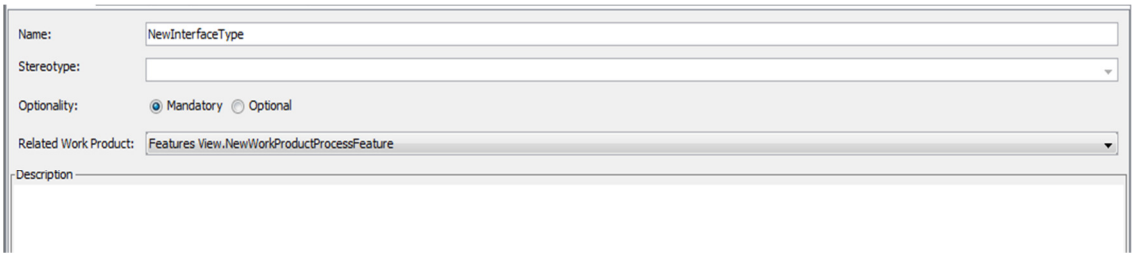


Figura 6.24 – Tela de Edição de Interfaces

6.4. HEURÍSTICAS DE MAPEAMENTO

O processo de mapeamento é dividido em três etapas. Após clicar na opção de mapeamento do nível de análise para o nível de projeto, uma tela é exibida informando as heurísticas de mapeamento disponibilizadas pela proposta (Figura 6.25).

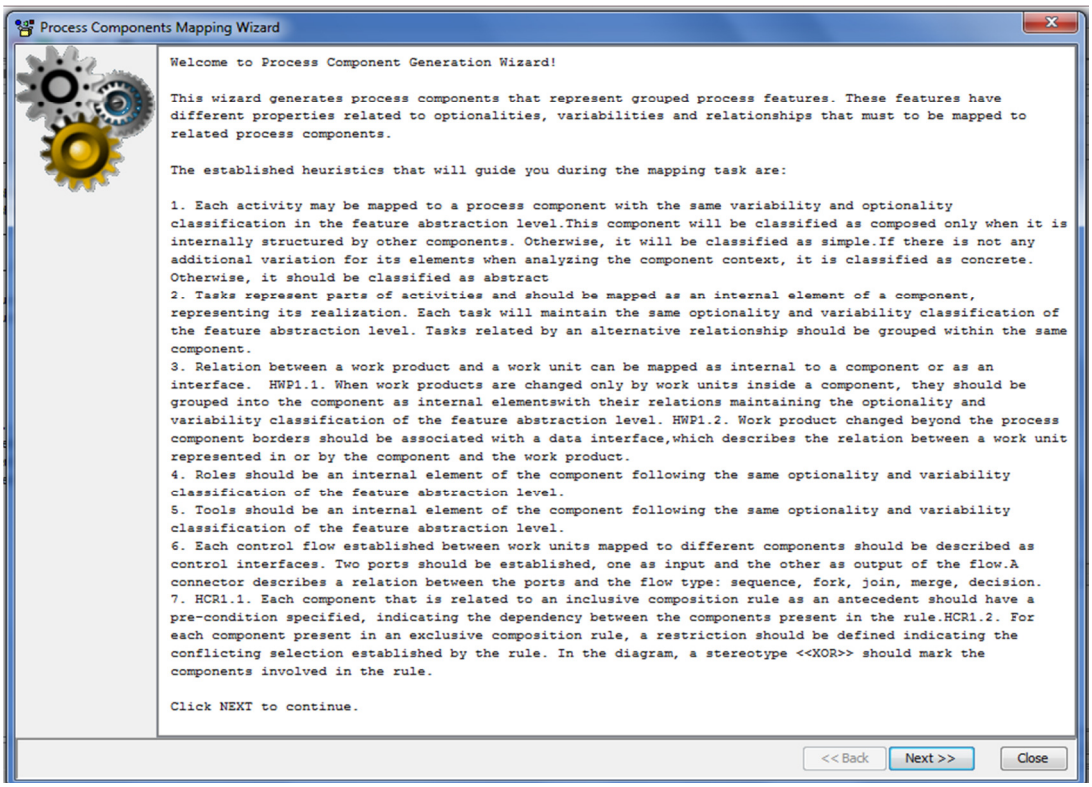


Figura 6.25 – Tela Inicial do Procedimento de Mapeamento do Modelo de Características para o Modelo de Componentes

Na continuidade do procedimento, é dada ao usuário a primeira possibilidade de interação com o processo, em que os nomes dos pacotes de componentes e interfaces a serem criadas devem ser especificados nesta tela (Figura 6.26).

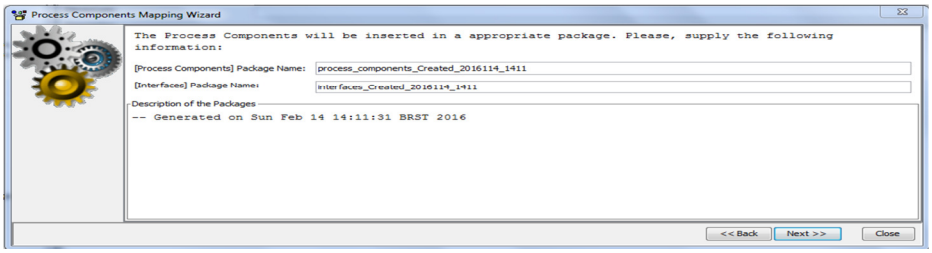
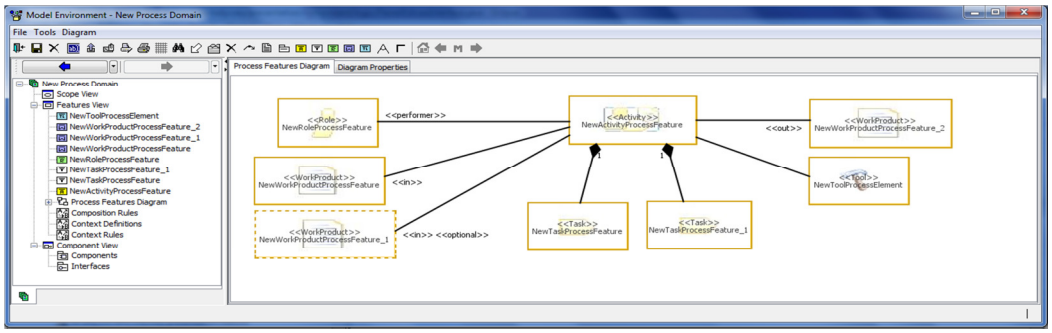


Figura 6.26 – Segunda Tela do Procedimento de Mapeamento do Modelo de Características para o Modelo de Componentes – Edição dos nomes dos pacotes

A última etapa apresenta uma árvore com os elementos que serão mapeados associados à heurística correspondente (Figura 6.27b). A árvore é organizada com o agrupamento dos elementos por cada atividade presente no modelo de característica que será mapeada como componente. Na Figura 6.27a é apresentado o modelo de característica a ser mapeado no exemplo da aplicação do mecanismo de mapeamento provido.



(a)

(b)

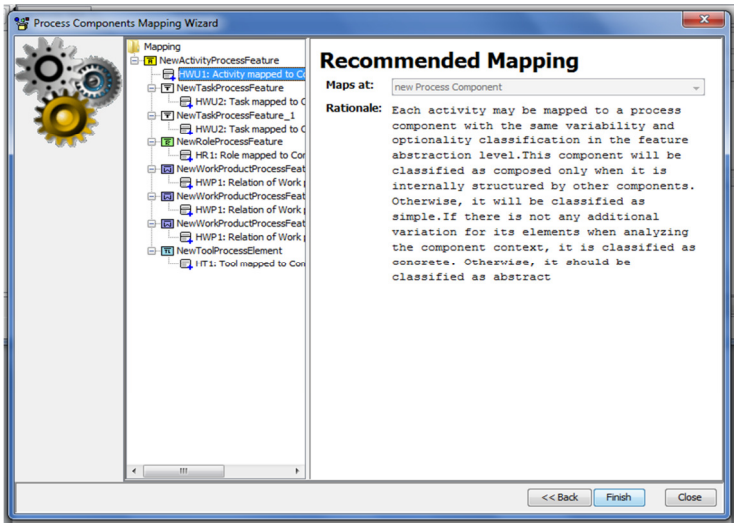


Figura 6.27 - Mapeamento de um Modelo de Características para um Modelo de Componentes

Ao final do mapeamento são criados dois pacotes no nível de componentes que correspondem a: (1) pacote com os componentes criados e (2) pacote com as interfaces criadas (Figura 6.28). As propriedades são mapeadas mantendo a consistência com o modelo de características.

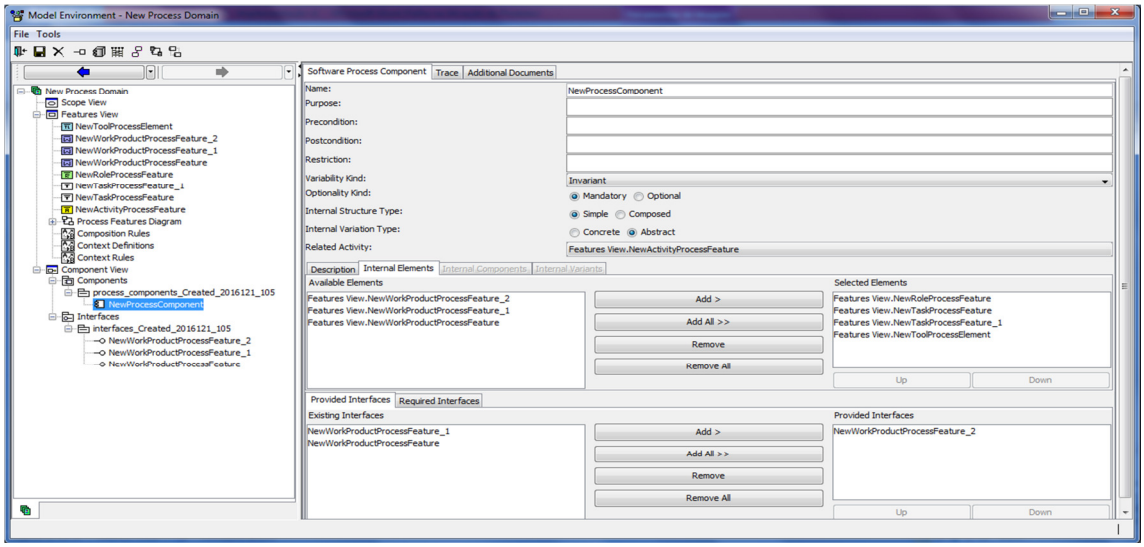


Figura 6.28 – Resultado do Mapeamento de um Modelo de Características para um Modelo de Componentes

6.5. CONSIDERAÇÕES FINAIS

Neste capítulo, foi a descrita a implementação realizada no contexto do ambiente Odyssey, com o intuito de viabilizar a adaptação do ambiente para apoiar o desenvolvimento de LPrS. Desta forma, o ambiente foi flexibilizado para trabalhar tanto com LPS quanto com LPrS. As principais atividades de modelagem foram implementadas no ambiente, assim como toda estrutura para detalhamento de cada elemento definido pelo metamodelo e notação *OdysseyProcess-FEX*. Desta forma, é possível modelar escopos, contexto, características e componentes no ambiente. O mapeamento entre os níveis de abstração também se encontra implementado.

O apoio ao agrupamento por meio dos critérios propostos e das visões de dependências, responsabilidades e instabilidade ainda precisam ser implementados. A viabilidade de implementação da proposta de agrupamento de componentes como uma adaptação do *plug-in GroupCriteria* (BLOIS, 2006) encontra-se em análise para atender esse item de implementação.

O apoio à fase de desenvolvimento para reutilização também deve ser implementado para aumentar o potencial de reutilização dos artefatos criados.

Capítulo 7 – Conclusão

7.1. EPÍLOGO

Organizações têm investido na melhoria de seus processos como forma de alcançar o alinhamento entre suas funções operacionais e seus objetivos estratégicos, fornecendo serviços e produtos de qualidade e obtendo ganhos de produtividade. A melhoria dos processos envolve desde o seu planejamento frente às necessidades do negócio, seu entendimento e acompanhamento frente aos impactos e benefícios resultantes de sua execução, e até a análise dos custos advindos das adaptações necessárias a sua evolução. Essas etapas constituem o ciclo de vida da Engenharia de Processos, que se inicia com a definição dos processos como forma de explicitar o que precisa ser feito e melhorar a forma pelo qual o trabalho de desenvolvimento de um software deve ser realizada. Tal atividade envolve o tratamento de diversos fatores, a identificação e combinação de diferentes elementos, precisa lidar com a especificidade de projetos e contextos organizacionais e, em alguns casos, com a conformidade com padrões ou modelos de referência, aspectos que provocam adaptações e inviabilizam a existência de processos genéricos diretamente aplicáveis a todas as situações. Desta forma, o acúmulo de conhecimento e experiência são aspectos fundamentais dos recursos envolvidos nesta atividade e em todo o ciclo de vida da Engenharia de Processos, o que acarreta a agregação de responsabilidades a gerentes de projetos e equipes de desenvolvimento de processos experientes.

Dentro desse contexto, técnicas de reutilização têm sido propostas como forma de auxiliar a equipe de processos de software a tomar decisões mais adequadas na definição de processos específicos de projeto. O propósito da área de reutilização de processos de software é facilitar a definição de processos, diminuindo o custo e o esforço associado à atividade, além de possivelmente aumentar a qualidade dos processos gerados, inclusive tornando a realização da atividade acessível a profissionais menos experientes (BARRETO, 2007). A dificuldade na reutilização de processos de software reside no fato que a variabilidade dos requisitos dos projetos frequentemente implica na variabilidade dos processos de software (ROUILLÉ *et al.*, 2013).

Para alcançar uma reutilização de processos efetiva, o uso de uma sistemática que combine técnicas, que gerenciem a variabilidade inerente a uma família de processos de forma a organizar a informação reutilizável em blocos de construção modulares, tende a ser promissora. A ideia central é unir os benefícios da área de LPrS com os da área de DPBC, promovendo a definição de processos específicos de projetos com base em conhecimento reutilizável, representado e disponibilizado de forma inteligível por parte dos envolvidos com o uso dessa informação .

A representação da informação reutilizável e a forma de disponibilização de dados de conhecimento e experiências relacionadas foram desafios tratados na metodologia aqui proposta. A tomada de decisão, em geral, realizada por gerentes de projeto, abrange o alinhamento de objetivos de qualidade e desempenho de processo estabelecido para o projeto com restrições impostas por múltiplos fatores e conhecimento das possibilidades de configuração de um domínio de processos de software. Uma estrutura que apoie a organização deste conhecimento durante o desenvolvimento dos artefatos a serem reutilizados é de extrema importância.

7.2. CONTRIBUIÇÕES

Diante do exposto, o objetivo central deste trabalho foi definir uma metodologia de apoio à representação e organização do conhecimento de domínios de processo de software, de forma a aumentar o potencial de reutilização com a identificação inteligível de alternativas para definições de processos de software específicos de projeto. A proposta de solução para este objetivo foi combinar as técnicas de LPrS e DPBC com mecanismos para: (1) gerenciar a representação explícita de variabilidade e opcionalidade de um domínio de processos de software em diferentes níveis de abstração e modelos; (2) modularizar tal conhecimento em blocos de construção denominados componentes; e (3) definir procedimentos para guiar a construção dos artefatos de processos de software reutilizáveis. Como contribuições alcançadas deste trabalho incluem-se:

1. **Caracterização das áreas de LPrS e DPBC**, pela condução de revisões da literatura e dos resultados descritos ao longo do Capítulo 2.
2. **Especificação de um método para Engenharia de LPrS, com foco no processo de desenvolvimento *para* reutilização**, visando fornecer apoio ao Engenheiro

de Processos de Software no desenvolvimento e organização dos artefatos do domínio de processos que serão reutilizados. Para isso, cada etapa envolvida foi descrita, contendo atividades, produtos esperados e mecanismos de solução e suporte à sua execução.

3. **Definição do apoio à modelagem de variabilidades de domínios de processos de software.** Um conjunto de requisitos necessários para modelagem de variabilidades em LPrS foi especificado. Um metamodelo e uma notação, denominados *OdysseyProcess-FEX*, foram definidos para apoiar a representação de artefatos de domínio de processos de software em níveis complementares de abstração. A representação proposta contempla tanto a parte estrutural de processos quanto comportamental (fluxos de controle e de dados), o tratamento das propriedades de variabilidade e opcionalidade, definição de restrições de dependências e mútua exclusividade via regras de composição, e organização da informação em unidades de composição de processos de software denominadas componentes. Modelos para caraterização dos contextos e fatores que impactam a seleção dos elementos reutilizáveis também foram especificados em conjunto com os modelos de características e de componentes.
4. **Definição de heurísticas para mapeamento dos diferentes artefatos do domínio e especificação de rastreabilidade**. A captura do relacionamento de um elemento em uma dada abstração com o mesmo elemento representado em outro nível de abstração deve ser feita por mapeamentos que permitem estabelecer a rastreabilidade bidirecional. Heurísticas foram definidas para guiar o mapeamento dos elementos que compõem um modelo de características para um modelo de componentes de processos de software, mantendo a consistência e tratando as propriedades de variabilidade e opcionalidade dos elementos e das relações.
5. **Representação de uma LPrS com módulos de informação denominados componentes que se comunicam via interfaces.** A definição do conceito de componente foi especificada indicando as propriedades que o caracterizam, como organizar os elementos que o compõem e como estabelecer as comunicações com outras componentes via interfaces de dados e de controle.

Alguns dados de execução podem ser especificados junto à definição de um componente, como a especificação de medidas e *baselines* de desempenho para análise do comportamento, capacidade e estabilidade do componente. Além disso, atores podem ser associados aos papéis envolvidos na execução do componente. Assim como *templates* podem ser associados a produtos de trabalho e *scripts* associados a ferramentas. Desta forma, informações e dados de projetos anteriores podem prover maior conhecimento a ser aplicado em novas definições.

6. **Especificação de critérios para apoiar o agrupamento de componentes.** A organização dos componentes em elementos que tratam o acoplamento dentro do domínio é importante para aumentar o entendimento dos módulos a serem reutilizados e melhorar a capacidade de manutenção da LPrS representada. Desta forma, critérios foram definidos para apoiar o engenheiro de domínio no agrupamento de componentes visando organizar os componentes em elementos arquiteturais de maior granularidade, reduzindo o número de componentes e o alto número de relações entre eles. Perspectivas de visualização dos componentes e suas relações foram definidas para apoiar o entendimento do grau de responsabilidades, dependências e instabilidades do domínio.
7. **A construção de apoio ferramental como infraestrutura de utilização da abordagem proposta**. Levando em consideração a complexidade da atividade de definição de processos, ainda que apoiada por uma sistemática de reutilização, uma infraestrutura de apoio ao desenvolvimento de LPrS foi desenvolvida como contribuição desta tese. O apoio visa prover uma visão complementar e conjunta dos diferentes elementos que compõem uma LPrS e fornecer algum apoio automatizado a utilização dos procedimentos definidos pelo método desta metodologia e a construção dos artefatos reutilizáveis e suas ligações. Este trabalho foi desenvolvido dentro do contexto do ambiente Odyssey (ODYSSEY, 2016). A ideia é criar um ambiente unificado de apoio as diferentes fases da reutilização de processos de software.
8. **Especificação de mecanismos de verificação dos artefatos do domínio construídos.** A consistência dos artefatos criados é de extrema importância

para evitar a propagação de erros na linha e dos processos a serem derivados. Desta forma, mecanismos de análise dos artefatos devem ser previstos como forma de verificação. Inicialmente, técnicas de inspeção baseadas em *checklists* foram especificadas com esta finalidade.

Algumas publicações foram realizadas ao longo do desenvolvimento deste trabalho:

- TEIXEIRA, E. N. E WERNER, C. (2013) "*OdysseyPrLE-CBArch*: A Process Reuse Approach Combining the Software Process Line and Process Component Based Architecture Approaches.", I*II Workshop de Teses e Dissertação de Software do CBSOFT* (WTDSOFT 2013), pp. 86 - 92.
- TEIXEIRA, E.N. (2014), "A Component-Based Software Process Line Engineering with Variability Management in Multiple Perspectives". In: *18th International Software Product Line Conference Doctoral Symposium*, 2014, Florence. Proceedings of Doctoral Symposium - 18th International Software Product Line Conference, 2014.
- MELLO, R. M. ; TEIXEIRA, E. N. ; SCHOTS, M. ; WERNER, C. ; TRAVASSOS, G. H., 2014, "Verification of Software Product Line Artefacts: A Checklist to Support Feature Model Inspections". *Journal of Universal Computer Science (J.UCS), Special Issue on Software Components, Architectures and Reuse* (Print), v. 20, no. 5, p. 720-745, 2014, ISSN 0948-695x.
- TEIXEIRA, E. N., de MELLO, R. M., MOTTA, R., WERNER, C., VASCONCELOS, A. (2015), "Verification of Software Process Line Models: A Checklist-based Inspection Approach". In: *XVIII Iberoamerican Conference on Software Engineering (CIbSE)*, v. 1. p. 81-91, 2015, Lima.
- TEIXEIRA, E. N., WERNER, C., VASCONCELOS, A. (2016) "*OdysseyProcess-FEX: Metamodelo e Notação para Representação de Variabilidades de Linha de Processos de Software*". Relatório Técnico ES-751/16, COPPE/UFRJ, Rio de Janeiro, Brasil, Março. Disponível em: http://www.cos.ufrj.br.

Ainda como resultado de pesquisas na área pode-se citar a participação na apresentação de Tutorial sobre "*Linha de Processos de Software: Explorando Variabilidades na Reutilização de Processos de Software*" no II Congresso Brasileiro de

Software (CBSOFT) (2011). Além disso, esta tese esteve inserida em projeto de pesquisa sobre reutilização de processos de software, denominado CDSOFT[17] (WERNER *et al.,* 2011).

Atualmente, uma publicação sobre a revisão *quasi*-sistemática realizada está sendo desenvolvida em parceria com pesquisadores do Rio Grande do Norte (UFRN) e do Paraná (UEM). Para isso, a revisão está sendo atualizada para coletar as publicações mais atuais.

7.3. LIMITAÇÕES

Algumas limitações podem ser identificadas dentro do conjunto de elementos que compõem a metodologia proposta. Com relação ao método proposto, a maior limitação é o apoio restrito oferecido à etapa de desenvolvimento *com* reutilização. Ainda que um conjunto de atividades tenha sido definido, a forma de utilização de uma LPrS não foi devidamente explorada. No entanto, essa fase da Engenharia de LPrS não foi foco desta tese e pretende ser pesquisada em oportunidades futuras.

Outro ponto a ser ressaltado é que para algumas atividades iniciais da fase de desenvolvimento *para* reutilização apenas foram indicados possíveis mecanismos de apoio, mas não foram definidos métodos concretos para sua realização. Por exemplo, a atividade de captura do conhecimento sobre um domínio de processo e análise de suas similaridades e variabilidades podem apresentar diferentes mecanismos e técnicas para sua efetiva realização. No entanto, entende-se que cada uma dessas atividades abrange um esforço e complexidade que se encontram fora do escopo de desenvolvimento de apenas uma tese e devem ser abordados em projetos de pesquisa e trabalhos colaborativos.

Com relação ao apoio à modelagem de processos, pode-se citar que a etapa de representação de processo de forma implementável e executável por máquinas de processos não foi abordado. Como o foco foi à etapa de definição de processos de software, aspectos de instanciação e execução não foram diretamente tratados como requisito para representação de conhecimento do domínio, mas entende-se que tais informações podem afetar o processo final a ser efetivamente executado e podem ser explorados em trabalhos futuros.

[17] CDSOFT - http://reuse.cos.ufrj.br/cdsoft/

Com relação ao agrupamento de componentes, se propôs um conjunto inicial de critérios para apoiar a análise do engenheiro de domínio, visando o acoplamento como propriedade a ser considerada. Esse conjunto inicial indicou que é possível agrupar componentes em elementos que trabalham com alta interação no domínio, visando aumentar o entendimento do domínio e a manutenção dos artefatos e da linha, já que elementos relacionados são mais facilmente identificados e impactos de alterações podem ser melhor gerenciados. Outros aspectos podem ser utilizados para análise e devem expandir o conjunto de critérios aqui proposto. Um apoio automatizado à análise pode viabilizar uma melhor organização desses componentes e ainda precisa ser explorado.

Mecanismos de verificação dos modelos de processos foram propostos por meio de técnicas de inspeção baseadas em *checklists* e das regras de boa formação do metamodelo. Esses mecanismos auxiliam a manter a consistência das propriedades do domínio e buscam minimizar a ocorrência de erros nos processos a serem derivados. No entanto, métodos mais formais podem ser pesquisados para garantir a ocorrência correta dos fluxos a serem definidos.

Quanto ao ferramental de apoio, uma limitação é que não foi possível implementar todos os mecanismos propostos na metodologia. A parte de verificação dos modelos e do agrupamento de componentes ainda precisa ser trabalhada. O *plug-in GroupCriteria* (BLOIS, 2006) encontra-se em análise para verificar sua adaptação e extensão para lidar com os critérios de agrupamento de componentes propostos. Além disso, o suporte computacional construído não foi avaliado. Desta forma, não há indicações da sua viabilidade de uso e necessidades de melhorias.

Ainda que estudos de viabilidade tenham sido conduzidos para análise dos diferentes elementos que compõem a metodologia proposto e a própria metodologia em si, tais estudos podem ser ainda considerados para captura de indícios dos retornos esperados com a aplicação da metodologia, devido ao escopo reduzido e parâmetros de avaliação apenas para caracterização. O desenvolvimento de um estudo com aplicação em cenários reais deve ser planejado para melhor entender a capacidade da metodologia proposta em melhorar o potencial de reutilização de processos de software.

7.4. CONSIDERAÇÕES FINAIS E OPORTUNIDADES FUTURAS

A área de processos ainda apresenta vários desafios nas diversas atividades que compõem seu ciclo de vida. Técnicas de reutilização podem ser aplicadas nas diferentes etapas como forma de melhorar a qualidade dos processos e dos produtos resultantes. Desta forma, um conjunto de oportunidades de melhorias foi identificado tanto para a metodologia proposta assim como extensões de pesquisa e trabalhos futuros.

O apoio à etapa de derivação de processos a partir da LPrS criada pode ser melhor explorado em pesquisas futuras. Mecanismos para apoio à seleção de alternativas mais adequadas ao contexto específico que o processo definido será aplicado podem ser desenvolvidos. A análise da similaridade entre projetos anteriores e o projeto atual, visando identificar oportunidades de reutilização mais próximas ao contexto em questão, é um campo de pesquisa que pode ser investigado. Além disso, o apoio à adaptação dos artefatos selecionados deve ser pensado de forma a trabalhar o impacto ao domínio especificado e evolução da LPrS definida. Além disso, ainda que determinados artefatos tenham sido escolhidos de acordo com os dados de execução, coletados de contextos específicos e armazenados como histórico, caso adaptações sejam necessárias, o comportamento do processo pode ser alterado. Apoio à análise desses impactos pode ser visto como um trabalho de pesquisa.

A definição de medidas que podem apoiar a seleção de componentes para contextos específicos pode ser outro campo de pesquisa. Por exemplo, o potencial de reutilização de um componente de processo é um dado relevante para apoiar a seleção entre as variantes do domínio associados a dados sobre sua estabilidade e capacidade histórica na execução de projetos anteriores. No entanto, a forma de representar esse potencial de reutilização ainda é um desafio e pode ser explorado com o intuito de definir tal conceito associado a componentes de processo de software.

Além da etapa de definição, devem ser especificados apoios às etapas de instanciação e execução de processos de software. Desta forma, a metodologia proposta pode ser estendida para tratar outros dados e elementos de processos que não foram considerados e são específicos dessas etapas do ciclo de vida de processos. A extensão pode considerar a representação dos componentes em níveis

implementacionais que tratam da execução. Transformações das possibilidades de configuração de cada componente para representações com linguagens de modelagens de processos existentes podem ser vistas como uma possibilidade para utilizar apoio a essas etapas citadas e já existentes na área de processos, com a integração de máquinas de processo com o ferramental desenvolvido.

Em um cenário onde exista uma LPrS complexa, com centenas de características, componentes e suas respectivas interfaces de comunicação, o número de alternativas possíveis para definição de um processo específico pode ser muito grande. Assim, mecanismos para visualização da informação a ser reutilizável é essencial para auxiliar o entendimento e tratar a complexidade da identificação e representação das informações envolvidas. Desta forma, recursos de visualização e interação podem ser explorados para representar diferentes camadas e perspectivas dos modelos de uma LPrS para atender as necessidades e aumentar a percepção e dos diferentes papéis envolvidos no ciclo de vida de processos como gerentes de projetos, engenheiros de processos, desenvolvedores de produtos e alta gerência.

Definição de LPrS para disciplinas específicas da área de processos de software com suporte ao atendimento de níveis de maturidade dos modelos de referência como MPS.Br (SOFTEX, 2016) podem ser considerados como necessidades para ampliar a capacidade de reutilização e disseminação do conhecimento sobre processos de software.

A análise da aplicabilidade da metodologia proposta para a área de processos de negócio pode ser realizada por meio de estudos. É importante destacar que processos de software podem ser entendidos como áreas de negócio de organizações, e como tal, os conceitos de modelagem de negócios aparecem, invariavelmente, nas iniciativas de definição de processos de software e, mais notadamente em sua posterior gestão e melhoria contínua (ARAUJO *et al*., 2004). Desta forma, estudos, que busquem analisar a aplicabilidade tanto das técnicas da área de processos de negócio para área de processo de software como o inverso, podem ser planejados para viabilizar o alcance dos benefícios já relatados da área de negócios para software e vice-versa. Um estudo para identificação das similaridades e diferenças entre esses tipos de processos ainda é um campo a ser explorado.

Referências

AALST, W. M. P.; GIINTHER, C. W., 2007,"Finding Structure in Unstructured Processes: The Case for Process Mining". In: *Application of Concurrency to System Design (ACSD)*, pp. 3–12, Bratislava, Slovakia.

AALST, W. VAN DER; WEIJTERS, T.; MARUSTER, L., 2004, "Workflow Mining: Discovering Process Models from Event Logs", *IEEE Transaction on Knowledge and Data Engineering*, v. 16, n. 9, pp. 1128–1142.

ACUÑA, S., FERRÉ, X., 2001, "Software Process Modelling". In: *Proceedings of the 5th. World Multiconference on Systemics, Cybernetics and Informatics (SCI 2001)*, pp. 1-6, Orlando Florida, USA.

ALEGRÍA, J. A. H., BASTARRICA, M. C., BERGEL, A., 2010, "Analyzing the Scrum Process Model with AVISPA". In: *XXIX International Conference of the Chilean Computer Science Society (SCCC)*, pp. 60-65, IEEE.

ALEGRÍA, J. A. H., BASTARRICA, M. C, 2012, "Building software process lines with CASPER". In*: International Conference on Software and System Process (ICSSP),* pp. 170-179, June, IEEE.

ALEIXO, F. A., FREIRE, M. A., DOS SANTOS, W. C., & KULESZA, U., 2010a, "A Model-driven Approach to Managing and Customizing Software Process Variabilities". In: *ICEIS 2010 - Proceedings of the 12th International Conference on Enterprise Information Systems*, v. 3 ISAS, pp. 92-100, Funchal.

ALEIXO, F., FREIRE, M. A., SANTOS, W., KULESZA, U., 2010b, "Uma Abordagem para Gerência e Customização de Variabilidades em Processos de Software". In: *Proceedings of the XXIV Simpósio Brasileiro de Engenharia de Software (SBES´2010),* Salvador, Brasil, Outubro.

ALEIXO, F. A., FREIRE, M., ALENCAR, D., CAMPOS, E., KULESZA, U., 2012, "A Comparative Study of Compositional and Annotative Modelling Approaches for Software Process Lines". In: *26th Brazilian Symposium on Software Engineering* (SBES), pp. 51-60, Natal, Rio Grande do Norte.

ALEIXO, F., 2013, *Uma Abordagem Anotativa para Gerência de Variabilidades em Linhas de Processos de Software: Concepção, Implementação e Avaliação*. Tese de D.Sc., UFRN, Natal, Brasil.

AMBLER, S.W., 1998, *Process Patterns: Building Large-Scale Systems Using Object Technology*, New York, United States, Cambridge University Press.

AOUSSAT, F., AHMED-NACER, M., OUSSALAH, M. C., 2010, "Reusing Approach for Software Processes based on Software Architectures". In *12th International Conference on Enterprise Information Systems,* pp. 366–369, Funchal, Madeira, Portugal.

AOUSSAT, F., OUSSALAH, M., NACER, M. A., 2011, "SPEM Extension with software process architectural concepts". In: *IEEE 35th Annual Computer Software and Applications Conference (COMPSAC)*, pp. 215-223, Munich.

AOUSSAT, F.; OUSSALAH, M.; AHMED NACER, M., 2014, "A SPEMOntology for Software Processes Reusing". *Computing and Informatics*, 2014, v. 33, n. 1, pp. 35-60.

ARANGO, G., PRIETO-DIAZ, R., 1991, "Introduction and Overview: Domain Analysis Concepts and Research Direction". In: *G.Arango, R.P.-D.A. (eds), Domain Analysis and Software Systems Modeling, IEEE Computer Society Press*, pp. 9-25, 1991.

ARAUJO, R. M., SANTORO, F. M., ROSA, M.G.P., BRÉZILLON, P., 2004a, "Context Models for Managing Collaborative Software Development Knowledge". In: *International Workshop on Modeling and Retrieval of Context (MRC)*, pp. 61-72, University of Ulm, Germany .

ARAUJO, R., CAPPELLI, C., JR, A. G., PEREIRA, M., DOS SANTOS IENDRIKE, H., IELPO, D., TOVAR, J. A., 2004b, "A Definição de Processos de Software sob o ponto de vista da Gestão de Processos de Negócio". In: *VI Simpósio Internacional de Melhoria de Processos de Software*, São Paulo.

ARMBRUST, O.; KATAHIRA, M.; MIYAMOTO, Y.; *et al*., 2008, "Scoping Software Process Models - Initial Concepts and Experience from Defining Space Standards", In: *Proceedings of the International Conference on Making Globally Distributed Software Development a Success Story*, pp. 160-172, Berlin / Heidelberg: Springer.

ARMBRUST, O.; KATAHIRA, M.; MIYAMOTO, Y.; *et al*., 2009, "Scoping Software Process Lines". In: *Software Process: Improvement and Practice*, v. 14, Issue 3, pp. 181-197, New York, NY, USA.

ARMBRUST, O., 2010, "Determining Organization-Specific Process Suitability". In: *New Modeling Concepts for Today's Software Processes*. Lecture Notes in Computer Science. Springer Berlin Heidelberg, p. 26–38.

AVRILIONIS, D., BELKHATIT, N., CUNIN, P.Y., 1996, "A Unified Framework for Software Process Enactment and Improvement". In: *Proceedings of the 4th International Conference on Software Process (ICSP4)*, pp. 102-111, Brighton, UK.

BARCELOS, R.F., 2006, *Uma Abordagem para Inspeção de Documentos Arquiteturais Baseada em Checklist*. Dissertação de M.Sc., Programa de Engenharia de Sistemas e Computação - COPPE, UFRJ, Rio de Janeiro, Brasil.

BARRETO, A., 2007, *Uma Abordagem para Definição de Processos de Software Baseada em Reutilização*. Exame de Qualificação, COPPE/UFRJ, Rio de Janeiro, RJ, Brasil.

BARRETO, A. S., MURTA, L. G. P., ROCHA, A. R., 2009, "Componentizando Processos Legados de Software Visando a Reutilização de Processos", In: *VIII Simpósio Brasileiro de Qualidade de Software*, pp. 189-203, Ouro Preto, MG, Brasil, Junho.

BARRETO, A. S.; NUNES, E.; ROCHA, A. R. C.; MURTA, L. G. P., 2010, "Supporting the Definition of Software Processes at Consulting Organizations via Software Process Lines". In: *Proceedings of the International Conference on the Quality of Information and Communications Technology (QUATIC)*, pp. 15-24, Porto, Portugal.

BARRETO, A., MURTA, L. G. P., DA ROCHA, A. R. C., 2011, "Software Process Definition: a Reuse-based Approach". *Journal of Universal Computer Science*, v. 17, n. 13, pp. 1765-1799.

BARRETO, A. S., 2011, *Uma Abordagem para Definição de Processos baseada em Reutilização Visando à Alta Maturidade em Processos*. Tese de D.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil.

BARROS, M., 2001, *Gerenciamento de Projetos baseado em Cenários: Uma abordagem de modelagem dinâmica e simulação*. Tese de D.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil.

BASILI, V., CALDIERA, G., ROMBACH, H., 1994, "Goal Question Metric Paradigm", *Encyclopedia of Software Engineering*, v. 1, John Wiley & Sons, pp. 528-532.

BECK, K., 2004, *Extreme Programming Explained: Embrace Change*. 2 ed. Boston, MA, USA, Addison-Wesley.

BERGER, P. M., 2003, *Instanciação de Processos de Software em Ambientes Configurados na Estação TABA*. Dissertação de M. Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil.

BERTOLLO, G., SEGRINI, B., FALBO, R. de A., 2006, "Definição de Processos de Software em um Ambiente de Desenvolvimento de Software Baseado em Ontologias". In: *V Simpósio Brasileiro de Qualidade de Software*, SBQS, 6, pp.72–86, Vila Velha, Espírito Santo.

BHUTA, J.; BOEHM, B.; MEYERS, S., 2005, "Process Elements: Components of Software Process Architectures." In: *Proceedings of the Software Process Workshop*, pp. 332-346, Beijing, China.

BIOLCHINI, J., MIAN, P., NATALI, A.C., *et al*., 2005, *Systematic Review in Software Engineering*, RT-ES 679/05, COPPE/UFRJ, Rio de Janeiro, RJ, Brasil.

BLOIS, A., WERNER, C., BECKER, K., 2005, "Towards a Components Grouping Technique within a Domain Engineering Process". In: *31th Euromicro Conference, Component-based Software Engineering Track*, pp.18-25, Porto, Portugal.

BLOIS, A. P., 2006, *Uma Abordagem de Projeto Arquitetural baseado em Componentes no contexto de Engenharia de Domínio*. Tese de D.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil.

BOLAÑOS, F., MARTÍNEZ J., 2013, *Análisis Visual de la Modularidad de Modelos de Procesos de Software AVIMO-PS*, Cauca University, Colombia (in Spanish).

BOSCH, J., 2000, *Design and use of software architectures: adopting and evolving a product-line approach*, 1st ed. New York, United States, Addison Wesley Professional.

BOSCH, J., 2004, "Software Variability Management". In: *Proceedings of the 26th International Conference on Software Engineering (ICSE'04)*, pp. 720-721, Scotland, UK.

BOSE, R.P.J.C., VAN DER AALST, W.M.P., 2009, "Abstractions in Process Mining: A Taxonomy of Patterns". In: *Business Process Management (BPM*). V. 5701 of LNCS, pp. 159–175, Germany.

BRAGA, R., WERNER AND C., MATTOSO, M., 1999, "Odyssey: A Reuse Environment Based on Domain Models", In: *2nd IEEE Symposium on Application-Specific Systems and Software Engineering Technology (ASSET'99*), pp.50-57, Richardson, USA, march.

BRERETON, P., KITCHENHAM, B. A., BUDGEN, D., TURNER, M., & KHALIL, M., 2007, "Lessons from applying the systematic literature review process within the software engineering domain". *Journal of systems and software*, v. 80, n.4, pp. 571-583.

BROWN, A., 2000, *Large-Scale Component-Based Development*, Prentice Hall.

CARDOSO, F.S., 2012, *Definição de Processos Reutilizáveis para Projetos com Aquisição*. Dissertação de M.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil.

CHRISSIS, M. B., KONRAD, M., SHRUM, S., 2006, *CMMI: Guidelines for Process Integration and Product Improvement*. 2 ed. Boston, USA, Addison-Wesley.

COLLOFELLO, J. S., 1988, *Introduction to software verification and validation*, No. CMU/SEI-CM-13-1-1, Carnegie-Mellon University, Pittsburgh, Soft. Eng. Institute.

COULETTE B., CRÉGUT X., DONG T. B. T. AND TRAN D. T., 2000, "RHODES, a Process Component Centered Software Engineering Environment", In: *Proceedings of the 2nd International Conference on Enterprise Information System (ICEIS2000*), pp. 173-182, Springer Netherlands, July.

COULETTE B., CRÉGUT X., DONG T. B. T. AND TRAN D. T., 2001, "Managing process through a base of reusable components", In: ICEIS2001, Setubal, July.

CURTIS *et al.*, 1992, "Process Modelling". In: *Communications of the ACM*, New York, v. 35, n. 9, pp. 75-90.

CZARNECKI, K. & ANTKIEWICZ, M., 2005, “Mapping features to models: a Template Approach Based on Superimposed Variants”. In: *Proceedings of the 4th international conference on Generative Programming and Component Engineering*, GPCE 2005, 422-437, Springer-Verlag, Berlin, Heidelberg., pp. 52, 110

DAI, F.; LI, T., 2007a, “Composing Software Evolution Process Component”. In: *Proceedings of the 7th International Symposium on Advanced Parallel Processing Technologies*, pp. 684-692.

DAI, F.; LI, T., 2007b, “Tailoring Software Evolution Process”. In: *International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing* (SNPD 2007), pp. 782 – 787.

DAI, F.; LI, T.; ZHAO, N.; YU, Y.; HUANG, B., 2008, “Evolution Process Component Composition Based on Process Architecture”. In: *International Symposium on* Intelligent Information Technology Application Workshops (IITAW’ 08), pp. 1097 – 110.

DANTAS, A.R., CORREA, A.L., WERNER, C.M.L., 2001, "Oráculo: Um Sistema de Críticas para a UML". In: *XV Simpósio Brasileiro de Engenharia de Software - SBES, Caderno de Ferramentas*, pp. 398-403, Rio de Janeiro, RJ , Brasil.

DANTAS, A.R., VERONESE, G.O., CORREA, A.L., *et al.*, 2002, "Suporte a Padrões no Projeto de Software". In: *XVI Simpósio Brasileiro de Engenharia de Software*, pp. 450-455, Gramado, RS, Brasil, Outubro.

DE CARVALHO, D. D., CHAGAS, L. F., LIMA, A. M., REIS, C. A. L., 2014, “Software Process Lines: A Systematic Literature Review”. In: *Software Process Improvement and Capability Determination,* pp. 118-130, Springer International Publishing.

DE MELLO, R. M., PEREIRA, W. M., TRAVASSOS, G. H., 2010, “Activity Diagram Inspection on Requirements Specification”. In: *Brazilian Symposium on Soft. Eng*, pp. 168-177, IEEE.

DE MELLO, R. M., & TRAVASSOS, G. H., 2013, “An ecological perspective towards the evolution of quantitative studies in software engineering”. In: *Proceedings of the 17th International Conference on Evaluation and Assessment in Software Engineering,* pp. 216-219, ACM.

DE MELLO, R. M., NOGUEIRA, E., SCHOTS, M., WERNER, C. M. L., TRAVASSOS, G. H., 2014, “Verification of Software Product Line Artefacts: A Checklist to Support Feature Model Inspections”. *Journal of Universal Computer Science*, v. 20, n. 5, pp. 720-745.

DENGER, C., KOLB, R, 2006, “Testing and inspecting reusable product line components: First empirical results”. In: *Proceedings of the 5th ACM-IEEE International Symposium on Empirical Software Engineering*, pp. 184-193, Rio de Janeiro, Brazil.

DERNIAME, J. C.; KABA, B. A.; WASTELL, D., 1999, (Ed.), “Software Process: Principles, Methodology, and Technology”. Berlin: Springer-Verlag, 1999. (Lecture Notes in Computer Science, 1500).

DEY, A. K.; ABOWD, G. D.; SALBER, D., 2001, "A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications", In: *Human-Computer Interaction*, v. 16, n. 2-4, p. 97–166.

DURÁN, A., BENAVIDES, D., BERMEJO, J., 2004, “Applying system families concepts to requirements engineering process definition”. In: *Software Product-Family Engineering*, pp. 140-151, Springer Berlin Heidelberg.

FALBO, R. A.; BERTOLLO, G., 2005, "Establishing a Common Vocabulary for Software Organizations Understand Software Processes". In: *EDOC International Workshop on Vocabularies, Ontologies and Rules for The Enterprise (VORTE)*, pp. 1-8, Enschede, The Netherlands.

FEILER, P.H., HUMPHREY, W.S., 1993, “Software process development and enactment: Concepts and definitions”. In: *Proceedings of the Second International Conference on Software Process*, pp. 28-40.

FERNANDES, P., TEIXEIRA, E. N., WERNER, C., 2011, “An Approach for Feature Modeling of Context-Aware Software Product Line.” *Journal of Universal Computer Science* (Print), v. 17, p. 807-815, 2011.

FIORI, D. R., GIMENES, I. M. S., MALDONADO, J., OLIVEIRA JUNIOR, E. A., 2012, “Variability Management in Software Product Line Activity Diagrams”. In: *International Conference on Distributed Multimedia Systems* (DMS), pp. 89-94.

FUGGETTA, A., 2000, “Software Process: A Roadmap”. In*: Proceedings of the Conference on The Future of Software Engineering. ICSE ’00*, p. 25–34, New York, NY, USA: ACM.

FUSARO, P., VISAGGIO, G., TORTORELLA, M., 1998, "REP - ChaRacterizing and Exploiting Process Components: Results of Experimentation". In: *Working Conference on Reverse Engineering*, pp. 20-29, Honolulu, United States, October.

GARY, K., DERNIAME, J.C., LINDQUIST, T., AND KOEHNEMANN, H., 1998, “Component-Based Software Process Support”. In: *Proceedings of the 13th Conference on Automated Software Engineering (ASE'98)*, pp. 196-199, Honolulu, Hawaii, October.

GARY, K.A., LINDQUIST, T.E., 1999a, "Cooperating Process Components". In: *International Computer Software and Applications Conference (COMPSAC)*, pp. 218-223, Phoenix, United States, October.

GARY, K., LINDQUIST, T., 1999b, “Distributed Architectures for Process Component Support”. In: *Proceedings of the 5th International Conference on Information Systems Analysis and Synthesis (ISAS'99)*, Orlando, August.

GRECO, G., GUZZO, A., PONTIERI, L., 2008, “Mining Taxonomies of Process Models”. In: *Data Knowl. Eng.*, v.67, n.1, pp. 74–102.

HANSSON, C.; DITTRICH, Y.; GUSTAFSSON, B.; *et al*., 2006, "How agile are industrial software development practices?", *Journal of Systems and Software (JSS*), v. 79, n. 9, p. 1295–1311.

HENNINGER, S., 1998, “An environment for reusing software processes”. In: *Proceedings of Fifth International Conference on Software Reuse,* pp. 103-112.

HERNANDES, E., ZAMBONI, A., FABBRI, S., & DI THOMMAZO, A., 2012, “Using GQM and TAM to evaluate StArt–a tool that supports Systematic Review”. In: *CLEI Electronic Journal*, Vol. 15 no. 1, pp. 13-25.

HOLLENBACH, C. R., 1996, “Experiences in process domain engineering at PRC Inc”. In: *Software Process Workshop, International*, pp. 78, IEEE Computer Society

HOLLENBACH, C., FRAKES, W., 1996, "Software Process Reuse in an Industrial Setting". In: *4th International Conference on Software Reuse*, pp. 22-30, Orlando, United States, April.

HUMPHREY, W. S., 1989, *Managing the Software Process*. Boston, MA, USA, Addison-Wesley.

HURTADO ALEGRÍA, J. A., BASTARRICA, M. C., QUISPE, A., & OCHOA, S. F. (2011, May). An MDE approach to software process tailoring. In Proceedings of the 2011 International Conference on Software and Systems Process (pp. 43-52). ACM.

HURTADO ALEGRÍA, J. A., 2012, *A Meta-process for Defining Adaptable Software Processes*. Doctoral dissertation, University of Chile.

HURTADO ALEGRÍA, J. A., BASTARRICA, M. C., OCHOA, S. F., SIMMONDS, J., 2013, "MDE software process lines in small companies". *Journal of Systems and Software*, v.86, n.5, pp. 1153-1171.

IEEE, 2008, "STD 1028-2008: IEEE Standard for Software Reviews and Audit".

IIDA H., TANAKA Y., 2002, "A Compositional Process Pattern Framework for Component-based Process Modeling Assistance", In [SDPP02].

ISO/IEC 15504-1, 2004, *INTERNATIONAL ORGANIZATION FOR STANDARDIZATION/INTERNATIONAL ELECTROTECHNICAL COMISSION*. ISO/IEC 15504-1: Information Technology – Process Assessment - Part 1: Concepts and Vocabulary. Genève: ISO, 2004.

ISO/IEC-12207, 2008, "Systems and Software Engineering - Software Life Cycle Process", The International Organization for Standardization and the International Electrotechnical Commission, v. ISO/IEC 12207, Genebra, Suíça.

JAFARINEZHAD, O., RAMSIN, R., 2012, "Development of Situational Requirements Engineering Processes: A Process Factory Approach". In: *Computer Software and Applications Conference (COMPSAC), 2012* IEEE 36th Annual, pp. 279-288, IEEE.

JAUFMAN, O., 2005, "Emergent process design". In: *Proceedings of the 27th international conference on Software engineering,* pp. 653-653, ACM.

JAUFMAN, O.; MÜNCH, J., 2005, "Acquisition of a Project-Specific Process", In: *Proceedings of the 6th International Conference on Product-Focused Software Process Improvement*, pp. 328-342, Oulu, Finland, June.

JØRGENSEN, H.; CARLSEN, S., 2001, "*Writings in Process Knowledge Management: Management of Knowledge Captured by Process Models*". In: Oslo: SINTEF Telecom and Informatics. Technical Report.

KALUS, G., & KUHRMANN, M., 2013, "Criteria for software process tailoring: a systematic review". In: *Proceedings of the 2013 International Conference on Software and System Process*, pp. 171-180, ACM Press.

KANG, K.C., COHEN, S.G., HESS, J.A., *et al*., 1990, *Feature-Oriented Domain Analysis (FODA) – Feasibility Study*, Technical Report CMU/SEI-90-TR-21, Software Engineering Institute (SEI).

KELLNER, M.I., 1996, "Connecting reusable software process elements and components". In: *Proceedings of 10th International Software Process Workshop*, pp. 8-11, Dijon, France, June.

KRUEGER, C. W., 1992, *Software reuse*. ACM Computing Surveys (CSUR), 24(2), 131-183.

KUHRMANN, M., 2014, "You can't tailor what you haven't modeled". In: *Proceedings of the 2014 International Conference on Software and System Process*, pp. 189-190, ACM.

KÜMMEL, G., 2007, *Uma Abordagem para a Criação de Arquiteturas de Referência de Domínio a partir da Comparação de Modelos Arquiteturais de Aplicações.* Dissertação de M.Sc., COPPE,UFRJ, Rio de Janeiro, Brasil.

LANNA, A. L. P. M, 2009, *Reuso de Processos de Software Baseado na Componentização de Processos e Conhecimento*. Dissertação de mestrado em Engenharia Elétrica, PUC-MG, Belo Horizonte, Brasil.

LANNA, A. L. P. M., PIETROBON, C. A. M., 2010, "Desenvolvimento de um Método e uma Ferramenta para a Reutilização de Processos de Desenvolvimento de Software". Qualidade e Produtividade em Software Projetos Ciclo.

LEITE, A. M. S., 2011, *Modelo de Contexto para Adaptação de Processos de Software*. Dissertação de Mestrado, PPGI–UNIRIO, Rio de Janeiro, Brasil.

LONCHAMP, J., 1993, "A Structured Conceptual and Terminological Framework for Software Process Engineering". In: *Proceedings of the Second International Conference on the Software Process*, Vol. 2 (1993), pp. 41-53, Berlin, Germany.

LOPES, M.A.M., MANGAN, M.A.S., WERNER, C.M.L., 2004, "MAIS: Uma Ferramenta de Percepção para apoiar a Edição Concorrente de Modelos de Análise e Projeto". In: *XVIII Simpósio Brasileiro de Engenharia de Software, Sessão de Ferramentas*, pp. 61-66, Brasília, DF, Brasil, Outubro.

LINDVALL, M.; RUS, I., 2000, "Process diversity in software development", Software, IEEE, v. 17, n. 4, (Aug.), pp. 14–18.

MACHADO, L. F. D. C., 2000, *Modelo para Definição de Processos de Software na Estação TABA*. Dissertação de M. Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil.

MAGDALENO, A. M., 2010, *Apoio à Decisão para o Balanceamento de Colaboração e Disciplina nos Processos de Desenvolvimento de Software*. Exame de Qualificação, COPPE/UFRJ, Rio de Janeiro, RJ, Brasil.

MAGDALENO, A. M.; WERNER, C. M. L.; ARAUJO, R. M., 2011, *A Criação de um Exemplo para Alimentar a Abordagem COMPOOTIM de Composição e Otimização de Processos de Software.* Relatório Técnico ES-744/11, PESC-COPPE. Disponível em: http://www.cos.ufrj.br.

MAGDALENO, A. M., DE ARAUJO, R. M., & WERNER, C., 2012, "COMPOOTIM: An Approach to Software Processes Composition and Optimization". In *CIbSE*, pp. 42-55.

MAGDALENO, A. M., 2013, *COMPOOTIM: Em Direção ao Planejamento, Acompanhamento e Otimização da Colaboração na Definição de Processos de Software*. Tese de D.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil.

MAGDALENO, A. M., DE OLIVEIRA BARROS, M., WERNER, C. M. L., DE ARAUJO, R. M., & BATISTA, C. F. A. (2015). Collaboration optimization in software process composition. *Journal of Systems and Software*, 103, pp. 452-466.

MAIA, N.E.N., BLOIS, A.P.B., WERNER, C.M., 2005, "Odyssey-MDA: Uma Ferramenta para Transformação de Modelos UML". In: *XIX Simpósio Brasileiro de Engenharia de Software*, Sessão de Ferramentas, pp. 51-56, Uberlândia, MG, Brasil, Outubro.

MARTÍNEZ-RUIZ, T., GARCÍA, F., PIATTINI, M., 2008, "Towards a SPEM v2. 0 extension to define process lines variability mechanisms". In: *Software engineering research, management and applications*, pp. 115-130. Springer Berlin Heidelberg.

MARTÍNEZ-RUIZ, T., GARCÍA, F., & PIATTINI, M., 2009, "Process Institutionalization using Software Process Lines". In: *ICEIS (3),* pp. 359-362.

MARTÍNEZ-RUÍZ, T., GARCÍA, F., PIATTINI, M., & MUNCH, J., 2011a, "Applying AOSE concepts to model crosscutting variability in variant-rich processes". *In: 37th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA),* 2011, pp. 334-338, IEEE.

MARTÍNEZ-RUIZ, T., GARCÍA, F., PIATTINI, M., & MUNCH, J., 2011b, "Modelling software process variability: an empirical study. Software", *IET*, 5(2), 172-187.

MARTÍNEZ-RUIZ, T., MÜNCH, J., GARCÍA, F., & PIATTINI, M., 2012, "Requirements and constructors for tailoring software processes: a systematic literature review". In: *Software Quality Journal*, 20(1), 229-260.

MARTÍNEZ-RUIZ, T., GARCÍA, F., PIATTINI, M., & DE LUCAS-CONSUEGRA, F., 2013, "Process variability management in global software development: a case study". In: *Proceedings of the 2013 International Conference on Software and System Process* (pp. 46-55). ACM.

MONTERO, PENA, RUIZ-CORTÉS, 2007, "Business Family Engineering: Does it make sense?", In: II Congreso Español de Informática (Cedi 2007), In: *Actas de los Talleres de las Jornadas de Ingeniería del Software y Bases de Datos (Tjisbd)*, pp. 34-40.

MURTA, L.G.P., BARROS, M.O., WERNER, C.M.L., 2001, "FrameDoc: Um Framework para a Documentação de Componentes Reutilizáveis". In: *IV International Symposium on Knowledge Management/Document Management (ISKM/DM'2001)*, pp. 241-259, Curitiba, PR, Brasil, Agosto.

MURTA, L., 2002, *CHARON: Uma Máquina de Processos Extensível baseada em Agentes Inteligentes*. Dissertação de M.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil.

MURTA, L.G.P., VASCONCELOS, A.P.V., BLOIS, A.P.T.B., *et al*., 2004, "Run-time Variability through Component Dynamic Loading". In: *XVIII Simpósio Brasileiro de Engenharia de Software. Caderno de Ferramentas*, pp. 67-72, Brasília, DF, Brasil, Outubro.

NORTHROP, L., 2002, "SEI's Software Product Line Tenets", IEEE Software, v.19, n. 4, pp. 32-40, July/August.

NUNES, V. T.; SANTORO, F.M.; BORGES, M.R.S. (2007) "Um Modelo para Gestão de Conhecimento Baseado em Contexto". In: *SBSC - Simpósio Brasileiro de Sistemas Colaborativos*, 2007, Rio de Janeiro. XXVII Congresso da SBC.

NUNES, V. T.; WERNER, C.; SANTORO, F. M., 2010, "Context-Based Process Line". In: *International Conference on Enterprise Information Systems* (ICEIS), pp. 277-282, Funchal, Madeira, Portugal.

OCAMPO, A., BELLA, F., & MÜNCH, J., 2005, "Software Process Commonality Analysis". In: *Software Process: Improvement and Practice*, *10*(3), 273-285.

ODYSSEY, 2016, "Projeto Odyssey", In: "http://reuse.cos.ufrj.br/odyssey".

OLIVEIRA, H., MURTA, L.G.P., WERNER, C.M.L., 2004, "Odyssey-VCS: Um Sistema de Controle de Versões Para Modelos Baseados no MOF". In: *XVIII Simpósio Brasileiro de Engenharia de Software, Sessão de Ferramentas*, pp. 85-90, Brasília, DF, Brasil, Outubro.

OLIVEIRA, M. S, 2011, *PREViA: Uma Abordagem para a Visualização da Evolução de Modelos de Software*. Dissertação (Mestrado em Engenharia de Sistemas e Computação) - Universidade Federal do Rio de Janeiro.

OLIVEIRA JUNIOR, E. A.; GIMENES, I. M. S.; MALDONADO, J. C., 2010, "Systematic Management of Variability in UML-based Software Product Lines". In: *Journal of Universal Computer Science* (Print), v. 16, 2010.p. 2374-2393.

OLIVEIRA JUNIOR, E. A., PAZIN, M. G., GIMENES, I. M., KULESZA, U., ALEIXO, F. A., 2013, "SMartySPEM: A SPEM-based approach for variability management in software

process lines". In: *Product-Focused Software Process Improvement* (pp. 169-183). Springer Berlin Heidelberg.

OMG, 2008, "Software Process Engineering Metamodel". Disponível em: "http://www.omg.org/technology/documents/formal/spem.htm".

OSTERWEIL, L., 1987, "Software processes are software too". In: *Proceedings of the 9th international conference on Software Engineering. ICSE '87*. Los Alamitos, CA, USA: IEEE Computer Society Press, p. 2–13.

PEDREIRA, O., PIATTINI, M., LUACES, M. R., *et al*., 2007, "A systematic review of software process tailoring", *SIGSOFT Software Engineering Notes*, v. 32, n. 3, pp. 1-6.

PAZIN, M.G., 2012, *SMartySPEM: Uma Extensão da Abordagem SPEM para Gerenciamento de Variabilidades em Linhas de Processo de Software*. Trabalho de Conclusão de Curso, Graduação em Bacharel em Informática, Universidade Estadual de Maringá.

PILLAT, R. M., OLIVEIRA, T. C., FONSECA, F. L., 2012, "Introducing software process tailoring to bpmn: Bpmnt". In: *Software and System Process (ICSSP), 2012 International Conference on* (pp. 58-62). IEEE.

PILLAT, R. M., OLIVEIRA, T. C., ALENCAR, P. S., COWAN, D. D., 2015, "BPMNt: A BPMN extension for specifying software process tailoring", In: *Information and Software Technology*, 57, pp. 95-115.

RAZAVIAN, M., KHOSRAVI, R., 2008, "Modeling Variability in Business Process Models Using UML". In: *Fifth International Conference on Information Technology: New Generations (ITNG'08)*, pp. 82 –87.

REIS, R. Q., 2002, *APSEE-Reuse: um metamodelo para apoiar a reutilização de processos de software*. Tese de D. Sc., Universidade Federal do Rio Grande do Sul (UFRGS), Porto Alegre, RS, Brasil.

REIS, C.A., 2003, *Uma Abordagem Flexível para Execução de Processos de Software Evolutivos.* Tese de Doutorado, Porto Alegre: PPGC- UFRGS, 2003.

ROCHA, R., FANTINATO, M. (2013), “The use of software product lines for business process management: A systematic literature review”. In: *Information and Software Technology* 55 (8), pp. 1355-1373.

ROLLAND, C., NURCAN, S., 2010, “Business Process Lines to Deal with the Variability”. In: *2010 43rd Hawaii International Conference on System Sciences (HICSS)*, pp. 1 – 10.

ROMBACH, D., 2006, "Integrated Software Process and Product Lines". In: *Unifying the Software Process Spectrum*, Berlin/Heidelberg: Springer-Verlag, pp. 83-90.

ROUILLÉ, E., COMBEMALE, B., BARAIS, O., TOUZET, D., JÉZÉQUEL, J. M., 2012, “Leveraging CVL to manage variability in software process lines”. In: *Software Engineering Conference (APSEC)*, 2012 19th Asia-Pacific (Vol. 1, pp. 148-157). IEEE.

ROUILLÉ, E., COMBEMALE, B., BARAIS, O., TOUZET, D., JÉZÉQUEL, J. M., 2013a, “Improving Reusability in Software Process Lines”. In: *Software Engineering and Advanced Applications (SEAA), 2013 39th EUROMICRO Conference on* (pp. 90-93).

ROUILLÉ, E., COMBEMALE, B., BARAIS, O., TOUZET, D., & JÉZÉQUEL, J. M., 2013b, “Integrating Software Process Reuse and Automation”. In: *Software Engineering Conference (APSEC), 2013 20th Asia-Pacific* (Vol. 1, pp. 380-387). IEEE.

RUIZ-RUBE, I., DODERO, J. M., COLOMO-PALACIOS, R., 2015, “A framework for software process deployment and evaluation”. In: *Information and Software Technology*, 59, pp. 205-221.

RU-ZHI *et al*., 2003*,* “Research on matching algorithm for XML-based software component query”, *Journal of Software*, July, pp.1195-1202.

RU-ZHI, X., TAO, H., DONG-SHENG, C*., et al.*, 2005, "Reuse-Oriented Process Component Representation and Retrieval". In: *International Conference on Computer and Information Technology*, pp. 911-315, Shangai, China, September.

RU-ZHI, X., PEIGUANG L., ZHIKUN Z., LEQIU Q., 2010, “An Approach of Reuse-based Software Process Improvement”. In: *Journal of Computational Information Systems Volume 6 Number 6*, pp. 1897-1906. Available at http://www.Jofcis.com

SÁNCHEZ-GONZÁLEZ, L., RUIZ, F., GARCÍA, F., PIATTINI, M., 2013, “Improving Quality of Business Process Model”. In: *Evaluation of Novel Approaches to Software Engineering* (pp. 130-144).

SAMETINGER, J., 1997, *Software Engineering with Reusable Components*, Springer-Verlag New York, Inc.

SCHWABER, K., 2004, *Agile Project Management with Scrum*. Washington, DC, USA, Microsoft Press.

SEGRINI, B.M. , 2009, *Definição de Processos Baseada em Componentes*. Dissertação de M.Sc., Universidade Federal do Espírito Santo, Vitória, ES, Brasil.

SILVESTRE, L., BASTARRICA, M. C., OCHOA, S. F., 2014, “A model-based tool for generating software process model tailoring transformations”. In: *Model-Driven Engineering and Software Development (MODELSWARD), 2014 2nd International Conference* on (pp. 533-540). IEEE.

SIMIDCHIEVA, B.I., CLARKE, L.A., OSTERWEIL, L., 2007, "Representing Process Variation with a Process Family". In: *International Conference on Software Process, v. Lecture Notes in Computer Science 4470*, pp. 109-120, Minneapolis, Estados Unidos, Maio.

SIMMONDS, J. e BASTARRICA, M.C., 2011, “*Modeling Variability in Software Process Lines”*. Santiago, Chile, Universidad de Chile.

SIMMONDS, J., BASTARRICA, M.C., SILVESTRE, L., QUISPE, A., 2011, “*Analyzing Methodologies and Tools for Specifying Variability in Software Processes”*. Santiago, Chile, Universidad de Chile.

SIMMONDS, J., BASTARRICA, M.C., SILVESTRE, L., QUISPE, A., 2012, “*Modeling Variability in Software Process Models”*. Technical Report. Santiago, Chile, Universidad de Chile.

SIMMONDS, J., BASTARRICA, M. C., SILVESTRE, L., QUISPE, A., 2013, “Variability in software process models: Requirements for adoption in industrial settings”. In: *Product Line Approaches in Software Engineering (PLEASE), 2013 4th International Workshop on* (pp. 33-36). IEEE.

SIMMONDS, J., PEROVICH, D., BASTARRICA, M. C., SILVESTRE, L., 2015, “A megamodel for Software Process Line modeling and evolution”. In: *Model Driven Engineering Languages and Systems (MODELS), 2015 ACM/IEEE 18th International Conference on* (pp. 406-415). IEEE.

SIMIDCHIEVA, B.I., CLARKE, L.A., OSTERWEIL, L., 2007, "Representing Process Variation with a Process Family". In: *International Conference on Software Process, v. Lecture Notes in Computer Science 4470*, pp. 109-120, Minneapolis, Estados Unidos, Maio.

SOFTEX, 2016, *Melhoria de Processo do Software Brasileiro – Guia Geral de Software, Modelo de Qualidade*. Disponível em: http://www.softex.br.

SOLINGEN, R.V., 2004, "Measuring the ROI of Software Process Improvement", *IEEE Software*, v. 21, n. 3, pp. 32-38

STOROLLI, A., GUIDINI, J., ZANOLLA, G., TEREZINHA, B., 2009, “Modelagem de Processos de Software”. In: *Octava Conferencia Iberoamericana en Sistemas, Cibernética e Informática (CISCI 2009)*, 10 - 13 de Julho, Orlando, Florida, EUA.

SUN, 2005, "Java Technology". In: http://www.java.sun.com.

SUTTON, S., OSTERWEIL, L., 1996, "Product families and process families". In: *Proceedings of the 10th International Software Process Workshop (ISPW)*, pp. 109-111, Dijon, France.

TEIXEIRA, E. N., VASCONCELOS, A., WERNER, C., 2009, “An Approach to Support a Flexible Feature Modeling”. In: *III Simpósio Brasileiros de Componentes, Arquiteturas e Reutilização de Software (SBCARS)*, p. 81-94, Natal, RN.

TEIXEIRA, E.N., 2011, *ODYSSEYPROCESS-FEX: uma abordagem para modelagem de variabilidades de linha de processos de software.* Dissertação de M.Sc., Programa de Engenharia de Sistemas e Computação, COPPE/UFRJ, Rio de Janeiro.

TEIXEIRA, E. N.; de MELLO, R. M. ; MOTTA, R. ; WERNER, C. ; VASCONCELOS, A., 2015, “Verification of Software Process Line Models: A Checklist-based Inspection Approach”. In: *XVIII Iberoamerican Conference on Software Engineering, 2015, Lima. XVIII Ibero-American Conference on Software Engineering*, v. 1. p. 81-91.

TEIXEIRA, E. N., VASCONCELOS, A., WERNER, C., 2016, "*OdysseyProcess-FEX: Metamodelo e Notação para Representação de Variabilidades de Linha de Processos de Software*", Relatório Técnico ES-751/16, COPPE/UFRJ, Rio de Janeiro, Brasil, Março. Disponível em: http://www.cos.ufrj.br.

TERNITÉ, T., 2009, "Process Lines: A Product Line Approach Designed for Process Model Development". In: *Software Engineering and Advanced Applications. SEAA '09. 35th Euromicro Conference on*, pp. 173-180, 27-29 Aug.

THRÄNERT, M., & WERNER, A., 2007, "A Process Family Approach for the reuse of development processes". In: *Innovations and Advanced Techniques in Computer and Information Sciences and Engineering* (pp. 309-313). Springer Netherlands.

TORTORELLA M., VISAGGIO G., 1997, "CREP - Characterising Reverse Engineering Process components methodology", In: *Proceedings of International Conference on Software Maintenance*, Bari, Italy, October 1 - 3, IEEE Comp. Soc. Press, pp. 222-231.

TRAVASSOS, G., SHULL, F., FREDERICKS, M., BASILI, V. R., 1999, "Detecting defects in object-oriented designs: using reading techniques to increase software quality". In *ACM Sigplan Notices*, Vol. 34, No. 10, pp. 47-56.

TRAVASSOS, G.H., DOS SANTOS, P.S.M., NETO, P.G.M., et al., 2008, "An Environment to Support Large Scale Experimentation in Software Engineering". In: *Engineering of Complex Computer Systems, 2008. ICECCS 2008. 13th IEEE International Conference on,* pp. 193-202, Belfast, March.

VASCONCELOS, A., & WERNER, C., 2011, "Evaluating reuse and program understanding in ArchMine architecture recovery approach". *Information Sciences*, v.181, n.13, pp. 2761-2786.

VERONESE, G.O., NETTO, F.J., CORREA, A., et al., 2002, "ARES – Uma Ferramenta de Engenharia Reversa Java-UML". In: *XVI Simpósio Brasileiro de Engenharia de Software - Caderno de Ferramentas*, pp. 347-352, Gramado, RS, Outubro.

WASHIZAKI, H., 2006a, "Building software process line architectures from bottom up". In: *Proceedings of the 7th International Conference on Product-Focused Software Process Improvement, PROFES 2006*, pp. 415-421, Amsterdam, Netherlands, June.

WASHIZAKI, H., 2006b, "Deriving project-specific processes from process line architecture with commonality and variability". In: *2006 IEEE International Conference on Industrial Informatics, INDIN'06, August 16, 2006 - August 18, 2006*, pp. 1301-1306, Singapore, Singapore.

WERNER, C., MATTOSO, M., BRAGA, R., et al., 1999, "Odyssey: Infra-estrutura de reutilização Baseado em Modelos de Domínios". In: *Caderno de Ferramentas do XIII Simpósio Brasileiro de Engenharia de Software*, pp. 17-20, Florianópolis, outubro.

WERNER, C. M. L.; ARAUJO, R. M. DE; SANTORO, F. M.; *et al*., 2011, *CDSoft: Balanceando Colaboração e Disciplina em Processos de Desenvolvimento de Software*. Projeto de Pesquisa, Conselho Nacional de Desenvolvimento Científico e Tecnológico, Rio de Janeiro, RJ, Brasil.

WOHLIN, C., RUNESON, P., HÖST, M., *et al*., 2000, *Experimentation in Software Engineering: an introduction*, USA, Kluwer Academic Publishers.

XAVIER, J.R., 2001, *Criação e Instanciação de Arquiteturas de Software Específicas de Domínio no Contexto de uma Infra-Estrutura de Reutilização*. Dissertação de M.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil.

ZAMLI, K. Z., LEE, P.A., 2001, "Taxonomy of Process Modelling Languages". In: *Proc. of the ACS/IEEE Inter. Conf. on Computer Systems and Applications (AICCSA'01)* Beirut, Lebanon, June.

Apêndice I - Protocolo Revisão Sistemática

- **Objetivo Geral**: O objetivo principal é caracterizar o estado da arte de abordagens de Linhas de Processos de Software. Este objetivo envolve coletar e caracterizar publicações científicas sobre processos de software com o propósito de caracterizar abordagens relacionadas ao uso de LPrS do ponto de vista de pesquisadores no contexto acadêmico e industrial.
- **Questão Principal:** Quais abordagens de LPrS podem ser identificadas e qual suporte é fornecido por essas abordagens para a reutilização de software?
    - **Questões Secundárias:**
        - **QPS1:** Quais são as abordagens de LPrS existentes?
        - **QPS2:** Qual(is) fase(s) da Engenharia de LPrS é(são) suportado(s) pela abordagem?
        - **QPS3:** Qual(is) atividade(s) do ciclo de vida de processos de software é(são) suportada(s) pela abordagem?
        - **QPS4:** Algum suporte ferramental é provido pela abordagem? Qual?
        - **QPS5:** A abordagem oferece alguma representação ou gerenciamento de variabilidade de processo?
        - **QPS6:** Que tipos de classificação de variabilidade de processos de software são propostas?
        - **QPS7:** A abordagem propõe alguma estratégia de componentização para elementos de processos de software reutilizáveis?
        - **QPS8:** Que tipos de exemplos práticos são usados na abordagem?
        - **QPS9:** Como a abordagem descrita foi avaliada?
        - **QPS10:** Que critérios ou medidas específicas são apresentados para avaliar a abordagem?
- **Escopo da Revisão Sistemática:** o escopo foi definido de acordo com a estrutura PICO (*Population, Intervention, Comparison and Outcomes* - População, Intervenção, Comparação e Resultado) (PAI *et al*., 2004).

- (P) População: Pesquisas na área de processos de software
- (I) Intervenção: Abordagens de LPrS
- (C) Comparação: Não é definido. O objetivo da revisão é de caracterização das abordagens.
- (O) Resultado: Técnicas, abordagens, métodos, metodologias, ferramentas, processos que suportam atividades do ciclo de vida de processos de software usando LPrS.

- **Artigos de Controle**: a seleção do conjunto de artigos de controle foi baseada em revisões prévias da literatura na área de reutilização de processos de software. Foram selecionados três artigos referenciados como controle em outras revisões e que apresentam semelhança com o tema de interesse e definições de base na área de LPrS.
    - ROMBACH, H.D., 2006, "Integrated Software Process and Product Lines". In: International Software Process Workshop, pp. 83-90, Beijing, China, May.
    - WASHIZAKI, H., 2006, "Building Software Process Line Architectures from Bottom Up". In: Product-Focused Software Process Improvement, pp. 415-421, Amsterdam, Netherlands, June.
    - ARMBRUST, O., KATAHIRA, M., MIYAMOTO, Y., et al., 2009, "Scoping Software Process Lines", Software Process: Improvement and Practice, v. 14, n. 3, pp. 181-197.
- **Definição da *String* de Busca**: as palavras chave que compõem o escopo das questões de pesquisa foram definidas considerando a estrutura PICO definida, os objetivos da revisão e o conhecimento dos pesquisados envolvidos. Termos que contemplam a população, intervenção e resultados esperados foram aplicados associados a alternativas de escrita e sinônimos conectados por operados lógicos. O conjunto de artigos de controle foi usado para refinar as *strings* inicialmente propostas. A *string* de busca final usada foi:

("process adaptation" OR "processes adaptation" OR "customization of processes" OR "software processes customization" OR "software process customization" OR "customizing software processes" OR "process definition" OR "processes definition" OR "process composition" OR "compose processes" OR "processes composition" OR

"process tailoring" OR "processes tailoring" OR “tailoring of processes” OR "process development" OR "processes development" OR "process engineering" OR "processes engineering" OR "process design" OR "software process modeling" OR "software process modelling" OR "process implementation" OR "managing processes")

AND ("family of software process" OR "family of software processes" OR "families of software process" OR "families of software processes" OR "software process line" OR "software process lines" OR "software processes line" OR "software processes lines" OR "process-line" OR "process-lines" OR "processes-line" OR "processes-lines" OR "software process family" OR "software processes family" OR "software process families" OR "software processes families" OR "process-family" OR "processes-family" OR "process-families" OR "processes-families" OR "software process variability" OR "software process variabilities" OR "software processes variability" OR "software processes variabilities" OR "variabilities in software processes" OR "process domain engineering" OR "processes domain engineering" OR "process feature" OR "process features" OR "processes feature" OR "processes features" OR "process asset reuse”)

- **Seleção das máquinas de busca a serem usadas**: as máquinas de busca foram selecionadas com base em análises como possibilidade de uso de expressões lógicas, recuperação do texto completo da publicação retornada, buscas automáticas pelo conteúdo total da publicação ou por partes específicas como resumo. As bases usadas foram: (1) Scopus: http://www.scopus.com; (2) IeeeXplore (IEEE): http://ieeexplore.ieee.org; e (3) Compendex: http://www.engineeringvillage2.org. Essas são algumas das bases mais relevantes na área de Engenharia de Software.
- **Critérios de Inclusão e Exclusão**: foram definidos para apoiar a seleção dos estudos primários considerados relevantes para esta revisão. Apenas um critério de inclusão principal foi definido e cinco critérios de exclusão foram especificados para identificar as publicações fora do escopo da revisão.
    - **Critério de Inclusão**: A leitura do conteúdo completo da publicação deve permitir identificar pelo menos um aspecto que caracterize uma abordagem de LPrS.
    - **Critério de Exclusão 1**: Publicações que não definem parcialmente ou totalmente uma abordagem de LPrS não devem ser incluídas.

- **Critério de Exclusão 2**: Publicações duplicadas devem ser removidas.
- **Critério de Exclusão 3**: Publicações do tipo apresentação, tutorial, curso, *workshop* ou de tipo similar não devem ser consideradas.
- **Critério de Exclusão 4**: Publicações sem acesso ao conteúdo completo devem ser desconsideradas.
- **Critério de Exclusão 5**: Publicações não escritas em inglês devem ser desconsideras. Para a realização desta pesquisa foi selecionado o idioma inglês, devido à sua adoção pela grande maioria das conferências, periódicos internacionais e bases eletrônicas relacionadas com o tema de pesquisa. A maioria das publicações em português não se encontram indexadas, embora se reconheça a sua importância.

- **Formulário para extração de informação:** Após a seleção dos estudos primários, cada publicação deve ser analisada e um conjunto de informações deve ser extraída segundo o formulário definido abaixo:

Extraction Form
Reference Identification **Title** **Author(s)** **Year** **Institution/Country** Type of publishing vehicle**:** Journal**/**Workshop**/**Conference**/**Symposium
Type of Contribution Method Graphical Notation **/**Metamodel**/**Process Line Modeled**/**Variability requirements Tool Guideline Comparative Study **Hypothesis or suggestion**
Phases of SPrL Engineering supported by the approach Domain Engineering (development *for* reuse) **Project Engineering (development *with* reuse)**
Phases of the software process lifecycle activity supported by the approach **Process Definition Process Simulation Process Enactment Process Execution Process Improvement**
Software process engine is used to support process execution? Yes /No Which engine? Execution data is collected? **Yes/**No **Some specific representation is used? State diagram; BPMN; Petri net; Other**
SPrL improvement/evolution is supported? Yes/No
SPrL verification and/or validation is supported? Yes/No
Type of supporting tool provided by the approach
Process Variability **SPrL Representation:** **Modeling Language: Feature Model** SPEM or Derivation of SPEM UML (Activity Diagram) BPMN or Derivation of BPMN Not mentioned **Other** **Process Variability Management:** Software Process Elements that are considered variable: Activity Task Step Work Product/Artifact Role Tool Software Process Flows are represented?

Restriction and Consistency rules: Dependency relations Exclusion relations Context Model Not mentioned
Process Variability Classification
Variation point (alternative) Optional Mandatory Invariant Crosscutting Not mentioned **Other**
Component Based Software Process aspects
SPrL Architecture Definition
Process Component Definition
Approach Evaluation
Example Case Study Industry Application Comparison among different approaches Survey Expert Assessment Not mentioned **Other**
Kinds of practical examples are used in the approaches
Criteria or metrics are presented to evaluate the approach

Apêndice II – Aplicação de Heurísticas de Mapeamento do Modelo de Características para o Modelo de Componentes de Processo de Software

LPrS: Planejamento de Projetos de Software

Esta LPrS descreve três atividades iniciais do planejamento de projetos de software: Desenvolver o Termo de Abertura do Projeto (*Develop Project Charter*); Determinar tamanho do Projeto (*Determine Project size*) e Determinar esforço do projeto (*Determine Project Effort*). O tamanho pode ser determinado por meio de duas técnicas diferentes: contagem indicativa PF ou base histórica (variantes do ponto de variação *Determine Project Size*). O esforço pode ser dimensionado usando duas técnicas diferentes: usando PF ou base histórica (variantes do ponto de variação *Determine Project Effort*). Cada uma das variantes do cálculo do esforço depende do resultado de cada uma das técnicas do cálculo do tamanho. As atividades ocorrem de modo sequencial.

Regras de Composição:

- R: Determine effort using size estimated by FP requires (Determine by Indicative Function Points (FP) AND FP Spreadsheet filled with Project Size)
- R_1: Determine effort using size estimated by historical base requires (Determine by projects historical base AND Project Size estimated by Historical Analysis)
- X: (Determine by Indicative Function Points (FP) AND FP Spreadsheet filled with Project Size) excludes (Determine by projects historical base AND Project Size estimated by Historical Analysis)

Tabela II.1 –Descrição Tabular da LPrS de Planejamento de Projetos de Software

Modelo de Característica		
Características da Categoria Atividade		
ID	**Classificação de Opcionalidade e Variabilidade**	**Nome**
A1	Invariante / Mandatório	*Develop Project Charter*
A2	Ponto de variação / Mantadório	*Determine Project Size*
A3	Variante / Opcional	*Determine by Indicative Function Points (FP)*
A4	Variante / Opcional	*Determine by projects historical base*
A5	Ponto de variação / Mantadório	*Determine Project Effort*
A6	Variante / Opcional	*Determine effort using size estimated by FP*
A7	Variante / Opcional	*Determine effort using size estimated by historical base*
Características da Categoria Tarefa		
ID	**Classificação de Opcionalidade e Variabilidade**	**Nome**
T1	Invariante / Opcional	*Calculate total indicative not adjusted FP*
T2	Ponto de variação / Opcional	*Determine and apply adjustment factor*
T3	Variante / Opcional	*Apply IFPUG adjustment factor*
T4	Variante / Opcional	*Apply organization adjustment factor*
Características da Categoria Produto de Trabalho		
ID	**Classificação de Opcionalidade e Variabilidade**	**Nome**
W1	Invariante / Mandatório / Entrada: A1	*Software Demand*
W2	Invariante / Mandatório / Saída: A1 / Entrada: A2 (A3 e A4) e A5 (A6 e A7)	*Project Charter*

W3	Invariante / Opcional / Saída: T1 -> Entrada: T2	*FP Spreadsheet filled with not adjusted FP*
W4	Invariante / Opcional / Saída: T2 (T3 ou T4) -> Entrada: A7	*FP Spreadsheet filled with Project Size*
W5	Invariante / Mandatório / Entrada: A4	*Organizational Historical Base*
W6	Invariante / Opcional / Saída: A4 -> Entrada:A6	*Project Size estimated by Historical Analysis*
W7	Invariante / Mandatório / Saída: A5 (A6 e A7)	*Project Effort (Man/Hour)*
Características da Categoria Ferramenta		
ID	**Classificação de Opcionalidade e Variabilidade**	**Nome**
Tl1	Invariante / Mandatório	*Tool for analyzing historical data*
Características da Categoria Papel		
ID	**Classificação de Opcionalidade e Variabilidade**	**Nome**
R1	Invariante / Mandatório	*Project Manager*

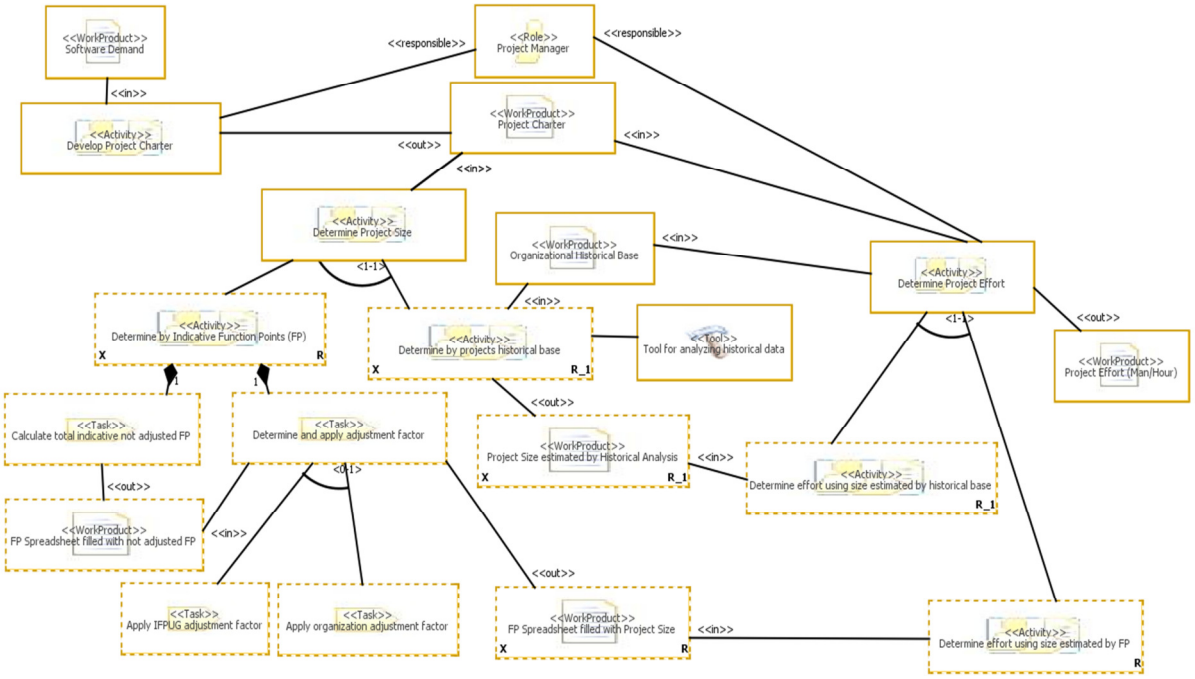


Figura II.1 – LPrS de Planejamento de Projetos de Software – Modelo de Características

A Tabela II.2 apresenta o mapeamento do modelo de característica para o modelo de componentes do domínio de Planejamento de Projetos de Software com a aplicação das heurísticas descritas na Seção 3.5. Partindo do modelo de características da Figura 19, sete componentes foram derivados. Dentre os sete, quatro componentes (C3, C4, C6 e C7) são elementos internos de outros componentes compostos (C2 e C5). Aplicando a heurística TH, todas as tarefas foram mapeadas como elementos internos dos componentes, mantendo a mesma classificação de variabilidade e opcionalidade do modelo de característica. O papel "*Project Manager*" foi mapeado como elemento interno dos componentes C1, C5, C6 e C7 e classificado como invariante e mandatório (aplicação da heurística RH). O elemento "Tool for analyzing historical data" foi mapeado como elemento interno do componente C4 e classificado como invariante e mandatório (aplicação da heurística TlH). O produto de trabalho "*Spreadsheet filled*

with not adjusted FP" como se caracteriza por um artefato trocado dentro dos limites de um componente, apenas entre unidades de trabalho internas ao componente C3, este é mapeado como elemento interno deste componente e classificado como invariante e opcional (aplicação da heurística WPH1).

Tabela II.2 – Mapeamento Características -> Componentes da LPrS de Planejamento de Projetos de Software

ID	Modelo de Característica Elemento	Heurísticas	Modelo de Componente Elemento
A1	*Develop Project Charter*	AH	**Componente** (**C1**): Invariante / Mandatório / Simples / Concreto
A2	*Determine Project Size*	AH / AH 1.1 / AH 2.1	**Componente** (**C2**): Ponto de variação / Mandatório / Composto / Abstrato
A3	*Determine by Indicative Function Points (FP)*	AH / AH2.2	**Componente** (**C3**)**:** Variante / Opcional / Simples / Abstrato
A4	*Determine by projects historical base*	AH	**Componente** (**C4**): Variante / Opcional / Simples / Concreto
A5	*Determine Project Effort*	AH / AH 1.1 / AH 2.1	**Componente** (**C5**): Ponto de variação / Mandatório / Composto / Abstrato
A6	*Determine effort using size estimated by FP*	AH	**Componente** (**C6**): Variante / Opcional / Simples / Concreto
A7	*Determine effort using size estimated by historical base*	AH	**Componente** (**C7**): Variante / Opcional / Simples / Concreto
T1	*Calculate total indicative not adjusted FP*	TH	**Elemento Interno de C3** : Invariante/Opcional
T2	Determine and apply adjustment factor	TH e TH1.1	**Elemento Interno de C3**: Ponto de Variação/Opcional
T3	*Apply IFPUG adjustment factor*	TH e TH1.1	**Elemento Interno de C3:** Variante/Opcional
T4	*Apply organization adjustment factor*	TH e TH1.1	**Elemento Interno de C3:** Variante/Opcional
R1	*Project Manager*	RH	**Elemento Interno de C1, C5, C6 e C7**: Invariante / Mandatório
Tl1	*Tool for analyzing historical data*	TlH	**Elemento Interno de C4:** Invariante /Mandatório
W3	FP Spreadsheet filled with not adjusted FP	WPH1	**Elemento Interno de C3:** Invariante /Opcional

Cada relação de produto de trabalho e unidade de trabalho que extrapola as fronteiras do componente de processo deve ser mapeada como uma interface de dados (WPH2) e as regras de composição são mapeadas como restrições dos componentes.

Tabela II.3 – Mapeamento das interfaces e das restrições nos componentes da LPrS de Planejamento de Projetos de Software

Modelo de Característica Relação	Heurísticas	Modelo de Componente Elemento
W1 **in** A1	WPH2.1	**Interface de Dado Requerida e Mandatória de C1** Produto de Trabalho associado: W1 (*Software Demand*)
W2 **out** A1	WPH2.2	**Interface de Dado Provida e Mandatória de C1** Produto de Trabalho associado: W2 (*Project Charter*): Invariante/Mandatório
W2 **in** A2	WPH2.1	**Interface de Dado Requerida e Mandatória de C2, C3 e C4** Produto de Trabalho associado: W2 (*Project Charter*): Invariante/Mandatório
W2 **in** A5	WPH2.1	**Interface de Dado Requerida e Mandatória de C5, C6 e C7** Produto de Trabalho associado: W2 (*Project Charter*): Invariante/Mandatório
W4 **out** T2 -> **out** A3	WPH2.2 WPH2.3	**Interface de Dado Provida e Mandatória de C3** **Interface de Dado Provida e Opcional de C2** Produto de Trabalho associado: W4 (*FP Spreadsheet filled with Project Size*): Invariante e Opcional
W4 **in** A6	WPH2.1	**Interface de Dado Requerida e Mandatória de C6**

Modelo de Característica Relação	Heurísticas	Modelo de Componente Elemento
	WPH2.3	**Interface de Dado Requerida e Opcional de C5** Produto de Trabalho associado: W4 (*FP Spreadsheet filled with Project Size*): Invariante e Opcional
W5 **in** A4	WPH2.1 WPH2.3	**Interface de Dado Requerida e Mandatória de C4** **Interface de Dado Requerida e Opcional de C2** Produto de Trabalho associado: W5 (*Organizational Historical Base*): Invariante e Mantadório
W5 **in** A5 (A6 e A7)	WPH2.1 WPH2.3	**Interface de Dado Requerida e Mandatória de C5, C6 e C7** Produto de Trabalho associado: W5 (*Organizational Historical Base*): Invariante e Mantadório
W6 **out** A4	WPH2.2 WPH2.3	**Interface de Dado Provida e Mandatória de C4** **Interface de Dado Provida e Opcional de C2** Produto de Trabalho associado: W6 (*Project Size estimated by Historical Analysis*): Invariante e Opcional
W6 **in** A7	WPH2.1 WPH2.3	**Interface de Dado Requerida e Mandatória de C7** **Interface de Dado Requerida e Opcional de C5** Produto de Trabalho associado: W6 (Project Size estimated by Historical Analysis): Invariante e Opcional
W7 **out** A5, A6 e A7	WPH2.2 WPH2.3	**Interface de Dado Provida e Mandatória de C5, C6 e C7** Produto de Trabalho associado: W7 (*Project Effort (Man/Hour)*): Invariante e Mandatório
Regra de Composição Inclusiva R	IRH	**Pré-condição em C6** Determine by Indicative Function Points (FP) component AND FP Spreadsheet filled with Project Size interface
Regra de Composição Inclusiva R_1	IRH	**Pré-condição em C7** Determine by projects historical base component AND Project Size estimated by Historical Analysis interface
Regra de Composição Exclusiva X	ERH	**Restrição em C3 e C4 e nas interfaces de C2** <<*XOR*>>: Determine by Indicative Function Points (FP) component e Determine by projects historical base component And FP Spreadsheet filled with Project Size interface and Project Size estimated by Historical Analysis

O modelo de componentes resultante do mapeamento com aplicação das heurísticas pode ser visualizada na Figura II.2.

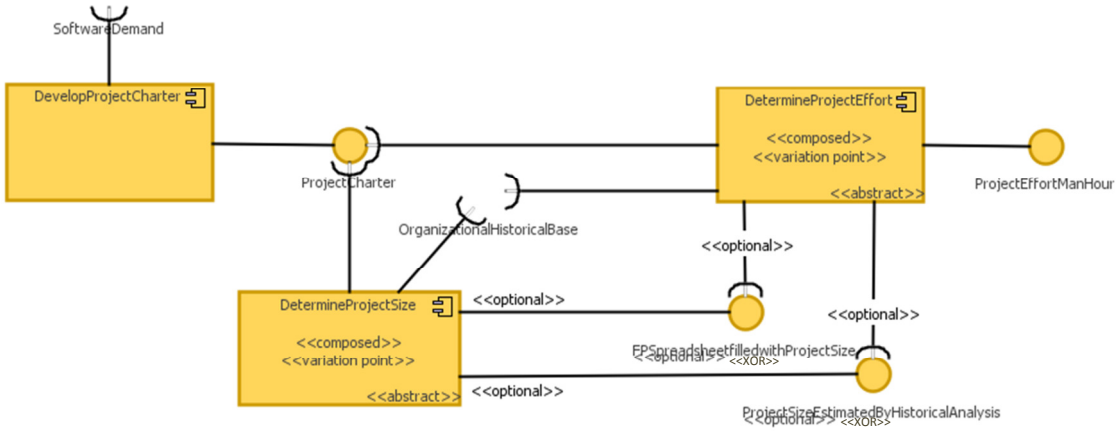


Figura II.2 – Modelo de Componentes da LPrS de Planejamento de Projetos de Software

A parte comportamental dos processos que corresponde a parte de fluxos de controle deve ser mapeada usando a heurística CFH. Tal heurística estabelece que cada fluxo de controle estabelecido entre unidades de trabalho mapeadas em diferentes

componentes devem ser descritos como interfaces de controle. Neste exemplo, todas as atividades escolhidas ocorrem de forma sequencial. Desta forma, uma porta deverá ser definida como entrada para fluxo de controle e outra porta deverá ser definida como saída para o fluxo de controle. Um conector deve ser estabelecido descrevendo o fluxo sequencial entre os componentes. O resultado deste mapeamento pode ser visto na Figura II.3.

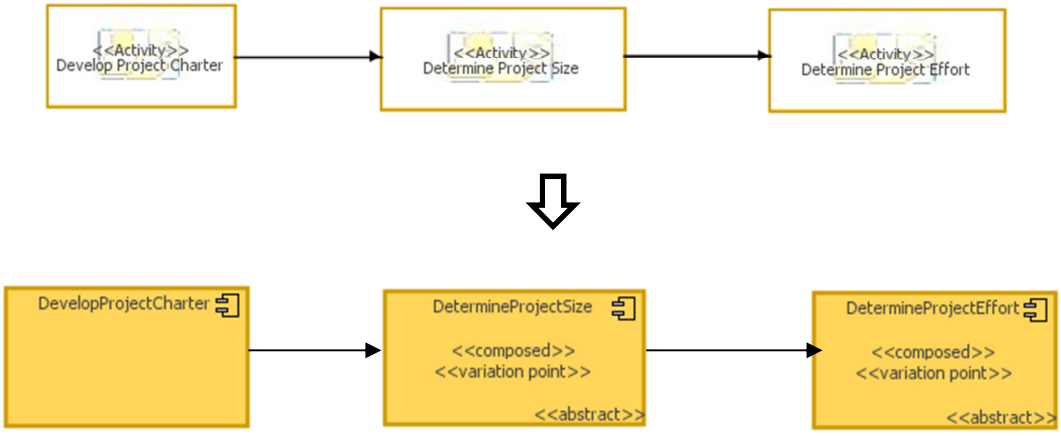


Figura II.3 – Visão Comportamental do Modelo de Componentes da LPrS de Planejamento de Projetos de Software

Apêndice III – Casos Discrepantes *PVMCheck*

Casos Discrepantes do pacote principal do *OdysseyProcess-FEX* (Grupo: Taxonomia)288

A. *Process Feature* (Característica de Processo)..... 288

B. *VariabilityType* (TipoVariabilidade) 289

C. *OptionalityType* (TipoOpcionalidade)..... 289

D. *WorkUnit* (Unidade de Trabalho) 290

E. *Activity* (Atividade)..... 291

F. *Task* (Tarefa)..... 291

G. *Step* (Passo) 292

H. *Role* (Papel)..... 293

I. *WorkProduct* (ProdutoDeTrabalho)..... 293

J. Tool (Ferramenta)..... 294

K. Discipline (Disciplina) 294

L. *Comment* (Nota)..... 295

Casos Discrepantes do pacote Relacionamentos do *OdysseyProcess-FEX* (Grupo: Relacionamentos)..... 296

A. *Relationship* (Relacionamentos)..... 296

B. *AlternativeRelationship*(Alternativo)..... 296

C. *Association* (Associação)..... 298

D. *Aggregation* (Agregação) 299

E. *Composition* (Composição) 299

F. *WorkUnitRoleRelationship* (LigaçãoUnidadeDeTrabalhoPapel)..... 300

G.*WorkUnitWorkProdutRelationship* - LigaçãoUnidadeDeTrabalhoProdutoDeTrabalho 301

H. *WorkProductRoleRelationship* - LigaçãoPapelProdutoDeTrabalho 302

I. *WorkUnitToolRelationship* - LigaçãoUnidadeDeTrabalhoFerramenta..... 303

Casos Discrepantes do pacote Regras de Composição do *OdysseyProcess-FEX* (Grupo: Regras de Composição) 304

A. RegraComposição 304

B. RegraComposiçãoInclusiva 305

C. RegraComposiçãoExclusiva 306

D. Expressão 306

E. ExpressaoLiteral 307

F. AND 307

G. OR 308

H. XOR 308

I. NOT 309

CASOS DISCREPANTES DO PACOTE PRINCIPAL DO *ODYSSEYPROCESS-FEX* (GRUPO: TAXONOMIA)

Estes casos foram descritos com base na taxonomia dos elementos de processos descritos no pacote principal da visão de características do metamodelo *OdysseyProcess-FEX* que define os elementos de processos de software reutilizáveis, seus atributos e propriedades.

A. *PROCESS FEATURE* (CARACTERÍSTICA DE PROCESSO)

Representa elementos estruturais de processo de software passíveis de reutilização na definição de novos processos.

Omissão

A01. Algum atributo de elemento de processo (*nome, tipoOpcionalidade, tipoVariabilidade*) não foi incluído ou está incompleto.

A02. Algum elemento de processo do domínio foi omitido do modelo.

Fato incorreto

A03. Um elemento de processo possui um nome diferente do definido pelo domínio.

A04. A descrição do elemento de processo não condiz com o significado do elemento modelado.

A05. Um elemento de processo possui classificações de opcionalidade e/ou variabilidade que não correspondem ao domínio modelado.

Inconsistência

A06. Mais de um elemento de processo possui o mesmo nome ou a mesma definição.

A07. Um elemento de processo possui mais de uma classificação de opcionalidade, ou seja, é simultaneamente opcional e mandatório.

A08. Um elemento de processo possui mais de uma classificação de variabilidade, ou seja, pode simultaneamente estar classificado como ponto de variação, variante e invariante.

A09. Um elemento de processo fixo no domínio não está classificado como invariante.

Ambiguidade

A10. A descrição dos atributos (*nome, descrição*) não está clara, sendo utilizados termos ambíguos.

Informação Estranha

A11. Um elemento de processo descrito não pertence ao domínio de processo de software modelado.

A12. Atributos (nome, descrição) foram descritos com informações desnecessárias.

B. *VariabilityType* (TipoVariabilidade)

TipoVariabilidade é uma classe do tipo "*ennumeration*", cujos literais determinam o tipo de variabilidade presente em um Elemento de Processo. Pode assumir os seguintes valores: ponto de variação (pontos configuráveis por meio das variantes), variante (são elementos NECESSARIAMENTE ligados a um ponto de variação que atuam como alternativas) e invariante (elementos "fixos", não).

Omissão

B01. Uma classificação de variabilidade de um elemento de processo que compõe o domínio não foi informada no modelo.

Fato Incorreto

B02. Um elemento de processo definido como um ponto de configuração no domínio não está classificado como ponto de variação.

B03. Um elemento de processo definido como uma alternativa de configuração no domínio não está classificado como variante.

B04. Um elemento de processo fixo no domínio não está classificado como invariante.

Inconsistência

-

Ambiguidade

-

Informação Estranha

-

C. *OptionalityType* (TipoOpcionalidade)

TipoOpcionalidade é uma classe do tipo "*ennumeration*", cujos literais determinam o tipo de opcionalidade presente em um Elemento de Processo. Pode assumir os seguintes valores: mandatório (representa elemento de processo que DEVERÁ estar presente nos processos derivados) e opcional (elemento que pode ou não estar presente nos processos derivados).

Omissão

C01. Uma classificação de opcionalidade de um elemento de processo que compõe o domínio não foi informada no modelo.

Fato Incorreto

C02. Um elemento de processo definido como opcional no domínio foi classificado como mandatório, ou vice-versa.

Inconsistência

-

Ambiguidade

-

Informação Estranha

-

D. *WorkUnit* (Unidade de Trabalho)

Elemento que representa o conceito de uma unidade de ação a ser executada para alcançar determinado resultado.

Levar em consideração todos os cenários descritos em *Elemento de Processo, TipoVariabilidade* e *TipoOpcionalidade*.

Omissão

D01. Atributos (*propósito, pré-condição, pós-condição*) não foram incluídos ou estão incompletos.

Fato Incorreto

D02. Atributo propósito com uma finalidade em desacordo com o resultado esperado para a unidade de trabalho dentro do domínio modelado.

Inconsistência

D03. Descrições de pré-condições não correspondem a condições que permitem o início da execução da unidade de trabalho.

D04. Descrições de pós-condições não correspondem a restrições que devem ser atendidas para permitir o término da execução da unidade de trabalho.

D05. Uma pré-condição modelada como pós-condição, ou vice-versa.

D06. Unidades de trabalho diferentes apresentam o mesmo propósito dentro do domínio modelado.

Ambiguidade

D07. A descrição de atributos (propósito, pré-condições, pós-condições) correspondentes à unidade de trabalho não está clara.

Informação Estranha

D08. Descrição de propósito não se aplica à unidade de trabalho correspondente

D09. Descrição de algum item de pré-condição não se aplica à unidade de trabalho correspondente.

D10. Descrição de algum item de pós-condição não se aplica à unidade de trabalho correspondente.

E. *ACTIVITY* (ATIVIDADE)

Elemento de processo que representa o agrupamento de unidades de trabalhos menores, representadas por outras atividades ou por tarefas.

Levar em consideração todos os cenários descritos em Unidade de Trabalho.

Omissão

-

Fato Incorreto

E01. Relações com outras atividades ou tarefas em desacordo com o domínio modelado.

Inconsistência

E02. Mais de uma atividade apresenta o mesmo conjunto de unidades de trabalho como sua composição.

Ambiguidade

-

Informação Estranha

-

F. *TASK* (TAREFA)

Elemento de processo que representa uma unidade descrita por meio de passos. Sua granularidade é definida como o menor agrupamento possível de ações para alcançar resultados,

Levar em consideração todos os cenários descritos em Unidade de Trabalho.

Omissão

-

Fato Incorreto

F01. Conjunto de passos descritos como composição da tarefa em desacordo com o domínio modelado.

Inconsistência

F02. Uma tarefa ser modelada como composição de outras tarefas.

F03. Mais de uma tarefa apresenta o mesmo conjunto de passos como sua composição.

Ambiguidade

-

Informação Estranha

-

G. *STEP* (PASSO)

O fluxo de passos descreve como atingir o propósito definido pela tarefa a qual pertence.

Omissão

G01. Algum atributo do passo (nome, tipoOpcionalidade) não foi incluído ou está incompleto.

G02. Algum passo pertencente ao domínio foi omitido do modelo.

Fato Incorreto

G03. Um passo possui um nome diferente do definido pelo domínio.

G04. A descrição do passo não condiz com o significado do elemento modelado.

G05. Um passo possui classificação de opcionalidade que não corresponde ao domínio modelado.

Inconsistência

G06. Um passo possui mais de uma classificação de opcionalidade, ou seja, é simultaneamente opcional e mandatório.

G07. O fluxo de passos não atinge o propósito definido pela tarefa a qual pertence.

G08. Mais de um passo possui o mesmo nome ou a mesma definição.

Ambiguidade

G09. A descrição dos atributos (nome, descrição) não está clara, sendo utilizados termos ambíguos.

Informação Estranha

G10. Um passo descrito não pertence ao domínio de processo de software modelado.

G11. Atributos (nome, descrição) foram descritos com informações desnecessárias.

H. *Role* (Papel)

Elemento de processo que representa um conjunto de habilidades, competências e responsabilidades de um indivíduo ou um conjunto de indivíduos.

Levar em consideração todos os cenários descritos em Elemento de Processo, TipoVariabilidade e TipoOpcionalidade.

Omissão

H01. O conjunto de competências e habilidades associado ao papel não foi incluído ou está incompleto.

Fato Incorreto

H02. O conjunto de competências e habilidades associado ao papel estão em desacordo com o domínio modelado.

Inconsistência

H03. Papéis diferentes apresentam o mesmo conjunto de competências e habilidades dentro do domínio modelado.

Ambiguidade

H04. A descrição do conjunto de competências e habilidades associado ao papel não está clara.

Informação Estranha

H05. A descrição do conjunto de competências e habilidades associado ao papel é não corresponde ao domínio modelado.

I. *WorkProduct* (ProdutoDeTrabalho)

Elemento de processo que representa um artefato consumido, modificado ou produzido por uma unidade de trabalho.

Levar em consideração todos os cenários descritos em Elemento de Processo, TipoVariabilidade e TipoOpcionalidade.

Omissão

-

Fato Incorreto

-

Inconsistência

-

Ambiguidade

-

Informação Estranha

-

J. TOOL (FERRAMENTA)

Elemento de processo que descreve capacidades de uma unidade de automação para apoiar a exceção da unidade de trabalho associada.

Levar em consideração todos os cenários descritos em Elemento de Processo, TipoVariabilidade e TipoOpcionalidade.

Omissão

-

Fato Incorreto

J01. O elemento ferramenta recebe alguma classificação de variabilidade diferente de invariante.

Inconsistência

-

Ambiguidade

-

Informação Estranha

-

K. DISCIPLINE (DISCIPLINA)

A disciplina é um conjunto de unidades de trabalho que estão relacionadas com uma grande área de interesse no âmbito do domínio como um todo.

Omissão

K01. Uma disciplina pertencente ao domínio não está incluída na modelagem.

K02. Unidades de trabalho que compõem a disciplina não estão incluídas na modelagem ou estão incompletas.

Fato Incorreto

K03. As unidades de trabalho não foram corretamente agrupadas em disciplinas de acordo com o descrito pelo domínio.

Inconsistência

K04. Uma disciplina criada sem unidades de trabalho associadas.

K05. Unidades de trabalho que compõem disciplina não possuem similaridades de interesses e propósitos de resultados.

Ambiguidade

K06. O nome que representa a disciplina no domínio não descreve com clareza o agrupamento indicado para as unidades de trabalho.

Informação Estranha

K07. A disciplina incluída não pertence ao domínio de processo modelado.

L. *COMMENT* (NOTA)

Um comentário / nota é uma anotação textual associada a um elemento com o objetivo de acrescentar alguma informação adicional para organização de conhecimento sobre o domínio.

Omissão

L01. Um comentário pertencente ao domínio não foi descrito no modelo.

Fato Incorreto

L02. Uma nota contém informações que contradizem o domínio modelado.

L03. Uma nota criada com associações incorretas.

Inconsistência

L04. O conteúdo da nota é relevante para o domínio do processo modelado, devendo ser representado como outro tipo de elemento.

Ambiguidade

L05. Nota não descrita com clareza.

Informação Estranha

L06. Nota desnecessária para o domínio modelado.

Tabela III.1. Tabela que apresenta a quantidade de cenários discrepantes por elemento do metamodelo (Pacote Principal)

Elemento	Cenários
Característica de Processo	12
Unidade de Trabalho	10
Atividade	02
Tarefa	03
Passo	11
Papel	05
ProdutoDeTrabalho	-
Ferramenta	01
Disciplina	07
Nota	06
TOTAL	**57**
Variabilidade	04
Opcionalidade	02

Tabela III.2 - Tabela que apresenta a quantidade de cenários discrepantes por tipo de defeito (Pacote Principal – Grupo: Elementos de Processo)

Defeitos	Quantidade
Omissão	11
Fato Incorreto	18
Inconsistência	18
Ambigüidade	06
Informação Estranha	10
TOTAL	**63**

CASOS DISCREPANTES DO PACOTE RELACIONAMENTOS DO *ODYSSEYPROCESS-FEX* (GRUPO: RELACIONAMENTOS)

O pacote Relacionamentos consiste na representação das relações disponibilizadas entre os elementos de processos descritos no pacote Principal.

A. *RELATIONSHIP* (RELACIONAMENTOS)

Relationship (Relacionamento) é um conceito abstrato que especifica algum tipo de relacionamento entre elementos (OMG, 2005). Classe Abstrata.

Omissão

-

Fato incorreto

-

Inconsistência

-

Ambiguidade

-

Informação Estranha

-

B. *ALTERNATIVERELATIONSHIP*(ALTERNATIVO)

Relacionamento existente entre um ponto de variação e suas variantes

Omissão

B01. Relacionamento Alternativo pertencente ao domínio não foi representado no modelo.

B02. Atributo Cardinalidade mínima definida no domínio não foi representada no modelo.

B03. Atributo Cardinalidade máxima definida no domínio não foi representada no modelo.

Fato Incorreto

B04. Um elemento de processo definido como uma alternativa de configuração no domínio não está classificado como variante.

B05. A origem do relacionamento não é um pontoVariação.

B06. O destino do relacionamento não é uma variante.

B07. Elemento de processo classificado como ponto de variação e opcional possui um relacionamento alternativo com um elemento variante classificado como mandatório.

B08. Relacionamento de mútua exclusividade com cardinalidade máxima diferente de 1.

B09. Elementos classificados como ponto de variação e mandatórios com cardinalidade mínima igual a 0.

B10. Elementos classificados como ponto de variação e opcionais com cardinalidade mínima diferente de 0.

B11. Um mesmo elemento de processo classificado como variante associado a dois ou mais elementos de processo classificados como ponto de variação.

B12. Um relacionamento com origem (ponto de variação) classificada como atividade NÃO tem como destino (variantes) elementos de processo classificados como atividade.

B13. Um relacionamento com origem (ponto de variação) classificada como tarefa NÃO tem como destino (variantes) elementos de processo classificados como tarefa.

B14. Um relacionamento com origem (ponto de variação) classificada como papel NÃO tem como destino (variantes) elementos de processo classificados como papel.

B15. Um relacionamento com origem (ponto de variação) classificada como produto de trabalho NÃO tem como destino (variantes) elementos de processo classificados como produto de trabalho.

B16. Um relacionamento com origem (ponto de variação) classificada como ferramenta NÃO tem como destino (variantes) elementos de processo classificados como ferramenta.

Inconsistência

B17. Atributos de cardinalidade (mínima e máxima) não estão de acordo com os definidos pelo domínio.

Ambiguidade

-

Informação Estranha

B18. Atributo Cardinalidade ou associações (variante, pontoVariação) informados não são aplicáveis ou não contribuem para a compreensão do domínio.

C. *ASSOCIATION* (ASSOCIAÇÃO)

Uma associação descreve um conjunto de tuplas cujos valores se referem a instâncias tipadas (OLIVEIRA *et al.*, 2005). É apresentado por meio de suas especializações Composição e Agregação.

Omissão

-

Fato Incorreto

C01. Um relacionamento com origem (extremidadeTodo) classificada como atividade NÃO tem como destino (extremidadeParte) um elemento de processo classificado como atividade ou tarefa.

C02. Um relacionamento com origem (extremidadeTodo) classificada como papel NÃO tem como destino (extremidadeParte) um elemento de processo classificado como papel.

C03. Um relacionamento com origem (extremidadeTodo) classificada como produto de trabalho NÃO tem como destino (extremidadeParte) um elemento de processo classificado como produto de trabalho.

C04. Um relacionamento com origem (extremidadeTodo) classificada como ferramenta NÃO tem como destino (extremidadeParte) um elemento de processo classificado como ferramenta.

Inconsistência

-

Ambiguidade

-

Informação Estranha

-

D. *AGGREGATION* (AGREGAÇÃO)

Uma associação pode ser representada por agregação (relacionamento do todo/parte).

Omissão

D01. Um relacionamento de agregação pertencente ao domínio não foi representado no modelo.

Fato Incorreto

D02. Associações não binárias estabelecidas como um relacionamento de agregação.

Inconsistência

D03. Elemento de processo que corresponde ao extremidadeTodo é representado como parte ou vice-versa.

Ambiguidade

-

Informação Estranha

-

E. *COMPOSITION* (COMPOSIÇÃO)

Uma associação pode representar uma composição (isto é, um relacionamento de todo/parte). A composição é um relacionamento mais forte do que agregação, e requer que em um dado momento, uma instância esteja incluída em no máximo uma composição (OLIVEIRA *et al*., 2005).

Omissão

E01. Um relacionamento de composição pertencente ao domínio não foi representado no modelo.

Fato Incorreto

E02. Associações não binárias estabelecidas como relacionamento de composição.

E03. Em um relacionamento de composição que possui o elemento que representa a extremidadeTodo é classificado como opcional não é permitido que o elemento que representa a extremidadeParte seja classificado mandatório.

Inconsistência

E04. Elemento de processo que corresponde ao extremidadeTodo é representado como parte ou vice-versa.

Ambiguidade

-

Informação Estranha

-

F. *WorkUnitRoleRelationship* (LigaçãoUnidadeDeTrabalhoPapel)

Estabelece um relacionamento de associação entre um elemento de processo da categoria Unidade de Trabalho (Atividade ou Tarefa) e um ou vários elementos de processo da categoria Papel participantes na execução da unidade de trabalho.

Omissão

F01. Atributos (opcional, estereótipo e tipoResponsabilidade) que constam no domínio não foram representados no modelo.

F02. Uma ligação entre um papel e uma unidade de trabalho que consta no domínio não foi representada no modelo.

Fato incorreto

F03. Um elemento de processo Papel sem relacionamento com alguma Unidade de Trabalho.

F04. Um elemento de processo Unidade de Trabalho sem relacionamento com algum Papel.

Inconsistência

F05. Declaração do tipo de responsabilidade de um papel descrito no modelo não é correspondente com a sua participação na execução do trabalho determinada no domínio.

F06. Participação do elemento de processo Papel classificada como opcional no modelo, sendo que é mandatória para a execução da atividade ou tarefa (unidade de trabalho) associada ou vice-versa.

F07. Um elemento de processo Papel classificado como mandatório sendo que é associado apenas a unidades de trabalho classificadas como opcionais.

F08. Um mesmo papel com mais de uma relação com uma mesma unidade de trabalho (com responsabilidade igual ou diferente).

Ambiguidade

F09. A descrição dos atributos (*opcional, estereótipo e tipoResponsabilidade*) não está clara, sendo utilizados termos ambíguos.

Informação Estranha

F10. Uma ligação entre um papel e uma unidade de trabalho que não constam no domínio foi representada no modelo.

G. *WorkUnitWorkProdutRelationship* - LigaçãoUnidadeDeTrabalhoProdutoDeTrabalho

Estabelece um relacionamento de associação entre um elemento de processo da categoria Unidade de Trabalho (Atividade ou Tarefa) e um ou vários elementos da categoria Produto de Trabalho que representam artefatos a serem consumidos, produzidos ou modificados pela execução da unidade de trabalho.

Omissão

G01. Atributos (opcional, estereótipo e tipoProdutoDeTrabalho) que constam no domínio não foram representados no modelo.

G02. Uma ligação entre um Produto de Trabalho e uma Unidade de Trabalho que consta no domínio não foi representada no modelo.

Fato incorreto

G03. Um elemento de processo Unidade de Trabalho sem relacionamento com algum Produto de Trabalho.

G04. Um elemento de processo Produto de Trabalho sem relacionamento com alguma Unidade de Trabalho.

Inconsistência

G05. Participação do elemento Produto de Trabalho classificada como opcional no modelo, sendo que é um produto de trabalho classificado como único resultado de uma Unidade de Trabalho associada.

G06. Um elemento de processo Produto de Trabalho classificado como mandatório sendo que é associado apenas Unidades de Trabalho classificadas como opcionais.

G07. Um produto de trabalho classificado como insumo (entrada) no domínio é representado como produto de trabalho classificado como resultado (saída) no modelo, ou vice-versa.

G08. Um produto de trabalho classificado como artefato modificado no domínio não é representado como tal (estereótipo <<inout>> do relacionamento estabelecido no modelo).

G09. Uma unidade de trabalho sem produto de trabalho associado classificado como resultado (saída).

Ambiguidade

G10. A descrição dos atributos (opcional, estereótipo e tipoProdutoDeTrabalho) não está clara, sendo utilizados termos ambíguos.

Informação Estranha

G11. Uma ligação entre um Produto de Trabalho e uma Unidade de Trabalho que não consta no domínio foi representada no modelo.

H. *WorkProductRoleRelationship* - LigaçãoPapelProdutoDeTrabalho

Estabelece um relacionamento de associação entre um elemento de processo da categoria Produto de Trabalho (*WorkProduct*) e um ou vários elementos de processo da categoria Papel (Role) com algum tipo de responsabilidade sobre o produto de trabalho.

Omissão

H01. Atributos (opcional, estereótipo e tipoResponsabilidade) que constam no domínio não foram representados no modelo.

H02. Uma ligação entre um Papel e um Produto de trabalho que consta no domínio não foi representada no modelo.

Fato incorreto

-

Inconsistência

H03. Declaração do tipo de responsabilidade de um papel descrito no modelo não é correspondente com a sua participação no desenvolvimento do produto de trabalho determinada no domínio.

H04. Participação do elemento de processo Papel classificada como opcional no modelo, sendo que é mandatória para o desenvolvimento do Produto de Trabalho.

H05. Um elemento de processo Papel classificado como mandatório sendo que é associado apenas a Produtos de trabalho classificados como opcionais.

H06. Um mesmo Papel com mais de uma relação com um mesmo Produto de Trabalho (com responsabilidade igual ou diferente).

Ambiguidade

H07. A descrição dos atributos (opcional, estereótipo e tipoResponsabilidade) não está clara, sendo utilizados termos ambíguos.

Informação Estranha

H08. Uma ligação entre um Papel e um Produto de Trabalho que não consta no domínio foi representada no modelo.

I. *WorkUnitToolRelationship* - LigaçãoUnidadeDeTrabalhoFerramenta

Estabelece um relacionamento de associação entre um elemento de processo da categoria Unidade de Trabalho (Atividade ou Tarefa) e um ou vários elementos de processo da categoria Ferramenta que podem prover algum tipo de apoio ferramental na execução da unidade de trabalho.

Omissão

I01. Atributos (opcional e estereótipo) que constam no domínio não foram representados no modelo.

I02. Uma ligação entre uma Ferramenta e uma Unidade de Trabalho que consta no domínio não foi representada no modelo.

Fato incorreto

-

Inconsistência

I03. Ligação da Ferramenta como opcional no modelo, sendo que é mandatória para a Unidade de Trabalho no domínio.

Ambiguidade

I04. A descrição dos atributos (opcional, estereótipo e tipoResponsabilidade) não está clara, sendo utilizados termos ambíguos.

Informação Estranha

I05. Uma ligação entre uma Ferramenta e uma Unidade de Trabalho que não consta no domínio foi representada no modelo

Tabela III.3 - Tabela que apresenta a quantidade de cenários discrepantes por elemento do metamodelo (Pacote Relacionamentos)

Elemento	Cenários
Relacionamentos	-
Alternativos	18
Associação	04
Agregação	03
Composição	04
Unidade/Papel	10
Unidade/Produto	11
Papel/Produto	08
Unidade/Ferramenta	05
TOTAL	**63**

Tabela III.4 - Tabela que apresenta a quantidade de cenários discrepantes por tipo de defeito (Pacote Relacionamentos)

Defeitos	Quantidade
Omissão	13
Fato Incorreto	24
Inconsistência	17
Ambigüidade	4
Informação Estranha	5
TOTAL	**63**

CASOS DISCREPANTES DO PACOTE REGRAS DE COMPOSIÇÃO DO *ODYSSEYPROCESS-FEX* (GRUPO: REGRAS DE COMPOSIÇÃO)

O pacote de Regras de Composição (*CompositionRules*) corresponde ao mesmo pacote da notação *Odyssey-FEX* (OLIVEIRA, 2006) com adaptações apenas do conceito de característica para elemento de processo. Esse pacote contém as classes que definem a semântica das regras de dependência e mútua exclusividade entre elementos de processo

A. REGRACOMPOSIÇÃO

Regras que definem restrições existentes entre elementos de processos. Tais regras incluem relações do tipo “exclui” e “requer” entre elementos de processos ou quaisquer conjuntos de elementos de processos.

Omissão

A01. Uma regra de composição não possui expressão antecedente e/ou consequente.

Fato incorreto

A02. Regra de composição contraditória, onde o antecedente é igual ao consequente.

A03. Regra de composição com o antecedente definido e consequente nulo ou vice-versa.

Inconsistência

A04. Uma expressão que deveria ser representada como antecedente está representada como consequente, ou vice-versa.

Ambiguidade

A05. Atributo que referencia o nome da Regra de Composição não é claro, ou utiliza termos ambíguos.

Informação Estranha

-

B. REGRACOMPOSIÇÃOINCLUSIVA

Regras de composição que indicam dependência entre duas ou mais elementos de processos. Indicam as regras do tipo “requer”.

Omissão

B01. Uma regra inclusiva do domínio não está incluída no modelo.

Fato incorreto

B02. Quando o elemento de processo que representa o antecedente de uma Regra de Composição Inclusiva simples (antecedente e consequente formados por expressões literais – apenas um elemento de processo) é classificado como opcional e o consequente classificado como mandatório.

Inconsistência

B03. Uma Regra de Composição Inclusiva representada como uma Regra de Composição Exclusiva.

Ambiguidade

-

Informação Estranha

-

C. REGRACOMPOSIÇÃOEXCLUSIVA

Regras de composição que indicam mútua exclusividade entre duas ou mais elementos de processos. Indicam as regras do tipo “exclui”.

Omissão

C01. Uma regra exclusiva do domínio não está incluída no modelo.

Fato incorreto

C02. Regra de Composição Exclusiva envolvendo elementos de processos classificados como mandatórios.

Inconsistência

C03. Uma Regra de Composição Exclusiva representada como uma Regra de Composição Inclusiva.

Ambiguidade

-

Informação Estranha

-

D. EXPRESSÃO

Expressões que constituem as Regras de Composição. Podem ser Booleanas ou Literais.

Omissão

-

Fato incorreto

-

Inconsistência

-

Ambiguidade

-

Informação Estranha

-

E. EXPRESSAOLITERAL

Expressão Literal é a expressão mais elementar de uma Regra de Composição. É representada por um único elemento de processo.

Omissão

E01. Uma expressão literal do domínio não está incluída no modelo.

E02. O elemento de processo que constitui a expressão literal descrito no domínio não está incluído no modelo.

Fato incorreto

-

Inconsistência

E03. O elemento de processo que constitui a expressão literal descrito no domínio não corresponde ao elemento definido no modelo para determinada regra de composição.

Ambiguidade

-

Informação Estranha

E04. Uma expressão literal constituída de elemento de processo que não está no domínio, mas foi incluído no modelo, ou contém informações que não contribuem para a compreensão do domínio.

F. AND

Expressão que representa o AND lógico.

Omissão

F01. Uma expressão do domínio não está incluída no modelo.

F02. A expressão antes do conector AND (expEsquerda) não foi representada no modelo.

F03. A expressão depois do conector AND (expDireita) não foi representada no modelo.

Fato incorreto

-

Inconsistência

F04. Expressão anterior (expEsquerda) representada como posterior (expDireita), ou vice-versa.

Ambiguidade

-

Informação Estranha

F05. A expressão AND ou as expressões que a formam não estão no domínio, mas foram incluídas no modelo, ou contém informações que não contribuem para a compreensão do domínio.

G. OR

Expressão que representa o OR lógico.

Omissão

G01. Uma expressão do domínio não está incluída no modelo.

G02. A expressão antes do conector OR (expEsquerda) não foi representada no modelo.

G03. A expressão depois do conector OR (expDireita) não foi representada no modelo.

Fato incorreto

-

Inconsistência

G04. Expressão anterior (expEsquerda) representada como posterior (expDireita), ou vice-versa.

Ambiguidade

-

Informação Estranha

G05. A expressão OR ou as expressões que a formam não estão no domínio, mas foram incluídas no modelo, ou contém informações que não contribuem para a compreensão do domínio.

H. XOR

Expressão que representa o XOR lógico.

Omissão

H01. Uma expressão do domínio não está incluída no modelo.

H02. A expressão antes do conector XOR (expEsquerda) não foi representada no modelo.

H03. A expressão depois do conector XOR (expDireita) não foi representada no modelo.

Fato incorreto

-

Inconsistência

H04. Expressão anterior (expEsquerda) representada como posterior (expDireita), ou vice-versa.

Ambiguidade

-

Informação Estranha

H05. A expressão XOR ou as expressões que a formam não estão no domínio, mas foram incluídas no modelo, ou contém informações que não contribuem para a compreensão do domínio.

I. NOT

Expressão que representa o NOT lógico.

Omissão

I01. Uma expressão do domínio não está incluída no modelo.

I02. A expressão depois do conector NOT não foi representada no modelo.

Fato incorreto

-

Inconsistência

-

Ambiguidade

-

Informação Estranha

I03. A expressão NOT ou a expressão que a forma não está no domínio, mas foi incluída no modelo, ou contém informações que não contribuem para a compreensão do domínio.

Tabela III.5 - Tabela que apresenta a quantidade de cenários discrepantes por elemento do metamodelo (Pacote Regras de Composição)

Elemento	Cenários
RegraComposição	05
Inclusiva	03
Exclusiva	03
Expressão	-
ExpressãoLiteral	04
AND	05
OR	05
XOR	05
NOT	03
TOTAL	**33**

Tabela III.6 - Tabela que apresenta a quantidade de cenários discrepantes por tipo de defeito (Pacote Regras de Composição)

Defeitos	Quantidade
Omissão	16
Fato Incorreto	4
Inconsistência	7
Ambigüidade	1
Informação Estranha	5
TOTAL	**33**

Apêndice IV – Modelagem da LPrS de Medição de Software com *OdysseyProcess-FEX*

LPrS Medição

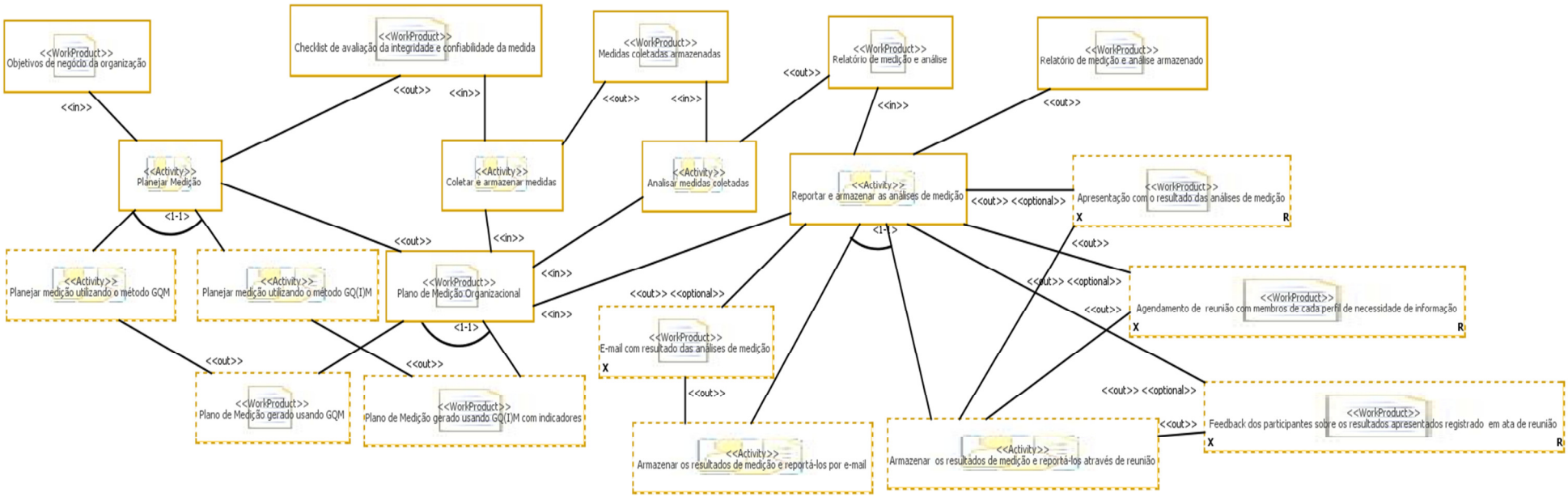
O Grupo de Medição da Organização é responsável pelo Planejamento e execução das atividades relacionadas à medição dos produtos desenvolvidos e dos processos executados na organização e em seus projetos, como forma de apoiar o alcance dos objetivos organizacionais. A organização desenvolve software em diferentes tecnologias e domínios de aplicação, além de também adquirir software desenvolvido externamente. Para cada projeto, equipes são montadas de acordo com as necessidades, variando de tamanho e tipo de proximidade dos integrantes, podendo estar reunidos em um mesmo local de trabalho ou dispersos geograficamente. Assim, o Grupo de Medição precisa atuar na definição de planos de medição que atendam a diferentes cenários e contexto específicos de projetos.

Devido a grande variedade de possibilidades de execução do processo de medição, a organização definiu uma LPrS para Medição no Contexto Organizacional. A linha de processos modelada abrange todos os componentes necessários à coleta, analise e divulgação das medidas relativas aos produtos desenvolvidos e aos processos executados na organização e em seus projetos, como forma de apoiar o alcance dos objetivos organizacionais. Dessa forma, os componentes presentes nesta linha são executados de forma periódica no contexto organizacional e à medida que os projetos vão sendo executados. Essa linha de processos é aderente ao MR-MPS Nível F.

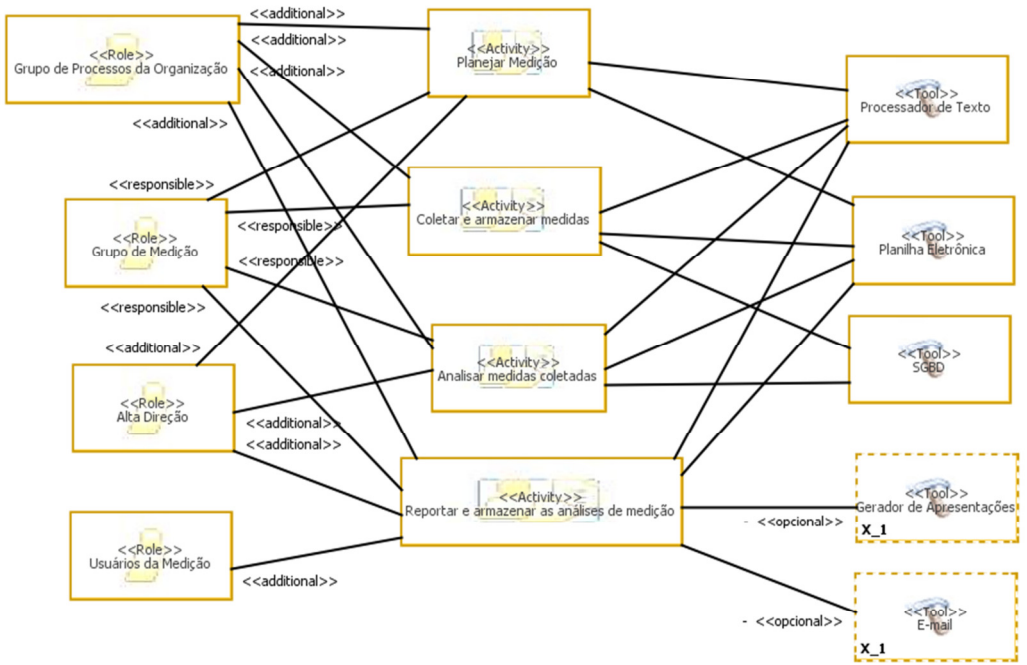
Nível de Análise - Modelo de Caraterísticas

Visão Estrutural

Unidades de Trabalho e Produtos de Trabalho

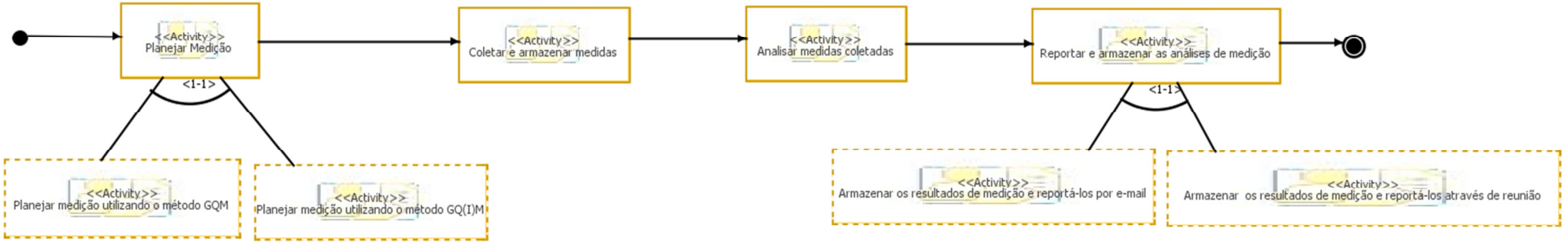


Unidades de Trabalho, Papéis e Ferramentas



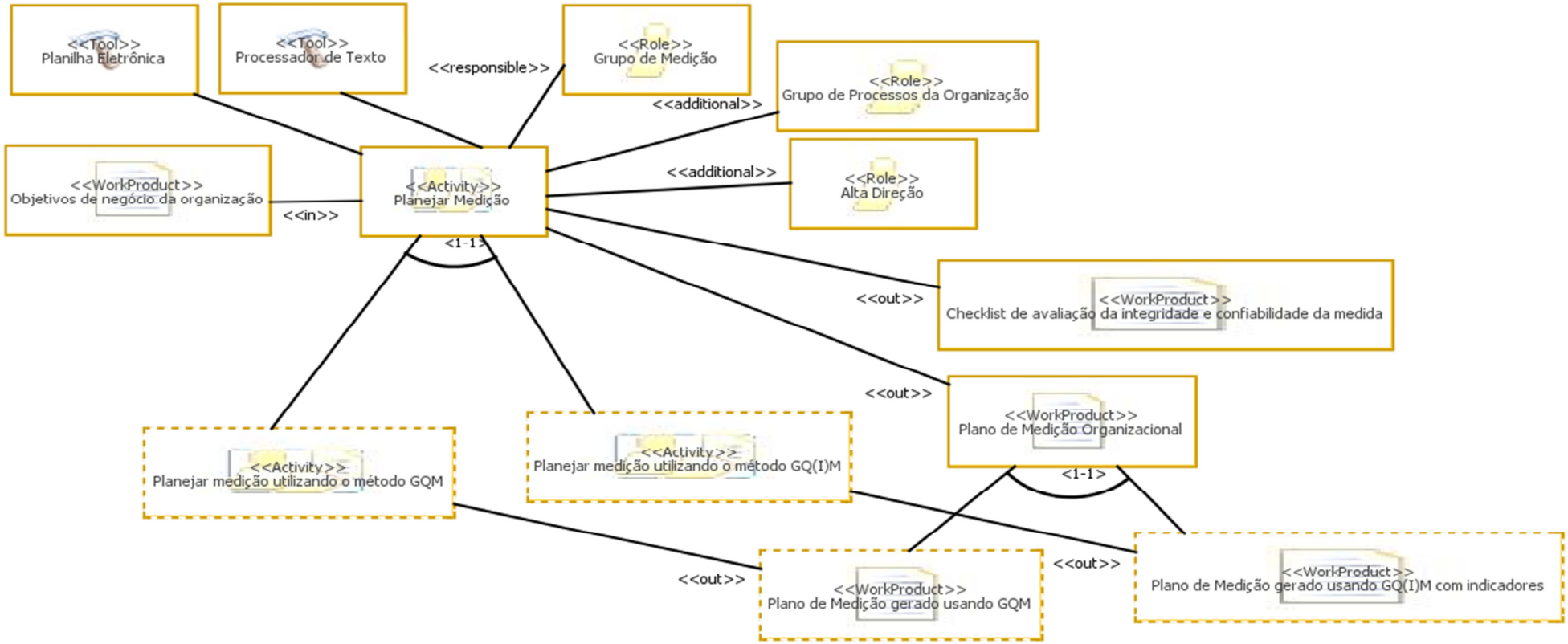
- **Regra de Composição Inclusiva:**
  - **R:** Apresentação com o resultado das análises de medição *requires* (Agendamento de reunião com membros de cada perfil de necessidade de informação AND Feedback dos participantes sobre os resultados apresentados registrado em ata de reunião)
- **Regra de Composição Exclusiva:**
  - **X**: E-mail com resultado das análises de medição *excludes* (Apresentação com o resultado das análises de medição AND (Agendamento de reunião com membros de cada perfil de necessidade de informação AND Feedback dos participantes sobre os resultados apresentados registrado em ata de reunião))
  - **X_1**: E-mail *excludes* Gerador de Apresentações

Visão Comportamental



Atividade: Planejar Medição

- Descrição: Estabelecer plano de medição organizacional pela definição dos objetivos de medição oriundos das necessidades de informação organizacional, das medidas a serem coletadas, dos procedimentos de coleta e armazenamento, bem como dos procedimentos de análise e divulgação dos resultados. Para a identificação dos objetivos e necessidades de informação, podem ser utilizadas diversas técnicas, tais como: GQM (*Goal Question Metric*) e GQ(I)M (*Goal Question (Indicator) Measure*).
- Variantes:
  - Planejar medição utilizando o método GQM;
  - Planejar medição utilizando o método GQ(I)M.

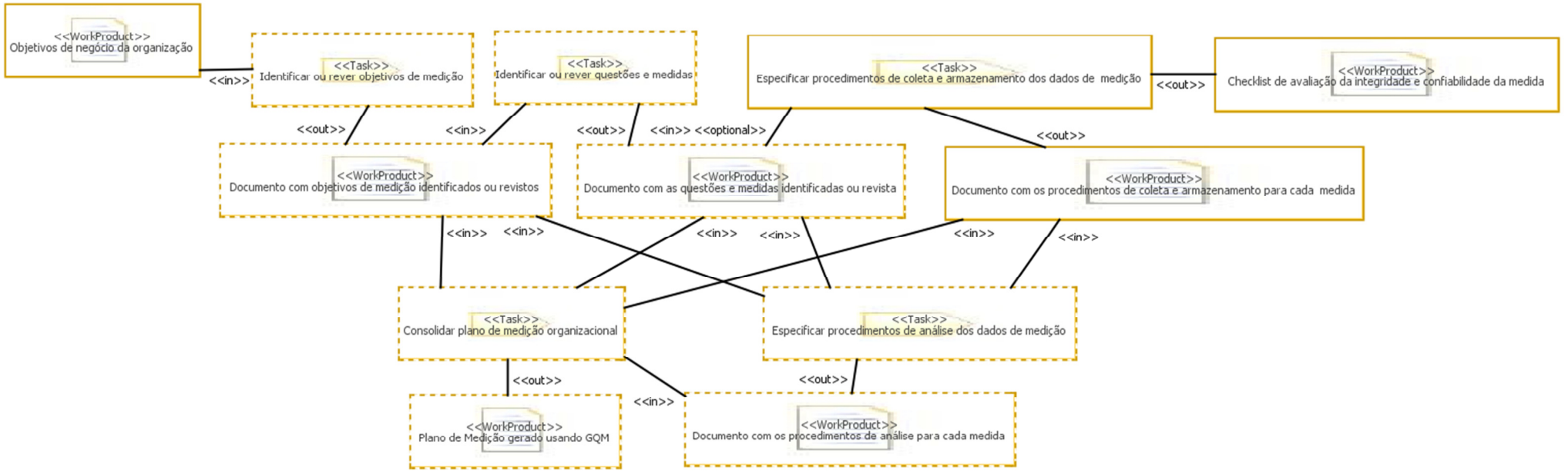


Atividade Variante: Planejar medição utilizando o método GQM

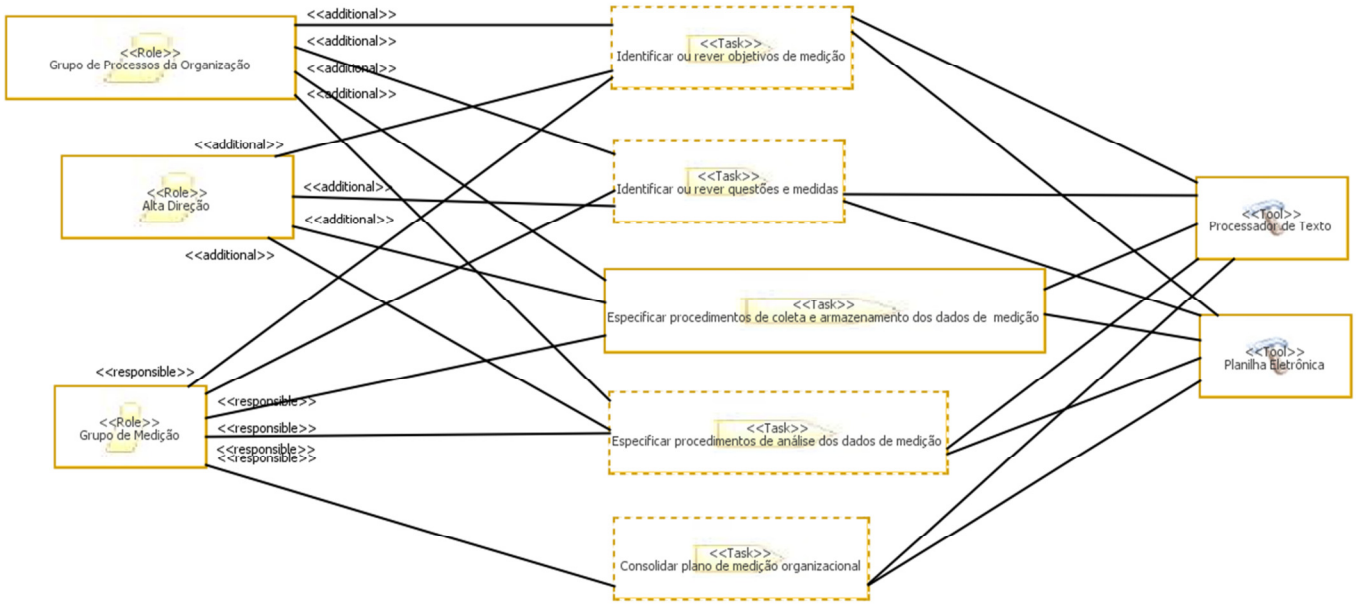
- Descrição: Estabelecer plano de medição organizacional pela definição dos objetivos de medição oriundos das necessidades de informação organizacional utilizando o método GQM (*Goal Question Metric*). Esse plano deve conter quais medidas deverão ser coletadas, quais os procedimentos de coleta e armazenamento deverão ser adotados para cada medida, bem como quais os procedimentos de análise e divulgação dos resultados de medição.
- Critérios de Entrada: Ter-se iniciado a implantação do processo de medição ou haver a necessidade de rever o plano de medição organizacional
- Critérios de Saída: Ter-se o plano de medição organizacional definido ou revisado.

Visão Estrutural

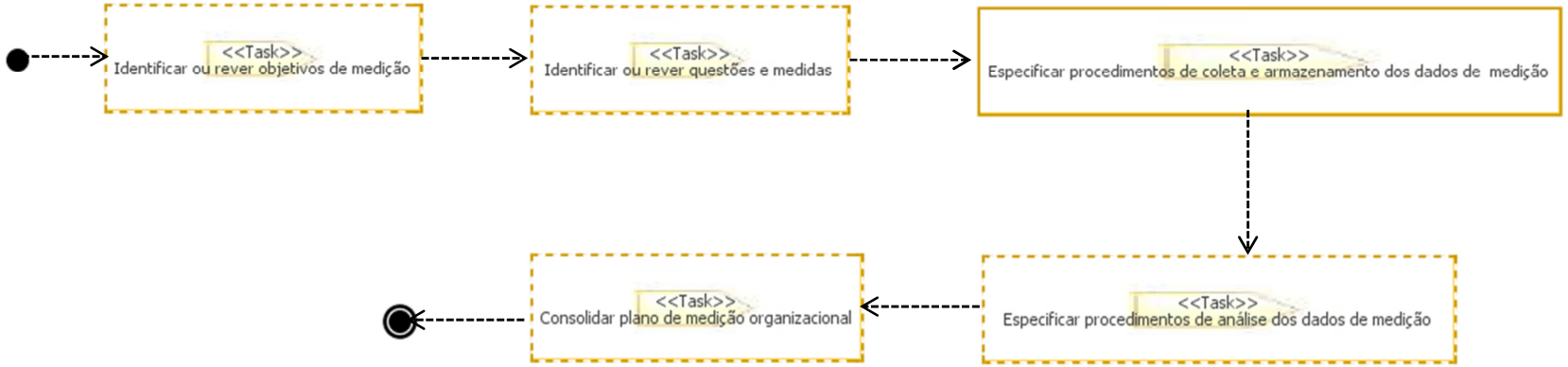
Unidades de Trabalho e Produtos de Trabalho



Unidades de Trabalho, Papéis e Ferramentas



Visão Comportamental



1. **Tarefa: Identificar ou rever objetivos de medição**
    - Descrição: Identificar ou revisar objetivos de medição derivados das necessidades e objetivos de informação da organização. Deve-se procurar ter um conjunto não muito grande de objetivos sendo tratados em um determinado momento. Desta forma, se o conjunto inicial de objetivos for grande ele precisa ser priorizado de forma a se poder selecionar os objetivos que serão efetivamente tratados. Os objetivos devem ser revistos periodicamente e atualizados, quando necessário.
    - Critérios de Entrada: Ter-se iniciado a implantação do processo de medição ou haver a necessidade de rever o plano de medição organizacional
    - Critérios de Saída: Ter-se os objetivos de medição definidos ou revisados.
2. **Tarefa: Identificar ou rever questões e medidas**
    - Descrição: Identificar ou revisar questões a serem respondidas pelas medidas para que os objetivos de medição da organização possam ser mensurados. Dessa forma, medidas devem ser especificadas para avaliar o alcance dos objetivos definidos. Devem ser estabelecidas definições operacionais para as medidas de forma a garantir que elas sejam coletadas e analisadas de forma clara, consistente e independente de quem fará a coleta e a análise. Portanto, cada medida deve conter as seguintes informações: (i) nome; (ii) descrição; (iii) Mnemônico – sigla; (iv) tipo da medida; (v) entidade da medida – o que ela mede; (vi) propriedade; (vii) unidade; (viii) tipo de escala; (ix) valor da escala; (x) intervalo esperado dos dados; (xi) fórmula de cálculo; (xii) responsável; (xiii) momento de coleta; (xiv) periodicidade de coleta. Essas questões e medidas devem ser revistas periodicamente e atualizadas, quando necessário.
    - Critérios de Entrada: Ter-se os objetivos de medição definidos ou revisados.
    - Critérios de Saída: Ter-se as questões e medidas definidas ou revisadas.
3. **Tarefa: Especificar procedimentos de coleta e armazenamento dos dados de medição**
    - Descrição: Especificar como os dados de medições serão obtidos e como e onde serão armazenados. Dessa forma, deve ser elaborado um *checklist* a ser utilizado logo após a coleta de uma medida de forma a avaliar e garantir sua integridade e confiabilidade. Também deve ser definido o local de armazenamento, que pode ser um repositório organizacional único ou não. Em ambos os casos, deve ser documentado o local, os procedimentos de inserção e os níveis de permissão de acesso aplicáveis.
    - Critérios de Entrada: Ter-se as questões e medidas definidas ou revisadas.
    - Critérios de Saída: Ter-se os procedimentos de coleta e armazenamento dos dados definidos.
4. **Tarefa: Especificar procedimentos de análise dos dados de medição**
    - Descrição: Especificar procedimentos que demonstrem como os dados coletados deverão ser analisados, representados e interpretados (se possível). Esses procedimentos devem conter que ferramentas analíticas serão utilizadas, em que momento devem

ser realizadas essas análises, sua periodicidade e quem será o responsável por executar o procedimento. Deve-se atentar para que a análise esteja explicitamente relacionada aos objetivos de medição documentados e que a apresentação dos resultados seja clara para os destinatários. Sempre que necessário, o conteúdo e formato desses relatórios devem ser revistos.

- Critérios de Entrada: Ter-se os procedimentos de coleta e armazenamento dos dados definidos.
- Critérios de Saída: Ter-se os procedimentos de análise dos dados definidos.

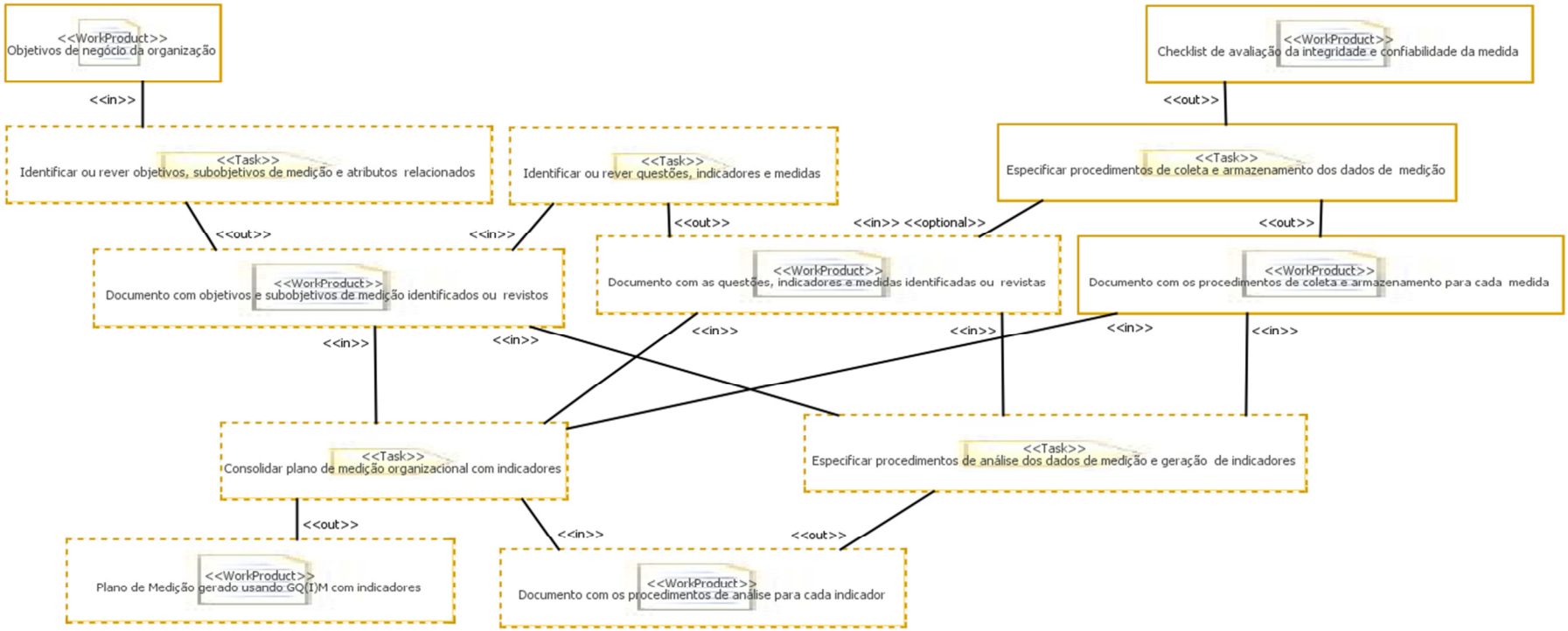
5. **Tarefa: Consolidar plano de medição organizacional**
   - Descrição: Consolidar o plano de medição organizacional através da integração, em um único documento, de todos os documentos relacionados à definição, coleta, análise e armazenamento das medidas gerados ou revisados até o momento. Esse plano deve conter, ainda, como será o processo de divulgação dos resultados de medição obtido e quem receberá essas informações.
   - Critérios de Entrada: Ter-se os objetivos de medição, questões, medidas, procedimentos de coleta, armazenamento e análise dos dados definidos ou revisados.
   - Critérios de Saída: Ter-se o plano de medição organizacional definido ou revisado

Atividade Variante: Planejar medição utilizando o método GQ(I)M

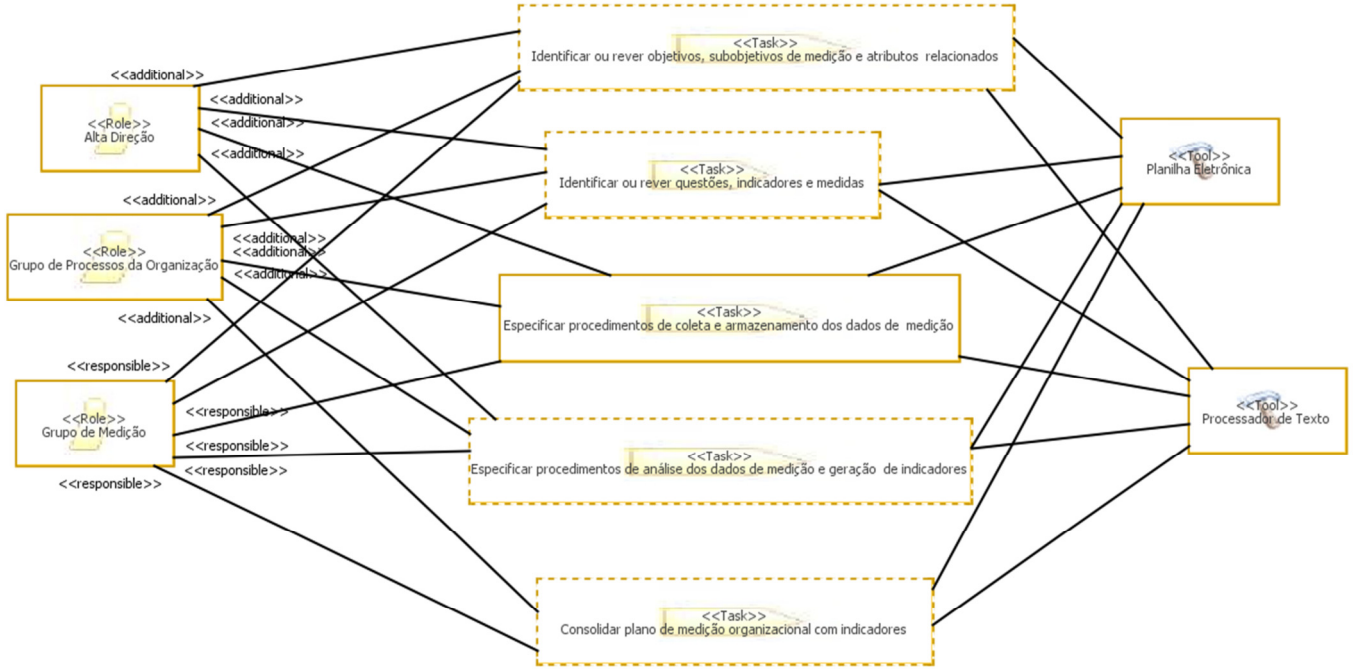
- Descrição: Estabelecer plano de medição organizacional pela definição dos objetivos de medição oriundos das necessidades de informação organizacional utilizando o método GQ(I)M (*Goal Question (Indicator) Measure*). Esse plano deve conter quais medidas deverão ser coletadas, quais os indicadores deverão ser gerados, quais os procedimentos de coleta e armazenamento deverão ser adotados para cada medida, bem como quais os procedimentos de análise e divulgação dos resultados de medição.
- Critérios de Entrada Ter-se iniciado a implantação do processo de medição ou haver a necessidade de rever o plano de medição organizacional.
- Critérios de Saída: Ter-se o plano de medição organizacional com indicadores definido ou revisado.

Visão Estrutural

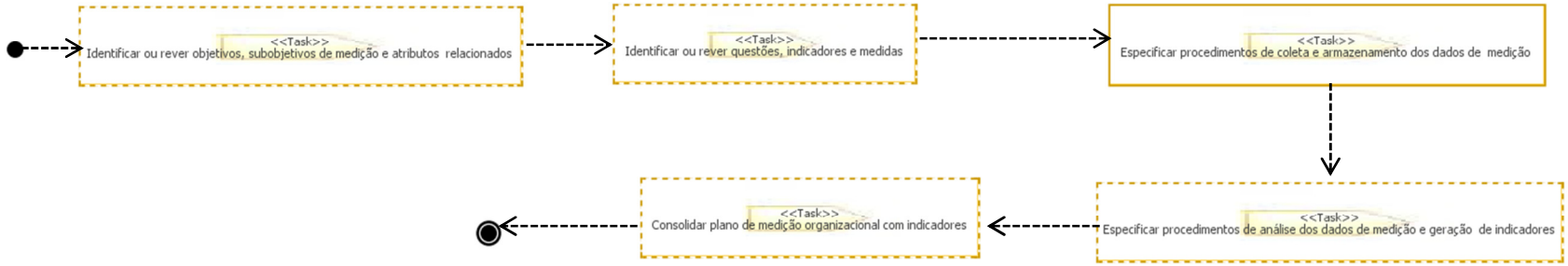
Unidades de Trabalho e Produtos de Trabalho



Unidades de Trabalho, Papéis e Ferramentas



Visão Comportamental



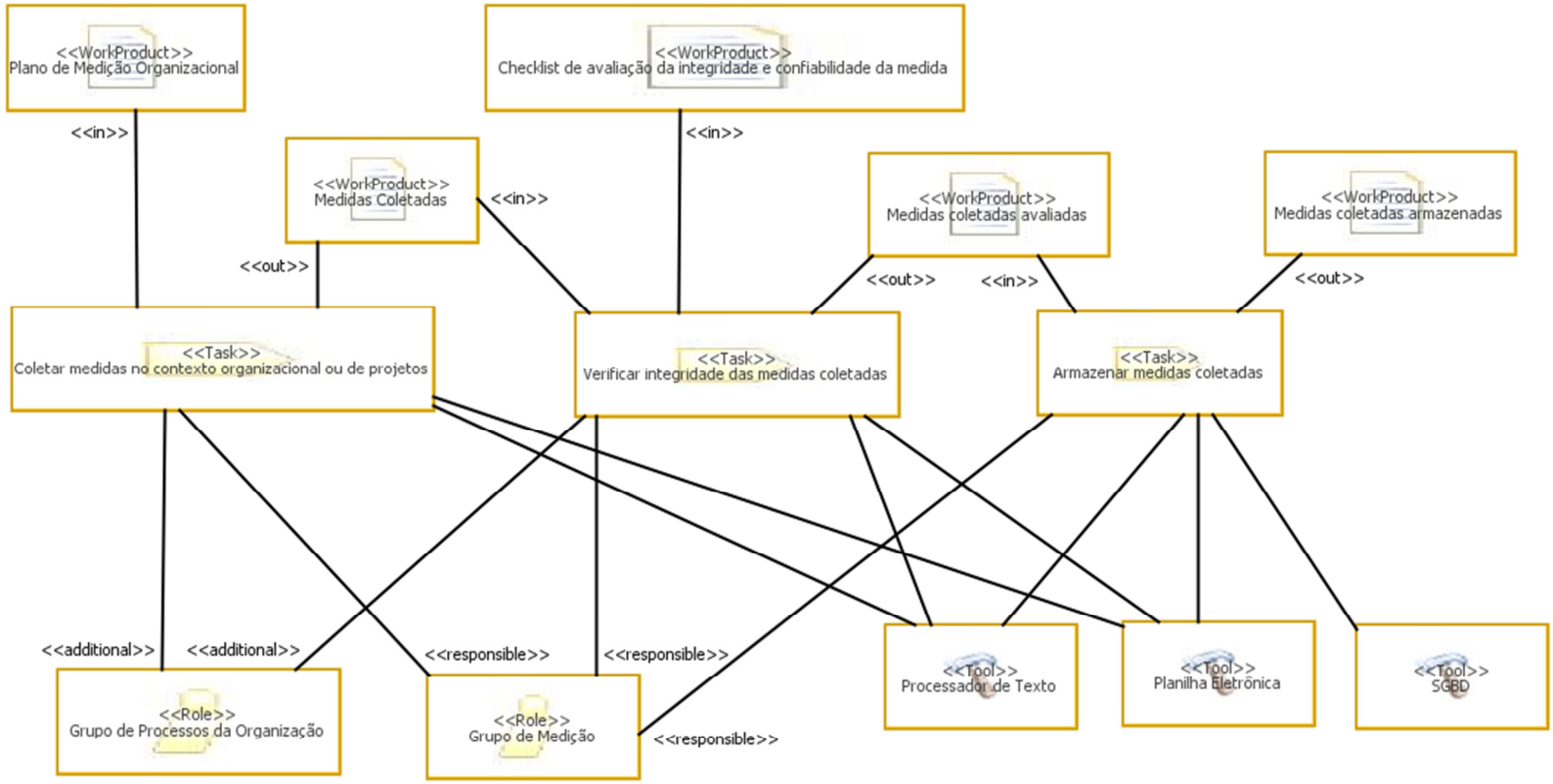
1. **Tarefa: Identificar ou rever objetivos, subobjetivos de medição e atributos relacionados**
    - Descrição: Identificar ou revisar objetivos de medição derivados das necessidades e objetivos de informação da organização. Para tal, deve-se: (i) identificar e priorizar os objetivos de negócio da organização; (ii) identificar os objetivos de conhecimento e aprendizado; (iii) identificar os subobjetivos, o que significa traduzir os objetivos de alto nível em subobjetivos relacionados às atividades; (iv) identificar entidades e atributos relacionados aos subobjetivos; (v) formalizar os objetivos de medição. Esses objetivos, subobjetivos e atributos devem ser revistos periodicamente e atualizados, quando necessário.
    - Critérios de Entrada: Ter-se iniciado a implantação do processo de medição ou haver a necessidade de rever o plano de medição organizacional
    - Critérios de Saída: Ter-se os objetivos e subobjetivos de medição definidos ou revisados
2. **Tarefa: Identificar ou rever questões, indicadores e medidas**
    - **Descrição:** Identificar ou revisar questões a serem respondidas pelas medidas para que os objetivos de medição da organização possam ser mensurados. Dessa forma, medidas e indicadores devem ser especificados para avaliar o alcance dos objetivos definidos. Para tal, deve-se: (i) identificar questões quantificáveis e indicadores relacionados que serão usados para apoiar o alcance dos objetivos; (ii) identificar os elementos de dados que serão coletados para compor os indicadores que ajudarão a responder às questões; (iii) definir as medidas que serão usadas, bem como suas definições operacionais. Portanto, cada medida deve conter as seguintes informações: (i) nome; (ii) descrição; (iii) Mnemônico – sigla; (iv) tipo da medida; (v) entidade da medida – o que ela mede; (vi) propriedade; (vii) unidade; (viii) tipo de escala; (ix) valor da escala; (x) intervalo esperado dos dados; (xi) fórmula de cálculo; (xii) responsável; (xiii) momento de coleta; (xiv) periodicidade de coleta. Essas questões, indicadores e medidas devem ser revistas periodicamente e atualizadas, quando necessário.
    - Critérios de Entrada: Ter-se os objetivos e subobjetivos de medição definidos ou revisados.
    - Critérios de Saída: Ter-se as questões, indicadores e medidas definidas ou revisadas.
3. **Tarefa: Especificar procedimentos de coleta e armazenamento dos dados de medição para indicadores**
    - Descrição: Especificar como os dados de medições serão obtidos e como e onde serão armazenados. Dessa forma, deve ser elaborado um *checklist* a ser utilizado logo após a coleta de uma medida de forma a avaliar e garantir sua integridade e confiabilidade. Também deve ser definido o local de armazenamento, que pode ser um repositório organizacional único ou não. Em ambos os casos, deve ser documentado o local, os procedimentos de inserção e os níveis de permissão de acesso aplicáveis.
    - Critérios de Entrada: Ter-se as questões, indicadores e medidas definidas ou revisadas.
    - Critérios de Saída: Ter-se os procedimentos de coleta e armazenamento dos dados definidos.

4. **Tarefa: Especificar procedimentos de análise dos dados de medição e geração de indicadores**
    - Descrição: Especificar procedimentos que demonstrem como os indicadores deverão ser gerados, analisados, representados e interpretados (se possível). Esses procedimentos devem conter que ferramentas analíticas serão utilizadas, em que momento devem ser realizadas essas análises, sua periodicidade e quem será o responsável por executar o procedimento. Deve-se atentar para que a análise esteja explicitamente relacionada aos objetivos de medição documentados e que a apresentação dos resultados seja clara para os destinatários. Sempre que necessário o conteúdo e formato desses relatórios devem ser revistos.
    - Critérios de Entrada: Ter-se os procedimentos de coleta e armazenamento dos dados definidos.
    - Critérios de Saída: Ter-se os procedimentos de análise dos indicadores definidos.
5. **Tarefa: Consolidar plano de medição organizacional com indicadores**
    - Descrição: Consolidar o plano de medição organizacional pela integração, em um único documento, de todos os documentos relacionados à definição, coleta, análise e armazenamento das medidas e indicadores gerados ou revisados até o momento. Esse plano deve conter, ainda, como será o processo de divulgação dos resultados de medição obtido e quem receberá essas informações.
    - Critérios de Entrada: Ter-se os objetivos e subobjetivos de medição, questões, indicadores, medidas, procedimentos de coleta, armazenamento e análise das medidas e dos indicadores definidos ou revisados.
    - Critérios de Saída: Ter-se o plano de medição organizacional com indicadores definido ou revisado.

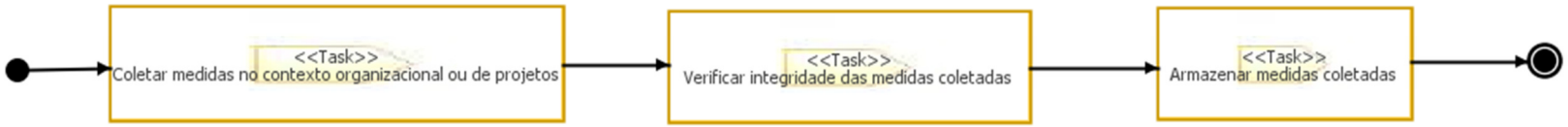
Atividade: Coletar e armazenar medidas

- Descrição: Coletar, avaliar e armazenar as medidas de acordo com a periodicidade e procedimentos constantes no plano de medição da organização.
- Critérios de Entrada: Ter-se o momento de coletar as medidas.
- Critérios de Saída: Ter-se as medidas coletas, avaliadas e armazenadas.

Visão Estrutural



Visão Comportamental

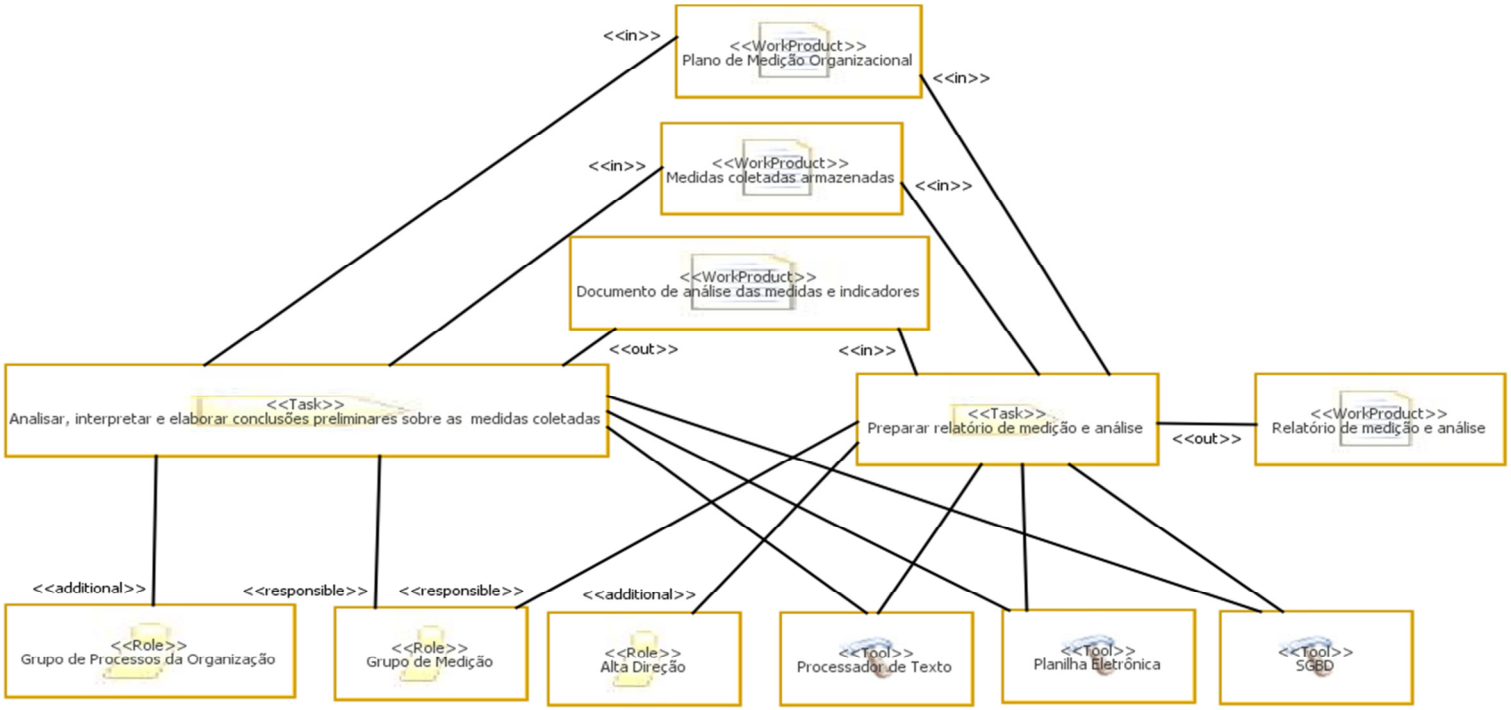


1. **Tarefa: Coletar medidas no contexto organizacional ou de projetos**
   - Descrição: Realizar a coleta das medidas de acordo com os procedimentos estabelecidos no plano de medição da organização.
   - Critérios de Entrada: Ter-se o momento de coletar as medidas
   - Critérios de Saída: Ter-se as medidas coletadas.
2. **Tarefa: Verificar integridade das medidas coletadas**
   - Descrição: Avaliar a integridade e confiabilidade de uma medida por meio da utilização de um *checklist* específico. Essa avaliação deve acontecer logo após a coleta de uma medida.
   - Critérios de Entrada: Ter-se as medidas coletadas.
   - Critérios de Saída: Ter-se as medidas coletadas avaliadas.
3. **Tarefa: Armazenar medidas coletadas**
   - Descrição: Realizar o armazenamento das medidas coletadas de acordo com o estabelecido no plano de medição da organização, incluindo suas informações de contexto.
   - Critérios de Entrada: Ter-se as medidas coletadas avaliadas.
   - Critérios de Saída: Ter-se as medidas coletadas armazenadas

Atividade: Analisar medidas coletadas

- Descrição: Realizar a análise das medidas e indicadores (se for o caso) de acordo com a periodicidade e procedimentos contidos no plano de medição organizacional. Essas análises devem apresentar informações que possibilitem a tomada de decisão por seus usuários. Uma boa análise deve utilizar gráficos adequados para a visualização dos valores das medidas; utilizar a ordenação adequada para demonstrar os dados (como por exemplo, utilizar a ordem cronológica para demonstrar dados históricos); relacionar os objetivos de medição com os valores obtidos; incluir as informações de contexto relacionadas às medidas e produzir pareceres e conclusões levando em consideração essas informações de contexto. Se necessário, novas coletas e análises podem ser realizadas com o objetivo de avaliar a consistência e validade das informações encontradas. Ao final, um relatório de medição e análise deve ser elaborado para posterior divulgação.
- Critérios de Entrada: Ter-se as medidas coletas, avaliadas e armazenadas..
- Critérios de Saída: Ter-se o relatório de medição e análise.

Visão Estrutural

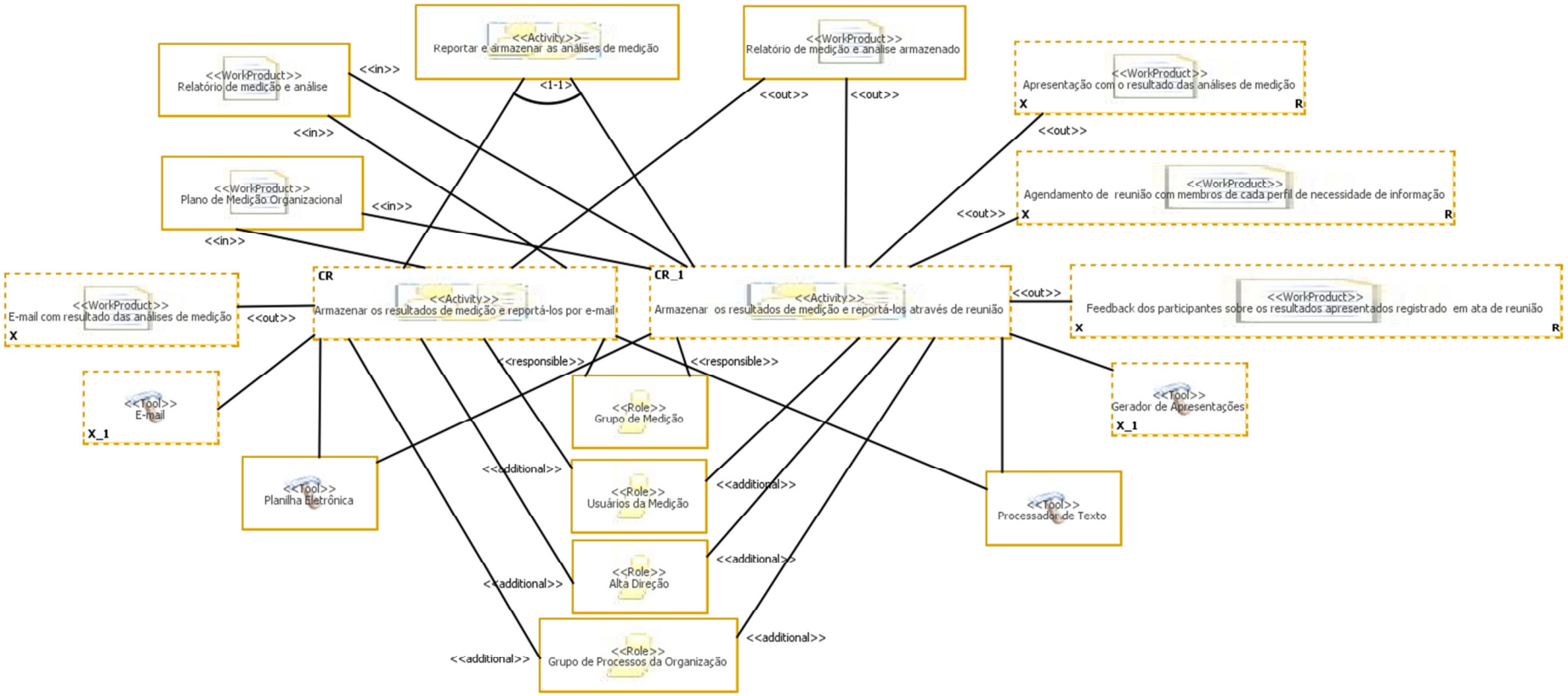


Visão Comportamental

1. **Tarefa: Analisar, interpretar e elaborar conclusões preliminares sobre** as medidas coletadas
   - Descrição: Realizar a análise das medidas e indicadores (se for o caso) de acordo com a periodicidade e procedimentos contidos no plano de medição organizacional. Essas análises devem apresentar informações que possibilitem a tomada de decisão por seus usuários. Uma boa análise deve utilizar gráficos adequados para a visualização dos valores das medidas; utilizar a ordenação adequada para demonstrar os dados (como por exemplo, utilizar a ordem cronológica para demonstrar dados históricos); relacionar os objetivos de medição com os valores obtidos; incluir as informações de contexto relacionadas às medidas e produzir pareceres e conclusões preliminares levando em consideração essas informações de contexto.
   - Critérios de Entrada: Ter-se as medidas coletas, avaliadas e armazenadas.
   - Critérios de Saída: Ter-se medidas e indicadores analisados e interpretados preliminarmente
2. **Tarefa: Preparar relatório de medição e análise**
   - Descrição: Preparar o relatório de medição e análise seguindo os procedimentos de análise constantes no plano de medição organizacional. Os resultados preliminares devem ser revistos junto aos envolvidos mais relevantes e ajustes na analise devem ser realizados, se pertinente, antes que o relatório com os resultados de medição seja divulgado aos demais interessados da organização.
   - Critérios de Entrada: Ter-se medidas e indicadores analisados e interpretados preliminarmente.
   - Critérios de Saída: Ter-se o relatório de medição e análise preparado e pré-avaliado

Atividade: Reportar e armazenar as análises de medição

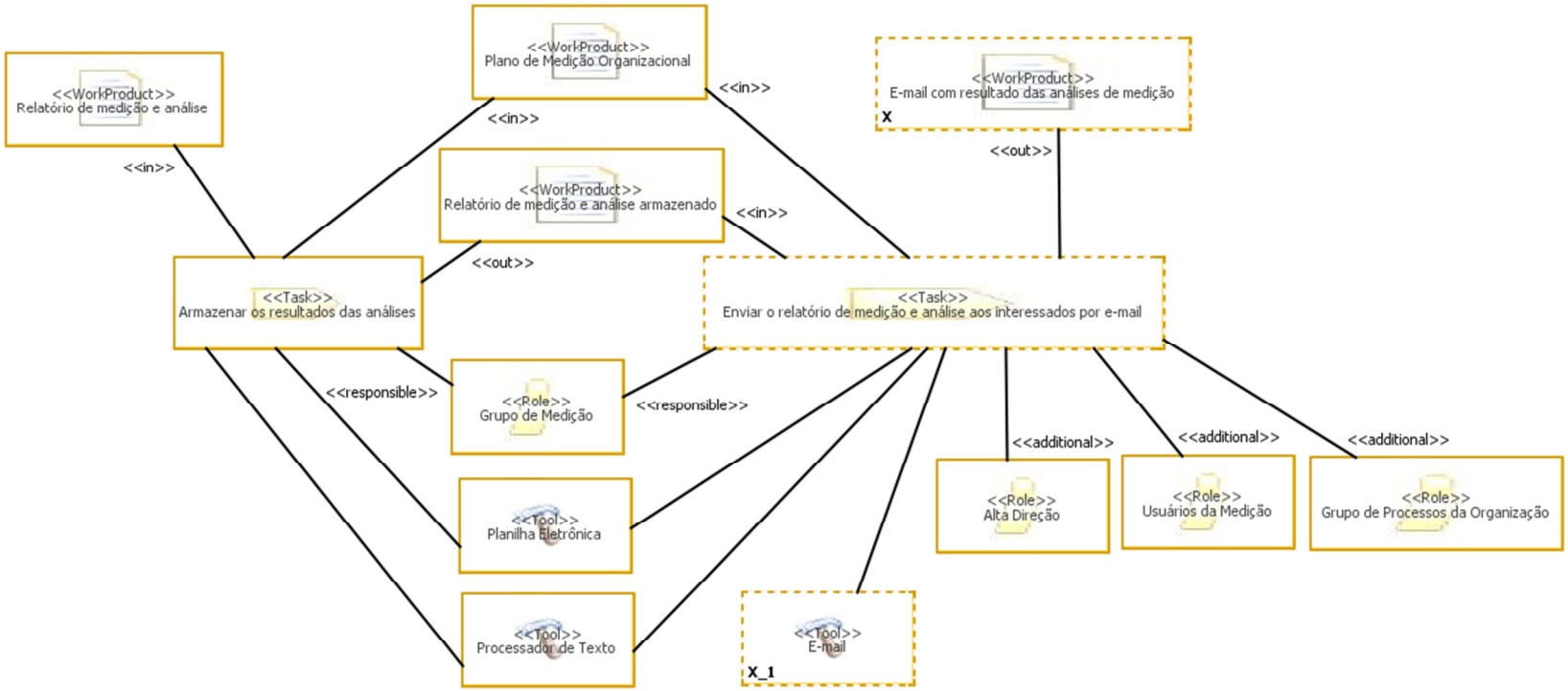
- Descrição: Armazenar os resultados das análises dos dados de forma a permitir seu uso futuro, o que deve ser feito de acordo com os procedimentos previamente especificados no plano de medição organizacional. Esse componente envolve, ainda, tornar esse conteúdo armazenado disponível para os demais membros da organização interessados nos resultados de medição, de acordo com as necessidades de informação e perfil de cada um. Para tal, esses resultados podem ser divulgados de forma eletrônica (através de e-mail) ou presencial (através de reuniões).
- Variantes:
  - Armazenar os resultados de medição e reportá-los por e-mail;
  - Armazenar os resultados de medição e reportá-los através de reunião.



Atividade Variante: Armazenar os resultados das análises e reportá-las por e-mail

- Descrição: Armazenar os resultados das análises dos dados de forma a permitir seu uso futuro, o que deve ser feito de acordo com os procedimentos previamente especificados no plano de medição organizacional. Esse componente envolve, ainda, tornar esse conteúdo armazenado disponível para os demais membros da organização interessados nos resultados de medição. Dessa forma, deve ser confeccionado um e-mail, contendo tais informações, para cada interessado, levando em consideração seu perfil e necessidade de informação.
- Critérios de Entrada: Ter-se o relatório de medição e análise.
- Critérios de Saída: Ter-se o resultado das análises armazenado e reportado por e-mail

Visão Estrutural



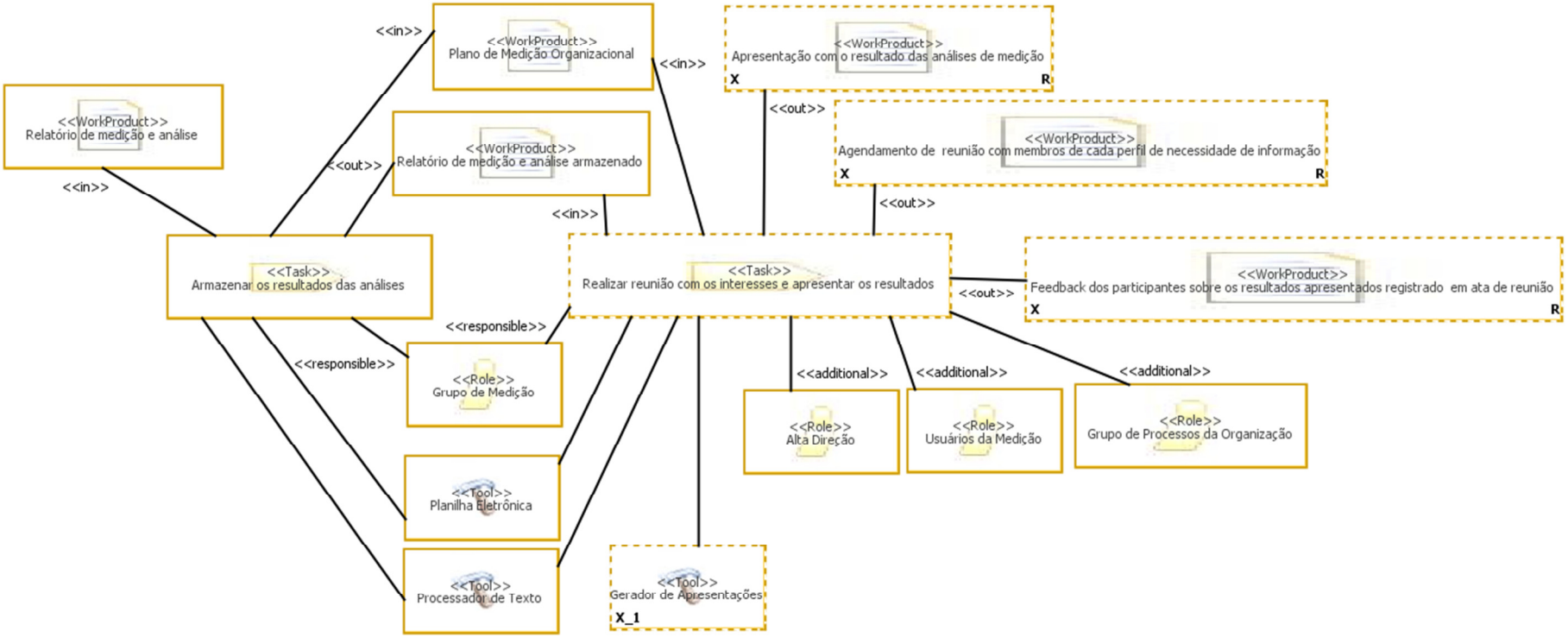
Visão Comportamental

1. **Tarefa: Armazenar os resultados das análises**
    - Descrição: Armazenar os resultados das análises dos dados de forma a permitir seu uso futuro, o que deve ser feito de acordo com os procedimentos previamente especificados no plano de medição organizacional. Dessa forma, todas as informações referentes às análises devem ser armazenadas, incluindo as informações de contexto relacionadas às medidas e as premissas e suposições adotadas na conclusão dos resultados.
    - Critérios de Entrada: Ter-se o relatório de medição e análise.
    - Critérios de Saída: Ter-se o resultado das análises armazenado.
2. **Tarefa: Enviar o relatório de medição e análise aos interessados por e-mail**
    - Descrição: Tornar o resultado das análises, feitas com os dados de medição, disponível para os demais interessados da organização. Dessa forma, deve ser confeccionado um e-mail contendo uma seleção desses resultados e análises, de acordo com o perfil e necessidade de informação de cada interessado.
    - Critérios de Entrada: Ter-se o resultado das análises armazenado.
    - Critérios de Saída: Ter-se o resultado das análises reportado por e-mail a cada interessado.

Atividade Variante: Armazenar os resultados de medição e reportá-los através de reunião

- Descrição: Armazenar os resultados das análises dos dados de forma a permitir seu uso futuro, o que deve ser feito de acordo com os procedimentos previamente especificados no plano de medição organizacional. Esse componente envolve, ainda, divulgar e tornar disponível esse conteúdo aos demais membros da organização interessados nos resultados de medição. Dessa forma, devem ser preparadas apresentações contendo somente os resultados das análises que sejam do interesse de cada publico alvo. Essas apresentações devem ser demonstradas em reuniões presenciais com membros de cada perfil de necessidade de informação, onde o *feedback* dos participantes sobre os resultados apresentados deve ser registrado.
- Critérios de Entrada: Ter-se o relatório de medição e análise.
- Critérios de Saída: Ter-se o resultado das análises armazenado e reportado através de reuniões presenciais.

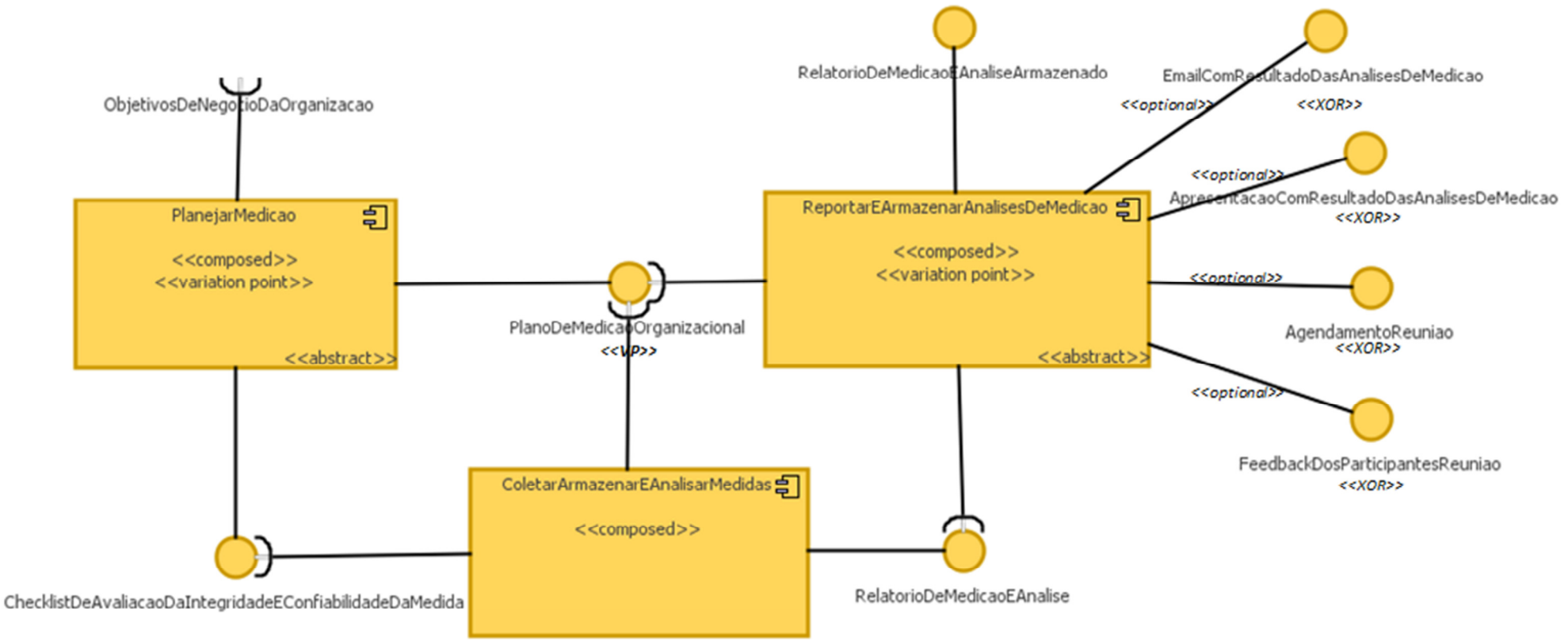
Visão Estrutural

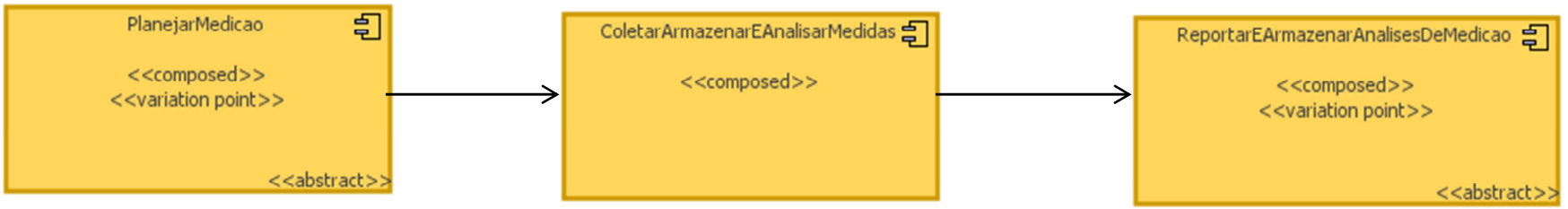


Visão Comportamental

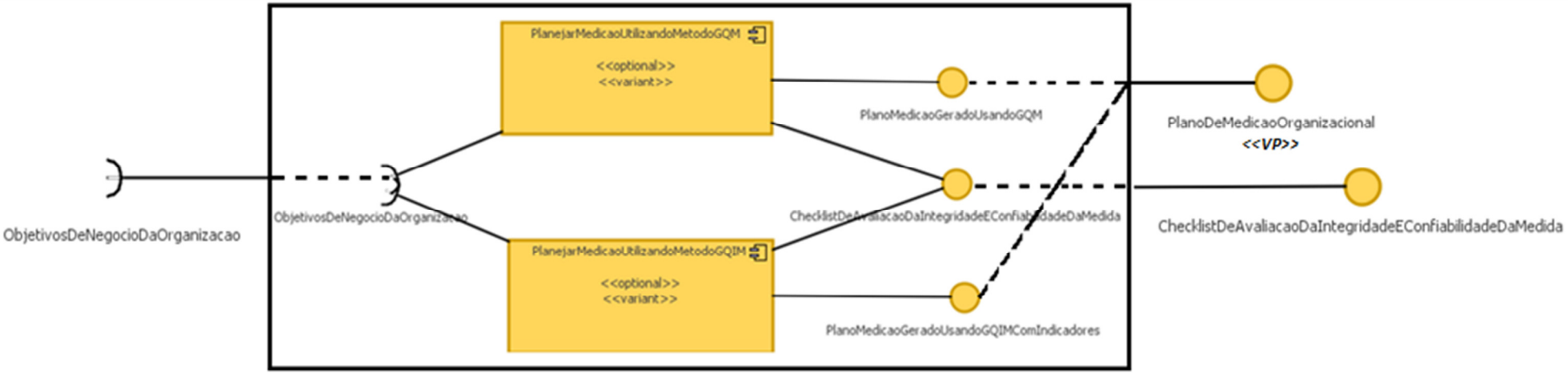
1. **Tarefa: Armazenar os resultados das análises**
   - Descrição: Armazenar os resultados das análises dos dados de forma a permitir seu uso futuro, o que deve ser feito de acordo com os procedimentos previamente especificados no plano de medição organizacional. Dessa forma, todas as informações referentes às análises devem ser armazenadas, incluindo as informações de contexto relacionadas às medidas e as premissas e suposições adotadas na conclusão dos resultados.
   - Critérios de Entrada: Ter-se o relatório de medição e análise.
   - Critérios de Saída: Ter-se o resultado das análises armazenado.
2. **Tarefa: Realizar reunião com os interessados e apresentar os resultados**
   - Descrição: Tornar o resultado das análises, feitas com os dados de medição, disponível para os demais interessados da organização. Dessa forma, devem ser preparadas apresentações contendo somente os resultados das análises que sejam do interesse de cada publico alvo. Essas apresentações devem ser demonstradas em reuniões presenciais com membros de cada perfil de necessidade de informação, onde o *feedback* dos participantes sobre os resultados apresentados deve ser registrado.
   - Critérios de Entrada: Ter-se o resultado das análises armazenado.
   - Critérios de Saída: Ter-se o resultado das análises armazenado e reportado através de reuniões presenciais.

Nível de Projeto - Modelo de Componentes



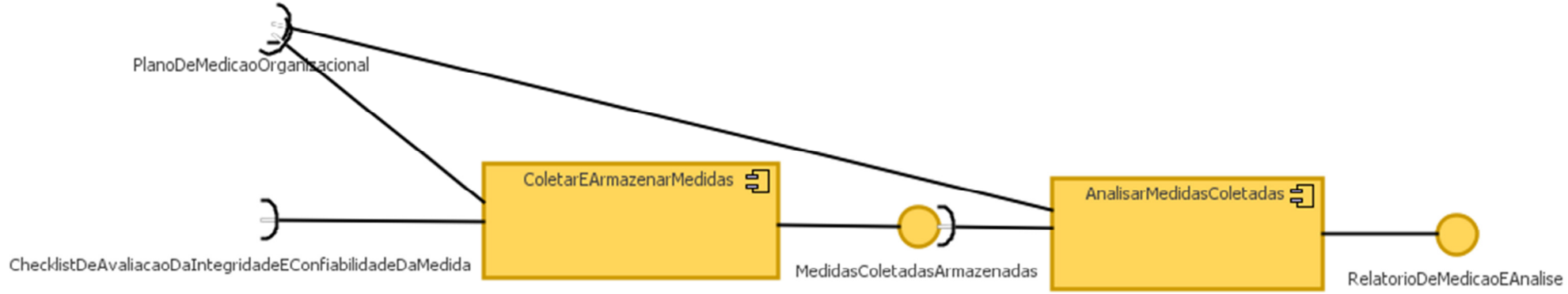


Componente Planejar Medição



- Medidas para cada componente interno:
  - **ESF_PLA_MED** – Esforço para realizar o planejamento do programa de medição da organização

Componente ColetarArmazenarEAnalisarMedidas

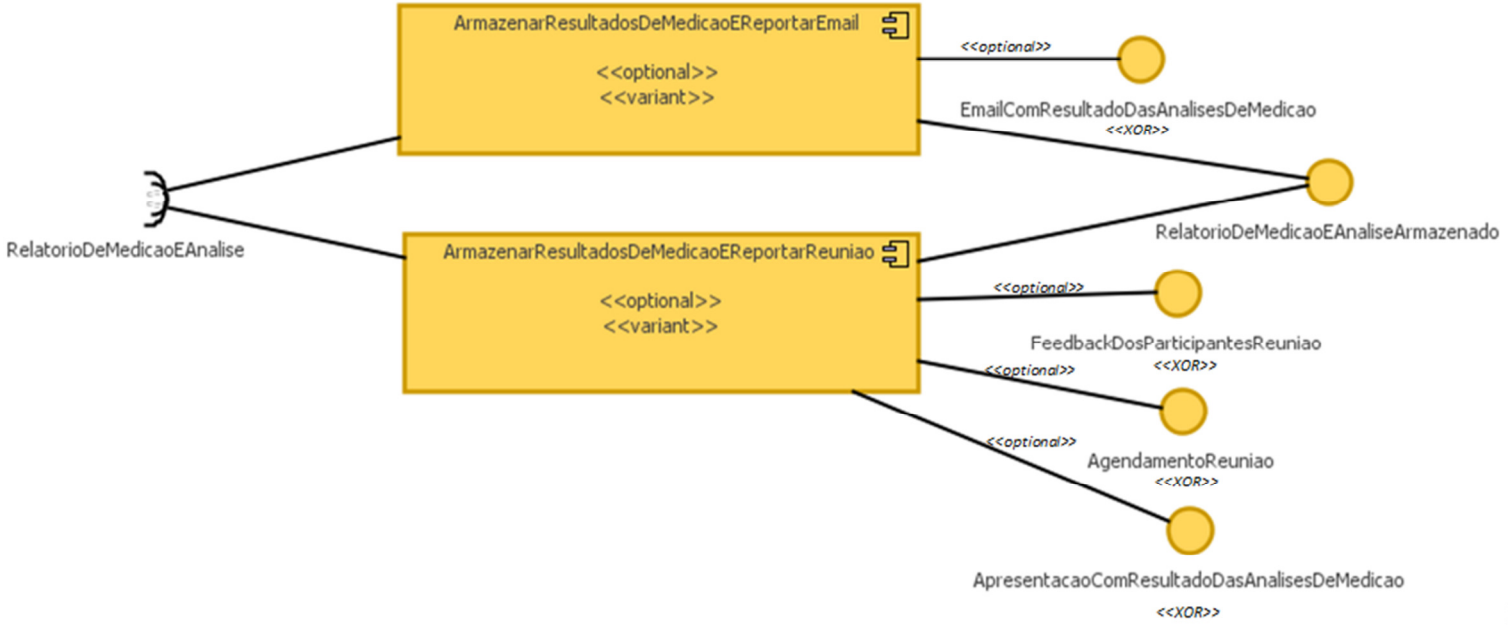


- Medidas Componente ColetarEArmazenarMedidas:
  - **TX_MED_COL –** Taxa de medidas efetivamente coletadas (Número de medidas coletadas / Número total de medidas presentes no plano de medição)

- **ESF_COL_MED –** Número de horas gastas pela equipe para realizar as atividades relacionadas à coleta das medidas que compõem o plano de medição

- **Medidas Componente AnalisarMedidasColetadas:**
  - **TX_MED_ANA –** Taxa de medidas efetivamente analisadas (Número de medidas analisadas / Número total de medidas presentes no plano de medição)
  - **ESF_ANA_MED –** Número de horas gastas pela equipe para realizar as atividades relacionadas à análise das medidas que compõem o plano de medição
  - **TX_MED_ACT –** Taxa de medidas que estão dentro das metas aceitáveis (Número de medidas dentro das metas aceitáveis / Número de medidas coletadas e analisadas)

ReportarEArmazenarAnalisesDeMedicao



Medidas para cada componente interno: **ESF_ARQ_RPT_MED –** Número de horas gastas pela equipe para realizar as atividades relacionadas ao arquivamento e divulgação das análises e resultados de medição

Apêndice V – Estudo de Viabilidade das Heurísticas de Mapeamento

Este documento apresenta o conjunto de informações fornecidas aos participantes para realização do estudo de viabilidade das heurísticas de mapeamento propostas nesta tese.

Heurísticas de Mapeamento de Modelo de Características para Modelo de Componentes de uma LPrS

ID	Elemento do Modelo de Características	Elemento do Modelo de Componentes	Heurísticas
Atividade			
HA1	Atividade	Componente de Processo de Software	Cada atividade deve ser mapeada para um componente de processo de software com a mesma classificação de opcionalidade e variabilidade da atividade. Estereótipos de opcionalidade (<<*optional*>>) e variabilidade (<<*variation point*>> e <<*variant*>>) devem ser utilizados.
HA2	Atividades relacionadas por relacionamento Alternativo	Componente de Processo de Software Composto	Atividades relacionadas por relacionamento alternativo devem ser agrupadas em um componente classificado como composto que representa a atividade classificada como ponto de variação. Os componentes internos representam cada uma das atividades que são as variantes do ponto de variação.
HA3	Atividades relacionadas por composição ou agregação	Componente de Processo de Software Composto	Agregação e Composição são tipos de associação que especificam uma relação todo/parte. Atividades relacionadas por composição ou agregação devem ser agrupadas em um componente composto que representa a atividade classificada como o todo em uma associação. Componentes internos representam cada uma das atividades que são as partes da relação.
Tarefa			
HT1	Tarefa	Elemento interno de um componente de processo de software	Tarefas representam partes da realização de uma atividade. Desta forma, cada tarefa deve ser mapeada, mantendo a mesma classificação de opcionalidade e variabilidade do modelo de característica, como elemento interno do

ID	Elemento do Modelo de Características	Elemento do Modelo de Componentes	Heurísticas
			componente que representa a atividade que a tarefa está associada.
HT2	Tarefas relacionadas por relacionamento alternativo	Elementos internos de um mesmo componente de processo de software	Tarefas relacionadas por relacionamento alternativo devem ser agrupadas em um mesmo componente.
Produto de Trabalho e relações de entrada e saída (<<in>> /<<out>>)			
HWP1	Produto de Trabalho	Elemento interno de um componente de processo de software ou interface de um componente de processo de software	Todo produto de trabalho será mapeado com a mesma classificação de variabilidade e opcionalidade do modelo de característica. Cada produto de trabalho será um elemento interno ou o produto de trabalho associado a cada interface que será gerada por relação deste produto de trabalho com cada unidade de trabalho mapeada nos componentes - Ver heurísticas HWP1.1 e HWP1.2
HWP1.1	Produto de Trabalho e relação de entrada ou saída de uma unidade de trabalho	Elemento interno de um componente de processo de software	Produtos de trabalho internos a um componente de processo de software (trocados apenas entre as unidades de trabalho que compõem o componente, não ultrapassando a fronteira do componente) devem ser representados como elementos internos do componente mantendo a classificação de opcionalidade e variabilidade do modelo de características.
HWP1.2	Relações de entrada ou saída de uma unidade de trabalho	Interface de dados de um componente de processo de software	Relações de produtos de trabalho externos a um componente de processo (produtos trocados além das fronteiras de um componente, ou seja, entre as unidades de trabalho que compõem diferentes componentes ou que são trocados com o ambiente/contexto de aplicação do processo) devem ser descritas como interfaces do componente mantendo a classificação de opcionalidade da relação no modelo de características. Para cada interface deve ser mantida uma descrição do produto de trabalho associado.
HWP2	Relação de entrada de um produto de	Interface de dados requerida de um componente de	Cada relação de um produto de trabalho externo classificada como uma entrada ao componente deve ser mapeada como uma interface requerida do

ID	Elemento do Modelo de Características	Elemento do Modelo de Componentes	Heurísticas
	trabalho em uma unidade de trabalho	processo de software	componente de processo.
HWP3	Relação de saída de um produto de trabalho em uma unidade de trabalho	Interface de dados provida de um componente de processo de software	Cada relação de um produto de trabalho externo classificada como uma saída do componente deve ser mapeada como uma interface provida do componente de processo.
HWP4	Relações de entrada ou saída de uma atividade que é um componente interno de um componente composto	Interface de dados de um componente interno mapeada como uma interface de dados do componente composto de processo de software	Cada interface de dados de um componente interno a um componente composto deve ser também mapeada como uma interface de dados do componente composto. Tais interfaces devem estar conectadas por uma relação de delegação. A opcionalidade da interface do componente externo (composto) deve ser observada com relação a sua participação no domínio e não somente com relação aos componentes internos. Desta forma, interfaces relacionadas a componentes internos, ainda que mandatórias internamente, podem ser mapeadas como opcionais para o componente externo. Por exemplo, produtos de trabalho associados como saídas de variantes excludentes entre si serão opcionais no domínio e ainda que a relação com a variante seja mandatória (interface provida mandatória do componente interno), as relações serão classificada como interfaces providas opcionais do componente externo.
Papel			
HR1	**Papel**	**Elemento interno de um componente de processo de software**	Cada papel deve ser definido como elemento interno de cada componente de processo de software que contém outro elemento de processo ao qual o papel se relaciona. Deve seguir a mesma classificação de opcionalidade e variabilidade do elemento correspondente no nível de características da LPrS. Relações envolvendo esse item devem ser descritas dentro do componente relacionado.
Ferramenta			
HTl1	**Ferramenta**	**Elemento interno de um componente**	Cada ferramenta deve ser definida como um elemento interno de cada componente de processo

ID	Elemento do Modelo de Características	Elemento do Modelo de Componentes	Heurísticas
		de processo de software	de software que contém outro elemento de processo ao qual a ferramenta se relaciona. Deve seguir a mesma classificação de opcionalidade e variabilidade do elemento correspondente no nível de características da LPrS. Relações envolvendo esse item devem ser descritas dentro do componente relacionado.

Classificação da Variação Interna de Componentes de processos de software (abstrato ou concreto)		
ID	**Heurísticas para análise do componente**	**Classificação**
IV1	Componente com nenhuma variação interna a ser resolvida, ou seja, todos os elementos de processo que compõem o componente estão definidos da maneira que serão instanciados, sem nenhuma decisão adicional de definição.	Componente de processo de software concreto
IV2	Componente com configuração interna a ser resolvida, ou seja, pelo menos um dos elementos de processo que o compõe requer alguma decisão para definição de um processo derivado. Por exemplo, um dos seus elementos internos é classificado como ponto de variação ou variante; ou existe algum tipo de relação opcional dentro do componente, por exemplo, unidades de trabalho relacionadas por agregação com outras unidades de trabalho opcionais ou relações opcionais com papéis, produtos de trabalho e ferramentas.	Componente de processo de software abstrato

Fluxo de Controle			
ID	**Elemento do Modelo de Características**	**Elemento do Modelo de Componentes**	**Heurísticas**
CF1	Fluxo de Controle	Interface de controle de um componente de processo de software	Cada fluxo de controle estabelecido entre unidades de trabalho mapeadas em diferentes componentes devem ser descritos como interfaces de controle. Uma porta deverá ser definida como entrada para fluxo de controle e outra porta deverá ser definida como saída para o fluxo de controle. Tais portas deverão ser conectadas por conectores descrevendo o tipo do fluxo: sequencial; paralelismo; síncrono; junção e decisão.

Regras de Composição			
ID	**Elemento do Modelo de Características**	**Elemento do Modelo de Componentes**	**Heurísticas**
IR1	Regra de Composição Inclusiva	Pré-condições de um componente de processo	Pré-condição deve ser especificada nos componentes que compõem o antecedente da regra de composição inclusiva. A descrição deve definir a dependência com os componentes e /ou interfaces que compõem o consequente da regra.
ER1	Regra de Composição Exclusiva	Restrições de um componente de processo	Restrição deve ser definida para cada componente que compõe a regra. Cada componente ou interface devem ser marcados com um estereótipo <<XOR>>.

FORMULÁRIO DE PREENCHIMENTO DO MAPEAMENTO A SER REALIZADO UTILIZANDO AS HEURÍSTICAS DE MAPEAMENTO

FM	Component Model		
ID Elemento [18]	**Categoria do elemento no modelo de componente** [19]	**Classificações dependendo da categoria do elemento** [20]	**Raciocínio de Mapeamento** [21]
	☐ Componente	☐ Invariante ☐ Ponto de Variação ☐ Variante ☐ Mandatório ☐ Opcional ☐ Simples ☐ Composto [22] ☐ Concreto ☐ Abstrato [23]	
	☐ Elemento Interno	☐ Invariante ☐ Ponto de Variação ☐ Variante ☐ Mandatório ☐ Opcional Do componente [24]:	
	☐ Interface	☐ Dados ☐ Requerida ☐ Provida ☐ Controle ☐ Sequencial ☐ Paralelismo ☐ Síncrono ☐ Junção ☐ Decisão Classificação de Opcionalidade ☐ Mandatório ☐ Opcional Do componente [7]:	
	☐ Pré-condição ☐ Restrição	Do(s) componente(s) [7]:	

[18] Referenciar o elemento do modelo de característica a ser mapeado (colocar ID ou explicitar a relação)
[19] Definir **Categoria**: Tipo do elemento no modelo de componente (componente/elemento interno/interface/restrições)
[20] Definir classificação do elemento no modelo de componentes.
[21] Raciocínio de Mapeamento: Apontar as heurísticas utilizadas e explicar em alto nível o que foi usado para mapeamento.
[22] Para componentes compostos identificar quais são os componentes internos
[23] Especificar o que motivou a classificação como abstrato (elemento ou relação do modelo de característica)
[24] Não esquecer de identificar o(s) componente(s) ao(s) qual(is) o elemento está relacionado

FORMULÁRIO DE AVALIAÇÃO DAS HEURÍSTICAS DE MAPEAMENTO E CRITÉRIOS DE AGRUPAMENTO DE COMPONENTES DE PROCESSOS DE SOFTWARE DE LPRS

Nome do Participante:

Avaliação Geral

1. A notação *OdysseyProcess-FEX* é de fácil entendimento e o material enviado foi suficientemente claro e permitiu a absorção de conhecimento necessário para entendimento da notação utilizada na modelagem do domínio. Justifique.

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

---

2. Não houve dificuldade para entender o domínio utilizado. Em caso negativo, ou parcialmente negativo, indique o que poderia ser apontado como complexidade no entendimento do domínio. Justifique.

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

---

3. O tempo para execução do estudo foi suficiente. Em caso negativo, ou parcialmente negativo, a falta de tempo está associada à complexidade: (1) das tarefas, (2) do domínio escolhido, (3) do entendimento dos formulários, (4) das heurísticas aplicadas, (5) outro? Justifique.

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

Avaliação Geral das Heurísticas

1. A ordem de apresentação das heurísticas foi apropriada para aplicação. Ou seja, a ordem de aplicação das heurísticas de mapeamento foi a mesma da apresentação das heurísticas no documento. Em caso negativo, ou parcialmente negativo, em que ordem as heurísticas foram aplicadas e o que pode facilitar? Justifique.

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

---

2. Alguma informação adicional poderia ser usada como suporte para aplicação das heurísticas. Em caso afirmativo ou parcialmente afirmativo, qual(is)? Justifique.

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

---

3. Foram identificadas heurísticas redundantes ou similares. Em caso afirmativo, ou parcialmente afirmativo, indique quais heurísticas poderiam ser agrupadas ou quais heurísticas poderiam ser removidas. Justifique.

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

---

4. Em algum momento você sentiu dificuldade na aplicação de uma ou mais heurísticas. Em caso afirmativo ou parcialmente afirmativo, qual(is)? Justifique indicando a dificuldade enfrentada e uma possível sugestão de melhoria.

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

---

5. Em algum momento você desejou mapear uma característica para um determinado elemento no modelo de componentes e esta forma de mapeamento não era sugerida dentro das heurísticas disponibilizadas. Em caso afirmativo ou parcialmente afirmativo, qual(is)? Justifique indicando o mapeamento desejado.

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

---

6. As decisões sobre classificação de opcionalidade (mandatório e opcional) e classificação de variabilidade (invariante, ponto de variação e variante) sugeridas pelas heurísticas são adequadas. Em caso negativo, ou parcialmente negativo,

aponte onde foi identificado o problema e qual heurística seria mais adequada. Justifique.

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

---

7. As heurísticas de agrupamento de características de unidades de trabalho relacionadas por relacionamentos alternativos, composição ou agregação, em um componente de processo de software composto são adequadas. Em caso negativo, ou parcialmente negativo, aponte qual heurística seria mais adequada. Justifique.

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

Suporte Computacional

1. Você acredita ser viável realizar o mapeamento manual de modelos de diferentes níveis de abstração de LPrS mais extensos e complexos. Em caso negativo, ou parcialmente negativo, aponte como o uso de um suporte computacional poderia auxiliar adequadamente a aplicação das heurísticas propostas? Justifique.

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

---

2. O uso de um suporte computacional para realizar o mapeamento de modelos de uma LPrS através da aplicação direta das heurísticas propostas sem consulta ao engenheiro do domínio seria considerado invasivo. Em caso afirmativo, ou parcialmente afirmativo, que ação poderia ser considerada mais apropriada? Justifique.

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

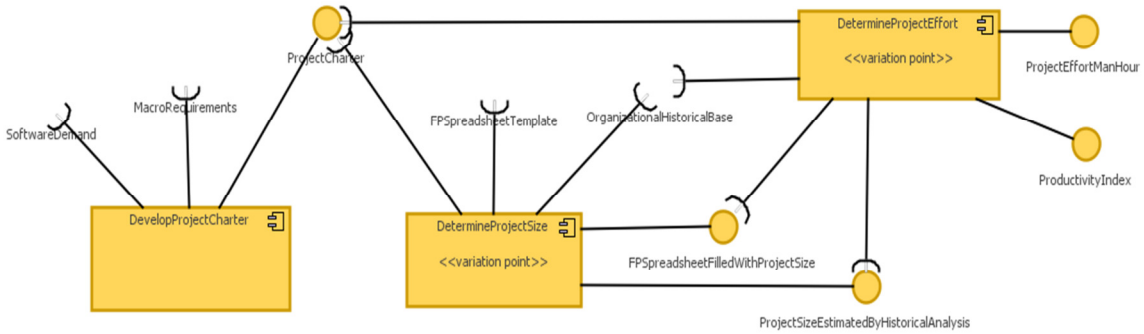
□ Concordo parcialmente □ Concordo totalmente

Obrigada pela colaboração!

Apêndice VI – Estudo de Viabilidade dos Critérios de Agrupamento de Componentes de Processos de Software

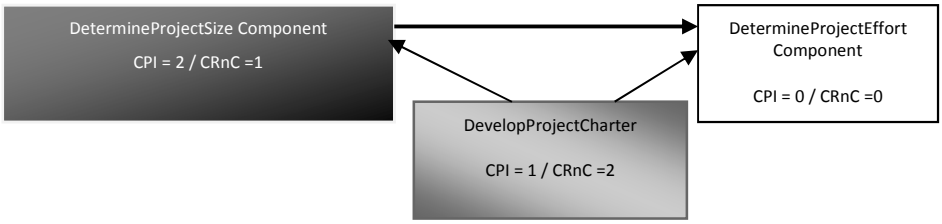
Este documento apresenta o conjunto de informações fornecidas aos participantes para realização do estudo de viabilidade dos Critérios de Agrupamento de Componentes de Processos de Software propostos nesta tese.

Modelos usados no estudo com as três perspectivas

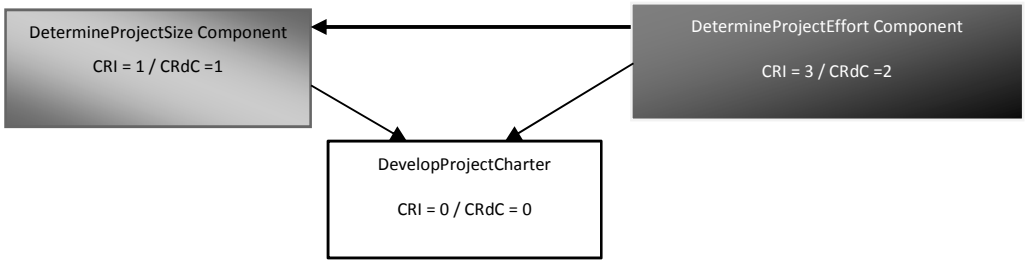


Componente	Valores dos critérios				
	CPI	CRnC	CRI	CRdC	I
DevelopProjectCharter	1	2	0	0	0
DetermineProjectSize	2	1	1	1	0,33
DetermineProjectEffort	0	0	3	2	1
Reference Value	1	1	1,33	1	-

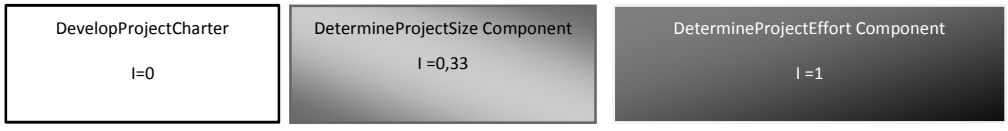
Visão de Responsabilidades



Visão de Dependências



Instabilidade



Critérios de Agrupamento de Componentes de Processo de Software de uma LPrS

Critério para Análise de Agrupamento de Componentes por Escopos do Domínio definidos **Análise de relações entre componentes de um mesmo escopo, descrevendo conceitos comuns e tópicos relacionados.**		
ID	**Critério**	**Descrição do critério**
C1	**Agrupamento de componentes relacionados a um mesmo escopo**	É sugerido o agrupamento de componentes mapeados de características que pertencem a um mesmo escopo dentro do domínio, baseado na suposição que esses componentes trocam mais informação entre si.

Critério para Análise de Agrupamento Componentes com relações entre si via interfaces de dados ou controle **Análise do grau de responsabilidade e dependência de cada componente. Cinco critérios podem ser aplicados com o objetivo de minimizar o acoplamento e aumentar a coesão de componentes relacionados.**		
ID	**Critério**	**Descrição do critério**
C2	**Análise de componentes relacionados por interfaces providas**	Análise do grau de responsabilidade. *Component Provided Interfaces* (CPI) - Cálculo do número de interfaces providas de cada componente. (Obs: considerar apenas interfaces trocadas entre componentes e descartar interfaces que trocam produtos de trabalho apenas com o ambiente) Valores de referência: CPI mínimo no domínio; CPI máximo no domínio; CPI médio no domínio. É sugerido a análise da possibilidade de agrupamento de um componente com aqueles que o requerem quando o valor CPI do componente for maior que o CPI médio. É necessário analisar se o componente possui um alto grau de responsabilidade que justifique o alto número de interfaces providas, fornecendo artefatos para diferentes componentes no domínio. Neste caso, o agrupamento não é aconselhável.
C3	**Análise de componentes relacionados por interfaces requeridas**	Análise do grau de dependência. *Component Required Interfaces* (CRI) - Cálculo do número de interfaces requeridas de cada componente. (Obs: considerar apenas interfaces trocadas entre componentes e descartar interfaces que trocam produtos de trabalho apenas com o ambiente) Valores de referência: CRI mínimo no domínio; CRI máximo no domínio; CRI médio no domínio. É sugerido a análise da possibilidade de agrupamento de um componente com aqueles que o requerem quando o valor CRI do componente for maior que o CRI médio. O agrupamento é fortemente sugerido nos casos em que uma interface é requerida por apenas outro componente ou quando

		dois componentes requerem interfaces entre si. Quando isso não acontece e existem múltiplos componentes requerendo uma mesma interface, o agrupamento não é aconselhável.
C4	**Instabilidade**	Análise da relação entre o grau de responsabilidade (CPI) e o grau de dependência de cada componente (CRI). Altos valores de CPI indicam a relevância de um componente dentro do domínio. Alterações neste componente tendem a afetar os componentes a ele relacionados. Valores altos de CRI afetam negativamente um componente, pois indicam que são componentes que podem ser afetados por outros. I= CRI / (CRI + CPI) Quando o valor de I ficar próximo de zero, o componente é considerado mais estável e com um grau de dependência mais baixo. Valor próximo de I for próximo de um, indica instabilidade e alto grau de dependência de um componente. Um balanço entre responsabilidade e dependência pode ser visto quando o valor de I for próximo de 0,5. Quando um componente for considerado instável (I próximo de 1), uma reorganização do modelo de componentes é sugerido para reduzir a dependência do componente.
C5	**Análise de componentes que requerem um componente**	Análise do grau de responsabilidade, indicando o número de componentes que requerem serviços do componente analisado (CRnC). Valores de referência: CRnC mínimo no domínio; CRnC máximo no domínio; CRnC médio no domínio. É sugerido a análise da possibilidade de agrupamento de um componente com aqueles que o requerem quando o valor CRnC do componente for maior que o CRnC médio.
C6	**Análise de componentes requeridos**	Análise do grau de dependência, indicando o número de componentes requeridos pelo componente analisado (CRdC). Valores de referência: CRdC mínimo no domínio; CRdC máximo no domínio; CRdC médio no domínio. Quando o valor CRdC do componente for maior que o CRdC médio, é sugerido a análise da possibilidade de agrupamento deste componente com os requerido.

FORMULÁRIO DE PREENCHIMENTO DO AGRUPAMENTO REALIZADO UTILIZANDO OS CRITÉRIOS PROPOSTOS

Elementos no Modelo Original de Componentes	Critérios considerados no agrupamento	Modelo Final de Componentes Gerado
	☐ C1 - Escopo ☐ C2 - CPI ☐ C3 - CRI ☐ C4 - I ☐ C5 - CRnC ☐ C6 - CRdC	☐ Invariante ☐ Ponto de Variação ☐ Variante ☐ Mandatório ☐ Opcional ☐ Simples ☐ Composto ☐ Concreto ☐ Abstrato Interfaces: **Dados** Requeridas Providas **Controle** Sucessor Predecessor Tipo: ☐ Sequencial ☐ Paralelismo ☐ Síncrono ☐ Junção ☐ Decisão Classificação: ☐ Mandatório ☐ Opcional **Restrição**
Elementos no Modelo Original de Componentes	**Critérios considerados no agrupamento**	**Modelo Final de Componentes Gerado**
	☐ C1 – Escopo ☐ C2 - CPI ☐ C3 - CRI ☐ C4 - I ☐ C5 - CRnC ☐ C6 - CRdC	☐ Invariante ☐ Ponto de Variação ☐ Variante ☐ Mandatório ☐ Opcional ☐ Simples ☐ Composto ☐ Concreto ☐ Abstrato Interfaces: **Dados** Requeridas Providas

Elementos no Modelo Original de Componentes	Critérios considerados no agrupamento	Modelo Final de Componentes Gerado
		Controle Sucessor Predecessor Tipo: ☐ Sequencial ☐ Paralelismo ☐ Síncrono ☐ Junção ☐ Decisão Classificação: ☐ Mandatório ☐ Opcional **Restrição**

Componente	Valores dos critérios				
	CPI	**CRnC**	**CRI**	**CRdC**	**I**
Reference Value					

Nome:

_____

Início:_____

Fim: _____

FORMULÁRIO DE AVALIAÇÃO DOS CRITÉRIOS DE AGRUPAMENTO DE COMPONENTES DE PROCESSOS DE SOFTWARE DE LPRS

Nome do Participante:

Avaliação Geral

1. A notação e representação visual utilizada para representação do modelo de componente é de fácil entendimento e o material apresentado foi suficientemente claro e permitiu a absorção de conhecimento necessário para entendimento da modelagem de componentes da LPrS. Justifique.

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

---

2. O tempo para execução do estudo foi suficiente. Em caso negativo, ou parcialmente negativo, a falta de tempo está associada à complexidade: (1) das tarefas, (2) do domínio escolhido, (3) do entendimento dos formulários, (4) dos critérios de agrupamento aplicados, (5) outro? Justifique.

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

Avaliação Geral dos Critérios de Agrupamento

1. Os critérios disponibilizados para apoiar o agrupamento de componentes de processos de software, visando reduzir o acoplamento, são adequados. Em caso negativo, ou parcialmente negativo, indique os critérios que na sua opinião não são adequados, justificando.

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

---

2. Em algum momento você desejou agrupar determinados componentes e esta forma de agrupamento não foi apoiada pelas análises realizadas usando os critérios disponibilizados. Em caso afirmativo, ou parcialmente afirmativo, aponte qual(is) critério(s) poderiam ser modificados, justificando.

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

---

3. Em algum momento você sentiu dificuldade na aplicação de um ou mais critérios. Em caso afirmativo ou parcialmente afirmativo, qual(is)? Justifique indicando a dificuldade enfrentada e uma possível sugestão de melhoria.

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

---

4. A forma de representação visual dos critérios e a separação por visões (responsabilidade, dependência e instabilidade) são adequadas e auxiliam na análise da organização dos componentes. Em caso negativo, ou parcialmente negativo, o que poderia ser melhorado? Justifique

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

---

5. Um modelo de componentes agrupados é mais compreensível quando comparado a um modelo original com mais componentes dispostos separadamente. Em caso negativo, ou parcialmente negativo, justifique.

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

---

6. A utilização de filtros para aplicação/visualização dos critérios em etapas seria desejável. Em caso afirmativo, ou parcialmente afirmativo, justifique.

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

---

7. Eu já utilizei outras abordagens de LPrS envolvendo agrupamentos de componentes de processo. Em caso afirmativo, ou parcialmente afirmativo, aponte qual mecanismo e se possui algum apoio ferramental.

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

Suporte Computacional

1. O agrupamento de componentes de processos de uma LPrS mais extensa e complexa seria viável sem o uso de suporte computacional. Em caso negativo, ou parcialmente negativo, aponte como o uso de um suporte computacional poderia auxiliar adequadamente a aplicação dos critérios propostos? Justifique

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

---

2. O uso de um suporte computacional para realizar o agrupamento de componentes de processos de uma LPrS através da aplicação direta dos critérios propostos sem consulta ao engenheiro do domínio seria considerado invasivo. Em caso afirmativo, ou parcialmente afirmativo, que ação poderia ser considerada mais apropriada? Justifique.

□ Discordo totalmente □ Discordo parcialmente □ Indiferente

□ Concordo parcialmente □ Concordo totalmente

_____***Obrigada pela colaboração***

Apêndice VII – Formulários da Avaliação do Resultado de Aplicação da Abordagem de Desenvolvimento de LPrS Proposta

Este documento apresenta o conjunto de formulários aplicados no estudo.

FORMULÁRIO DE CARACTERIZAÇÃO DO PARTICIPANTE

Nome Completo: _____

Formação Acadêmica

☐**Pós-Doutorado** ☐**Doutorado** ☐**Mestrado** ☐**Especialização** ☐**Graduação** ☐**Técnico**
Outra: ___

Linha de Pesquisa: _____ (para Mestrado ou Doutorado)
Atualmente eu trabalho na
☐ Academia ☐ Indústria ☐ Academia e Indústria
Tempo na academia: _____
Tempo da indústria:_____

Experiência em Desenvolvimento de Software

Por favor, indique o grau de sua experiência para cada item nesta seção seguindo a escala de 5 pontos abaixo:

1 = nenhum
2 = conhecimento teórico (estudei em aula ou em livro ou apliquei em projeto em sala de aula)
3 = participei em um ou mais projetos na indústria, mas sem ser o responsável
4 = participei em até 03 oportunidades na indústria como responsável
5 = participei em mais de 03 oportunidades na indústria como responsável incluindo diferentes organizações e/ou diferente tipos de processos.

Por favor, explique sua resposta incluindo o número de semestres ou número de anos de cada experiência relevante e o número de projetos que participou (E.g. "Eu trabalhei por 10 anos como gerente de projeto na indústria em 20 projetos").

EXPERIÊNCIA COM PROCESSO DE DESENVOLVIMENTO DE SOFTWARE

Experiência implantando processos de desenvolvimento de software 1 2 3 4 5
Tempo: _____ Número de Projetos: _____

Experiência definindo processos de desenvolvimento de software 1 2 3 4 5
Tempo: _____ Número de Projetos: _____

Experiência revisando (atividades de validação/verificação) processos de desenvolvimento de software 1 2 3 4 5
Tempo: _____ Número de Projetos: _____

Experiência gerenciando processos de desenvolvimento de software 1 2 3 4 5

Tempo: _____ Número de Projetos: _____

EXPERIÊNCIA COM REUTILIZAÇÃO DE SOFTWARE

Experiência aplicando reutilização de software de maneira *ad-hoc* 1 2 3 4 5
Tempo: _____ Número de Projetos: _____

Experiência com reutilização de software aplicando técnicas como componentes, *frameworks*, padrões, bibliotecas ou repositórios 1 2 3 4 5
Tempo: _____ Número de Projetos: _____

Experiência com reutilização de software aplicando técnicas como Engenharia de Domínio ou Linha de Produtos de Software (LPS) 1 2 3 4 5
Tempo: _____ Número de Projetos: _____

Experiência com reutilização de software como responsável pela elaboração de ativos reutilizáveis ou pela definição da forma de se reutilizar software em alguns projetos 1 2 3 4 5
Tempo: _____ Número de Projetos: _____

Experiência na representação/modelagem de LPS 1 2 3 4 5
Número de Projetos: _____ Técnica/Linguagem: _____

Experiência com modelagem de características 1 2 3 4 5
Número de Projetos: _____ Notação: _____

Experiência com a notação *Odyssey*-FEX 1 2 3 4 5
Número de Projetos: _____ Domínios:_____

EXPERIÊNCIA COM REUTILIZAÇÃO DE PROCESSOS DE SOFTWARE

Experiência com aplicação de partes existentes de processos anteriores para definir novos processos, de maneira *ad-hoc*. 1 2 3 4 5
Tempo: _____ Número de Projetos: _____

Experiência com reutilização de processos de software aplicando a técnica de Linha de Processo de Software (LPrS) 1 2 3 4 5
A Linha de Processos de Software (LPrS) é uma abordagem cujo principal objetivo é definir artefatos de processos de software com o propósito de serem reutilizados em novos processos específicos de projeto com a análise de aspectos de variabilidade e opcionalidade em domínios de processos.
Tempo: _____ Número de Projetos: _____ Domínio: _____

Experiência com reutilização de processos de software aplicando a técnica de componentes de processo de software 1 2 3 4 5

Desenvolvimento de processos baseados em componentes: Consiste em definir/compor processos a partir de elementos reutilizáveis menores (componentes de processo), que encapsulam informações e comportamentos de processo.

Tempo: _____Número de Projetos: _____ Tipo do Processo: ____

Experiência na representação/modelagem de LPrS 1 2 3 4 5

Tempo: _____Número de Projetos: _____ Notação: _____

Experiência na definição de componentes de processos de software 1 2 3 4 5

Tempo: _____ Número de Projetos: _____ Tipo do Processo: _____

Experiência com reutilização de processos como responsável pela elaboração de ativos de processo reutilizáveis ou pela definição da forma de se reutilizar processos em alguns projetos 1 2 3 4 5

Tempo: _____ Número de Projetos: _____ Técnica de reutilização: _____

Experiência em Abordagens de Desenvolvimento Específicas

Por favor, indique o grau de experiência nesta seção seguindo a escala de 3 pontos abaixo:

1 = Eu não tenho familiaridade com isto. Eu nunca fiz isto.

3 = Eu apliquei isto algumas vezes, mas não sou um especialista.

5 = Eu sou muito familiar com isto. Eu me sentiria confortável fazendo isto.

<u>Quanto você sabe sobre…</u>

MPS-Br?	1	3	5
Scrum?	1	3	5
RUP?	1	3	5
Processo de Medição de Software?	1	3	5
Processo de Gerência de Portfólio?	1	3	5
Processos de Engenharia de Requisitos?	1	3	5
Processos de Gerenciamento de Projetos?	1	3	5
Processo de Gerência para Reutilização?	1	3	5
Processo de Desenvolvimento para Reutilização?	1	3	5

FORMULÁRIO PÓS-ESTUDO

Nome do Participante:

Avaliação Geral

1. A notação *OdysseyProcess-FEX* é de fácil entendimento e o material enviado foi suficientemente claro e permitiu a absorção de conhecimento necessário para entendimento da notação utilizada na modelagem do domínio. Justifique.

[ ] Discordo totalmente [ ] Discordo parcialmente [ ] Indiferente [ ] Concordo parcialmente [ ] Concordo totalmente

2. Não houve dificuldade para entender o domínio utilizado. Em caso negativo, ou parcialmente negativo, indique o que poderia ser apontado como complexidade no entendimento do domínio. Justifique.

[ ] Discordo totalmente [ ] Discordo parcialmente [ ] Indiferente [ ] Concordo parcialmente [ ] Concordo totalmente

3. O tempo para execução do estudo foi suficiente. Em caso negativo, ou parcialmente negativo, a falta de tempo está associada à complexidade: (1) das tarefas, (2) do domínio escolhido, (3) do entendimento dos formulários, (4) da notação utilizada, (5) de múltiplos modelos para representação do domínio, outros? Justifique.

[ ] Discordo totalmente [ ] Discordo parcialmente [ ] Indiferente [ ] Concordo parcialmente [ ] Concordo totalmente

Avaliação dos Benefícios Esperados com o Uso da Abordagem Utilizada

O valor Alto: é de se esperar que a abordagem utilizada forneça o benefício na maioria das situações.
O valor Médio: é de se esperar que a abordagem utilizada tenha algum potencial de fornecer o benefício esperado.
O valor Baixo: é de se esperar que a abordagem utilizada forneça o benefício apenas em poucos casos.
O valor Nenhum: é de se esperar qualquer benefício a partir da utilização da abordagem utilizada.

1. **Aumento de produtividade**

   Aumento de produtividade na definição de processos de software, ou seja, a organização seria capaz de definir mais processos, sendo necessária a utilização de menos recursos (ex.: tempo, dinheiro), assumindo a mesma qualidade, quando comparado a não utilização da abordagem de reutilização.

   [ ] Alto [ ] Médio [ ] Baixo [ ] Nenhum

2. **Diminuição de retrabalho**

   Diminuição da necessidade de se definir ou documentar novamente processos ou partes de processos já definidos anteriormente, quando comparado a não utilização da abordagem de reutilização.

   [ ] Alto [ ] Médio [ ] Baixo [ ] Nenhum

3. **Aumento da qualidade**

   No contexto de processos, o aumento da qualidade pode ser visto como sendo o aumento na adequação dos processos definidos aos contextos para os quais foram estabelecidos. Ou seja, serão definidos processos mais adequados a seus usuários, quando comparados àqueles gerados sem a utilização da abordagem de reutilização.

   [ ] Alto [ ] Médio [ ] Baixo [ ] Nenhum

4. **Diminuição de custos/esforço**

   Diminuição do esforço (custo) necessário para se definir um processo, quando comparado a não utilização da abordagem de reutilização.

   [ ] Alto [ ] Médio [ ] Baixo [ ] Nenhum

5. **Diminuição do tempo de desenvolvimento**

   Diminuição do tempo necessário para se definir um processo, quando comparado a não utilização da abordagem de reutilização.

   [ ] Alto [ ] Médio [ ] Baixo [ ] Nenhum

Avaliação do Nível de Dificuldades esperado decorrente da adoção da abordagem utilizada

O valor Alto: é de se esperar que a adoção da abordagem utilizada apresente a dificuldade na maioria das situações.

O valor Médio: é de se esperar que a adoção da abordagem utilizada tenha algum potencial de apresentar a dificuldade.

O valor Baixo: é de se esperar que a adoção da abordagem utilizada apresente a dificuldade apenas em poucos casos.

O valor Nenhum: é de se esperar que a adoção da abordagem utilizada nunca apresente dificuldade.

1. **Dificuldades na Identificação e recuperação de elementos reutilizáveis**

   Dificuldade para encontrar os elementos reutilizáveis necessários e identificar se estão ou não disponíveis.

   [ ] Alto [ ] Médio [ ] Baixo [ ] Nenhum

   Falta de informação para guiar a seleção dos elementos reutilizáveis.

   [ ] Alto [ ] Médio [ ] Baixo [ ] Nenhum

2. **Dificuldades no entendimento dos elementos de processos reutilizáveis**

   Dificuldade para entender os elementos reutilizáveis necessários e identificar a sua possibilidade de aplicação, seu propósito e suas propriedades.

   [ ] Alto [ ] Médio [ ] Baixo [ ] Nenhum

3. **Falta de qualidade dos elementos reutilizáveis**

   Elementos reutilizáveis existentes podem ser de baixa qualidade, ou seja, não serem adequados para reutilização, estarem incompletos, estarem modelados de maneira errada, etc.

[ ] Alto [ ] Médio [ ] Baixo [ ] Nenhum

4. **Existência de Barreiras psicológicas, legais e econômicas**

   Existência de resistências para se reutilizar processos ao invés de começar cada definição do zero. Estas podem ser psicológicas, legais ou econômicas.

   [ ] Alto [ ] Médio [ ] Baixo [ ] Nenhum

5. **Necessidade da criação de incentivos à reutilização**

   Dificuldade para que ocorra reutilização de processos sem que sejam criados incentivos à reutilização.

   [ ] Alto [ ] Médio [ ] Baixo [ ] Nenhum

6. **Alto custo de Implantação**

   Custos altos para implantação da reutilização de processos.

   [ ] Alto [ ] Médio [ ] Baixo [ ] Nenhum

Avaliação da Abordagem Utilizada

1. **A abordagem utilizada auxilia a definição de processos**.

   [ ] Discordo totalmente [ ] Discordo parcialmente [ ] Indiferente [ ] Concordo parcialmente [ ] Concordo totalmente

2. **A abordagem utilizada torna a atividade de definição de processos mais fácil.**

   [ ] Discordo totalmente [ ] Discordo parcialmente [ ] Indiferente [ ] Concordo parcialmente [ ] Concordo totalmente

3. **A abordagem utilizada permite a reutilização de conhecimento para a definição de processos.**

   [ ] Discordo totalmente [ ] Discordo parcialmente [ ] Indiferente [ ] Concordo parcialmente [ ] Concordo totalmente

4. **A abordagem utilizada facilita a seleção dos elementos de processos mais adequados para compor um processo em um dado contexto, considerando as situações do contexto e requisitos a serem atendidos pelo processo.**

   [ ] Discordo totalmente [ ] Discordo parcialmente [ ] Indiferente [ ] Concordo parcialmente [ ] Concordo totalmente

5. **A abordagem utilizada permite a definição de processos com base na escolha de componentes (módulos de processos)?**

   [ ] Discordo totalmente [ ] Discordo parcialmente [ ] Indiferente [ ] Concordo parcialmente [ ] Concordo totalmente

6. **A abordagem utilizada permite a definição de processos considerando dados de estabilidade e desempenho dos componentes, incluindo informações sobre medidas e *baselines* de desempenho.**

   [ ] Discordo totalmente [ ] Discordo parcialmente [ ] Indiferente [ ] Concordo parcialmente [ ] Concordo totalmente

7. **Qual a sua opinião sobre a necessidade dos artefatos definidos na abordagem? Justifique.**

[ ] Todos os artefatos da abordagem são necessários

[ ] Alguns artefatos da abordagem são necessários

[ ] Nenhum artefato da abordagem é necessário

8. **No seu ponto de vista, as opcionalidades e variabilidades ficaram explícitas na linha de processos criada? Justifique.**

   [ ] As opcionalidades e variabilidades ficaram totalmente explícitas

   [ ] As opcionalidades e variabilidades ficaram parcialmente explícitas

   [ ] As opcionalidades e variabilidades não ficaram explícitas

9. **Você acredita que linhas de processos criadas com a abordagem utilizada possam ser aplicadas na indústria? Justifique.**

   [ ] Sim, acredito que LPrSs criadas com a abordagem utilizada possam ser aplicadas na indústria.

   [ ] Sim, acredito que LPrSs criadas com a abordagem utilizada possam ser aplicadas na indústria com adaptações.

   [ ] Não, acredito que LPrSs criadas com a abordagem utilizada não possam ser aplicadas na indústria.

10. **De acordo com a sua opinião, liste os aspectos positivos da aplicação da abordagem para apoiar a reutilização de conhecimento de processos em novas definições.**

11. **De acordo com a sua opinião, liste os aspectos negativos da aplicação da abordagem para apoiar a reutilização de conhecimento de processos em novas definições.**

12. **Você possui alguma sugestão para melhoria para a abordagem?**

13. **Este espaço é reservado para quaisquer comentários adicionais (dificuldades, críticas, e/ou sugestões) a respeito da abordagem utilizada. Contamos com sua contribuição para que o trabalho seja aprimorado.**

Suporte Computacional

1. **Você acredita ser viável reutilizar o conhecimento de domínios mais extensos e complexos sem auxílio de uma infraestrutura de reutilização. Em caso negativo, ou parcialmente negativo, aponte como o uso de um suporte computacional poderia auxiliar adequadamente a reutilização, considerando ambas as etapas envolvidas (desenvolvimento de uma LPrS e utilização de uma LPrS)? Justifique.**

   [ ] Discordo totalmente [ ] Discordo parcialmente [ ] Indiferente [ ] Concordo parcialmente [ ] Concordo totalmente

Novamente, gostaríamos de agradecer pela sua disponibilidade e participação neste estudo.

Obrigada pela colaboração