VISAR3D - UMA ABORDAGEM BASEADA EM TECNOLOGIAS EMERGENTES 3D PARA O APOIO À COMPREENSÃO DE MODELOS UML

Claudia Susie Camargo Rodrigues

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 2012

VISAR3D - UMA ABORDAGEM BASEADA EM TECNOLOGIAS EMERGENTES 3D PARA O APOIO À COMPREENSÃO DE MODELOS UML

Claudia Susie Camargo Rodrigues

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:

Prof.ª Cláudia Maria Lima Werner, D.Sc.

Prof. Guilherme Horta Travassos, D.Sc.

Prof. Jano Moreira de Souza, Ph.D.

Prof.ª Rosa Maria Esteves Moreira da Costa, D.Sc.

Prof. Rafael Prikladnicki, D.Sc.

RIO DE JANEIRO, RJ – BRASIL

MARÇO DE 2012

Rodrigues, Claudia Susie Camargo

VisAr3D - Uma Abordagem Baseada em Tecnologias Emergentes 3D para o Apoio à Compreensão de Modelos UML/Claudia Susie Camargo Rodrigues. – Rio de Janeiro: UFRJ/COPPE , 2012.

XV, 166 p. 29,7 cm.

Orientadora: Cláudia Maria Lima Werner

Tese (doutorado) – UFRJ/ COPPE/ Programa de Engenharia de Sistemas e Computação, 2012.

Referências Bibliográficas: p. 134-146.

1. Ensino de Engenharia de Software. 2. Modelos UML. 3. Realidade Virtual e Realidade Aumentada. 4. Sistemas Complexos. I. Werner, Cláudia Maria Lima. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Titulo.

Aos meus pais José (*in memorian*) e Magnólia,

ao meu marido Paulo Fernando e

aos meus filhos Thiago e Marina.

Agradecimentos

A Deus em primeiro lugar, por me guiar por toda essa caminhada.

Ao meu marido, Paulo, minha âncora, minha força, pelo apoio, pelo amor e pela compreensão em todos os momentos.

Aos meus filhos, Thiago e Marina, pelo carinho e compreensão, mesmo privados de uma dedicação plena.

À minha mãe, por todo carinho e dedicação.

À minha tia Elodi, que sempre torceu e acreditou em mim.

À minha família, por todos os pensamentos positivos e força para continuar.

À professora Cláudia Werner, pela orientação, disponibilidade, competência, apoio demonstrados nesses anos em que trabalhamos juntas e por ter aceitado o desafio.

Ao professor Guilherme Travassos, pela participação na minha banca e pelas valiosas sugestões na avaliação da minha abordagem.

Aos professores Jano Moreira de Souza, Rosa Costa e Rafael Prikladnicki, que generosamente deram o seu tempo e aceitaram fazer parte desta banca.

Ao professor Leonardo Murta, por ter contribuído na proposta inicial desta tese.

Ao professor Luiz Alfredo pelas valiosas contribuições.

A amiga Karla Engelke pelo ajuda nas questões pedagógicas.

Aos amigos Sérgio Duque Estrada Meyer e Luís Felipe Gouveia, pela confecção do vídeo de treinamento do estudo experimental.

Aos amigos do grupo de reutilização da COPPE Sistemas, pelo convivência e amizade.

A todos aqueles que participaram do estudo experimental, pelo tempo, empenho e seriedade dedicados. Em especial aos colegas Jobson Massolar e Breno França, pela ajuda nas análises finais.

Aos amigos Kely Villacorta e Felipe Garcia, pela amizade e inestimável apoio, especialmente nos últimos meses deste trabalho.

Aos meus amigos Josiete Souza, Erika Coelho, Hebert Coelho, Fabiana Marinho, Rogério Borba e Vitor Lopes cujo incentivo e amizade foram muito significativos.

A todos os Funcionários do PESC: Taísa Guidini, Cláudia Prata, Solange Santos, Sônia Galliano, Gutierrez da Costa, Ana Paula Rabello, Maria Mercedes, Rosa, Itamar e Adilson pelo apoio e pelos serviços prestados ao longo de todo o período.

À CAPES, pelo apoio financeiro deste trabalho.

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

VISAR3D - UMA ABORDAGEM BASEADA EM TECNOLOGIAS EMERGENTES 3D PARA O APOIO À COMPREENSÃO DE MODELOS UML

Claudia Susie Camargo Rodrigues

Março/2012

Orientadora: Cláudia Maria Lima Werner

Programa: Engenharia de Sistemas e Computação

Novos desafios e novas demandas à educação de engenharia de software são apresentados pelas rápidas mudanças e crescente complexidade dos sistemas de software. Esta tese apresenta a abordagem VisAr3D – Visualização de Arquitetura de Software em 3D, que foi desenvolvida, como uma proposta inovadora a ser introduzida em sala de aula, com o objetivo de proporcionar a exploração e interação de modelos UML através da visualização 3D. O aluno é convidado a, intuitivamente, compreender os elementos de modelagem e suas relações neste ambiente tridimensional. Ela estabelece uma atividade de aprendizagem prática e agradável, focando em sistemas de grande escala. É uma nova forma de visualizar e compreender os modelos UML, combinando as tecnologias de Realidade Virtual e Realidade Aumentada. Diagramas 3D são gerados automaticamente a partir de um diagrama 2D existente e são capazes de fornecer uma semântica mais rica do que seu correspondente em 2D. Um estudo experimental foi realizado para avaliar a viabilidade do apoio oferecido pelo visualizador UML 3D construído a alunos da disciplina de Modelagem de Sistemas, bem como a contribuição da inserção da terceira dimensão.

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

VISAR3D - AN APPROACH BASED ON EMERGING TECHNOLOGIES IN 3D FOR SUPPORTING THE COMPREHENSION OF UML MODELS

Claudia Susie Camargo Rodrigues

March/2012

Advisor: Cláudia Maria Lima Werner

Department: Computer and Systems Engineering

New challenges and new demands on software engineering education are presented by rapid changes and increased complexity of software systems. This thesis presents the VisAr3D (Software Architecture Visualization in 3D) approach which has been designed, as an innovative proposal to be introduced in the classroom, to provide a 3D visualization of UML models. The user is invited to intuitively understand the model elements in this 3D environment. It includes exploration and interaction to establish a practical and pleasant learning activity, focusing in large scale systems. It is a new way to visualize and understand UML models, by combining the technologies of Virtual Reality and Augmented Reality. A 3D diagram is automatically generated from an existing 2D diagram and is able to provide richer semantics than its corresponding 2D diagram. An experimental study was conducted to evaluate the feasibility of the support offered by the 3D UML visualization built to students of System Modeling discipline, as well as the contribution of the insertion of the third dimension.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Siglas e Abreviaturas

AS: Arquitetura de Software
EA: *Enterprise Architect*
ES: Engenharia de Software
HMD: *Head-mounted* display
OMG: *Object Management Group*
RA: Realidade Aumentada
RV: Realidade Virtual
SAI: *Scene Access Interface*
UML: *Unified Modeling Language*
VisAr3D: Visualização de Arquitetura de Software em 3D
X3D: *eXtensible 3D*
XMI: *XML Metadata Interchange*

Capítulo 1 – Introdução

"Aprender é um processo que pode deflagrar no aprendiz uma curiosidade crescente, que pode torná-lo mais e mais criador"

Paulo Freire

1.1. Motivação

Os profissionais de Engenharia de Software (ES) que trabalham na indústria apresentam uma insatisfação quanto ao nível de preparação dos universitários recém-formados que entram no mercado de trabalho (SHAW, 2000), (HILBURN e TOWHIDNEJAD, 2007), (THOMPSON e EDWARDS, 2009). A raiz do problema parece ser a forma que a ES é ensinada. Nos últimos anos, a academia tem investido muito esforço para mitigar esse problema através da elaboração de novas formas de ensinar ES. Algumas instituições chegam a inovar ao desenvolverem algumas ferramentas que têm como objetivo mobilizar os alunos e ensinar ou apoiar o ensino de ES ((MOHRENSCHILDT e PETERS, 1998), (BAKER *et al.*, 2005), (DE LUCENA *et al.*, 2006)).

Dentro da ES, a Arquitetura de Software[1], como uma disciplina importante no desenvolvimento de software, torna-se fundamental, para atender a demanda de sistemas cada vez mais sofisticados e complexos. Faz-se necessário, portanto, um programa de educação que, além de ensinar as bases teóricas, possibilite a experiência prática e que tente aprofundar o entendimento das competências não apenas técnicas, na formação de um bom arquiteto de software (HUANG e DISTANTE, 2006), (CHENOWETH *et al.*, 2007), (CREIGHTON e SINGER, 2008).

As linguagens de programação não estão em um nível de abstração suficientemente alto para facilitar as discussões sobre a etapa de projeto em geral, muito menos sobre Arquitetura de Software. Por isso, a necessidade de uma linguagem gráfica de modelagem. A UML, como linguagem de modelagem de sistemas, pode ser empregada para a representação de arquiteturas, embora apresente um baixo formalismo possui a vantagem de representar um padrão acadêmico e industrial para a modelagem de software OO, o que facilita a

[1] Arquitetura de Software é a descrição dos elementos a partir dos quais os sistemas são construídos, interações entre esses elementos, padrões que guiam a sua composição e restrições sobre esses padrões (SHAW e GARLAN,1996).

comunicação das decisões arquiteturais entre os diferentes *stakeholders*. Em função desses fatores, a UML é adotada nesta tese.

Reconhecendo que a comunicação visual é um fator-chave no processo de ensino-aprendizado do futuro arquiteto de software ((PERRY e WOLF, 1992), (GARLAN e PERRY, 1995)), esta tese investe nas novas tecnologias emergentes de visualização 3D, Realidade Virtual e Realidade Aumentada. Recentemente, percebe-se o aumento do interesse nestas tecnologias nas aplicações de jogos e na indústria dos cinemas ((PIEKARSKI e THOMAS, 2002), (WAGNER *et al.*, 2005), (SECOND LIFE, 2011)). Contudo, o que se pretende é mostrar o potencial da aplicação das tecnologias interativas na Educação, ou mais especificamente, no apoio ao ensino de Modelagem de Sistemas ou Projeto de Sistemas, onde o tema Arquitetura de Software costuma ser ensinado. Neste contexto, pode-se receber um impulso destas novas opções tecnológicas através da visualização e análise de sistemas de software. Enfatizando aqueles compostos por um número grande de elementos com muitas interações e definidos, para fins deste trabalho, como sistemas complexos.

Durante o desenvolvimento de um projeto arquitetural é produzida uma grande quantidade de documentação que facilita a comunicação entre os *stakeholders*, registra as decisões iniciais acerca do projeto de alto-nível, e permite o reúso do projeto ((PERRY e WOLF, 1992), (GARLAN e PERRY, 1995)). Estas informações são de grande importância para o arquiteto de software e, principalmente, para o aluno iniciante, no entendimento de todo o processo de criação e desenvolvimento de software. No entanto, muitas vezes estas informações não são utilizadas devidamente. A ideia é organizar todo tipo de documentação existente e relevante relacionada aos elementos de modelagem de um sistema. Esses dados importantes estarão associados aos elementos de modelagem correspondentes, e o usuário poderá explorar e interagir com eles na terceira dimensão.

1.1.1. Cenários

De uma maneira geral, três cenários podem ilustrar a motivação para este trabalho. No primeiro cenário, podemos verificar que os alunos participam, muitas vezes, de forma passiva em sala de aula, onde ouvem palestras ministradas por seus professores. A eles são solicitados uma série de leituras e ao final do curso é cobrado um exame final, como teste do seu aprendizado. O aluno reage mal a esta abordagem de aprendizagem e em face de um grande volume de documentos necessários para o bom desenvolvimento de um projeto de Engenharia de Software, se desinteressam pelo tema.

Num segundo cenário, para atender as necessidades da indústria, o professor oferece ao aluno uma real experiência sobre modelagem de sistemas de software complexos. Segundo KARAM et al. (2004), os alunos aprendem mais se os conceitos importantes aprendidos em sala de aula podem ser aplicados fora dela. Contudo, preparar os alunos para estes desafios, requer muito tempo. Ele precisa de um programa educacional que preveja o ensino deste conhecimento mais específico, que inclua uma boa base teórica e bastante experiência prática, diferente das aulas regularmente oferecidas nos seus cursos. Outro problema enfrentado pelo professor é que ao criar oportunidades para que os alunos experimentem sob a sua supervisão, ele deve incentivar o trabalho em equipe, a colaboração, discussões e debates sobre decisões de projeto entre os diversos alunos.

Em um terceiro cenário, a necessidade de evolução do software torna os produtos propensos a defeitos, atrasos na de entrega e custos além da expectativa. O problema se agrava quando novos desenvolvedores são integrados a equipe de produção, tendo como função continuar a criar modelos de sistemas de software, sem estarem envolvidos desde a criação do aplicativo original. Esta nova equipe deve ser capaz de utilizar modelos criados por outros desenvolvedores, reutilizar seus componentes, entender as relações complexas num conjunto grande de elementos modelados e, ainda, acessar toda a documentação, que muitas vezes é insuficiente e pouco disponível. Estes novos colaboradores acham difícil ou, às vezes, impossível participar destes projetos sem ajuda dos colegas.

Cada um desses cenários caracteriza a necessidade de apoiar o professor, o aluno e o desenvolvedor de software que enfrentam dificuldades relacionadas à descrição e compreensão de modelos de software com muitos elementos.

1.2. Questão de Pesquisa

Visando atender aos problemas descritos nos cenários acima, este trabalho destina-se, preferencialmente, aos alunos de graduação e aos professores. Contudo, não se restringe apenas a este nível de alunos, podendo ser útil àqueles mais experientes, assim como pode apoiar, também, aos desenvolvedores de software.

De uma maneira geral, o trabalho de pesquisa se propõe a apoiar o professor a dar uma aula prática com modelos UML e ao mesmo tempo apoiar, também, o aluno a aprender sobre a modelagem UML em sistemas com muitos elementos de modelagem e com muita documentação.

Portanto, a questão de pesquisa foi elaborada da seguinte forma:

"Como a visualização 3D contribui como tecnologia de apoio à compreensão de modelos UML num sistema com muitos elementos de modelagem?"

1.3. Objetivo da Pesquisa

Para responder a questão de pesquisa, este trabalho tem como objetivo propor uma abordagem, denominada VisAr3D - Visualização de Arquitetura de Software em 3D – para apoiar a compreensão de modelos UML com muitos elementos de modelagem, utilizando tecnologias emergentes de visualização 3D, como a Realidade Virtual e Realidade Aumentada. E ao desenvolver o ferramental de apoio de mesmo nome, investigar a contribuição da inserção da terceira dimensão.

A ideia é desenvolver um ambiente de visualização 3D, com a utilização da Realidade Virtual e da Realidade Aumentada, para a exibição dos modelos com muitos elementos de modelagem numa nova perspectiva, análogos aos desenvolvidos na indústria. Um espaço para apresentar uma nova forma das pessoas interagirem com o conhecimento. Permitir a exploração e a interação do aluno da disciplina de Modelagem de Sistemas, utilizando os recursos e facilidades presentes neste ambiente virtual, tais como: acesso à documentação multimídia, no formato de áudio, imagens e vídeos, associada aos elementos de modelagem, possibilitando a visualização dos modelos em diversos ângulos e a interação dos mesmos em um espaço ilimitado com todos os diagramas modelados disponíveis.

Neste trabalho, propõe-se disponibilizar a documentação através de um acesso mais fácil e rápido, permitindo sua visualização sobreposta aos modelos no ambiente 3D.

É previsto, ainda, um sistema de comunicação entre os alunos e professores, favorecendo a construção do conhecimento coletivo. E disponibilizar recursos para o exercício da prática. Alguns trabalhos na literatura técnica mostram alguns pontos em comuns, como analisar e compreender informações complexas numa representação visual especial de seus diagramas, explorando a terceira dimensão; mas nenhum tem a proposta de um ambiente de apoio ao ensino-aprendizagem, facilitando o ensino por parte do professor e o aprendizado por parte do aluno, como é o caso da abordagem desta tese.

A visualização de software em 3D tem o potencial de ajudar no processo de manutenção e melhoria da compreensão do usuário de um sistema de software (YOUNG e MUNRO, 1998). Ela pode ajudar a produzir uma imagem do software, criando um objeto (visual) físico que representa o sistema de software. Engenheiros de

software podem ganhar algum *insight* inicial em como ele está estruturado e do que ele é composto. Isso os ajuda a usar suas habilidades de percepção na investigação e compreensão do software, além de materializar/concretizar algo que é abstrato/lógico.

O enfoque visual desta abordagem, empregando a Realidade Virtual e a Realidade Aumentada, na disciplina de Modelagem de Sistemas, utilizando muitos elementos de modelagem e suas relações, trata-se de um tema atual, inovador e com importância prática no ensino de ES.

1.4. Etapas da Pesquisa

A elaboração deste trabalho teve como ponto de partida um primeiro levantamento e estudo da base teórica. Através de uma *quasi* revisão sistemática, foram identificadas algumas iniciativas relevantes que tinham o objetivo de ensinar Arquitetura de Software. Identificaram-se, também, algumas necessidades para sua prática, com enfoque na utilização de sistemas complexos. Esta necessidade motivou a definição do objetivo geral do trabalho.

Como sequência do trabalho, foram realizados estudos relativos às tecnologias emergentes de visualização 3D. Bem como o desenvolvimento de protótipos de realidade virtual e aumentada. Foram testadas uma série de implementações, com diferentes linguagens de programação, assim como uma combinação de técnicas para escolher a mais adequada para a criação da visualização 3D. Foram também estudados os aspectos pedagógicos que fundamentariam todo o trabalho.

Escolhida a linguagem de programação, foi desenvolvido, ainda, outro protótipo de realidade virtual, voltado para a pesquisa da tese. Ele foi utilizado como primeiro teste com a interface, movimentação no mundo virtual e funcionalidades.

Ainda foi realizado um levantamento dos trabalhos relacionados à abordagem proposta. Para isso, foi realizada uma segunda *quasi* revisão sistemática. Os requisitos da abordagem foram definidos e um protótipo da abordagem foi desenvolvido, com o objetivo de mostrar a viabilidade da tecnologia, mostrando algo concreto e executável aos usuários. Este protótipo implementou parte das características da abordagem como prova de conceito.

Como útlima etapa foi planejado e realizado um estudo experimental para avaliar a viabilidade do apoio oferecido pelo visualizador UML 3D a alunos no contexto de uma disciplina de Modelagem de Sistemas, bem como a contribuição da inserção da terceira dimensão, antes mesmo de vislumbrar sua utilização completa no cenário proposto que seria sua utilização em sala de aula.

1.5. Organização da Tese

Partindo desta Introdução, esta tese está organizada em mais 6 capítulos, da seguinte forma:

O Capítulo 2 descreve sobre a Engenharia de Software e a Educação, as suas novas demandas e as novas propostas de ensino introduzidas na sala de aula. Apresenta a importância de se utilizar a Arquitetura de Software no desenvolvimento de sistemas, justifica a necessidade da utilização da Modelagem UML e finaliza com a primeira *quasi* Revisão Sistemática conduzida durante a pesquisa.

O Capítulo 3 descreve alguns recursos que uma ferramenta de visualização deve ter, trata sobre a Visualização e a Compreensão de Modelos e discorre sobre a Visualização 3D. Aborda as tecnologias de pesquisa e desenvolvimento escolhidas para desenvolver esta tese: Realidade Virtual e Realidade Aumentada e apresenta estas tecnologias no contexto da Educação.

O Capítulo 4 descreve os trabalhos relacionados de autores presentes na literatura, após a condução da segunda *quasi* Revisão Sistemática.

O Capítulo 5 apresenta a abordagem VisAr3D – Visualização de Arquitetura de Software em 3D – e o protótipo implementado.

O Capítulo 6 detalha a condução da avaliação da abordagem VisAr3D.

O Capítulo 7 resume a pesquisa desta tese de doutorado, apresenta as suas contribuições, uma análise das suas limitações e descreve sugestões para trabalhos futuros.

Os Apêndices de A a D contêm os formulários utilizados no estudo experimental descrito no Capítulo 6.

Capítulo 2 – Engenharia de Software e a Educação

"A primeira tarefa da educação é ensinar a ver..."

Rubens Alves

2.1. Introdução

O Software, a cada dia, torna-se cada vez mais importante em nossas vidas e, na mesma proporção, cresce a necessidade de programadores bem preparados.

Atualmente, o ensino da Engenharia de Software (ES) está passando por vários tipos de questionamentos, principalmente, pelos estudantes e pela indústria (SHAW, 2000), (HILBURN e TOWHIDNEJAD, 2007), (THOMPSON e EDWARDS, 2009). A academia ensina tipicamente teorias e conceitos que estão presentes numa série de leituras e aos estudantes, geralmente, é solicitado, em termos práticos, o desenvolvimento de um pequeno projeto para colocar todo este conhecimento em prática num espaço de tempo muitas vezes restrito. Apesar destes componentes serem necessários e úteis para a educação de futuros engenheiros de software, não são suficientes para atender todo o processo de ES. Os estudantes, em face de um grande volume de documentos necessários para o bom desenvolvimento de um projeto de ES, segundo DE LUCENA *et al.* (2006), se desinteressam e associam-no como assunto “extremamente” teórico. Muitas vezes, durante as aulas, os conceitos não são muito bem ilustrados, tornando-se monótono quando o professor fala na maior parte do tempo. Os alunos tendem a decorar o conteúdo sem dar a devida importância para o assunto. Estes alunos parecem preferir escrever programas e ver seu código funcionando, a documentar formalmente o desenvolvimento de suas aplicações.

Paralelo a isso, de acordo com CONN (2002), os profissionais de ES, na indústria, estão insatisfeitos com a falta de preparo dos universitários que ingressam no mercado de trabalho. A indústria se vê obrigada a complementar sua educação com treinamentos e preparações que lhes forneçam conhecimento que suprem esta deficiência. Geralmente, os alunos saem da faculdade sem terem participado de um processo de ES próximo ao que vão encontrar no mercado de trabalho, que envolve: sistemas grandes e complexos; a participação em uma equipe também grande e distribuída geograficamente, com mudanças de objetivos durante o projeto; problemas com os clientes; pressão com tempo de entrega do produto; uma maior demanda por qualidade de software; e outros fatores como gerenciamento, questões de espaço de trabalho e cultura corporativa.

Segundo MEYER (2001), a academia não deve assumir toda esta responsabilidade, pois a universidade não é uma empresa e nem deve ser. Mas deve preparar seus estudantes para os reais desafios que encontrarão, sendo este um projeto em longo prazo. Recentemente, a academia colocou muito esforço em diminuir este problema, encontrando novas maneiras de ensinar a ES, utilizando diferentes abordagens que compartilham o mesmo objetivo: diminuir a distância entre a teoria e a prática. Há um consenso que diz que a maneira como a ES é ensinada deve ser mudada para refletir esta nova demanda por desenvolvimento de software mais sofisticado (BAKER *et al.*, 2005). HUANG e DISTANTE (2006) dizem, ainda, que o ensino de ES tradicional focalizado pesadamente em metodologias de ES, não é adequado. E acrescenta que os estudantes precisam aprender tanto aspectos técnicos, quanto aqueles não-técnicos no desenvolvimento de sistemas de software. Segundo VARMA e GARG (2005), esta busca por métodos alternativos, convencionais e não-convencionais, pode fazer o ensino da ES mais efetivo e interessante, e, ainda, estreitar a distância entre a indústria e a academia.

Baseados nestes objetivos e nas demandas descritas acima, novas propostas de ensino de ES foram introduzidas na sala de aula, principalmente aquelas que tornam o ensino mais atraente para o aluno, que serão apresentadas na próxima Seção.

Este capítulo está organizado da seguinte forma: a Seção 2.2 mostra algumas ferramentas de apoio ao ensino de Engenharia de Software. A Seção 2.3 define Sistemas Complexos e apresenta algumas técnicas para lidar com estes sistemas. A Seção 2.4 destaca a importância de utilizar a Arquitetura de Software no desenvolvimento de sistemas. Na Seção 2.5, é justificada a necessidade da utilização da Modelagem UML. Na Seção 2.6, a primeira *quasi* Revisão Sistemática conduzida durante a pesquisa desta tese, é detalhada. As Considerações Finais fecham o capítulo na Seção 2.7.

2.2. Ferramentas de Apoio ao Ensino de Engenharia de Software

Devido à sua finalidade, é muito importante que as instituições de ensino forneçam educação apropriada e oportunidades de pesquisa para prepararem os estudantes para esses novos desafios e oportunidades. O ensino da ES deve prepará-los para participações efetivas e produtivas de uma forma colaborativa e interdisciplinar (POUR, 2006). Esta Seção tem como objetivo apresentar ferramentas propostas por algumas destas instituições que, de alguma forma, tentam tornar o ensino de ES em tarefas atrativas de laboratório. Na literatura existem muitos exemplos, contudo são citados aqui apenas três representativos, que têm em comum

a utilização de recursos de ensino, às vezes, externas ao computador, que desafiam e mobilizam os alunos em tarefas mais próximas de objetivos reais. A seguir, serão descritos alguns projetos responsáveis pelo desenvolvimento de ferramentas de apoio ao ensino de ES. São eles: *Problems and Programmers*, Projeto Unibral e *Draw-Bot*.

2.2.1. *Problems and Programmers* (BAKER *et al.,* 2005)

Baseado na observação de que os alunos, em um curso de ES, possuem pouca experiência prática em processo de software, devido, principalmente, a questão de tempo, pesquisadores da Universidade da Califórnia (Irvine nos Estados Unidos) desenvolveram a abordagem *Problems and Programmers.* Trata-se de um jogo de cartões educativo que simula o processo de ES a partir da especificação de requisitos até a entrega do produto. Ele fornece aos estudantes uma visão geral de alto nível, possibilitando a experiência prática do processo de ES, num espaço de tempo reduzido e competitivo. Cada jogador assume o papel de um gerente de projeto e deve completar o mesmo antes de qualquer oponente. Além disso, a sua natureza física, ou seja, a utilização de cartões (Figura 2.1) e interação face-a-face incentivam a interação entre os estudantes, motiva os jogadores, encorajando o aprendizado colaborativo. No jogo, as leituras propostas ensinam os conceitos e teorias fundamentais da ES.

O jogador, que completar o projeto primeiro, será o vencedor. No entanto, eles devem se preocupar com questões como o orçamento ou a demanda do cliente por segurança no software a ser produzido. Ou seja, eles devem seguir práticas de ES apropriadas para evitar qualquer consequência adversa que poderá levá-los a falhar perante o seu oponente na corrida para completar o projeto.

O jogo utiliza 85 regras de ES. Estas regras representam as boas práticas seguidas tanto pela academia quanto pela indústria.

Problems and Programmers é um jogo interessante para o aluno, pois combina conceitos de ES com simulação. Segundo o artigo, esta abordagem representa a primeira tentativa em ensinar o processo de ES utilizando um jogo com cartões. Sua maior contribuição é a motivação dos alunos, disponibilizada, principalmente, pela interação face-a-face.

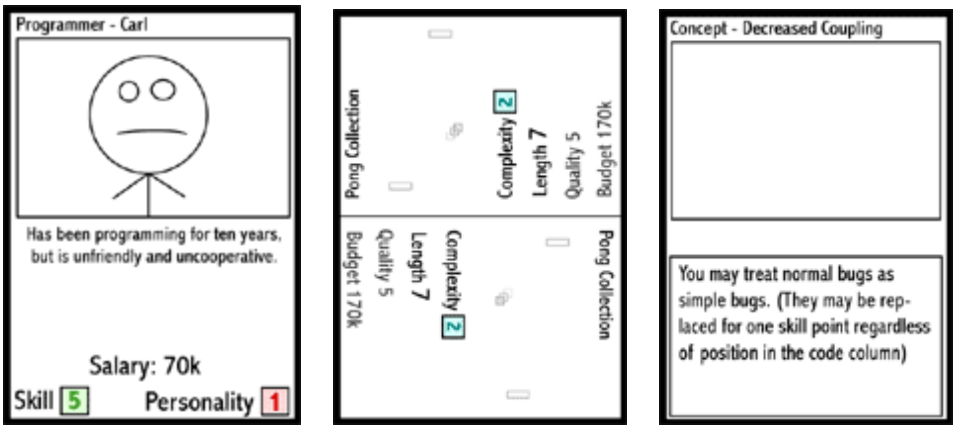


Figura 2.1: Alguns cartões do *Problems and Programmers*
(BAKER *et al.*, 2005)

2.2.2. Projeto Unibral (DE LUCENA *et al.*, 2006)

O Projeto Unibral é resultado de uma cooperação científica e acadêmica entre a Universidade Federal do Amazonas e o Instituto de Automação Industrial e Engenharia de Software da Universidade de Stuttgart. O projeto, iniciado em 2003, incentiva a interação entre estudantes de engenharia elétrica e estudantes de ciência da computação.

A ideia do projeto é expor aos estudantes os mesmos problemas que são encontrados em situações profissionais reais. Portanto, o objetivo do laboratório é fazer com que os alunos, a partir de uma plataforma de hardware existente (um robô), desenvolvam um sistema de software que resolva um problema específico. Os alunos têm disponível o robô (Figura 2.2), seu sistema de controle e de comunicação. O desafio proposto é desenvolver uma estratégia (algoritmo) para controlar o robô, que deve partir de um ponto e se mover de maneira autônoma por um caminho com obstáculos desconhecidos, no prazo mais curto em relação ao seu competidor.

O resultado dessa experiência foi um aumento significativo no interesse dos alunos. A corrida de robôs foi a razão para a motivação observada, à medida que os estudantes competiram para mostrar que sua equipe produziu o melhor software.

As fases do projeto a serem seguidas são: Definição, Projeto, Implementação, Integração e Aceitação, com documentos de garantia de qualidade para cada fase. No início do curso, os alunos recebem uma descrição textual dos requisitos dos usuários, que é incompleta e ambígua. Baseados nestes requisitos, eles devem formular cerca de 20 requisitos funcionais e 10 não-funcionais.

O Projeto Unibral é uma iniciativa importante, principalmente, por pertencer a duas universidades de culturas diferentes, garantindo o intercâmbio entre os pesquisadores para o aprimoramento do ensino da ES.

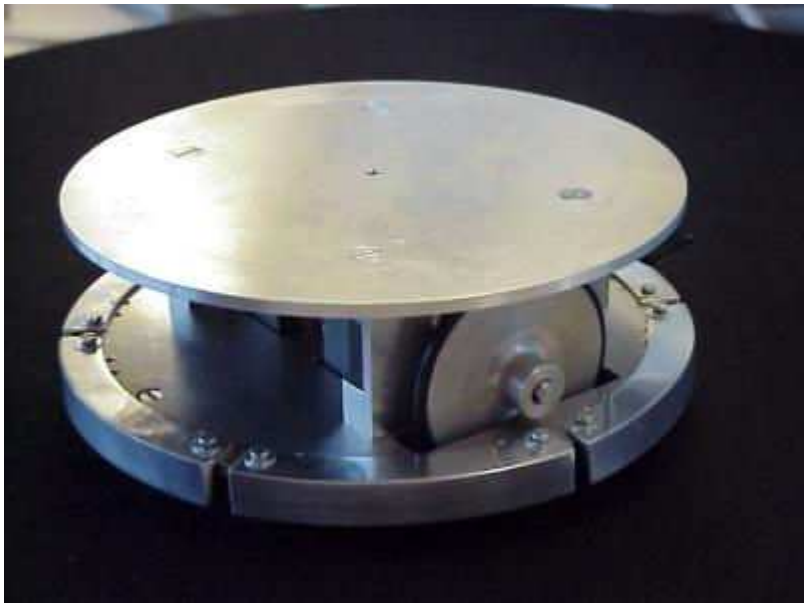


Figura 2.2: Robô utilizado no projeto Unibral (DE LUCENA *et al.*, 2006)

2.2.3. *Draw-Bot* (MOHRENSCHILDT e PETERS, 1998)

Com o objetivo de ensinar os princípios de ES como especificação de software, projeto e teste de sistemas com requisitos de segurança e em tempo real, num projeto concreto, um curso da Universidade de McMaster desenvolveu o projeto *Draw-Bot*. Durante o curso, os estudantes têm como objetivo controlar um robô para traçar um simples caminho através de um labirinto (no papel). O robô é construído através de um pequeno kit de robô educacional que é controlado por uma interface de software, ou seja, os alunos não devem se preocupar com aspectos de hardware para construí-lo. Eles acreditam que os seus requisitos podem e devem ser atingidos e que, para isso, a quantidade de esforço requerido deve ser apropriada para que este objetivo seja cumprido no tempo restrito do curso. Segundo MOHRENSCHILDT e PETERS (1998), percebeu-se um aumento significativo do interesse pelos estudantes.

O controle de um dispositivo, neste caso, o robô, para completar uma tarefa bem definida, possui um número grande de vantagens, entre eles: o entusiasmo dos estudantes e o aprendizado dos conceitos de ES num projeto real.

Tanto o projeto *Draw-Bot*, quanto o Unibral, compartilham a ideia de que seu projeto, ao exigir o controle de um dispositivo para concluir uma tarefa clara, gera entusiasmo nos alunos e os ajuda a compreender os princípios ensinados no curso. O

projeto Unibral acrescenta a competição entre os alunos no processo de aprendizagem que, segundo eles, junto com o uso do robô, foi um fator decisivo para a motivação observada.

2.3. Sistemas Complexos

A sociedade depende cada vez mais de software com operações críticas, cujas falhas são intoleráveis. As falhas de sistemas complexos podem custar vidas ou, no caso de um produto de software comercial, pode resultar em um “recall” muito caro de milhares de unidades. Estas aplicações apresentam um conjunto muito rico de comportamentos, como manter a integridade de centenas de milhares de registros de informações, com atualizações simultâneas e consultas, ou são sistemas de comando e controle de entidades do mundo real, tais como o controle do tráfego aéreo ou ferroviário. Sistemas de software como estes tendem a ter uma vida longa, e muitos usuários dependem de seu bom funcionamento.

BOOCH (1994) afirma que a característica marcante de um sistema grande de software é ser muito difícil, senão impossível, para um desenvolvedor individualmente compreender todas as sutilezas de seu projeto. Ele também diz que a complexidade parece ser uma propriedade inerente a todos os sistemas grandes de software, e que devemos dominar essa complexidade. O estudo de sistemas complexos é uma tarefa desafiadora, que lida com problemas que são difíceis de resolver e, muitas vezes, de difícil compreensão, porque são abertos a variadas interpretações. Em particular, os iniciantes, como estudantes, por exemplo, podem enfrentar sérias dificuldades em compreender os detalhes de tais sistemas.

Perante esta situação, SIMON (1996, citado em (JENNINGS, 2001)) também concorda que o papel da ES é fornecer estruturas e técnicas que tornem mais fácil lidar com a complexidade. Os engenheiros de software têm desenvolvido uma série técnicas para ajudar a gerenciar a complexidade, dentre elas (BOOCH, 1994):

Decomposição: esta é a técnica básica para enfrentar grandes problemas, que é dividi-los em partes menores, mais simples. Cada parte pode ser tratada, independentemente, das demais e compreendida, também, independentemente, das demais. Para entender um nível de um sistema, precisa-se apenas compreender algumas partes (em vez de todas as partes) de uma só vez.

Abstração: é a definição de um modelo simplificado do sistema que enfatiza alguns detalhes ou propriedades, enquanto suprime outros. Se um observador for incapaz de dominar a totalidade de um objeto complexo, pode-se optar por ignorar

seus detalhes não essenciais, lidando com o modelo idealizado deste objeto, generalizando. Assim, limita-se o número de coisas que pode ser compreendido de uma só vez, e através da abstração, são usadas partes da informação com mais conteúdo semântico.

Estrutura: é a gestão das relações entre os vários componentes do sistema. Identificar a estrutura dentro de um sistema complexo de software muitas vezes não é fácil, porque exige a descoberta de padrões entre os muitos objetos, com comportamento complicado. A capacidade de identificar as relações dentro de um sistema torna sua compreensão simplificada.

Se os seus modelos são difíceis de entender, são difíceis, também, de analisar, modificar, ampliar, integrar-se com outros modelos, e de serem reutilizados. Para alcançar os benefícios do ensino de sistemas deste porte, é necessário oferecer um suporte a compreensão destes modelos.

JACOBSON *et al.* (2003) descrevem um conjunto de princípios gerais de *design* para criar ambientes de aprendizagem e ferramentas de aprendizagem que ajudam os alunos a compreender as perspectivas científicas sobre sistemas complexos:

a) **Conectando-se com os interesses e experiências dos alunos**: é importante desenvolver atividades de ensino sobre sistemas complexos que tenham relevância imediata para o educando. Os alunos se interessam mais quando estão envolvidos com atividades do "mundo real";

b) **Experimentando fenômenos de sistemas complexos**: os alunos precisam de oportunidades para experimentar diretamente ou conduzir experimentos e observações sistêmicas de fenômenos de sistemas complexos. No passado, os materiais educativos destinados a cobrir fenômenos científicos contavam quase que exclusivamente com representações textuais, complementado por experimentos utilizando instrumentos e técnicas científicas. No entanto, novas visualizações científicas e ferramentas computacionais de modelagem e simulação estão permitindo que os próprios cientistas experimentem qualitativamente as representações destes fenômenos e, assim, de forma iterativa, explorar questões e hipóteses nas representações virtuais e modelos;

c) **Tornar os conceitos básicos explícitos**: é fazer com que os conceitos fundamentais sejam explícitos para o aluno. Especialistas, em contraste com os alunos novatos ou intermediários, são capazes de "cognitivamente ver" a relação entre as características dos fenômenos e problemas. A ideia é ajudar os alunos a obterem entendimentos conceituais mais avançados sob a perspectiva de sistemas complexos;

d) **Incentivando a colaboração, discussão e reflexão**: ambientes de aprendizagem em que os alunos experimentam e constroem seu conhecimento sobre sistemas complexos, podem ser, significativamente, mais poderosos, interessantes, envolventes e motivadores, quando envolvem interações colaborativas e cooperativas. Essas discussões podem ser entre alunos ou com a presença do instrutor;

e) **Construindo teorias, modelos e experimentos**: Um princípio central das abordagens de aprendizagem construtivista é que um aluno está ativamente construindo novos entendimentos, ao invés de receber passivamente e absorvendo "fatos". Uma forma de implementar essa perspectiva, como parte de atividades educativas relacionadas a sistemas complexos, é envolver os alunos com problemas e questões relacionadas com tais sistemas, para que eles gerem perguntas, teorias e hipóteses, e depois, executem experimentos observacionais ou criem modelos computacionais relacionados; e

f) **Aprendizagem como trajetória de aprofundamento do entendimento e da exploração**: o objetivo do aprendizado não é apenas "cobrir" vários conceitos de sistemas complexos. Pelo contrário, a integração de conhecimentos e metodologias de sistemas complexos promoverá trajetórias de aprendizagem para os alunos que irão levar a um crescimento conceitual e a um aprofundamento do seu entendimento ao longo do tempo.

2.3.1. Abstração

Segundo HAZZAN e KRAMER (2007), a abstração é apresentada como um conceito central da Ciência da Computação e ES e, por essa razão, merece destaque nesta Seção. Eles citam muitas explicações para a importância da noção de abstração, entre outras, a intangibilidade dos sistemas de software, a necessidade de abstrair, a fim de lidar com a complexidade, e a capacidade de analisar muitos tópicos em Ciência da Computação e ES em diferentes níveis de detalhe. Apesar disso, este não é um conceito fácil de ensinar.

Segundo os mesmos autores, a abstração possui um significado cognitivo segundo o qual, para superar a complexidade em um estágio específico de uma situação de resolução de problemas, nós nos concentramos nas características essenciais do nosso objeto de pensamento, e ignoramos detalhes irrelevantes. KRAMER (2007) simplifica destacando dois aspectos: ênfase no processo de remoção de detalhes para simplificar e concentrar a atenção e ênfase no processo de generalização para identificar o núcleo comum ou essência.

Ao tentar compreender as exigências dos clientes, os desenvolvedores devem ter uma visão global da aplicação (alto nível de abstração) e ao codificar uma classe específica, devem adotar uma perspectiva local (em um nível mais baixo de abstração).

Software é entidade cuja complexidade exige do profissional o domínio da expressão de ideias na forma de modelos. Assim, entra a abstração que é um conceito fundamental para a criação destes modelos. Esta habilidade de abstração deve ser exercitada pelos estudantes, assim como a confecção de modelos correspondentes.

2.4. Arquitetura de Software

Um dos assuntos que merece destaque no ensino de ES, a Arquitetura de Software (AS) procura orientar o processo de desenvolvimento de software, fornecendo uma sólida base para o reúso (SHAW e CLEMENTS, 2006). À medida que os sistemas baseados em software vêm atingindo complexidade e alcances inimagináveis, e se tornando cada vez mais essenciais à sociedade e às organizações, a importância da especificação e do projeto destes sistemas vai aumentando. A necessidade da aplicação de um maior formalismo das representações estruturais de sistemas e o estudo dos possíveis benefícios alcançados e consequências diretas de sua aplicação levaram a AS ao status de disciplina, sendo uma área de pesquisa que tem recebido grande atenção (GARLAN e SHAW, 1994). LAGO e VAN VLIET (2005) citam, ainda, que a AS está se tornando um dos tópicos centrais da ES.

Encontram-se na literatura técnica algumas definições relacionadas ao termo Arquitetura de Software (MENDES, 2002), (BASS *et al.*, 2003), (SEI, 2011). Segundo SHAW e GARLAN (1996), AS é a descrição dos elementos a partir dos quais os sistemas são construídos (componentes), interações entre esses elementos (conectores), padrões que guiam a sua composição e restrições sobre esses padrões.

Ela é um elemento de ligação fundamental entre os requisitos de um software e a implementação. Uma ponte que interliga um mundo não-técnico, envolvendo os *stakeholders* (o problema), e um mundo técnico, envolvendo os desenvolvedores e *designers* de software (a solução) (LAGO e VAN VLIET, 2005). Ou melhor, a AS se tornou o meio principal pelo qual os requisitos são transformados em sistemas implementados que funcionam (GRISHAM *et al.*, 2007).

Apesar do fato de que a aplicação de um bom projeto arquitetural ser, imensamente, importante para a prática de ES, a prática comum, muitas vezes ainda é

um projeto arquitetural informal, *ad-hoc*, que não é analisado, não manutenível e artesanal (GARLAN e PERRY, 1995). Por outro lado, o projeto arquitetural de um sistema de larga escala é um fator determinante para o sucesso do sistema. A escolha de uma arquitetura não apropriada pode ter um efeito desastroso.

Uma AS reflete as grandes decisões de um projeto de software. E estas decisões são tomadas por arquitetos de software, baseadas, também, nas considerações de diferentes *stakeholders* envolvidos. Muitos profissionais da área de desenvolvimento recebem o título de arquitetos de software. Geralmente, estes possuem larga experiência na criação de software em suas organizações e bom conhecimento de tecnologias, no entanto, muitas vezes, as arquiteturas criadas pelos mesmos, não possuem uma representação adequada. O papel do arquiteto de software deve abranger competências tanto técnicas como não-técnicas. Ele deve ser o responsável por criar uma atmosfera de confiança e de motivação para se obter sucesso neste ambiente complexo, trabalhando em uma equipe formada por pessoas com diferentes *backgrounds* e pensamentos. Atender estas necessidades, segundo CREIGHTON e SINGER (2008), é indispensável.

A literatura ((PERRY e WOLF, 1992) e (GARLAN e PERRY, 1995), dentre outros) destaca os seguintes aspectos que justificam a importância de utilizar AS:

a) **Compreensão**: a AS simplifica a habilidade de compreender sistemas de larga escala ao apresentá-los num alto nível de abstração, gerenciando questões complexas do projeto e trazendo vantagens para todas as etapas do ciclo de vida do sistema. Para apresentar novos projetos de sistemas para outras pessoas, a arquitetura auxilia a ação entre os *stakeholders* e, também, representa a união da visão dos *stakeholders* sobre o projeto. Isto significa que a AS representa uma abstração comum em alto nível de um sistema a qual a maioria dos participantes (*stakeholders*) podem usar como base para criar um entendimento mútuo, formar consenso e comunicar-se uns com os outros (BASS *et al.*, 2003);

b) **Reutilização:** um projeto arquitetural facilita o reconhecimento de estruturas comuns entre sistemas, possibilitando a reutilização ao longo de um mesmo projeto ou em projetos de outros sistemas. Ele suporta diferentes granularidades de reúso: em pequena escala, no nível de componentes, e em larga escala (subsistemas, produtos ou *frameworks*), reduzindo o esforço e melhorando a produtividade;

c) **Evolução**: a AS propicia a longevidade de um sistema, permitindo a facilidade de sua evolução e de sua escalabilidade;

d) **Análise**: as descrições arquiteturais fornecem novas oportunidades para análise, gerando documentação mais clara e expõem vários tipos de problemas que podem não ser detectados. Elas favorecem a tomada de decisão pelos *stakeholders* sobre alternativas de projetos. A divisão do esforço de desenvolvimento dentro da equipe fica mais clara, pois com os itens dos sistemas identificados, o trabalho de desenvolvimento se torna simplificado. Ela, também, pode simplificar a análise de fatores como consistência, qualidade da aplicação e conformidade com as restrições previamente estabelecidas. O custo das mudanças e a identificação das possíveis alterações podem ser mais facilmente analisáveis; e

e) **Gerência**: fornece uma base gerencial para estimativa de custo e gerência de processo.

SHAW e GARLAN (1996) enfatizam a importância de projetar sistemas complexos por meio de ferramentas intelectuais necessárias para fazê-lo de forma eficaz. Eles já frisavam que os cursos de AS deviam estar preparados para lidar com o desenvolvimento das competências voltadas para a compreensão das arquiteturas. Devido às demandas do mercado por sistemas de larga escala, ao longo do tempo, os engenheiros de software têm procurado maneiras melhores de compreender o seu software e novas formas de construir sistemas de software maiores e mais complexos.

Outra visão de ensino também é abordado por BOER *et al.* (2009), que estimula os alunos a refletirem sobre o *feedback* obtido depois de uma discussão sobre uma solução de arquitetura com outros estudantes. KARAM *et al.* (2004), por outro lado, afirma que os alunos, geralmente, aprendem mais, se eles aplicam os conceitos aprendidos na prática, num projeto real. Contudo, DENZLER e GRUNTZ (2008), com longa experiência no ensino de ES, diz perceber que os alunos têm problemas ao aplicar o conhecimento teórico em projetos concretos.

LAGO e VAN VLIET (2005) seguem alguns objetivos em seus cursos: (i) os alunos devem saber como desenvolver diferentes visões de arquitetura, abordando questões específicas das partes interessadas; (ii) por haver diferentes soluções aceitáveis de uma AS, a solução, finalmente, escolhida depende do equilíbrio entre as preocupações das partes interessadas; e (iii) os alunos devem saber como fazer uma avaliação de uma arquitetura. A avaliação fornece *insights* sobre os limites das soluções arquiteturais, as consequências da solução escolhida, bem como uma impressão da sua qualidade. A avaliação, ainda, envolve explicar a arquitetura para seus *stakeholders*.

Em geral, existem vários desafios relacionados ao ensino de projeto de arquitetura de software para estudantes universitários, incluindo pós-graduandos. Segundo MAENNISTOE *et al.* (2008), os alunos não têm tanta experiência com os desafios da vida real sobre AS, e eles normalmente têm acesso limitado a tarefas de alto nível de *design*. Eles, raramente, têm a chance de desempenhar o papel de arquiteto de software. Além disso, eles não dominam o domínio da aplicação para obter êxito necessário na concepção da AS. Portanto, devido à limitada capacidade e tempo, os projeto nos cursos universitários, muitas vezes começam do zero. Ao contrário do contexto industrial, onde os arquitetos têm de dar conta de uma grande quantidade de software e sistemas existentes, com um porte maior e com muito mais usuários (CHENOWETH *et al.*, 2007). DONGSUN *et al.* (2008) diz que as empresas estão ansiosas para contratar engenheiros com este tipo de preparação.

2.5. Modelagem UML

As linguagens de programação não estão em um nível de abstração suficientemente alto para facilitar as discussões sobre o projeto, muito menos sobre AS. Por isso, a necessidade de uma linguagem gráfica de modelagem.

A UML (*Unified Modeling Language*) é uma notação gráfica padrão, proposta pela OMG (*Object Management Group*), que ajuda na descrição e no projeto de sistemas de software orientados a objetos. Como descrito por BOOCH *et al.* (1998), "é um padrão gráfico para visualizar, especificar, construir e documentar os artefatos de um sistema de software".

A UML pode ser empregada para a representação de arquiteturas, contudo ainda encontra-se distante da representatividade provida pelas linguagens de descrição arquitetural (ADLs - *Architecture Description Languages*). A falta de uma semântica formal tem dificultado o desenvolvimento rigoroso baseado em modelos de aplicações que por sua natureza necessitam de ênfase na especificação e na verificação de seus componentes. Embora a UML apresente um menor formalismo de especificação do que as ADLs, ela facilita a comunicação das decisões arquiteturais entre os diferentes *stakeholders*.

A UML é uma linguagem visual universalmente utilizada para o desenvolvimento de software, tanto em nível acadêmico quanto industrial. Ela tornou-se um padrão comercialmente aceito para a visualização de sistemas de software e tem sido contemplada em diversos trabalhos encontrados na literatura. Além disso, diversas ferramentas e ambientes são compatíveis com a UML.

Atualmente, a especificação da linguagem UML encontra-se em sua versão 2.0. Classes, métodos, objetos e as mensagens são exemplos de elementos de modelo UML. Estes elementos representam entidades de programas de software ou de relações entre eles. A UML tem 13 tipos de diagramas para visualizar modelos. E cada tipo de diagrama descreve uma projeção de um modelo UML numa certa perspectiva.

Uma das grandes vantagens da UML é o fato dela ser extensível e flexível. O objetivo não é adaptar a sua modelagem à UML, mas sim estendê-la, se necessário, para não perder a compreensão. Além disso, um mesmo diagrama pode ser utilizado nas fases de análise e projeto, mudando-se, apenas, sua visão.

A UML facilita a compreensão de sistemas complexos durante todo o ciclo de vida e não depende da plataforma. Ela dá suporte a diversas áreas de aplicação, recebe suporte de inúmeras ferramentas e tem uma grande aceitação pelos desenvolvedores de aplicações.

A UML inclui estruturas gráficas que fornecem uma visão geral tanto da estrutura como do comportamento do programa. Ela pode ajudar até mesmo não-programadores a obter uma melhor compreensão da funcionalidade geral do sistema.

No entanto, um diagrama UML não é forte o suficiente para mostrar todas as informações necessárias para o projeto, como: as alternativas tecnológicas discutidas, mudanças no objetivo do projeto etc. Estas informações importantes não ficam em lugar nenhum e podem se perder (TREHARNE, 2002).

Um dos problemas das ferramentas existentes, que permitem criar e projetar software com UML, é que elas não ajudam a compreensão da conexão entre os vários diagramas UML. Elas, geralmente, fornecem uma janela para cada diagrama. Isto pode causar dificuldades em relacionar informações que estejam em diagramas separados (HUOTARI, 2004).

A principal razão pela escolha da UML nesta tese é proveniente de seu uso amplo, da padronização dentro da comunidade de desenvolvimento orientado a objetos e por ser uma notação rica que combina várias técnicas de diagramação.

Por ser uma notação consistente, clara, acessível para arquitetos, desenvolvedores, gerentes e estudantes, esta tese utiliza a UML para descrever a AS de um sistema, mesmo sabendo de suas limitações quanto a sua representatividade.

2.6. Primeira *quasi* Revisão Sistemática

Visando investigar na literatura o material relevante sobre as práticas de ensino da disciplina de AS, realizadas pela comunidade acadêmica, houve a necessidade de se conduzir uma *quasi* Revisão Sistemática. A íntegra desta *quasi* Revisão Sistemática, realizada de março a abril de 2009, pode ser encontrada em (RODRIGUES e WERNER, 2009b) e (RODRIGUES e WERNER, 2009c).

Como o objetivo deste estudo foi realizar uma caracterização da área, não houve comparação. Portanto, pode-se definir este tipo de estudo secundário, apesar de sistemático, como uma *quasi* revisão sistemática (TRAVASSOS *et. al.*, 2008).

Ela foi conduzida em três etapas: Planejamento da Revisão, Condução da Revisão e Publicação dos Resultados, de acordo com BIOLCHINI *et al.* (2005), conforme detalhado a seguir.

2.6.1. Planejamento da Revisão

2.6.1.1. Questão de Pesquisa e Estrutura PICO

Objetivo:

Identificar as iniciativas, práticas ou experiências realizadas para o ensino de Engenharia de Software que caracterizam uma tentativa de ensinar a disciplina de Arquitetura de Software, ou parte dela, não representando, necessariamente, um curso completo de Arquitetura de Software.

Questão de pesquisa:

Quais iniciativas foram realizadas no ensino de Arquitetura de Software?

Foi adotada a abordagem PICO que estrutura a questão de pesquisa em quatro elementos básicos: população, intervenção, comparação e resultado (PAI *et al*., 2004):

População (P):

Publicação tendo em vista a Arquitetura de Software

Intervenção (I):

Ensino de Arquitetura de Software

Comparação (C):

Não há

Resultados (O):

Iniciativas identificadas

2.6.1.2. Fontes para Busca, Termos e Sinônimos

Fontes:

Por esta ter sido a primeira *quasi* revisão sistemática realizada durante a pesquisa de tese, serviu como base de estudo e, por isso, decidiu-se utilizar uma gama maior de fontes de pesquisa. Foram utilizadas, portanto, as seguintes bases de dados eletrônicas: IEEE, Elsevier, ACM, Springer, Compendex e Scopus.

Termos e sinônimos utilizados na pesquisa:

Software Architecture – architecture design, architectural representation, architectural model;

Education – educational, training, teaching, learning, mentoring, course;

Initiative – experience, best practices, benefit, guideline, tool, method, technique, curriculum.

2.6.1.3. *String* de Busca

A string de busca utilizada foi montada da seguinte forma:

P:

("software architecture" **OR** "architecture design" **OR** "architectural representation" **OR** "architectural model")

I:

(education **OR** educational **OR** training **OR** learning **OR** teaching **OR** mentoring **OR** course)

O:

(initiative **OR** experience **OR** "best practices" **OR** benefit **OR** guideline **OR** tool **OR** method **OR** technique **OR** curriculum **OR** experiences **OR** initiatives **OR** benefits **OR** guidelines **OR** tools **OR** methods **OR** techniques **OR** curricula)

(P) AND (I) AND (C) AND (O):

("software architecture" **OR** "architecture design" **OR** "architectural representation" **OR** "architectural model")

AND

(education **OR** educational **OR** training **OR** learning **OR** teaching **OR** mentoring **OR** course)

AND

(initiative **OR** experience **OR** "best practices" **OR** benefit **OR** guideline **OR** tool **OR** method **OR** technique **OR** curriculum **OR** experiences **OR** initiatives **OR** benefits **OR** guidelines **OR** tools **OR** methods **OR** techniques **OR** curricula)

2.6.1.4. Critérios para Inclusão e Exclusão de Estudos

Os critérios definidos para inclusão e exclusão do estudo foram:

- Os documentos devem estar disponíveis publicamente na Web;
- Os estudos devem ser sobre iniciativas que foram realizadas, especificamente, no ensino de AS, ou parte dela, não representando, necessariamente, um curso completo da disciplina;
- Os artigos devem apresentar o resultado da *string* de busca no seu título ou no seu resumo;
- Os artigos devem ser escritos em inglês;
- Os artigos que apresentavam iniciativas de ensino de engenharia de software, sem o enfoque devido à AS, devem ser excluídos.

2.6.1.5. Processo de Seleção dos Estudos

- O pesquisador executou a busca nas fontes selecionadas utilizando a string de busca elaborada.
- Os artigos retornados pela busca foram inseridos na ferramenta JabRef (JABREF, 2009).
- O conjunto de artigos foi selecionado a partir da verificação dos critérios de inclusão e exclusão. Esta verificação se deu pela leitura do resumo e do título do artigo.

Os artigos incluídos e excluídos foram documentados no Formulário de Seleção de Estudos, apresentado em (RODRIGUES e WERNER, 2009c).

Apesar do protocolo formal, do critério empenhado e das iterações realizadas, esta *quasi* revisão sistemática foi feita por apenas um pesquisador. Mesmo correndo o risco de aumentar o viés do estudo, na época ela representou um material valioso, como embasamento teórico, para a proposta de tese.

2.6.1.6. Avaliação da Qualidade dos Estudos

Não foram preparados procedimentos explícitos para avaliação da qualidade do material. A revisão se concentrou em procurar por estudos que descrevessem iniciativas de ensino de AS. A única questão considerada foi que o artigo deveria incluir uma descrição da prática de ensino, pois esta faz parte dos dados a serem extraídos. Foi considerado que as fontes dos documentos eram confiáveis, e que os textos tinham passado por revisões externas que serviram de filtragem para que os mesmos tenham qualidade suficiente para contribuir com a *quasi* revisão sistemática.

2.6.2. Condução da Revisão

2.6.2.1. Análise dos Documentos Recuperados

A ferramenta JabRef version 2.4.2 (JABREF, 2009) foi o gerenciador de referências utilizado para manipular as publicações recuperadas pelas máquinas de busca.

Durante a etapa de Condução da Revisão, as fontes foram selecionadas, os estudos primários, identificados, selecionados e avaliados de acordo com os critérios de inclusão e de exclusão e de qualidade estabelecidos durante o protocolo da revisão (MAFRA e TRAVASSOS, 2006).

As *strings* de busca preparadas foram executadas nas respectivas máquinas de busca das editoras selecionadas, como fontes no protocolo (IEEE, Elsevier, ACM, Springer, Compendex e Scopus). No entanto, algumas máquinas de busca apresentaram limitações que impediram uma correta execução das *strings*. Por esta razão, a máquina de busca da biblioteca digital da ACM foi descartada, pois impossibilitou a execução da seleção conforme o protocolo definido.

2.6.2.2. Resultado das Buscas nas Bibliotecas Digitais

A Tabela 2.1 expõe a quantidade de referências recuperadas de acordo com as máquinas de busca utilizadas.

2.6.2.3. Análise dos Documentos Recuperados

Numa primeira avaliação superficial, foi feita a exclusão das referências sem disponibilidade de acesso pela *Web* e dos artigos repetidos acessados por máquinas de busca diferentes. A nova situação quantitativa resultou em 479 artigos. Posteriormente, em uma avaliação mais apurada e detalhada, foram selecionados os documentos candidatos a fazer parte da *quasi* revisão sistemática. Foram excluídas as

referências que nitidamente tratavam de outros assuntos não pertinentes à pesquisa. Finalmente, a seleção quantitativa ficou em 28 artigos selecionados.

Tabela 2.1. Artigos retornados nas respectivas máquinas de busca

Maquinas de busca	Artigos retornados
Compendex	436
Elsevier	21
IEEE	124
Scopus	430
Spring	12
Total (*)	**479**

(*) O total de artigos retornados exclui os artigos repetidos em mais de uma máquina de busca

2.6.2.4. Extração da Informação

Após a seleção dos estudos, os dados dos mesmos foram extraídos e sintetizados para serem, finalmente, publicados durante a etapa de Publicação dos Resultados (MAFRA e TRAVASSOS, 2006).

2.6.3. Publicação da Revisão

2.6.3.1. Categorização das Iniciativas Encontradas

Posteriormente à execução da seleção dos resultados da *quasi* revisão sistemática, as iniciativas encontradas foram classificadas de acordo com alguns critérios para facilitar a sua análise. Os critérios propostos foram:

- Curso (quantidade): apontava se a iniciativa proposta era apresentada no formato de um ou mais cursos. 43% dos artigos descreveram experiência de ensino no formato de curso.
- Projeto em larga escala (sim/não): informava se o enfoque do ensino era utilizar projetos em larga escala ou não. 25% das iniciativas deram destaque à abrangência de sistemas complexos.
- *Design Pattern* (sim/não): informava se era utilizado *Design Pattern* (COPLIEN, 1995) no curso. 36% das iniciativas descreveram vantagens na utilização de *Design Pattern.*
- Estudo ativo (tipo de estudo ativo): apontava o enfoque dado às atividades dos alunos no processo de assimilação do conhecimento e habilidades, como a conversação dirigida, a discussão, o estudo dirigido individual e em grupo, os exercícios etc. (LIBÂNEO, 1990). 29% das iniciativas enfatizaram

a importância de estudos ativos com destaque para maior inserção do instrutor, colaboração, aprendizado cooperativo, mudança de papéis, habilidades sociais, trabalho em grupo, atrativo ao aluno e comunicação.

- Abordagem utilizada (tipo da abordagem): mostrava a abordagem utilizada para o ensino de AS, que pode ser uma ferramenta, um projeto etc. As abordagens retornadas foram: ferramenta, método de ensino, avaliação de software, protótipo, método de análise baseado em cenários, arquitetura de software Web, análise global, toolkit, experiência de uma companhia, editor de arquitetura de software.

2.6.3.2. Discussão dos Resultados

Com este levantamento sistemático, percebeu-se que a academia já está se preparando para entrar em sintonia com as necessidades do mercado de trabalho, oferecendo cursos com projetos em larga escala. Apesar de ser predominante a utilização da sala de aula com o auxílio de material teórico e alguma prática. Iniciativas como as de (KAZMAN *et al.,* 1996), (DIKEL *et al*, 1997), (SCHAUER e KELLER, 1998), (HOFMEISTER *et al.*, 2005), (WANG *et al.*, 2007), (GRISHAM *et al.*, 2007) e (MAENNISTOE *et al.*, 2008) estão em sintonia com as idéias de CONN (2002), preparando os futuros profissionais de Engenharia de Software para o mercado de trabalho. Alguns artigos utilizam o recurso de casos de estudo para ensinar requisitos não funcionais em projetos reais.

Outro destaque foi que muitas iniciativas (36%) sugerem a utilização de Padrões de Projeto (*Design Pattern*), com o objetivo de atingir qualidade de software em diferentes níveis de abstração, especialmente nos quesitos reutilização, modularidade e flexibilidade. Os artigos explicam que Padrões de Projeto têm recebido muita atenção, recentemente, e é um tópico importante que tem sido usado de forma eficiente no ensino. Os artigos mencionam que os alunos têm problemas para aplicar o conhecimento teórico em projetos concretos e adotando a abordagem de utilizar Padrões de Projeto em projetos contínuos, diminui este problema. Os artigos que deram destaque aos Padrões de Projeto foram (SCHAUER e KELLER, 1998), (NAVEDA, 1999), (AL-TAHAT *et al*., 2001), (LAGO e VAN VLIET, 2005), (WANG *et al.,* 2007), (DENZLER e GRUNTZ, 2008) e (GAST, 2008).

Alguns artigos compartilham a preocupação da monitoração mais presente por parte do instrutor ((MCGREGOR *et al.*, 2007), (DONGSUN *et al.,* 2008)). E outros voltados para o incentivo à comunicação dentro da equipe e o aprendizado cooperativo. Muitos deles utilizaram recursos que incentivam a mudança de papéis

dentro do grupo ((CHENOWETH *et al.*, 2007), (CREIGHTON e SINGER, 2008), (DONGSUN *et al.*, 2008), (MAENNISTOE *et al.*, 2008)) e uma abordagem atrativa ao aluno (DONGSUN *et al.*, 2008). Segundo HUANG e DISTANTE (2006), os estudantes precisam aprender tanto aspectos técnicos, quanto aqueles não-técnicos no desenvolvimento de sistemas de software e os artigos selecionados enfocam a importância destes estudos ativos.

CHENOWETH *et al.* (2007) descrevem um curso de AS que prioriza a mudança de papéis dentro das equipes: clientes, arquitetos e desenvolvedores. Este curso mostra os benefícios do aprendizado cooperativo enquanto os alunos são expostos a estes três tipos de perspectivas diferentes. O aprendizado cooperativo é um método próprio para dar este tipo de experiência porque suporta um tipo de trabalho em grupo empregado por arquitetos de software e projetistas. Este método enfatiza competências de grupos pequenos, a prática da comunicação face-a-face e reconhecer a responsabilidade individual para o sucesso do grupo. CREIGHTON e SINGER (2008) confirmam esta abordagem e acrescentam a necessidade de aumentar a percepção das forças e fraquezas individuais.

O artigo produzido por DIKEL *et al.* (1997) descreve os princípios críticos de sucesso de AS, baseado em seus estudos: focar na simplificação, minimização e clareza; adaptar a arquitetura para as necessidades de futuros clientes, tecnologia, competição e objetivos de negócio; estabelecer um ritmo de arquitetura consistente e difundida e lançamento de produto que ajudem a coordenar as ações e expectativas de todas as partes; parceria com os *stakeholders*; manter uma visão da arquitetura clara; e fazer gerenciamento de riscos e oportunidades.

Num ciclo de aprendizado desta disciplina, os estudantes devem criar uma AS, discutir sua solução com outros estudantes e podem descobrir que a sua solução ainda não esteja satisfatória. Segundo BOER *et al.* (2009), isto estimula os alunos a refletirem sobre o *feedback* obtido e encontrar alternativas.

As iniciativas identificadas utilizavam abordagens que ensinam a partir de métodos novos de ensino e de análise de AS, do desenvolvimento de sistemas, de editores de AS, de ferramentas desenvolvidas, da utilização de livros técnicos e de metodologias diversas. Cada relato contribui com as lições aprendidas com a utilização daquela ferramenta, metodologia ou estratégia.

BUCCI *et al.* (1998) seguem a seguinte posição: “É possível ensinar questões de nível arquitetural o mais cedo possível nos cursos de Ciência da Computação. Mas, para o sucesso desta abordagem, devem estar disponíveis ferramentas apropriadas

para ajudar os alunos a construírem os seus modelos". Para isso, eles desejam construir um editor que ajude os estudantes nos seus modelos mentais iniciais que seja sofisticado o bastante para envolver noções de alto nível relacionadas à AS e não à linguagem de programação.

O artigo escrito por LEE (2003) descreve um curso de Arquitetura de Software Web com tempos de duração diferentes. Seu objetivo é ensinar tanto fundamentos quanto conhecimento prático, para atender as necessidades da indústria.

WANG e STAELHANE (2005) utilizam um método chamado "Análise *Post Mortem*" (PMA) para suscitar pontos fortes e fracos do projeto durante sua avaliação.

WANG e SCANNELL (2005) descrevem uma ferramenta de modelagem de confiabilidade que incorpora estilos arquiteturais, desenvolvida para facilitar a predição de qualidade pelos estudantes. Esta ferramenta fornece aos estudantes *feedbacks* instantâneos e diminui as curvas de aprendizado. Com uma interface gráfica fácil de usar, ela trabalha com os estilos: sequencial, paralelo, tolerância à falha e cliente/servidor.

O artigo de WANG e SOERENSEN (2006) apresenta o método "*writing as a tool for learning*". Este é um método simples, usado nas leituras, as quais são introduzidas pausas. Os alunos pensam no tópico e escrevem tudo que sabem sobre ele, compartilham informações com o grupo e, finalmente, comparam com o livro texto.

Como parte de um curso, é descrito em (VALLIESWARAN e MENEZES, 2007), o ArchKriti - *A Software Architecture Based Design and Evaluation Tool Suite*. Esta ferramenta dá suporte a passos importantes dentro do desenvolvimento baseado em arquitetura: requisitos dos *stakeholders*, projeto de arquitetura de um sistema, avaliação de arquitetura e documentação. Este é um projeto aberto que facilita a adição de novos estilos arquiteturais e visões.

A avaliação de AS está se tornando uma ferramenta muito importante durante os vários processos de decisão no desenvolvimento de software. Decisões estratégicas são tomadas baseadas nos resultados das avaliações de AS. As avaliações são também usadas para garantir que os objetivos de qualidade sejam atingidos. Para se tornar um avaliador de AS eficiente, é importante ganhar experiência em conduzi-las. Os artigos de (WANG e STAELHANE, 2005), (HOFMEISTER *et al*., 2005), (WANG *et al*., 2007), (SVAHNBERG e MAERTENSSON, 2007) e (BOER *et al.,* 2009) mostram experiências de cursos que dão destaque a avaliação de AS.

HOFMEISTER *et al*. (2005) utilizam a Análise Global, que segundo o mesmo, serve para guiar o processo de projeto arquitetural, para capturar o raciocínio e apoiar a rastreabilidade entre a fase de requisitos e a AS.

O *Architecture Expert* (ArchE) (MCGREGOR *et al.*, 2007) é uma ferramenta de software que serve como um assistente de arquiteto de software que ajuda a criar arquiteturas que tem níveis específicos de qualidades requeridas. O ArchE incorpora teoria de atributos de qualidade, técnicas para resolver modelos destes atributos associados a um projeto arquitetural para dar respostas para determinadas situações e a habilidade de usar projetos legados como entrada. Neste artigo, a ferramenta ArchE é descrita, produzindo arquiteturas e ensinando sobre AS.

Resumindo, os resultados representam um indicador da necessidade de aprofundar o entendimento da prática e de todas as competências ligadas à disciplina de AS, visando a formação de bons arquitetos. Foi observada, a importância da prática no processo de aprendizado da disciplina. Os especialistas demonstraram que ao ensinarmos somente métodos e técnicas, os conhecimentos e as competências dos estudantes tornam-se frágeis. E ressaltaram, também, a necessidade de ser oferecida, a oportunidade dos estudantes projetarem sistemas mais complexos antes de saírem da faculdade, atendendo à demanda da indústria.

As contribuições desta *quasi* revisão sistemática foram significativas para a proposta desta tese, extraindo requisitos como lidar com projetos maiores, utilização de mais prática aliada aos tópicos teóricos e a necessidade de investimento no trabalho em equipe, na discussão em grupo e no apoio do instrutor.

2.7. Considerações Finais

Nossa sociedade está cada vez mais dependente do software, com uma demanda grande pela sua qualidade. Esta sociedade, conectada via Internet, em busca de soluções ainda mais rápidas, requer melhores maneiras de produzir sistemas. À medida que sistemas maiores e mais complexos vão surgindo, a ES, assim como o seu ensino, torna-se vital. Estas são algumas das pressões sofridas pela educação de ES, que com o passar dos anos, tem tido sua comunidade mobilizada para atender estas demandas. Particularmente, a disciplina de AS assume um papel relevante neste cenário, incorporando a reutilização e suas vantagens e permitindo aos engenheiros de software tomar decisões sobre alternativas de projeto. Ao longo deste capítulo foram apresentadas algumas técnicas para lidar com estes sistemas complexos; foram relatados alguns problemas e propostas de ensino de ES, e ainda, citadas algumas ferramentas de apoio, que tentam tornar esse aprendizado

mais efetivo e interessante para os alunos. Ele apresentou, também, uma breve introdução sobre a disciplina de AS, com foco na modelagem em UML. E detalhou a primeira *quasi* Revisão Sistemática conduzida que investigou práticas de ensino da disciplina de AS.

Capítulo 3 – Visualização e Tecnologias 3D

“O pensamento é impossível sem imagens”

Aristóteles

3.1. Introdução

Segundo um velho provérbio chinês, uma imagem vale mais que mil palavras, assim, esta pesquisa de tese investiu na visualização dos modelos de sistemas de software com o principal objetivo de aumentar a percepção destes sistemas pelo usuário.

A visualização da informação oferece vários benefícios para a identificação e comunicação das propriedades da informação. Ela pode reduzir a sobrecarga cognitiva através da filtragem de informações desnecessárias e, ainda, melhorar a percepção dos usuários e exploração do conteúdo e estrutura do espaço de informação (HUOTARI, 2004).

GERSHON (1994) define a visualização da seguinte forma: “Visualização é mais que um método de computação. Visualização é um processo de transformar a informação numa forma visual, permitindo que indivíduos observem a informação. A exibição visual resultante permite que cientistas e engenheiros percebam características que estavam escondidas nos dados, todavia, são necessárias para a exploração e análise de dados”. Em relação à visualização de software, DIEHL (2007) diz que seu objetivo é ajudar a compreender os sistemas de software e melhorar a produtividade no processo de desenvolvimento de software. É mostrar quantidades abstratas e relações de uma forma natural e intuitiva, a fim de ajudar o usuário a entender melhor e obter *insights* sobre os dados. Segundo TEYSEYRE e CAMPO (2009), a visualização de software tem como principal objetivo melhorar, simplificar e esclarecer a representação mental que um engenheiro de software tem de um sistema de computador.

O capítulo está estruturado da seguinte forma: a Seção 3.1 apresentou a Introdução. A Seção 3.2 descreve alguns Recursos que uma ferramenta de visualização deve ter. A Seção 3.3 trata sobre a Visualização e a Compreensão de Modelos. A Seção 3.4 discorre sobre a Visualização 3D. A Seção 3.5 apresenta a Realidade Virtual e Aumentada como tecnologias 3D emergentes. A Seção 3.6 termina o capítulo com as Considerações Finais.

3.2. Recursos de Visualização

A utilização de técnicas de visualização de software pode auxiliar as atividades que envolvem o raciocínio humano, fornecendo uma melhor representação das informações obtidas.

Em SHNEIDERMAN (1996), são apresentadas sete tarefas ou recursos que uma ferramenta de visualização de informações deve ter:

a) **Visão geral**: a ferramenta deve fornecer um mapa geral claro e organizado.

b) **Busca**: ela deve fornecer busca utilizando palavras-chave ou índice, navegador ou um agente de busca.

c) ***Zoom* e filtro**: a ferramenta deve permitir *zoom* em itens de interesse e filtro sobre itens irrelevantes.

d) **Detalhes sobre demanda**: é importante gerenciar a granularidade, escondendo e mostrando detalhes, para que o contexto não seja perdido.

e) **Relacionar**: ela deve fornecer a visualização de relacionamentos entre os itens.

f) **Histórico**: é importante evitar que um usuário se perca no "hiperespaço". Assim, auxílios à navegação, tais como marcas, histórico e facilidades de rastreamento devem ser incluídos como recursos de suporte.

g) **Extração**: é a opção de salvar o estado atual da visualização de alguma forma. Depois de o usuário obter o item desejado, ele deve ser capaz de salvá-lo num formato qualquer.

ITOH *et al.* (2004) também sugeriram uma lista de características que uma técnica de visualização deve ter que complementa a lista de Shneiderman, conforme a seguir:

a) **Uso eficiente de espaços de exibição**: é útil colocar todos os itens de dados em um espaço limitado de exibição para fornecer uma boa visão. Critica-se os sistemas de árvores tradicionais, tais como o sistema de arquivos, que precisam de um espaço de exposição grande quando há muitos nós ou quando há uma hierarquia de profundidade.

b) **Sem sobreposições entre nós**: não deve-se permitir a sobreposição entre os nós, de modo a fornecer uma visão uniforme dos dados.

c) **Proporção de subespaços**: ao subdividir um espaço de exposição para representar partes dos dados fornecidos, é preferível a utilização de subespaços quadrados a subespaços finos, para que os usuários possam visualmente reconhecer estas partes. Portanto, os tamanhos dos subespaços devem ser considerados.

d) **Colocação flexível de nós de forma arbitrária**: sugere-se a unificação do tamanho dos ícones utilizados. E, além disso, que estes ícones sejam alterados pelos usuários. Isto torna possível enfatizar visualmente itens importantes de dados.

e) **Similaridade**: é desejável que os dados semelhantes sejam igualmente representados.

f) **Semântica da colocação:** é desejável que as posições dos itens de dados sejam calculadas de acordo com a semântica do usuário.

3.3. Visualização e a Compreensão de Modelos

Conforme será apresentado mais adiante, no Capítulo 5, esta pesquisa de tese sugere a utilização de uma grande quantidade de recursos de visualização, anteriormente mencionados, para apoiar a compreensão de modelos de software.

LANGE *et al.* (2007) dizem que a visualização pode ajudar na compreensão e desenvolvimento de um modelo. Quando um novo desenvolvedor se junta a uma equipe existente, ele precisa compreender o sistema antes que ele seja capaz de fazer suas próprias contribuições. A visualização ajuda a identificar classes principais, classes que implementam uma funcionalidade, relações entre as classes e interações complexas. Acrescentam, ainda, que a visualização pode apoiar a criação de modelos, que muitas vezes é um processo incremental e iterativo, que inclui muitas mudanças.

Esse mesmo autor também diz que a visualização deve prover informação que está diretamente presente no modelo ou identificar as propriedades associadas com os elementos do modelo. As informações internas são informações derivadas do modelo, como por exemplo, as métricas. E as informações externas são as de fora do modelo, como por exemplo, artefatos, tais como código fonte, os requisitos de documentos ou documentos de teste.

Há uma série de trabalhos de pesquisa e projetos que oferecem soluções promissoras como técnicas e aplicações de visualização, no entanto, segundo HUOTARI (2004), estas soluções raramente são comercializadas. Ele diz que isto deve-se à dificuldade de implementação de um suporte abrangente para visualização da informação.

HOUTARI (2004) identificou diversas áreas problemáticas relacionadas com a compreensão de um modelo de sistema de informação em geral e integração de diagramas e objetos dentro deles:

a) Necessidade de entender um único diagrama e como ele se relaciona com outros diagramas;

b) Necessidade de ter uma visão geral do diagrama;

c) Necessidade de pistas visuais para ajudar na identificação das inter-relações entre os diagramas;

d) Múltiplas visões muitas vezes levam a inconsistências entre as visões, ou omissões, ou até erros dos modelos de sistema de informação;

e) Necessidades de encontrar e mostrar componentes reutilizáveis. HOUTARI (2004) menciona que pode-se compartilhar ou reutilizar uma propriedade (por exemplo, um nome), um objeto (por exemplo, uma classe), um diagrama (por exemplo, um diagrama de classes), uma visão (por exemplo, uma visão estática), ou mesmo todo o modelo;

f) A grande quantidade de dados em um modelo de sistema de informação, que pode esconder os pontos de interesse essenciais. Assim, um sistema deve esconder e mostrar os detalhes dos modelos quando necessário e deve permitir *zoom in* e *out*; e

g) Problema de gerenciamento de diferentes dimensões. A dimensão vertical está relacionada com níveis de abstração. Numa abordagem *top-down*, significa ver o modelo a partir de requisitos através projeto arquitetural para a implementação (código). A dimensão horizontal lida com diferentes diagramas, mas no mesmo nível, por exemplo, diagrama de caso de uso e diagrama de classe.

3.4. Visualização 3D

Por muitos anos, a visualização de software no espaço 2D tem sido ativamente estudada, mas na última década, os pesquisadores começaram a explorar novas representações em 3D. TEYSEYRE e CAMPO (2009) apresentam uma visão geral das pesquisas atuais na área, descrevendo vários aspectos importantes, como representações visuais, problemas de interação, métodos de avaliação e ferramentas de desenvolvimento.

A visualização de software em 3D tem o potencial para ajudar o processo de manutenção e melhorar a compreensão do usuário de um sistema de software. A visualização pode ajudar a produzir uma imagem do software. Criando um objeto

(visual) físico que representa o sistema de software, engenheiros de software podem ganhar algum *insight* inicial em como ele está estruturado e de que ele é composto. Isso os ajuda a usar suas habilidades de percepção na investigação e compreensão do software, além de materializar/concretizar algo que é abstrato/lógico.

YOUNG e MUNRO (1998) destacam, ainda, algumas qualidades que são importantes considerar ao projetar uma visualização 3D:

a) **Navegação simples**: as visualizações devem ser projetadas com o usuário em mente, adicionando recurso para ajudá-los na navegação.

b) **Conteúdo com muita informação**: as visualizações devem apresentar o máximo de informações possíveis sem sobrecarregar o usuário.

c) **Baixa complexidade visual**: apesar da complexidade da informação apresentada, deve-se fazer um esforço para reduzir a complexidade visual das visualizações. A forma como as informações são mostrados graficamente sobre os vários componentes do sistema de software deve ser um aspecto que deve ser decidido com cuidado.

d) **Diferentes níveis de detalhes**: o nível de detalhe, o conteúdo de informação e o tipo de informação apresentado devem variar para atender os interesses dos usuários. A visualização deve apoiar sua mudança de interesse e proporcionar cada vez mais detalhes e informações enquanto o usuário se move em direção a um componente ou expressa interesse sobre um componente.

e) **Resistência à mudança**: pequenas adições ou alterações no conteúdo da informação da visualização ou mudanças nos interesses dos usuários não deveriam resultar em grandes diferenças na visualização. Por exemplo, um reposicionamento completo das representações pode fazer com que o usuário fique desorientado ou tenha que reaprender a estrutura do ambiente.

f) **Bom uso de metáforas visuais**: as metáforas introduzem conceitos visuais familiares ao usuário e proporcionam um bom ponto de partida para a obtenção de um entendimento da visualização.

g) **Interface de usuário amigável**: a interface do usuário deve ser flexível o suficiente para proporcionar uma navegação intuitiva e um controle, mas não deve desencorajar o usuário ou introduzir recursos a mais, desnecessários. Em muitos casos, um monitor de computador pode ser suficiente para utilizar a Realidade Virtual, sem que seja necessário investir numa CAVE, por exemplo. É importante que os

usuários possam navegar facilmente sem se perderem. Recursos como sinalização, pontos de referência, marcadores, mapas ajudam na interação.

h) **Integração com outras fontes de visualizações**: deve-se fornecer a possibilidade de um ponto de vista diferente sobre as informações que estão sendo apresentadas. É desejável correlacionar informações entre diferentes possibilidades de visualizações. Esta correlação deve ser compreensível para que a visualização tenha alguma utilidade.

i) **Bom uso da interação**: Ao permitir a interação do usuário com a informação, o sistema de visualização proporciona mecanismos para a obtenção de mais informações e também ajuda a manter o interesse do usuário. Considerações devem ser feitas a respeito de como o usuário irá interagir com a visualização e como eles podem manipular o seu conteúdo. Esta interação não pode ser mais complicada do que navegar pela visualização.

j) **Automação adequada**: um bom nível de automação é necessário para fazer a visualizações valer à pena. É necessário determinar o quanto da visualização é gerada automaticamente e quanto controle o usuário tem sobre o processo ou o resultado. Apoiar a investigação do usuário nas áreas do sistema de software pode ser mais útil na sua compreensão do que automatizar totalmente o processo.

3.4.1. Por que 3D?

Segundo ALFERT e FRONK (2000), a grande vantagem dos desenhos 2D é que eles requerem apenas papel e lápis. No entanto, de acordo com alguns aspectos, pode-se mostrar onde o 3D supera a visualização 2D:

a) **Efeito de profundidade**: segundo ALFERT e FRONK (2000), é mais adequado usar o efeito de profundidade em um espaço 3D. Com o ambiente 3D é possível fazer um layout dos objetos de forma mais consistente do que pode ser feito em um único plano (FEIJS e JONG,1998);

b) **Ordem no espaço**: ALFERT e FRONK (2000) dizem que a ordem no espaço pode ser expresso e compreendido muito melhor do que em 2D;

c) **Animação**: apesar de ser bastante utilizado em 2D, ALFERT e FRONK (2000) ressaltam sua importância, devido aos efeitos e destaques no ambiente 3D;

d) **Semântica:** Modelos de software permitem a visualização de semântica mais rica do que os dos modelos 2D ((GIL e KENT, 1998), (SANATNAMA e BRAHIMI, 2010)). A semântica que transporta aumenta, consideravelmente, o poder expressivo

das notações gráficas, sem recorrer ao emaranhado de anotações textuais (GIL e KENT, 1998).

e) **Percepção**: em relação à visualização 2D, a 3D reduz a sobrecarga cognitiva e melhora a percepção dos usuários. Seu objetivo é promover *insight*, segundo HUOTARI (2004).

f) **Visão numa tela**: Na visão 2D, apenas um diagrama pode ser visto por vez. Segundo BERGHAMMER e FRONK (2003), a visualização de grandes sistemas levanta muitos problemas devido às limitações da tela no espaço e resolução. Na visão 3D, todos os diagramas são exibidos em uma única cena, sem a presença de *scroll*.

g) **Relacionamentos**: Visualizações tridimensionais são importantes para certas aplicações, especialmente para estudar as relações entre vários modelos ou diagramas (PILGRIM e DUSKE, 2008).

h) **Disponibilidade de hardware:** A visualização tridimensional de informações é um campo em rápida evolução na Ciência da Computação. Isto deve-se à crescente disponibilidade de aceleração de hardware de baixo custo e o desenvolvimento de interfaces de programação tais como VRML (STANEK, 1996), entre outros. A visualização de software 3D está se tornando cada vez mais viável e atraente para muitas aplicações ((RADFELDER e GOGOLLA, 2000), (PILGRIM *et al.*, 2009)).

i) **Estereoscopia:** projeções estereoscópicas expõem mais informações do que um diagrama simples planar (GIL e KENT, 1998).

j) **Apelo visual:** Segudo FEIJS e JONG (1998), projetos em 3D possuem um apelo visual e podem ser mais atraentes, como exemplo, citam a apresentação de um projeto para clientes potenciais.

3.5. Realidade Virtual e Aumentada

Em algum momento, num programa de televisão, num noticiário ou num filme de ficção científica, o espectador deve ter visto uma pessoa de capacete experimentando sensações de um mundo simulado artificialmente pelo computador. Esta tecnologia que tenta recriar ao máximo a sensação da realidade para um indivíduo através de interações em tempo real com o uso de técnicas e de equipamentos computacionais é chamada de Realidade Virtual (RV). A Realidade Aumentada (RA), por sua vez, tem como base o mundo real, colocando sobre ele informações ou objetos virtuais. Ela pode ser reconhecida em jogos de futebol exibidos pela televisão, quando uma seta ou um círculo marca a distância entre a bola e o gol, ou entre a bola e os outros

jogadores numa cobrança de falta. Ela também é utilizada nos sistemas de posicionamento global, ou GPS (*Global Positioning System*), utilizados nos carros, dando sugestões de rotas de um ponto a outro na cidade, indicações de nomes de ruas, acessos alternativos, distância até uma curva, velocidade etc.

A RV transporta o usuário para o ambiente virtual, diferentemente da RA, que mantém o usuário no seu ambiente físico e transporta o ambiente virtual para o espaço do usuário, permitindo a interação com o mundo virtual, de maneira mais natural e sem necessidade de treinamento ou adaptação (KIRNER e TORI, 2006).

A **Realidade Virtual** (*Virtual Reality*) pode ser explicada como uma simulação em terceira dimensão do mundo real ou de um mundo imaginário qualquer. Esta simulação é mais imersiva, ou seja, através de dispositivos, o usuário tem a impressão de estar em outro mundo. Os usuários têm uma sensação de presença física e podem manipular os objetos 3D. Aplicações de RV propiciam a visualização, movimentação e interação do usuário, em tempo real, em ambientes tridimensionais gerados por computador. Além da visão, outras sensações como o tato e a audição podem ser usadas para enriquecer a experiência do usuário. Enquanto imerso, o usuário não pode ver o mundo real em torno dele (MILGRAM e KISHINO, 1994). A RV ganhou muita popularidade devido a jogos e aplicações como o *Second Life* (SECOND LIFE, 2011).

A **Realidade Aumentada** (*Augmented Reality*) é uma tecnologia nova e emergente, uma evolução da RV. Segundo AZUMA *et al*. (2001), ela suplementa o mundo real com objetos virtuais que parecem coexistir no mesmo espaço do mundo real, através de algum dispositivo tecnológico. A RA pode adicionar gráficos, sons, tato e cheiro ao mundo natural. Como o nome já diz, a RA aumenta o ambiente para o usuário, sendo capaz de ampliar sua percepção e a sua interação com o mundo real. Ela enriquece o ambiente real com informações virtuais que ajudarão no desempenho de suas tarefas. AZUMA *et al*. (2001) também definiram algumas características de um sistema para que o mesmo seja considerado de RA: “combina objetos reais e virtuais num ambiente real”; “opera interativamente, e em tempo real”, e “registra (alinha) objetos reais e virtuais, uns com os outros”.

Nos últimos anos o conceito de RV tem recebido considerável publicidade. Mas teve sua origem com os trabalhos de Ivan Sutherland, que na década de 1960 criou o capacete de visão estereoscópica, ou *head-mounted* display (HMD) (Figura 3.1) e fez diversos experimentos de imersão.

A RV utiliza equipamentos especiais, como luvas de dados, capacetes e óculos estereoscópicos, para oferecer ao usuário a possibilidade de navegar por um ambiente virtual tridimensional de maneira bem próxima a um ambiente real (TORI *et al.*, 2006).

Outra solução de dispositivo é utilizar a própria mão. Estudos incentivam a utilização deste tipo de interação (interação via gestos) muito mais intuitivo nos ambientes virtuais. E ainda, os usuários com necessidades especiais poderiam se beneficiar com este tipo de interação. A mão é gravada por uma câmera de vídeo e digitalizada por um hardware especial. Um software de reconhecimento de imagem usa os dados da imagem para determinar a posição atual e a forma ("gesto") da mão (PAVLOVIC *et al.*, 1997).

Figura 3.1: HMD baseado em vídeo (RODRIGUES e WERNER, 2008)

Um dos equipamentos mais sofisticados para implementação de RV é a chamada CAVE, um espaço cúbico no qual o participante entra e se movimenta livremente enquanto é envolvido por projeções tridimensionais nas paredes e, em algumas implementações mais sofisticadas, no chão e no teto. A RV imersiva é aquela na qual o usuário é totalmente isolado, ao menos visual e auditivamente, do mundo real (TORI, 2010). A RV não imersiva, em sua configuração mais convencional exige apenas um monitor, como equipamento necessário.

A estereoscopia acrescenta a dimensão de profundidade às telas de projeção dos mundos virtuais, tornando-os mais próximos e realistas da forma que os usuários os vêem no mundo real. A visualização de imagens 3D é obtida através de duas imagens diferentes que são geradas, uma para o olho esquerdo e outra para o olho

direito. O processo pode ser simulado através de duas câmeras organizadas com a mesma distância interocular dos olhos humanos (que varia em torno de 6,5 cm). Logo, colocando-se as câmeras separadas uma da outra com base nessa distância, simula-se o sistema visual humano. Há várias técnicas de estereoscopia, entre elas, a Técnica de Polarização da Luz. Nesta técnica, dois projetores comuns são usados para projetar imagens diferentes para os olhos esquerdo e direito. Filtros de luz com polarização oposta são colocados sobre a lente de cada projetor. Cada usuário deve usar óculos com as lentes polarizadas de forma correspondente, de modo que cada olho enxergará somente a imagem correta. Uma tela de projeção prateada (ou aluminizada) é necessária para preservar a polarização da luz (SISCOUTTO *et al.*, 2006).

A RV e a RA são tecnologias bastante promissoras, mas que também apresentam muitos desafios e espaços para aprimoramento, principalmente no que se refere à interação e ao *design* de interfaces. A seguir, são discutidos alguns aspectos relacionados à sua participação na Educação, que têm merecido atenção especial dos pesquisadores.

3.5.1. Realidade Virtual e Aumentada na Educação

A RV e RA contribuem de maneira significativa em atividades de aprendizagem, como processo de exploração, descoberta, observação e construção de uma nova visão do conhecimento, oferecendo ao aprendiz a oportunidade de melhor compreensão do objeto de estudo. Essas tecnologias, portanto, têm potencial de colaborar no processo cognitivo do aprendiz, proporcionando não apenas a teoria, mas também a experimentação prática do conteúdo em questão (CARDOSO e LAMOUNIER, 2008).

A introdução da RA na matemática, por exemplo, pode eliminar uma das principais dificuldades do aluno: visualizar um problema complexo de geometria. Ela fornece a professores e estudantes um método intuitivo e colaborativo de aprender. Os alunos vêem os objetos tridimensionais, sem precisar imaginá-los ou desenhá-los numa folha de papel (LOPES, 2005). Quando envolvido e totalmente imerso no ambiente virtual, o usuário desenvolve um comportamento natural e intuitivo, buscando agir como agiria no mundo real e através da interação, receber resposta ideal para suas ações. No entanto, deve-se levar em conta, também, que o uso por tempo prolongado de seus dispositivos pode causar fadiga e efeitos colaterais (TORI, 2010).

Aqui, a RV e RA não são tratadas apenas como "mais uma ferramenta" para melhorar a aprendizagem, e sim como um poderoso instrumento de aprendizagem. Segundo BELL e FOGLERL (1995), a principal vantagem de utilizar a RV é a sua capacidade de visualizar situações e conceitos que não poderiam ser vistos de outra forma, e ainda imergir o aluno dentro dessa visualização. Segundo ele, o interesse e o entusiasmo do aluno são também evidentes benefícios da RV. De acordo com TORI (2010), as tecnologias de RV e RA costumam chamar a atenção dos alunos e motivá-los, mas deve-se considerar a curiosidade pela novidade.

A potencialidade destas novas tecnologias está exatamente no fato de permitir a exploração de alguns ambientes, processos ou objetos, não através de livros, fotos, filmes ou aulas, mas através da manipulação e análise virtual do próprio alvo do estudo.

A introdução da RV, e consequentemente da RA, na educação demonstra um novo paradigma que relata uma educação de forma dinâmica, criativa, colocando o aluno no centro dos processos de aprendizagem e buscando uma formação de um ser crítico, independente e construtor de seu conhecimento. A grande preocupação é com o investimento em hardware e software, mas hoje em dia é possível montar uma boa plataforma com custo, relativamente, baixo. Há poucos anos, os equipamentos e software de RV eram acessíveis apenas a grandes empresas, hoje é um recurso bastante viável de ser aplicado em larga escala em atividades educacionais (TORI, 2010).

3.6. Considerações Finais

Avanços atuais na visualização estão fornecendo novas maneiras de olhar para o software. Neste capítulo foram descritos alguns recursos que uma ferramenta de visualização deve ter e discorreu sobre a Visualização 3D. Ele pretendeu mostrar o potencial da aplicação das tecnologias interativas 3D na Educação, principalmente no apoio à compreensão de modelos. Ele destaca a Realidade Virtual e a Realidade Aumentada como uma possibilidade de promover o enriquecimento da prática educacional, facilitando o ensino e promovendo o aprendizado por meio de representações dimensionais de objetos.

Capítulo 4 - Trabalhos Relacionados

4.1. Introdução

Há muitas áreas de pesquisa que têm interesse na visualização em 3D, como a visualização de software, interação homem-computador, computação gráfica, realidade virtual, engenharia de software, entre outras. Contudo, um artigo de TEYSEYRE e CAMPO (2009), relativamente recente, fornece uma visão geral de estudos realizados na área de visualização de software 3D, contendo apenas 194 registros.

Este capítulo restringe e enfoca a questão da visualização 3D de diagramas UML. Ele descreve trabalhos relacionados de autores presentes na literatura, após a condução de uma Segunda *quasi* Revisão Sistemática, realizada em julho de 2011. Neste sentido, ele apresenta como a abordagem VisAr3D se difere destas outras propostas existentes.

Este capítulo descreve detalhes sobre a condução desta revisão, assim ele é organizado da seguinte forma: a Seção 4.2 exibe o objetivo da Segunda *quasi* Revisão Sistemática; na Seção 4.3, é apresentado todo o Processo para Realização da Revisão; e o capítulo é finalizado com as Considerações Finais, na Seção 4.4.

4.2. Objetivo da Segunda *quasi* Revisão Sistemática

A abordagem VisAr3D propõe uma solução diferenciada para o apoio à compreensão de diagramas UML em sistemas com muitos elementos de modelagem, ao combinar recursos de tecnologias emergentes de visualização 3D, como a Realidade Virtual e a Realidade Aumentada. O Objetivo desta revisão é relacionar os trabalhos existentes na literatura que se preocupam com estas mesmas questões e destacar as semelhanças e diferenças entre eles.

4.3. Processo para Realização da Revisão

Conforme BIOLCHINI *et al.* (2005), esta *quasi* Revisão Sistemática foi conduzida em três etapas: Planejamento da Revisão, Condução da Revisão e Publicação dos Resultados.

4.3.1. Planejamento da Revisão

4.3.1.1. Questões de Pesquisa e Estrutura PICO

Objetivo:
Identificar as iniciativas que utilizam a visualização 3D de diagramas UML.

Questão de pesquisa:
Quais iniciativas realizadas utilizam a visualização 3D de diagramas UML?

População (P):
Publicação tendo em vista a visualização 3D de diagramas UML

Intervenção (I):

Visualização 3D de diagramas UML

Comparação (C):
Não houve comparação

Resultados (O):
Iniciativas identificadas

4.3.1.2. Fontes para Busca, Termos e Sinônimos

Fonte:
A pesquisa foi feita na base de dados eletrônica Scopus. O critério de escolha foi por esta ser considerada, pelo pesquisador, como a mais utilizada e mais estável.

Scopus: <http://www.scopus.com>

Termos e sinônimos utilizados na pesquisa:

UML diagram – *UML diagrams, UML model, UML models, UML, UML modeling, software architecture, graphical notation, software systems modelling;*

3D view – *3D visualization, third dimension, three dimensional, three dimensions;*

Initiative – *experience, best practices, benefit, guideline, tool, method, technique, project, approach.*

4.3.1.3. Strings de Busca

A string de busca padrão utilizada nesta *quasi* Revisão Sistemática foi a seguinte:

P:

(“UML diagram” **OR** “UML diagrams” **OR** “UML model” **OR** “UML models” **OR** UML **OR** “UML modeling” **OR** “software architecture” **OR** “graphical notation” **OR** “software systems modelling”)

I:

(“3D view” **OR** “3D visualization” **OR** 3D **OR** “third dimension” **OR** “three dimensional” **OR** “three dimensions”)

O:

(initiative **OR** experience **OR** “best practices” **OR** benefit **OR** guideline **OR** tool **OR** method **OR** technique **OR** project **OR** approach)

No formato do Scopus ((P) AND (I) AND (C) AND (O)):

TITLE-ABS-KEY((("UML diagram" OR "UML diagrams" OR "UML model" OR "UML models" OR UML OR "UML modeling" OR "software architecture" OR "graphical notation" OR "software systems modelling") AND ("3D view" OR "3D visualization" OR 3D OR "third dimension" OR "three dimensional" OR "three dimensions") AND (initiative OR iniciatives OR experience OR experiences OR "best practices" OR benefit OR benefits OR guideline OR guidelines OR tool OR tools OR method OR methods OR technique OR techniques OR project OR approach)))

4.3.1.4. Critérios para Inclusão e Exclusão de Estudos

Os critérios definidos para inclusão e exclusão de estudo foram:

- Os documentos devem estar disponíveis na Web;
- Os artigos que apresentavam palavras da *string* de busca no seu título ou no seu resumo;
- Os artigos escritos em inglês.

Durante a revisão inicial da literatura foi identificada a seguinte tese de doutorado, que serviu como artigo de controle para refinar a string de busca e garantir que todos os artigos relevantes para este estudo fossem recuperados:

MCINTOSH, P., 2009, “X3D-UML : user-centred design, implementation and evaluation of 3D UML using X3D”, PhD. Thesis, RMIT University, Australia.

4.3.1.5. Processo de Seleção dos Estudos

1. O pesquisador executa a busca na fonte selecionada, utilizando a string de busca elaborada.

2. Os artigos retornados pela busca são inseridos na ferramenta JabRef versão 2.6 (JABREF, 2011).
3. O conjunto de artigos é selecionado a partir da verificação dos critérios de inclusão e exclusão. Esta verificação se dá pela leitura do resumo e do título do artigo.
4. Os artigos incluídos e excluídos são documentados no Formulário de Seleção de Estudos.

O processo de seleção dos estudos foi revisado por quatro pesquisadores, dentre eles a autora desta tese.

4.3.1.6. Avaliação da Qualidade dos Estudos

Será considerado que as fontes dos documentos são confiáveis, e que os textos tenham passado por revisões externas que serviram de filtragem para que tenham qualidade suficiente para contribuir com a *quasi* revisão sistemática. Procedimentos explícitos para avaliação da qualidade do material não foram preparados. A revisão se concentrou em procurar por estudos que descrevam iniciativas de visualização 3D de diagramas UML. A única questão considerada é que o artigo deve incluir uma descrição da abordagem utilizada, que constará como dados a serem extraídos.

4.3.1.7. Estratégia de Extração de Informações

Para cada estudo selecionado, após a execução do processo de seleção, são extraídas as seguintes informações:

- Autor
- Título do documento
- Ano de publicação
- Nome da Conferência ou Revista
- DOI/URL

A Figura 4.1 ilustra parte do Formulário de Seleção de Estudos com as informações extraídas.

4.3.2. Condução da Revisão

4.3.2.1. Análise dos Documentos Recuperados

A ferramenta JabRef versão 2.6 (JABREF, 2011) foi o gerenciador de referências utilizado para manipular as publicações recuperadas pela máquina de busca. O JabRef identificou repetições, categorizou referências, priorizou leituras, recuperou o

documento completo, disponibilizou campos para comentários e revisões etc. Novos campos foram customizados nesta ferramenta, bem como a elaboração de layouts específicos de exportação de relatórios.

Durante a etapa de Condução da Revisão, as fontes para a *quasi* Revisão Sistemática são selecionadas, os estudos primários são identificados, selecionados e avaliados de acordo com os critérios de inclusão e de exclusão estabelecidos durante o protocolo da revisão (MAFRA e TRAVASSOS, 2006).

QuickSearch: clear Number of matching entries: 154/154. Search Settings

Author	Title	Year	Journal/Proceedings	Reftype	DOI/URL
Allani-Bouhoula, N.a, P.J.-P.	Architectural rules for three-dimensional reconstruction [Abstract] [Review] [BibTeX]	2008	Proceedings of the 10th IASTED International Conference on Computer Graphics and Imaging, CGIM 2008, pp. 237-242	conference	URL
Badesa, F.J.a b, P.M.S.J. b.A.J. b.S.J.C.P.	Efficient collision algorithm for the 3D haptic interaction with solid organs in medical environments [Abstract] [Review] [BibTeX]	2009	Proceedings of the 2nd International Conferences on Advances in Computer-Human Interactions, ACHI 2009, pp. 187-192	conference	DOI URL
Bagby, M., R.R.S.B.U.H.J.J.H.J.F.	DiRT - Dust in Real-time: The specification process [Abstract] [Review] [BibTeX]	2005	Proceedings of the 2005 International Conference on Software Engineering Research and Practice, SERP'05 Vol. 2, pp. 807-813	conference	URL
Bai, X., S.C.	Research on modeling of SoS and large system development process [Abstract] [Review] [BibTeX]	2009	6th International Conference on Information Technology and Applications, ICITA 2009, pp. 146-150	conference	URL
Balogh, E.a f, D.A.S.R.K.A.V.A.M.C.C.J.F.G. c.	Visualization, planning, and monitoring software for MRI-guided prostate intervention robot [Abstract] [Review] [BibTeX]	2004	Lecture Notes in Computer Science Vol. 3217(1 PART 2), pp. 73-80	conference	URL
Belharet, K., F.D.F.A.	3D MRI-based predictive control of a ferromagnetic microrobot navigating in blood vessels [Abstract] [Review] [BibTeX]	2010	2010 3rd IEEE RAS and EMBS International Conference on Biomedical Robotics and Biomechatronics, BioRob 2010, pp. 808-813	conference	DOI URL
Berghammer, R.a, F.A.	Applying relational algebra in 3D graphical software design [Abstract] [Review] [BibTeX]	2004	Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) Vol. 3051, pp. 62-73	article	URL
Bier, H.a, D.J.A.V.D.H.G.B.N.H.M.V.M.H.	Prototypes for automated architectural 3D-layout [Abstract] [Review] [BibTeX]	2008	Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) Vol. 4820 LNCS, pp. 203-214	article	DOI URL

Figura 4.1 - Formulário de Seleção de Estudo

4.3.2.2. Resultado das Buscas na Biblioteca Digital

A *string* de busca preparada foi executada na máquina de busca Scopus e foi revisada mais de uma vez. 154 artigos foram retornados nesta máquina de busca.

4.3.2.3. Análise dos Documentos Recuperados

Numa primeira avaliação superficial, foi feita a exclusão das referências sem disponibilidade de acesso pela Web, ou que disponibilizavam o texto numa língua diferente do inglês (muitos artigos descartados foram redigidos em chinês). Foram

excluídas as referências que nitidamente tratavam de outros assuntos não pertinentes à pesquisa.

Foram incluídos quatro artigos não selecionados pela máquina de busca eletrônica, mas que são relevantes para o presente estudo. Ou seja, eles são artigos não retornados pelo Scopus, mas que atendem ao objetivo deste estudo. Eles foram recuperados a partir das referências bibliográficas de trabalhos relacionados do artigo de controle. Dois deles não constam da base de dados utilizada, que é o caso dos artigos DWYER (2001) e THADEN e STEIMANN (2003). Apesar da pertinência do conteúdo dos outros dois artigos, eles não foram recuperados pela *string* de busca utilizada. As palavras contidas, tanto no seu título, quanto no seu resumo, não contribuíram para que eles fossem selecionados. Por este motivo e por sua importância neste estudo, os artigos LANGE *et al.* (2007) e FEIJS e JONG (1998) foram adicionados.

Finalmente, a seleção quantitativa ficou em 14 artigos selecionados. Considerando que quatro destes artigos mencionam o mesmo projeto, esta revisão identificou 12 iniciativas. Os artigos PILGRIM e DUSKE (2008) e PILGRIM *et al.* (2009) representam a mesma abordagem “GEF3D - Editor de diagramas 3D” e os artigos MCINTOSH *et al.* (2005) e MCINTOSH *et al.* (2008) representam a mesma abordagem “X3D-UML – visualizador UML 3D”.

4.3.2.4. Extração da Informação

Após a seleção dos estudos, os dados são extraídos e sintetizados para serem finalmente publicados durante a etapa de Publicação dos Resultados (MAFRA e TRAVASSOS, 2006).

Na Tabela 4.1 são listados os artigos selecionados, com uma breve descrição da sua proposta e indicação do número da Figura correspondente (na Tabela representada por “Fig.”) a um exemplo de tela da sua ferramenta. O artigo RODRIGUES (2010), que faz referência a presente pesquisa de tese de doutorado, foi retornado pela máquina de busca, mas não foi listada nesta Tabela.

Tabela 4.1 – Resultados Selecionados

Referência	Iniciativas identificadas	Fig.
(IRANI e WARE, 2000)	Baseado na teoria de percepção do objeto estrutural, o artigo propõe uma nova maneira de apresentar a informação, através de Diagramas Geon 3D, em comparação com a UML 2D.	2
(PILGRIM e DUSKE, 2008) e (PILGRIM *et al*., 2009)	O artigo apresenta o GEF3D editor 3D, baseado no *framework Eclipse Graphical Editing Framework* (GEF).	3
(DUGERDIL e ALAM, 2008)	O artigo propõe a representação de uma cidade moderna no espaço 3D, onde as classes e arquivos são prédios e os seus relacionamentos são canos entre os prédios, chamado de EvoSpace.	4
(BERGHAMMER e FRONK, 2003)	Apresenta a ferramenta RelView, que checa a validade de diagramas 3D sintaticamente, utilizando álgebra relacional.	5
(RADFELDER e GOGOLLA, 2000)	Apresenta uma notação que estende UML para a terceira dimensão, colocando aspectos estáticos e dinâmicos em uma única exibição.	6
(GIL e KENT, 1998)	O artigo apresenta uma série de notações gráficas 3D, que demonstram um uso efetivo da terceira dimensão na modelagem.	7
(MCINTOSH *et al*., 2005) e (MCINTOSH *et al*., 2008)	A ferramenta X3D-UML utiliza a linguagem X3D como apoio à visualização UML dentro de um ambiente 3D.	8
(LANGE *et al.*, 2007)	O artigo propõe o *framework* Metricview que desenvolve visões e novas técnicas de visualização em 3D, que apóiam a modelagem orientada a tarefas.	9
(DWYER, 2001)	Explora o uso do algoritmo Força Direta que organiza diagramas de classe UML na terceira dimensão para representar a arquitetura de sistemas de software orientados a objetos. Utiliza a ferramenta Wilma.	10
(FEIJS e JONG, 1998)	Apresenta o ArchView, um protótipo para visualização de arquitetura de software em 3D, utilizando formas, cores e objetos LEGO.	11
(THADEN e STEIMANN, 2003)	É uma abordagem que apresenta animações em gráficos 3D para enfatizar o entendimento intuitivo da execução de programas de uma forma didática.	12

4.3.2.5. Categorização das Iniciativas Encontradas

Posteriormente à execução da seleção dos resultados da *quasi* revisão sistemática, as iniciativas encontradas foram classificadas de acordo com critérios propostos para facilitar a sua análise e expostas na Tabela 4.2. Os critérios propostos foram:

- Diagramas estáticos ou dinâmicos: indica a utilização de diagramas estáticos ou dinâmicos (indicação na Tabela: “Estático”/”Dinâmico”/”Estático e Dinâmico).
- Sistemas em larga escala: informa se o enfoque é utilizar sistemas em larga escala (indicação na Tabela: “Sim”).
- Objetos virtuais: menciona o tipo de objetos virtuais utilizados para representar os diagramas (indicação na Tabela: tipo de objetos virtuais utilizados), ou se inclui objetos virtuais para destacar informações adicionais nos diagramas (indicação na Tabela: ”Inclui”).
- 2D = 3D: indica se valoriza a visualização do diagrama 3D semelhante ao diagrama 2D (indicação na Tabela: “Sim”).
- Abordagem utilizada: mostra a abordagem utilizada para a visualização 3D da UML (indicação na Tabela: nome da abordagem utilizada).

Na Tabela 4.2, pode ser encontrada a classificação das iniciativas de visualização 3D da UML de acordo com os critérios propostos. Após a categorização das iniciativas, foi possível realizar a seguinte análise em relação a cada critério:

Tabela 4.2 – Resultados Classificados

Referências	Diagramas Estáticos ou Dinâmicos	Sistemas em larga escala	Objetos virtuais	2D = 3D	Abordagem Utilizada
(IRANI e WARE, 2000)	Estáticos	Sim	Caixas, círculos, linhas e setas	Sim	Diagrama Geon, visualização em 3D
(PILGRIM e DUSKE, 2008) e (PILGRIM *et al.*, 2009)	Estáticos	Sim		Sim	GEF3D - Editor de diagramas 3D
(DUGERDIL e ALAM, 2008)	Estáticos e Dinâmicos	Sim	Prédios, Distritos e Canos		EvoSpace, representação da arquitetura de software no espaço 3D
(BERGHAMMER e FRONK, 2003)	Estáticos	Sim	Caixa, Cone, Esfera, Cubo, Parede		ferramenta RelView que checa a validade de diagramas 3D sintaticamente,
(RADFELDER e GOGOLLA, 2000)	Estático e Dinâmico	Sim	Inclui	Sim	Notação de UML 3D
GIL e KENT, 1998)	Estático e Dinâmico			Sim	Notação gráfica 3D da UML
(MCINTOSH *et al.*, 2005) e (MCINTOSH *et al*., 2008)	Estático	Sim		Sim	X3D-UML – visualizador UML 3D
(RODRIGUES, 2010)	Estático e Dinâmico	Sim	Inclui	Sim	VisAr3D – Visualizador de Arquitetura de Software em 3D
(LANGE *et al.*, 2007)	Estático	Sim	Inclui	Sim	Metricview – framework de visualização em 3D
(DWYER, 2001)	Estático	Sim	Inclui		Wilma – ferramenta de visualização de modelos UML 3D
(FEIJS e JONG, 1998)	Estático	Sim	Formas, cores e objetos LEGO.	Sim	ArchView – arquitetura de software em 3D
(THADEN e STEIMANN, 2003)	Dinâmico				Abordagem que apresenta animações em gráficos 3D

- Diagramas estáticos ou dinâmicos: 58,3% das abordagens trabalham com somente diagramas estáticos, 8,3% trabalham com somente diagramas dinâmicos e 33,3% com diagramas dinâmicos e estáticos.
- Sistemas em larga escala: 83,3% das iniciativas deram destaque a sistemas em larga escala.
- Objetos virtuais: 33,3% representam seus diagramas com objetos virtuais, enquanto 33,3% inclui objetos virtuais para destacar informações adicionais nos diagramas.
- 2D = 3D: 67% valorizam a visualização 3D semelhante ao diagrama 2D correspondente.

- Abordagem utilizada: 12 abordagens.

4.3.3. Publicação dos Resultados

Durante a sua condução, foram retornados 154 registros. Após a execução da seleção dos resultados, feitas as devidas filtragens baseadas nos critérios definidos para inclusão e exclusão de estudos, os dados foram extraídos de 14 registros selecionados. Dentre os 14 artigos, destacam-se 12 abordagens que apóiam de maneiras diferentes, o engenheiro de software, ao analisar e compreender informações complexas presentes na representação visual de seus diagramas. Entre elas, a abordagem VisAr3D (RODRIGUES, 2010) foi recuperada pela máquina de busca.

Todas concordam que a terceira dimensão permite a visualização de relacionamentos mais complexos do que a provida por representações existentes em duas dimensões. E de formas diferentes propõem suas soluções.

O Diagrama Geon (IRANI e WARE, 2000) é uma abordagem alternativa para UML utilizando primitivas geon 3D. Ela fornece evidências que suas estruturas são identificadas mais rapidamente e mais precisamente pelo engenheiro de software. E que são facilmente interpretadas e lembradas. As formas Geons representam seus diagramas 3D com caixas retangulares, círculos, linhas e setas desenvolvidas com a API OpenGl. Existe equivalência de um para um entre Geons e os objetos UML e no lugar de texto, utilizam cores e texturas.

O framework GEF3D ((PILGRIM e DUSKE, 2008) e (PILGRIM *et al.*, 2009)) é uma extensão do editor gráfico 2D do Eclipse (GEF) e como tal permite a edição de diagramas 3D. Os diagramas 2D são projetados em planos 3D. Ele não altera a notação gráfica com o objetivo de reutilizar a edição 2D com pouca ou nenhuma modificação. Uma primeira versão do GEF3D foi implementado em Java3D, mas esta abordagem foi cancelada por causa dos problemas de memória e outros problemas técnicos. Segundo os autores, Java3D é bem adequado para a implementação de ferramentas de visualização, mas é difícil implementar um editor baseado em Java3D.

O sistema EvoSpace (DUGERDIL e ALAM, 2008) representa a arquitetura de software através de prédios que são as classes, distritos que são os pacotes e canos que são os relacionamentos entre os prédios no espaço 3D. As métricas são utilizadas para alterar os tamanhos e as texturas dos prédios. Ele possui uma visão diurna que representa a informação estrutural e uma visão noturna, que representa a execução do programa. EvoSpaces foi implementado em Java no Eclipse.

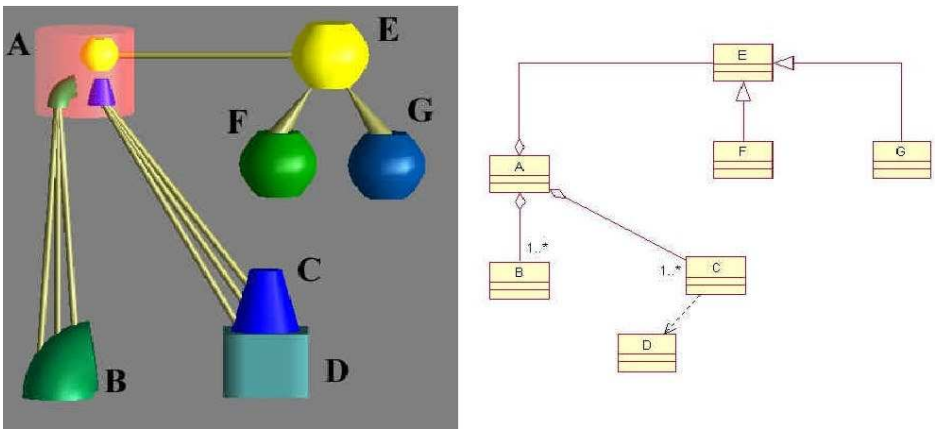


Figura 4.2: Diagrama Geon comparado ao diagrama correspondente (IRANI e WARE, 2000)

O sistema RelView (BERGHAMMER e FRONK, 2003) é um sistema que usa a álgebra relacional para checar a validade sintática de diagramas 3D. Ele utiliza caixas e esferas para visualizar classes e interfaces, respectivamente. E canos fazem a conexão entre eles. Na hierarquia de classes, utilizam árvores de cones. O sistema RelView foi implementado em Java.

A notação UML 3D proposta por RADFELDER e GOGOLLA (2000) estende a UML para a terceira dimensão, apresentando tantos aspectos estáticos como dinâmicos de um sistema numa única visão. Para facilitar a visualização de estruturas complexas, ele propõe a supressão de atributos e operações de algumas classes enquanto mostra a estruturas de outras. Ele propõe a edição interativa do diagrama. No ambiente 3D, ele destaca pontos de interesse, trazendo-os para frente, enquanto move outros, menos interessantes para o fundo. Eles mencionam que o comportamento é melhor visualizado através de diagramas animados, onde símbolos de mensagens se movem de um objeto emissor para o objeto receptor.

A notação UML 3D proposta por GIL e KENT (1998) apresenta uma alternativa aos diagramas 2D conhecidos, no formato de uma série de notações gráficas 3D. Ela combina, por exemplo, diagramas de sequência e de colaboração UML em um único diagrama de sequência 3D.

As vantagens, segundo os autores, destas duas notações é que os aspectos estáticos e dinâmicos de um sistema podem ser visualizados em um único diagrama.

A ferramenta X3D-UML (MCINTOSH *et al.*, 2005) combina a linguagem X3D e a UML para prover visualização avançada de software num ambiente 3D. Ela utiliza informações existentes do software para gerar sua visualização. Segundo o artigo, seu maior benefício é facilitar a compreensão de sistemas de software em larga escala. O

sistema inteiro pode ser visualizado em uma única cena. A VisAr3D, também, é implementada com a linguagem X3D, mas difere na maneira de exibir os diagramas e como explorar o ambiente 3D para fornecer os meios para a compreensão deste tipo de sistema.

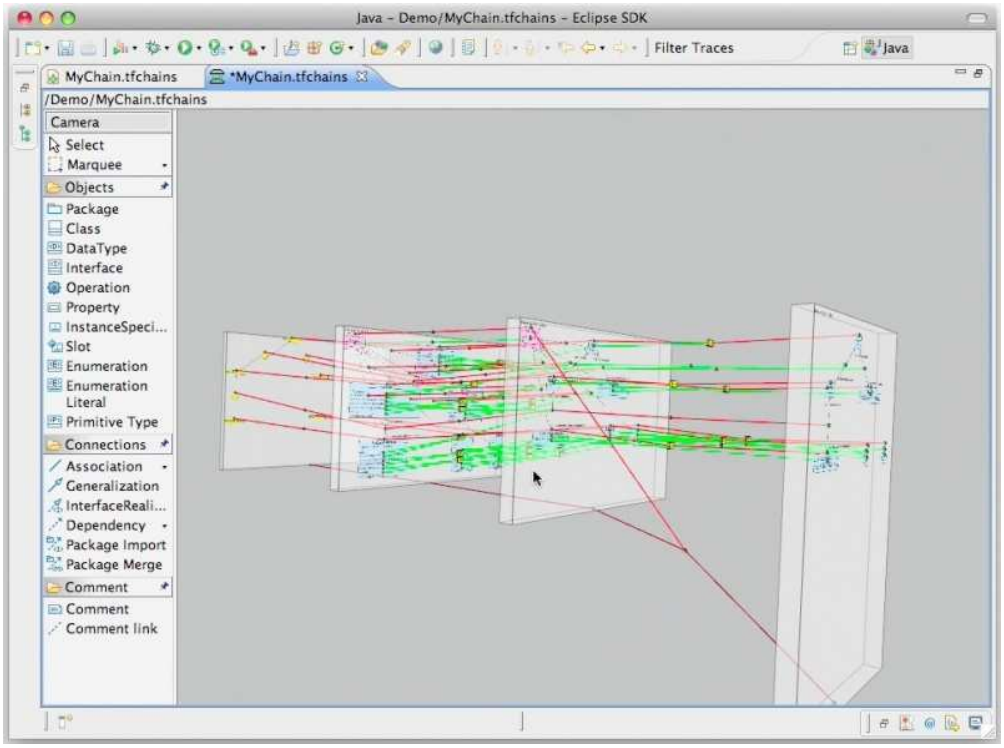


Figura 4.3: GEF3D (PILGRIM e DUSKE, 2008) e (PILGRIM *et al.*, 2009)

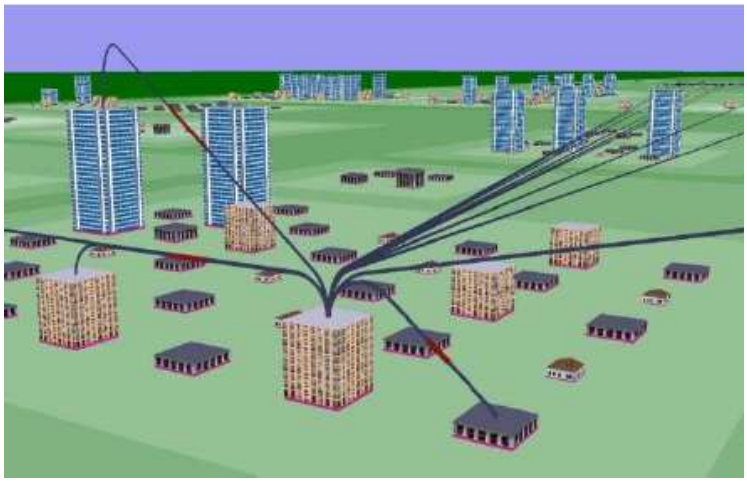


Figura 4.4: Relacionamento entre Classes no Evospace (DUGERDIL e ALAM, 2008)

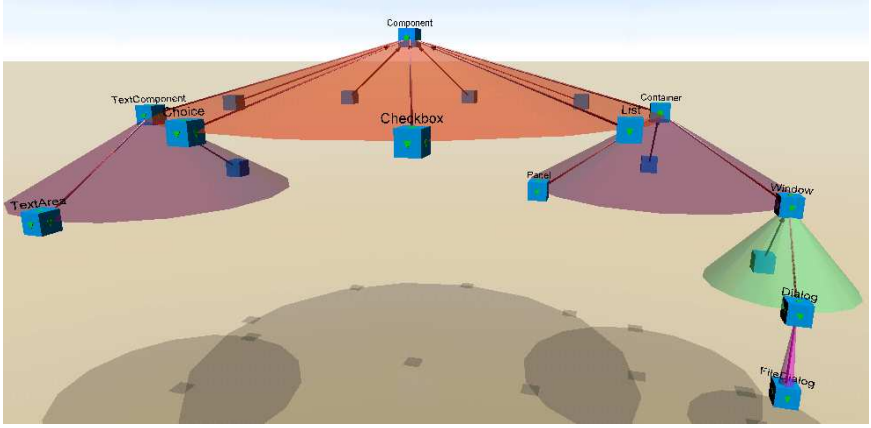


Figura 4.5: RelView (BERGHAMMER e FRONK, 2003)

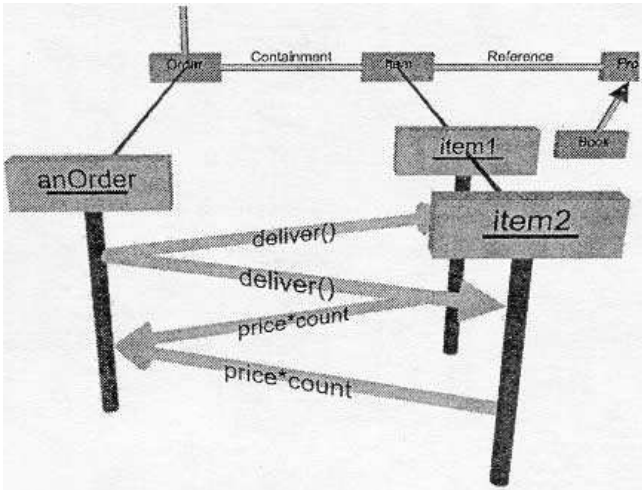


Figura 4.6: Notação de UML 3D (RADFELDER e GOGOLLA, 2000)

3D UML State Machine Diagrams (MCINTOSH *et al.*, 2008) fornece uma visão da máquina de estado para facilitar o entendimento do comportamento do sistema em diferentes níveis de abstração. Este artigo faz parte da abordagem X3D-UML, portanto, neste estudo será considerado como uma única abordagem.

A MetricView (LANGE *et al.*, 2007) é um *framework* que disponibiliza visões e técnicas de visualização em 3D que apóiam a modelagem orientada a tarefas. Suas visões combinam informações de métricas com diferentes tipos de diagramas no formato de cidades em 3D. Seu objetivo é apoiar tarefas como entendimento de programas, desenvolvimento de modelos, testes, manutenção de modelos, avaliação da qualidade, entre outros.

A ferramenta MetricView pode ser considerada a abordagem com características mais semelhantes à VisAr3D, pois utiliza também visões de diagrama para visualizar o sistema sob diferentes perspectivas, do ponto de vista dos diagramas estáticos. Sobre

o aspecto dinâmico, a notação de RADFELDER e GOGOLLA (2000) é a que mais se assemelha à abordagem VisAr3D, ao destacar os pontos de interesse e animar mensagens entre os objetos no diagrama.

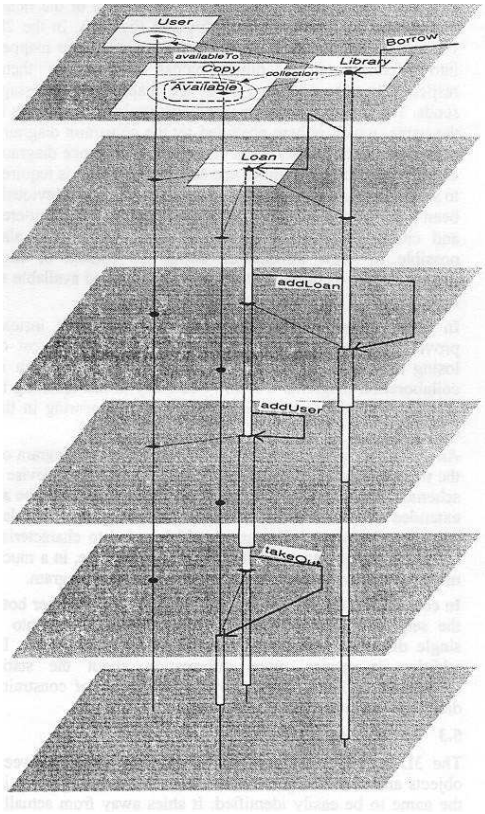


Figura 4.7: Diagrama de sequência da notação UML 3D (GIL e KENT, 1998)

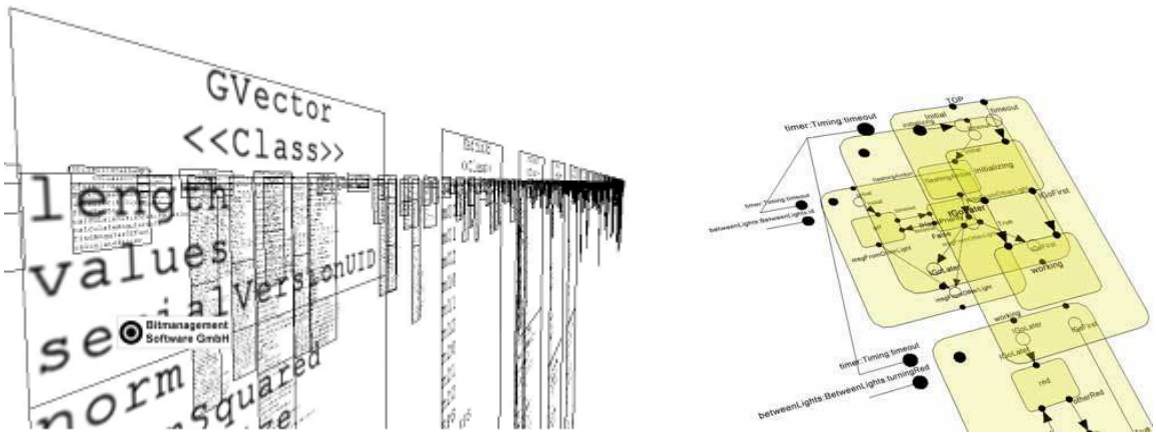


Figura 4.8: X3D-UML (MCINTOSH *et al.*, 2005) e *State Machine Diagram* (MCINTOSH *et al*., 2008)

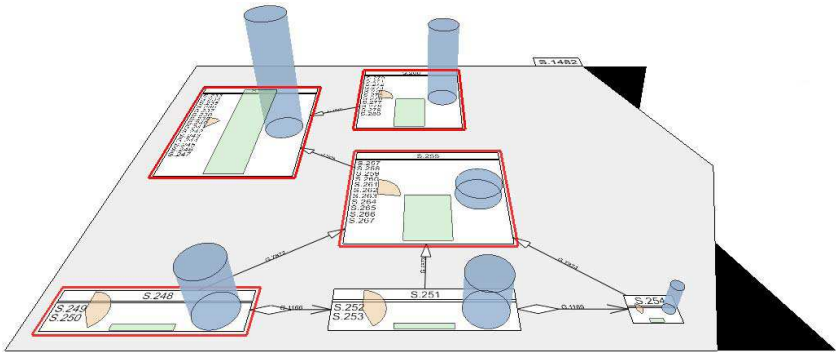


Figura 4.9: MetricView (LANGE *et al.*, 2007)

O trabalho de DWYER (2001) propõe a visualização estática de diagramas UML com o objetivo de melhorar sua compreensão num ambiente 3D. A ferramenta Wilma utiliza o algoritmo “Force Directed” e constroi e edita modelos UML. Foi desenvolvida em Java usando Java3D.

Nos artigos resultantes do estudo predominam a visualização e não a edição dos modelos 3D. As propostas de edição são apresentadas pela notação de RADFELDER e GOGOLLA (2000) e as ferramentas Wilma (DWYER, 2001) e GEF3D (PILGRIM e DUSKE, 2008) (PILGRIM *et al.*, 2009).

O trabalho de FEIJS e JONG (1998) apresenta o protótipo ArchView desenvolvido com VRML 1.0. Ele usa formas e cores através de tipos de objetos LEGO. Ao gerar um mundo virtual, o ArchView, para cada nó, deriva uma coordenada 3D a partir de sua coordenada 2D e, examinando o tipo deste nó, escolhe um objeto correspondente na biblioteca LEGO.

THADEN e STEIMANN (2003) apresenta uma abordagem que utiliza animações em gráficos 3D para enfatizar o entendimento intuitivo da execução de programas de uma forma didática. Ele enfatiza o entendimento intuitivo utilizando VRML para animar elementos UML num ambiente tridimensional.

Das 4 abordagens que utilizam diagramas estáticos e dinâmicos ao mesmo tempo, somente uma abordagem tinha até o momento, desenvolvido uma ferramenta ou protótipo com a implementação dos diagramas dinâmicos. Este tipo de diagrama ainda está no formato de proposta, inclusive na presente pesquisa de tese.

Pelo alto índice de abordagens com enfoque em sistemas em larga escala no resultado deste estudo, mostram-se evidências qualitativas que existem benefícios no uso de visualização 3D da UML ao utilizar sistemas em larga escala. Diminuir a complexidade da compreensão de sistemas deste porte foi o principal desafio da abordagem proposta nesta tese.

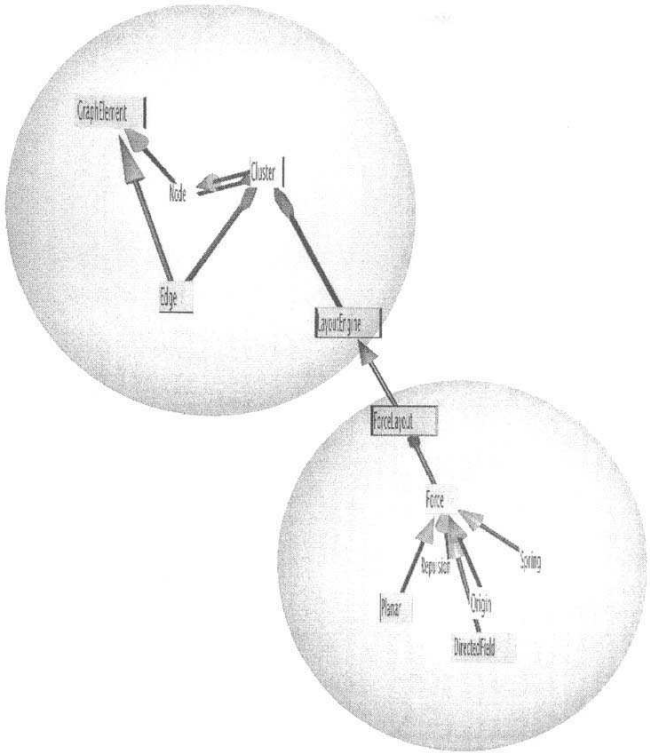


Figura 4.10: Wilma (DWYER, 2001)

O estudo revelou que 33,3% dos registros são baseadas em metáforas, tais como caixas coloridas, cones e esferas ((FEIJS e JONG, 1998), (IRANI e WARE, 2000), (BERGHAMMER e FRONK, 2003), (DUGERDIL e ALAM, 2008)), que se aproveitam da capacidade humana de lembrar e distinguir formas em 3D. A VisAr3D está inserida nos 33,3% que utiliza a valorização visual para diferenciar a informação adicional que será explorada.

Os diagramas 3D modelados devem ter uma aparência semelhante aos diagramas 2D de origem, para que se tornem um ambiente familiar ao seu usuário. A pesquisa mostra que a maioria dos registros (67%) valorizam a compatibilidade entre as duas dimensões, permitindo uma melhor compreensão do novo ambiente 3D com muitos elementos de modelagem.

Os diagramas descrevem uma projeção de um modelo UML de uma certa perspectiva. Além disso, os modelos possuem também um conjunto de dados relacionados que podem estar disponíveis, por exemplo, no formato de métricas. Muitos registros combinam estas informações com seus diagramas utilizando cores, formas, tamanhos e valores ((LANGE *et al.*, 2007), (DUGERDIL e ALAM, 2008), (MCINTOSH *et al.*, 2008), (PILGRIM *et al.*, 2009), (RODRIGUES, 2010)).

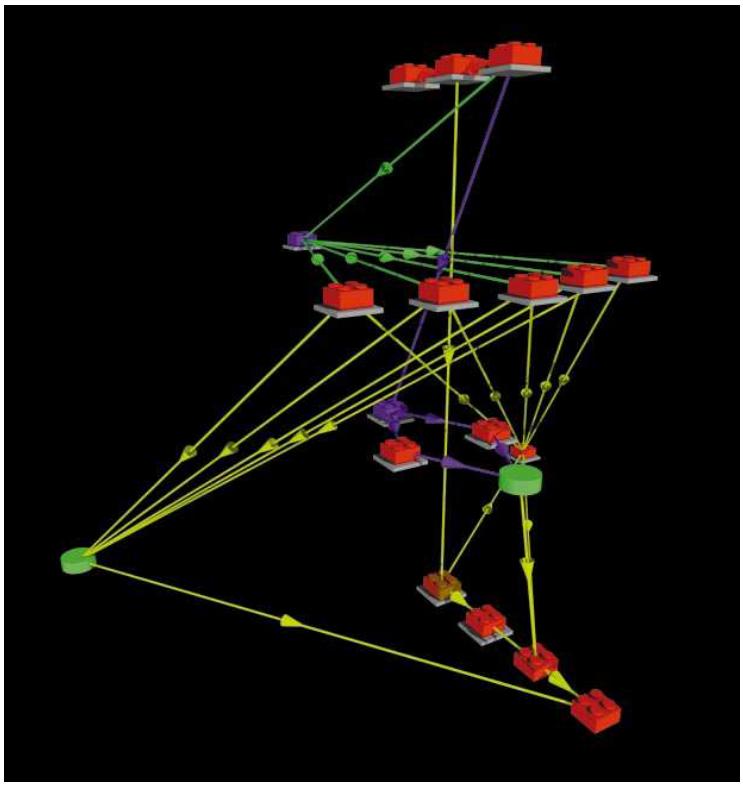


Figura 4.11: ArchView (FEIJS e JONG, 1998)

A poluição visual também é uma preocupação quando o objetivo é visualizar sistemas muito grandes. É preciso ter clareza. Por isso, os registros RADFELDER e GOGOLLA (2000) e RODRIGUES (2010) possuem a possibilidade de esconder os detalhes como atributos, métodos e nomes de associações, quando necessário.

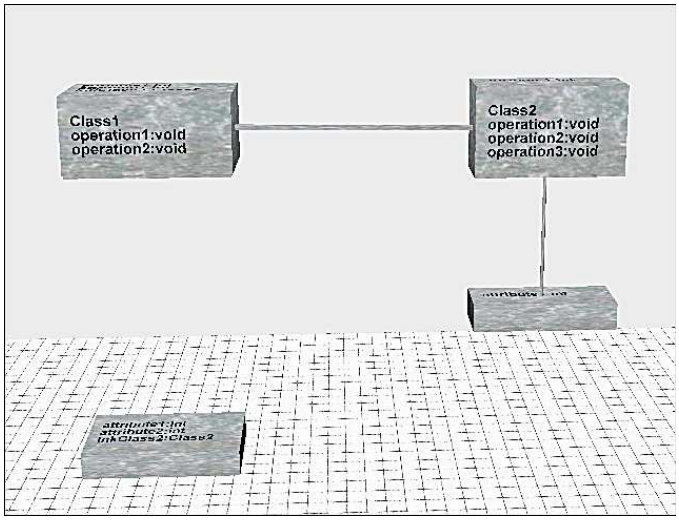


Figura 4.12: Animação em gráficos 3D (THADEN e STEIMANN, 2003)

Os artigos retornados pelo estudo começaram em 1998 (com dois artigos), continuaram frequentes nos anos seguintes, com um artigo em cada ano e teve quatro artigos publicados em 2008. Isto deve-se, provavelmente, pela disponibilidade de investimento em hardware e software, com custo, relativamente, baixo, incentivando os grupos de pesquisa em realidade virtual.

Em comparação à abordagem VisAr3D, não foi encontrada uma abordagem que atenda a todos os seus requisitos. O que se nota é que algumas iniciativas apresentam alguns pontos em comuns, mas nenhuma tem a proposta de ser um ambiente de apoio ao ensino-aprendizagem, facilitando o ensino por parte do professor, o aprendizado por parte do aluno e a comunicação entre eles como recurso importante que proporcione isto tudo, como é o caso da abordagem desta tese.

A abordagem VisAr3D tem, como os artigos selecionados pelo estudo, a preocupação de fornecer uma abordagem que possibilite ao futuro arquiteto de software, analisar e compreender informações complexas numa representação visual especial de seus diagramas, explorando a terceira dimensão. Ela pode identificar a informação, reduzir a sobrecarga cognitiva por filtragem de informações desnecessárias, melhorar a percepção dos usuários, possibilitar a exploração do conteúdo e do ambiente 3D e, ainda, possibilitar os *insights* e ampliar a cognição. Mas a VisAr3D também tem o objetivo de possibilitar o apoio às aulas práticas interativas, incluindo a utilização de vídeos, animações e documentações e permitir a criação de um ambiente de troca de informações entre os alunos e professores, permitindo a integração dos resultados do indivíduo com uma atividade do grupo de trabalho e a colaboração na resolução de problemas complexos, através da cooperação.

4.4. Considerações Finais

Este capítulo apresentou os trabalhos relacionados com a abordagem VisAr3D, mostrando suas características e particularidades, recuperados através de uma *quasi* Revisão Sistemática. Todo o processo para realização desta revisão foi detalhado.

A ferramenta MetricView (LANGE *et al.*, 2007) é a abordagem com características mais semelhantes à abordagem VisAr3D, por causa das suas visões de diagrama, ao analisar o ponto de vista dos diagramas estáticos. Sob o aspecto dinâmico, a notação de RADFELDER e GOGOLLA (2000) se compara ao VisAr3D, ao destacar os pontos de interesse e animar mensagens entre os objetos no diagrama. Contudo, não foi encontrada uma abordagem que atendesse a todos os requisitos da VisAr3D. Muitas mostram alguns pontos em comuns, como analisar e compreender informações complexas numa representação visual especial de seus diagramas,

explorando a terceira dimensão; mas nenhuma tem a proposta de ser um ambiente de apoio ao ensino-aprendizagem, facilitando o ensino por parte do professor, o aprendizado por parte do aluno e a comunicação entre eles como recurso importante que proporcione isto tudo, como é o caso da abordagem desta tese.

Capítulo 5 - Abordagem VisAr3D

"Os olhos têm de ser educados para que nossa alegria aumente"

Rubens Alves

5.1. Introdução

Tendo introduzido a motivação e a fundamentação teórica, propõe-se agora as soluções e a metodologia do trabalho desta pesquisa de tese que lidaram com os problemas e questões desta pesquisa. Este capítulo tem como objetivo apresentar a abordagem proposta VisAr3D – Visualização de Arquitetura de Software em 3D e os fatores que motivaram a sua elaboração.

A abordagem VisAr3D traz uma nova dimensão à prática da disciplina Modelagem de Software e propõe uma solução diferenciada para o apoio à compreensão de diagramas UML em sistemas com muitos elementos de modelagem, combinando recursos de tecnologias emergentes de visualização 3D, como a Realidade Virtual e a Realidade Aumentada (RODRIGUES e WERNER, 2011). Ela visa atender aos seguintes desafios:

1. possibilitar o apoio a aulas práticas interativas, incluindo a utilização de vídeos, animações, documentações etc.;
2. permitir a criação de um ambiente de troca de informações entre os alunos e professores;
3. possibilitar uma nova forma de disposição das informações contextualizadas a um modelo de software 3D;
4. se propor a surpreender os alunos com novas maneiras de ressaltar informações e relações existente e relevantes; e
5. Permitir o aluno interagir, intuitivamente, no ambiente 3D para compreender um sistema com muitos elementos de modelagem proposto pelo professor em sala de aula.

A principal característica da abordagem VisAr3D é disponibilizar a visualização 3D de um modelo UML, gerado automaticamente a partir de um diagrama 2D existente. Esta solução explora diferentes perspectivas, relacionamentos, abstrações e documentações contextualizadas, onde o usuário possa, intuitivamente, compreender os elementos de modelagem de um ambiente 3D.

A ideia é organizar todo tipo de documentação existente e relevante relacionada aos elementos de modelagem de um sistema. Esses dados importantes estarão associados aos elementos de modelagem correspondentes, e o usuário poderá explorar e interagir com eles na terceira dimensão. A abordagem propõe ao aluno conflitos com seu atual nível de desenvolvimento, que o desafia a aprender e avançar. O resultado deste tipo de aprendizagem exige do educando um salto no sentido de apropriar-se de novos conhecimentos que lhe estão sendo propostos (LUCKESI, 2005).

A visualização de diagramas em uma nova perspectiva possibilita a utilização de projetos mais elaborados e complexos, semelhantes aos desenvolvidos pela indústria. E, ainda, a manipulação destes modelos em um ambiente de aprendizagem simulado por meio de elementos virtuais 3D pode trazer uma experiência atraente para o processo de aprendizagem. Uma exploração e interação intuitiva feita pelos alunos de modelagem de software e pelo professor, utilizando recursos e facilidades, tais como a manipulação de modelos em um ambiente de aprendizagem simulado por meio de elementos virtuais, traz uma experiência atraente para o processo de aprendizagem.

O capítulo é organizado da seguinte forma: a Seção 5.1, como Introdução, apresentou a Abordagem VisAr3D. A Seção 5.2 dedica-se a descrever os aspectos pedagógicos que fundamentou toda a pesquisa. A Seção 5.3 descreve os seus Requisitos. A Seção 5.4 mostra uma Visão Geral da Abordagem. A Seção 5.5 apresenta o Público-alvo. A Seção 5.6, a Utilização do Espaço 3D. A Seção 5.7 mostra os Recursos da VisAr3D. A Seção 5.8 detalha o Protótipo Implementado e a Seção 5.9 termina com as Considerações Finais do capítulo.

5.2. Aspectos Pedagógicos

Em relação à ação educativa desta tese de doutorado, tem-se o interesse que o aluno aprenda e se desenvolva, individualmente e coletivamente. No entanto, esta ação se submete, diretamente, ao desempenho do sistema educativo e à conduta individual dos professores. Por essa razão, esta Seção dedica-se a descrever os aspectos pedagógicos que conduziu toda a pesquisa, desde a concepção da abordagem VisAr3D, a implementação do protótipo e até a execução do estudo experimental.

O que importa aqui é a aproximação do aluno do conhecimento elaborado e assim, torná-lo receptivo a este conhecimento, porém não passivo. De forma que ele assimile ativamente o que está sendo exposto, ou seja, ele receba a informação, medite sobre ela e faça relacionamentos com suas experiências e saberes.

Ensinar significa criar condições para que o educando, efetivamente, entenda aquilo que se está querendo que ele aprenda. Para que a aprendizagem se efetue, os conteúdos necessitam ser compreendidos e internalizados (LUCKESI, 2005). Ele se desenvolve enquanto aprende e, para que isto ocorra, é preciso que haja um ensino intencional. Segundo LUCKESI (2005), existe ainda outro tipo de aprendizagem, que se dá espontânea e informalmente e que, apesar de significativa, é insuficiente para dar conta da assimilação ativa dos conteúdos.

Dos estudos de Piaget, HAYDT (2006) infere alguns pressupostos pedagógicos e diretrizes para a ação docente, que são apresentados a seguir:

a) Respeitar as características de cada etapa do desenvolvimento do aluno, estimulando a atividade funcional;

b) Propor atividades desafiadoras, contribuindo para ampliar seus esquemas mentais de pensamento;

c) Utilizar métodos ativos de ensino-aprendizagem (como solução de problemas, pesquisa, experimentação, atividades de manipulação e construção e trabalho em grupo), permitindo que o conhecimento não seja apenas transmitido, mas reinventado ou reconstruído pelo aluno;

d) Prover a sala de aula de materiais variados que o aluno possa ver, tocar e manipular, tendo em vista a resolução de problemas;

e) Proporcionar aos alunos situações nas quais tenham possibilidade de manipular objetos concretos, aplicando seus esquemas mentais às situações reais e exercitando as operações concretas nas soluções de problemas;

f) Utilizar o jogo como um recurso útil para a aprendizagem. O jogo é capaz de absorver o aluno de forma intensa e total, criando um clima de entusiasmo;

g) Fazer com que a interação social e a linguagem tenham um lugar proeminente na programação diária de ensino, estimulando a interação verbal entre os alunos e provendo atividades de grupo que envolvam cooperação e troca de ideias; e

h) Adotar uma atitude de *feedback* estimulando a atividade mental dos alunos e seu trabalho de descoberta e investigação, bem como reforçando, positivamente, suas iniciativas, de modo a incentivá-lo no processo de construção do conhecimento.

Para atender estes pressupostos, a pesquisa investiu na utilização das tecnologias emergentes 3D, como Realidade Virtual e Realidade Aumentada, que

segundo o relatório *The Horizon Report*[1] de 2011 (TNMC, 2011), a Realidade Aumentada representa a terceira posição entre as tendências-chave, como tecnologia potencial, que pode ter um grande impacto ao longo dos próximos cinco anos para a prática de ensino e aprendizagem. Por ordem de prioridade foram elas: livros eletrônicos, dispositivos móveis, realidade aumentada, aprendizado baseado em jogos, computação baseados em gesto e análise de aprendizagem.

Com a utilização destes recursos, a intenção é aproximar o aluno do conteúdo da aprendizagem proposto pelo professor, e obter uma série de vantagens cognitivas, que segundo TORI (2010), destacam-se:

a) **Evolução humana**: o ser humano evolui interagindo com um ambiente puramente tridimensional; a interface 3D é, portanto, natural e intuitiva, mesmo para quem a utiliza pela primeira vez;

b) **Modelos mentais**: a todo instante nosso cérebro processa modelos mentais tridimensionais para interpretar o meio ambiente e com este interagir. Dessa forma, ainda que o volume de dados e a complexidade visual sejam maiores, em geral as estruturas espaciais são mais facilmente interpretadas e organizadas, mentalmente, pela maioria das pessoas;

c) **Orientação**: é mais fácil para o ser humano se orientar numa navegação baseada em ambientes tridimensionais que em uma estrutura de navegação baseada em menus e/ou ícones;

d) **Menor sobrecarga cognitiva**: graças à naturalidade da interação com ambientes tridimensionais, mais capacidade cognitiva fica à disposição do usuário;

e) **Interação direta**: se em vez de clicar em um menu para acessar um objeto, o usuário puder “pegá-lo” diretamente (como faz no mundo real), a interface será mais direta e, portanto, mais intuitiva e simples para o usuário; e

f) **Imersão**: ambientes 3D facilitam a sensação de imersão e, consequentemente, aumentam as percepções de presença e proximidade.

A abordagem VisAr3D sugere, ainda, uma série de procedimentos de ensino que podem se ajustar aos objetivos propostos pelo professor para o processo instrucional. Os procedimentos de ensino dizem respeito às formas de intervenção na sala de aula, como por exemplo: (HAYDT, 2006):

[1] A série *The Horizon Report’s*, internacionalmente reconhecida, faz parte do *The New Media Consortium* (NMC), um empreendimento de pesquisa abrangente, criada em 2002, que analisa as tecnologias emergentes pelo seu impacto potencial e utilização no ensino, aprendizagem e investigação criativa em uma variedade de setores ao redor do mundo.

a) **Aprendizagem por descoberta**: o professor cria situações de ensino nas quais o aluno observa, manipula materiais, experimenta, coleta dados e informações, para depois sistematizá-los e chegar às conclusões e generalizações necessárias que permitirão formular os conceitos e princípios;

b) **Método de solução de problemas**: que consiste em apresentar ao aluno uma situação problemática para que ele proponha uma solução satisfatória, utilizando os conhecimentos de que já dispõe ou buscando novas informações através da pesquisa;

c) **Jogo**: o jogo canaliza as energias no sentido de um esforço total para a consecução de seu objetivo;

d) **Trabalho em grupo**: os principais objetivos do trabalho em equipe são facilitar a construção do conhecimento, permitir a troca de ideias e opiniões e possibilitar a prática da cooperação para conseguir um fim comum.

e) **Estudo de caso**: é uma técnica que consiste em apresentar aos alunos uma situação real, dentro do assunto estudado, para que analisem e, se for necessário, proponham alternativas de solução. É uma forma de os alunos aplicarem os conhecimentos teóricos a situações práticas.

Através desse conjunto de recursos e procedimentos, a abordagem VisAr3D se propõe a envolver e estimular o aluno para a aprendizagem, com o apoio bem próximo do professor. No entanto, o sucesso não é garantido, pois a motivação é um fenômeno psicológico, segundo HAYDT (2006), pessoal e interno, que varia de acordo com as diferenças individuais. E o que o professor pode fazer é incentivar o aluno, isto é, despertar e polarizar sua atenção e seu interesse, orientando e canalizando positivamente as fontes motivacionais, utilizando recursos ou procedimentos incentivadores.

5.3. Requisitos da Abordagem

JACOBSON *et al.* (2003) descrevem um conjunto de princípios gerais de *design* para criar ambientes de aprendizagem e ferramentas para ajudar os alunos a compreender as perspectivas científicas sobre sistemas complexos, baseados em recentes modelos construtivistas da aprendizagem e na ponderação dos êxitos e desafios identificados em projetos de sistemas complexos e de ensino. Eles acreditam que a integração de conhecimentos e metodologias de sistemas complexos no currículo promoverá trajetórias de aprendizagem de alunos que levem ao crescimento dos conceitos e entendimentos aprofundados ao longo do tempo. Esses princípios,

assim como o resultado da primeira *quasi* Revisão *S*istemática (RODRIGUES e WERNER, 2009c) conduzida anteriormente, norteiam os requisitos da abordagem VisAr3D, descritos a seguir, que atendem tanto o aluno quanto o professor:

(1) A abordagem deve apoiar o desenvolvimento e a participação dos alunos em projetos complexos. Enquanto os projetos encontrados no mercado de trabalho são, geralmente, sistemas grandes, com requisitos como desempenho, confiabilidade, baixo custo e alta qualidade, os alunos saem da universidade sem ter tido oportunidade de lidar com este tipo de problema. Os alunos, muitas vezes, aprendem a teoria melhor quando ela aparece em exemplos concretos e têm relevância imediata para eles.

Segundo Piaget, a aprendizagem torna-se mais significativa ao proporcionar aos alunos aplicar seus esquemas mentais às situações reais (HAYDT, 2006). HAYDT (2006) diz ainda que a partir da correlação com o real, chega-se à abstração, à generalização e a elaboração teórica, por meio da mobilização dos esquemas operatórios do pensamento, que geram a reflexão e o raciocínio.

Assim, esta abordagem deve apoiar a modelagem de um sistema feita pelos estudantes em sistemas com muitos elementos de modelagem, facilitando o entendimento das partes e a compreensão do todo.

(2) A abordagem deve reduzir a distância entre a teoria e a prática. Aliado à teoria, apoiando a prática, a abordagem deve trazer um maior dinamismo à sala de aula e resultar em uma participação mais significativa dos alunos e um maior rendimento na aprendizagem. Segundo DONGSUN *et al.* (2008), num projeto, exemplos práticos, consistentes e repetidos de um mesmo contexto devem ser introduzidos para se obter um ensino efetivo.

Combinar as atividades intelectuais com trabalho manual, fazendo os dois caminharem juntos é apresentado por HAYD (2006) como uma contribuição para o desenvolvimento intelectual do aluno.

(3) A abordagem deve apoiar as atividades dos estudantes no processo de assimilação de conhecimentos e habilidades, e assim propagar o conhecimento entre os participantes. JACOBSON *et al.* (2003) afirmam que, por meio de interações colaborativas e a construção de artefatos compartilhados, as oportunidades surgem naturalmente para os alunos. Oportunidades para obterem *feedback* sobre suas ideias, e evoluí-las e refiná-las ao longo de seus projetos. Com a abordagem, pretende-se ajudar a melhorar a comunicação entre os membros do grupo, ajudar na organização e divisão de tarefas dentro da equipe, permitindo a integração dos

resultados individuais como uma atividade de grupo de trabalho, e colaborar com a resolução de problemas complexos, através da cooperação, ou seja, propagar conhecimento entre os participantes.

HAYDT (2006) diz que a interação humana tem uma função educativa, pois convivendo com seus semelhantes que o ser humano é educado e se educa. Diz ainda que no processo de construção do conhecimento, o valor pedagógico da interação humana é ainda mais evidente, pois é por intermédio da relação professor-aluno e da relação aluno-aluno que o conhecimento vai sendo coletivamente construído.

(4) A abordagem deve ser interessante para o usuário, proporcionando um ambiente exploratório e intuitivo, propício a descobertas e desafios, que responda a interação do aluno. Através de experiências bem sucedidas, a sua motivação é reforçada para promover um aprendizado futuro (BAR-YAM *et al.*, 2004). Recentemente, percebe-se o aumento do interesse nas tecnologias tridimensionais nas aplicações de jogos e na indústria dos cinemas. Ambos ligados ao entretenimento oferecem diferentes formas de lidar com a percepção do usuário. Com o requisito de ser atrativa para o aluno, a abordagem se propõe a fornecer um ambiente com esta perspectiva.

(5) A simplicidade e a facilidade de uso são requisitos importantes ao utilizar a ferramenta. Tanto professores, como os estudantes de engenharia de software, têm pouco tempo para aprender novos conceitos e, provavelmente, perderiam o interesse em utilizar a ferramenta se imposta uma curva de aprendizagem como um pré-requisito para familiarizar-se com ela. É fato sabido que as pessoas não gostam de passar muito tempo aprendendo como usar um sistema. Preferem utilizá-lo logo e tornarem-se competentes para realizar tarefas sem muito esforço (PREECE *et al.*, 2005).

Segundo TORI (2010), as pessoas gostam de previsibilidade, facilidade e simplicidade de uso quando interagindo com qualquer mídia.

(6) Os diagramas 3D modelados devem ter uma aparência semelhante aos diagramas 2D de origem, para que se tornem um ambiente familiar ao seu usuário. O usuário deve reconhecer facilmente o seu próprio diagrama, mesmo sob uma roupagem diferente. A terceira dimensão deve complementar os diagramas 2D, permitindo uma visualização mais rica, semanticamente.

(7) A abordagem deve utilizar a valorização visual, necessária para diferenciar a informação adicional que será explorada. Usando profundidade e cor, novos pontos de

vista são fornecidos. É importante que os conceitos relevantes de sistemas complexos sejam salientados e explícitos para o aluno.

(8) Detalhes de informações devem ser escondidos, e exibidos quando solicitados. Atributos, métodos e nome de associações são omitidos para diminuir a poluição visual e obter clareza, principalmente em diagramas grandes.

A VisAr3D propõe-se a ser um ambiente de visualização 3D e não permite edição de modelos. A edição de diagramas 3D é mais árdua do que a edição de diagramas 2D (GEF3D, 2011) e poderia exigir do usuário uma habilidade que prejudicaria um requisito importante da abordagem, que é a sua simplicidade de uso. TORI (2010) confirma, dizendo que esta faz parte das diretrizes básicas para o *design* de mídia interativa, sob uma perspectiva de aplicação na educação.

5.4. Visão Geral da Abordagem VisAr3D

Uma visão geral da abordagem VisAr3D, apresentada na Figura 5.1, é composta de três módulos: Módulo Arquitetural, Módulo Realidade Aumentada e Módulo Realidade Virtual.

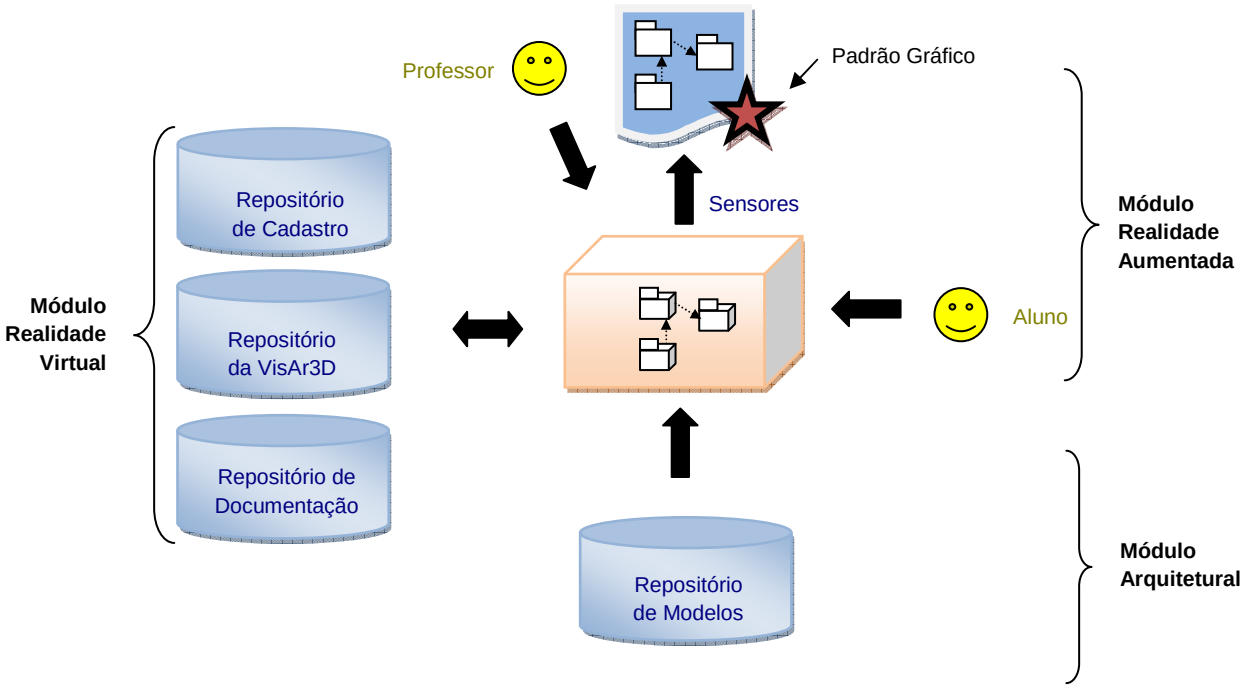


Figura 5.1: Visão Geral da Abordagem VisAr3D

Em um curso regular voltado para as novas demandas do mercado, o professor ensina exibindo parte de um diagrama UML de um sistema grande e complexo, projetando-o na parede ou em um cartaz impresso. Ele detalha elementos de

modelagem e seus relacionamentos em um nível de abstração, possivelmente utilizando vários estilos arquiteturais.

No **Módulo Arquitetural** da abordagem VisAr3D, estes diagramas são criados e documentados por uma ferramenta UML externa e exportados como um arquivo XMI. A VisAr3D possui a capacidade de ler estes diagramas UML armazenados.

No **Módulo Realidade Aumentada**, a VisAr3D usa a tecnologia 3D para capturar e reconhecer a projeção 2D (diagrama do professor), ajudando o professor e os alunos a identificarem e acessarem, rapidamente, o modelo. Ao utilizar as bibliotecas ARToolKit (KATO, 2011), um padrão gráfico presente na projeção é capturado pela *webcam* no computador do aluno. A VisAr3D identifica o diagrama que está sendo estudado e o ponto de vista exato deste diagrama. Este Módulo torna o objeto de estudo mais acessível para o usuário, ao permitir acesso ao arquivo XMI relativo a este diagrama através da rede de computadores, garantindo a exibição dos modelos na versão mais recente. A Figura 5.2 mostra o diagrama exibido pelo professor na sala de aula, contendo o padrão gráfico (o diagrama do professor corresponde à parte do diagrama original criado em 2D) e as Figuras 5.3 e 5.4 mostram os alunos capturando este padrão gráfico com sua webcam no seu computador.

Depois de reconhecer o diagrama do software exibido em 2D, o **Módulo Realidade Virtual** gera, automaticamente, um equivalente, em 3D (a Figura 5.5 mostra a geração automática de um modelo 3D a partir de um modelo 2D). Este diagrama resultante é equivalente ao diagrama 2D, exibido em um espaço ilimitado, com uma pequena profundidade.

5.4.1. Funcionamento Geral dos Três Módulos

1) No Módulo Realidade Aumentada, o usuário, a partir de um padrão gráfico presente num diagrama impresso ou projetado, permite o acesso do Módulo Realidade Virtual.

2) Os dados de um projeto de modelos de sistemas de software são importados ou lidos através de arquivos XMI de um repositório, criados, inicialmente, no Módulo Arquitetural.

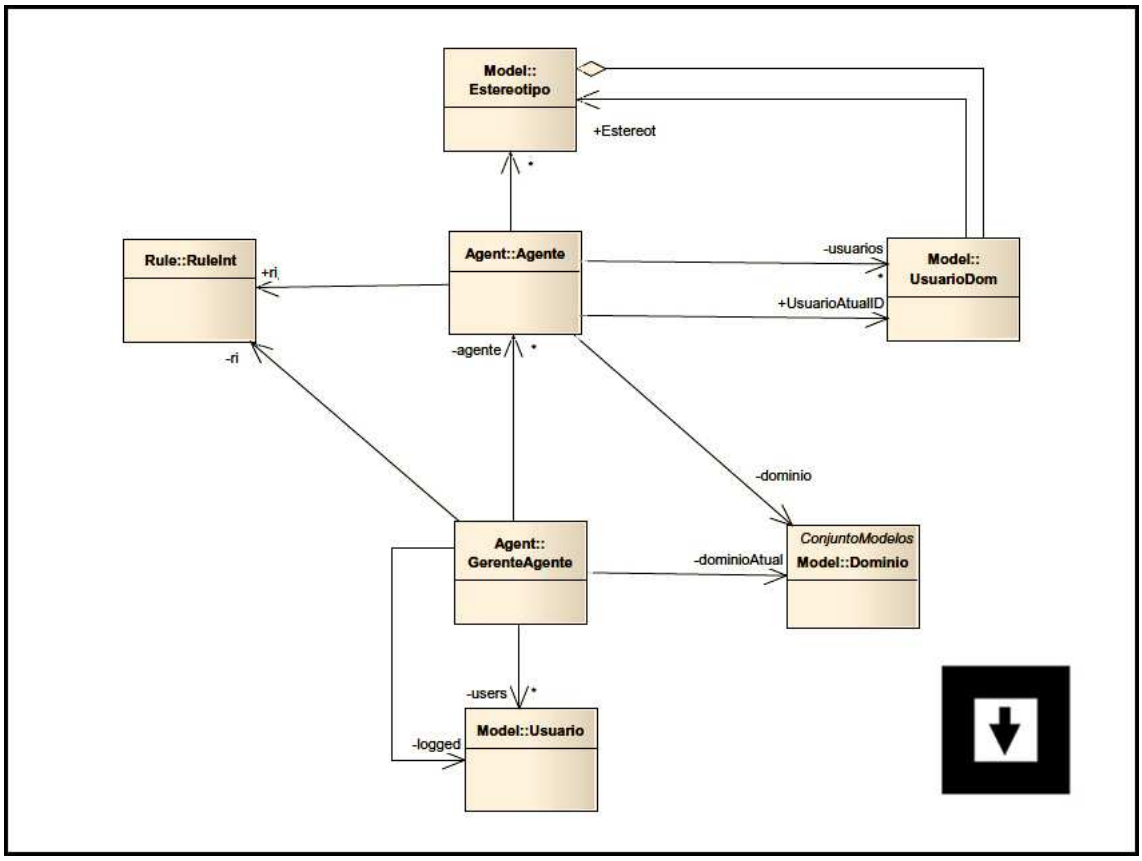


Figura 5.2: Exemplo de arquitetura em 2D com padrão gráfico

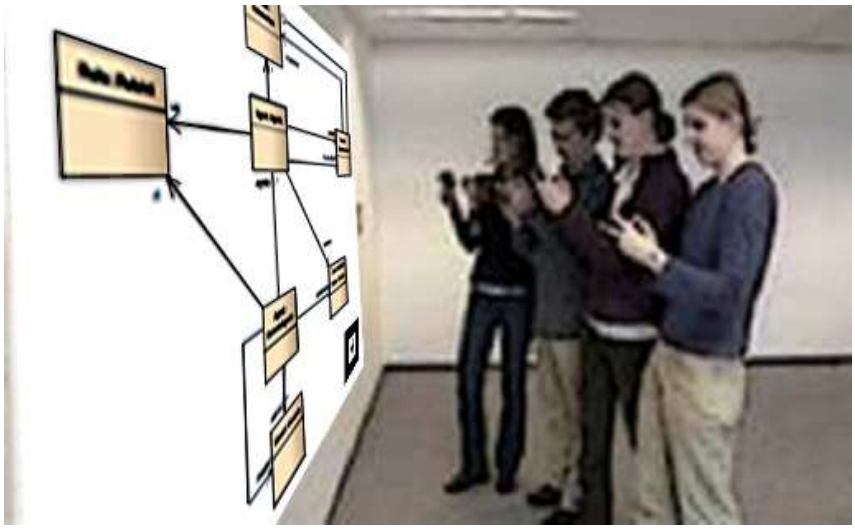


Figura 5.3: Projeção da arquitetura de software na parede (simulação)

3) O Módulo Realidade Virtual compila as informações vindas do Módulo Arquitetural e, em seguida, posiciona os diagramas no espaço 3D, esconde atributos e operações, cria uma pequena profundidade nas classes com diferentes cores para

indicar a presença de informações importantes contextualizadas. Por esta razão, deve-se considerar que a qualidade da visualização 3D está diretamente ligada à qualidade do modelo construído.

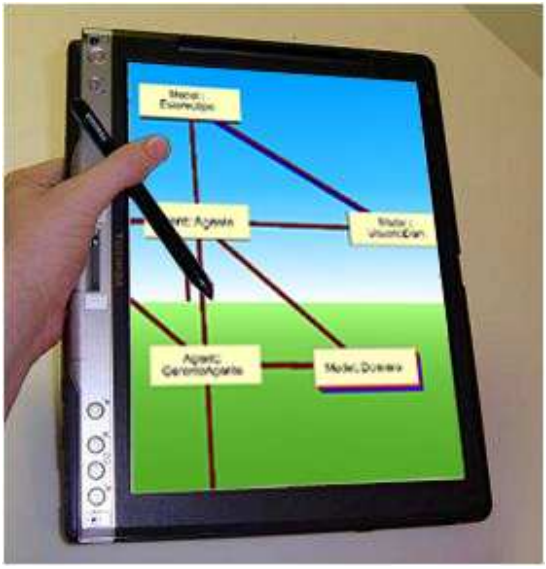


Figura 5.4: Dispositivo móvel dos alunos (simulação)

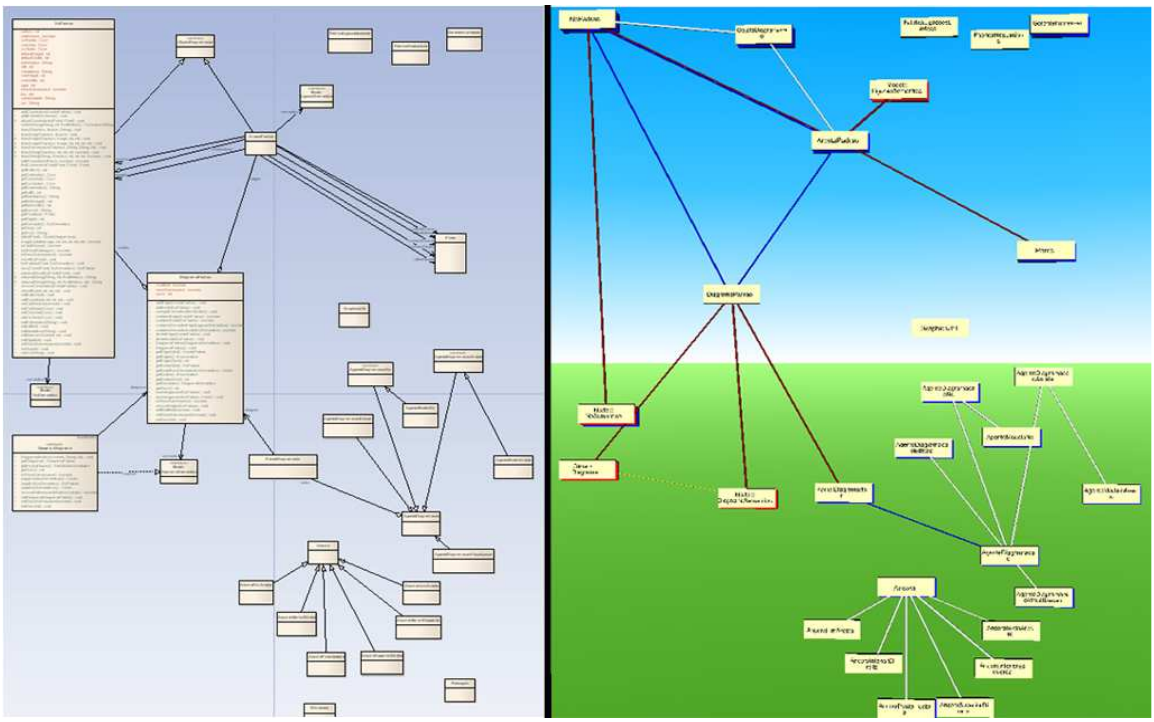


Figura 5.5: A VisAr3D, automaticamente, gera um modelo 3D (à direita) a partir do modelo 2D (à esquerda)

4) O espaço 3D é totalmente preenchido por todos os tipos de diagramas disponíveis modelados: diagramas de classes, casos de uso, diagramas de sequência etc. Centralizado na tela, é apresentado o primeiro diagrama de classes do sistema.

5) Para acessar diferentes tipos de diagramas UML, como por exemplo, um diagrama de sequência ou caso de uso, o usuário deve mover no espaço 3D numa posição abaixo na tela. Numa posição acima na tela, tem-se o acesso aos diagramas de pacotes.

6) A profundidade da terceira dimensão disponibiliza outras versões daqueles diagramas. O eixo Z mostra a variação dos modelos no tempo. A ideia é ter um relógio, que ao rodar os ponteiros possa navegar no eixo Z e visualizar outra versão daquele mesmo diagrama.

7) Para examinar cada modelo, o usuário deve aplicar o *Zoom* ou rodar o mesmo, utilizando além do mouse, as setas do teclado.

8) Durante a visualização de um diagrama, o usuário pode querer explorar algum tipo de informação, especificamente. Como por exemplo, métodos da classe, a documentação disponível ou o relacionamento de uma classe em outro diagrama. Para isso, basta acessar o menu.

5.4.2. Repositórios

A VisAr3D utilliza quatro repositórios, conforme Figura 5.1 da Visão Geral da Abordagem VisAr3D, que estão disponíveis em rede e no formato de arquivos XMI:

1) Um repositório de modelos, que identifica e disponibiliza os diagramas modelados nas ferramentas UML em 2D nas suas diversas versões;
2) Um repositório de documentação sobre estes modelos gerada dentro do espaço virtual;
3) Um repositório da VisAr3D com as informações do tipo: anotações, exercícios e o log das ações dos usuários feitas durante a sua utilização;
4) Um repositório de cadastro dos usuários.

5.4.3. Recursos Mínimos Necessários

A implementação completa da abordagem VisAr3D requer os seguintes recursos mínimos: um diagrama UML projetado na parede ou em um papel impresso com um padrão gráfico, e um dispositivo móvel com webcam, ligado à rede de computadores.

Os dispositivos móveis sugeridos para serem utilizados com esta abordagem são a tablet PC e o PDA, devido à sua portabilidade. No entanto, a tablet PC é o mais adequado dos dois, devido à limitada capacidade da tela do PDA. Seu uso implicará

na adoção de uma estratégia de exibição da aplicação direcionada a esta plataforma e a adaptação de conteúdo às suas restrições.

5.5. Público-alvo

A abordagem destina-se, preferencialmente, aos alunos de graduação e a professores da disciplina de Modelagem de Sistemas. Contudo não se restringe apenas a este nível de alunos, podendo ser útil àqueles mais experientes, assim como pode apoiar, também, aos desenvolvedores de software. Estes alunos já conhecem, através dos seus professores, ou mesmo, até por experiência profissional particular na área, a modelagem de sistemas como uma etapa importante no desenvolvimento de software, determinante para o seu sucesso.

Os alunos, como nativos digitais[2], são pessoas que já passam a maior parte do tempo lidando com diversões em ambientes tridimensionais de computadores, videogames, vídeos e celulares. Por isso, a adaptação dos alunos ao ambiente virtual é rápida. Eles não têm quaisquer dúvidas ou receios quanto ao uso de tecnologia de informação e comunicação em atividades do dia-a-dia (TORI, 2010).

Além de utilizarem a linguagem dos jogos, ou seja, a linguagem dos jovens, segundo TORI (2010), estes ambientes são uma forma segura de envolver e motivar alunos a colaborarem entre si, se comunicarem, desempenharem papéis e vivenciarem situações diferentes daquelas a que estão habituados.

No Brasil, o perfil desse novo aluno conectado exige mudanças de paradigmas no processo de ensino e aprendizagem. Segundo TORI (2010), não vai ser fácil para eles se adaptarem às escolas que não tiverem integrado as novas tecnologias à sua rotina.

HAYDT (2006) afirma que a construção do conhecimento é um processo interpessoal. O ponto principal desse processo interativo é a relação educando-educador. E essa relação não unilateral, pois não é só o aluno que constroi seu conhecimento. O professor é atingido nessa relação. De certa forma, ele aprende com seu aluno. A abordagem VisAr3D promete possibilitar uma situação de intercâmbio bastante proveitosa para ambos, em que o conhecimento será construído em conjunto.

[2] Expressão criada em 2001 por Marc Prensky, pensador e desenvolvedor de games, o termo Nativos Digitais corresponde às pessoas que cresceram com a tecnologia digital, tais como computadores, Internet, telefones celulares e MP3. Um nativo digital experimenta novos aplicativos, tem facilidade com blogs, em lidar com múltiplos links e sites e de interagir com os outros (PRENSKY, 2001).

A ideia não é automatizar as atividades da disciplina de Modelagem de Sistemas, mas apoiar o ensino do professor e, principalmente, a aprendizagem do aluno ao utilizar sistemas com muitos elementos de modelagem. É fornecer suporte ao acesso das inúmeras informações e fazer com que os alunos trabalhem juntos sob a orientação do professor.

A abordagem disponibiliza diferentes participações, assim como permissões, para sua utilização pelo aluno e professor.

5.6. Utilização do Espaço 3D

Durante a criação de modelos dentro de sala de aula, são elaborados diagramas de vários tipos, com documentação correspondente, são feitos diversos exercícios, envolvendo uma discussão entre os alunos e entre os alunos e o professor. Para organizar este cenário e apoiar o ensino de modelagem, foi proposta uma divisão do espaço 3D para que todas estas informações não sejam perdidas e que ainda sejam aproveitadas durante todo o ciclo de vida do projeto. Este espaço ilimitado para exploração dos dados foi dividido de acordo com visões e tipos de diagramas (a Figura 5.6 mostra um esquema desta organização), seguindo alguns critérios:

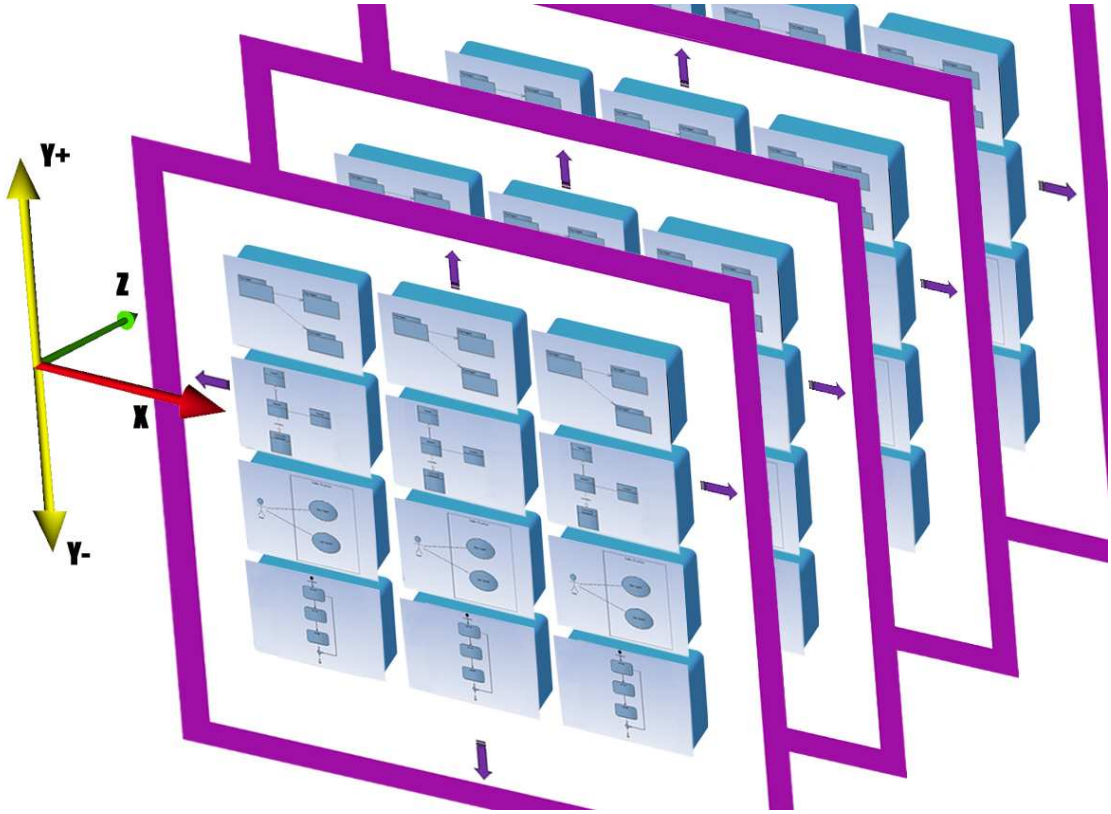


Figura 5.6: Esquema da organização do espaço 3D

1) Todos os diagramas são exibidos e explorados em uma só cena em vez de múltiplas janelas, como acontece nas ferramentas 2D. Alguns dos impactos (como por exemplo, inconsistências) e facilidades na leitura são perdidos quando impressos, ou dispostos em muitas janelas com *scroll*.

2) Diagramas do mesmo tipo são exibidos ao longo do eixo X do Sistema Cartesiano, ou seja, numa mesma altura são exibidos somente diagramas de Caso de Uso. Numa outra altura somente, Diagramas de Classe e assim por diante;

3) O eixo Y exibe diferentes tipos de diagramas, seguindo o seguinte critério:

i) Y = 0: Diagrama de Classes, devido à sua relevância e maior frequência de uso.

ii) Y positivo: Diagrama de Pacotes, que exibe um outro nível de abstração da arquitetura.

iii) Y negativo: para cada altura diferente, é reservado um espaço para diferentes tipos de diagramas: Diagrama de Casos de Uso, Diagrama de Sequência, Diagrama de Estados etc.

iv) O eixo Z é utilizado para exibir os mesmos diagramas numa outra versão. Este eixo é reservado para comparação e versionamento de modelos.

v) Para cada diagrama, ficam disponíveis diferentes tipos de visões (acessadas através do menu) que mostram informações existentes e disponíveis para cada elemento de modelagem, como: documentação, anotação, exercício, autoria, atributos/operações, pacotes, outros diagramas de um mesmo tipo, ou outros diagramas de tipos diferentes, onde o elemento também pode ser encontrado.

Apesar dos diagramas terem apenas uma pequena profundidade e parecer com um plano, quase 2D, o 3D é explorado quando a este é acrescentado novos elementos gráficos (além das cores) e com novas camadas (documentação, por exemplo), que são reconhecidos como informações extra. Além disso, existe ainda a exploração do espaço ilimitado 3D, como foi explicado acima.

5.7. Recursos da Abordagem

Segundo os critérios para uma boa visualização 3D descritos no capítulo 3, a VisAr3D apresenta as seguintes recursos:

5.7.1. Visão Geral

Um mapa geral é fornecido com todos os diagramas disponíveis, seguindo uma organização do espaço 3D apresentada, anteriormente. Todos os diagramas são simplificados, escondendo informações, a princípio, desnecessárias, deixando-o limpo, sem poluição visual. Em primeiro plano, é exibido o primeiro diagrama de classes, ou ponto de vista capturado com a *webcam*.

Ao passar o mouse sobre os elementos de modelagem, obtém-se informações básicas sobre eles. Por exemplo, o seu nome, se for uma classe, ou detalhes como tipo, nome, multiplicidade e sentido, no caso de ser um relacionamento.

Essas informações são úteis, principalmente, quando o diagrama está à distância e o seu texto não pode ser lido.

5.7.2. Informações Contextualizadas

Toda informação importante que deve ser explorada pelo aluno, como por exemplo, a indicação de que o elemento de modelagem possui documentação, aparece de alguma forma sobreposta ao elemento correspondente. Estas, especificamente, aparecem no formato de uma pequena camada de espessura presente numa classe, na cor vermelha e na cor azul, indica que a mesma pertence a mais de um diagrama e que contém documentações associadas a ela, respectivamente.

Elas podem, também, ser ícones coloridos que só aparecem quando solicitados através do menu. São de fácil percepção e ressaltam a sua correspondência ao seu conteúdo. Este recurso é interessante por exibir informações associadas a objetos gráficos, que apóiam a aula, ajudam na sua compreensão, ou constituem-se num material rico para discussão. Tornam-se, também, imprescindíveis para o estudo de um grande volume de dados dispersos na rede. Ao clicar sobre estes ícones, o usuário poderá acessar: descrições sobre detalhes do modelo, anotações sobre pontos de impasse numa reunião, indicadores de qualquer tipo, mensagens entre usuários ao discutir sobre aquela modelagem, vídeos de reunião, áudio de relatos ou qualquer documentação que contribua para o aprendizado do aluno sobre aquele diagrama ou elemento de modelagem.

Em muitos casos, informações como estas podem estar soltas ou perdidas, sem estarem associadas àquele modelo. Esta é uma maneira de resgatar informações pertinentes, que se perderiam com o tempo, e disponibilizá-las para utilização.

Este recurso explora as capacidades de percepção do aluno, permite a análise de modelos de software numa nova perspectiva e, portanto, torna possível a descoberta de semelhanças entre as partes desse diagrama, para compreender as relações mais complexas, diferentes técnicas e estilos arquiteturais.

ITOH *et al.* (2004) afirmam que é conveniente que ícones que relacionam dados em grande escala sejam utilizados (e tenham aparência) de acordo com a semântica conhecida pelo usuário para facilitar a sua busca dentre um número grande de ícones exibidos.

5.7.3. Exploração do Ambiente 3D

O Módulo Realidade Virtual da abordagem VisAr3D é exibido numa janela de um *browser* que possui botões de navegação na sua parte inferior. Através destes botões, o aluno, como se estivesse imerso no ambiente virtual, pode explorar o ambiente com a ajuda do mouse, movendo-se no espaço para a direita, para a esquerda, para cima, para baixo, para longe ou para próximo do plano, com ângulos de rotação. A Realidade Virtual como interface avançada para aplicações computacionais pode ainda fazer uso de dispositivos multisensoriais, que capturam seus movimentos e comportamento e reagem a eles, como luvas, capacetes, *caves,* entre outros, por exemplo.

A técnica de *Zoom* também está disponível, fornecendo mais detalhes sobre os dados apresentados em uma determinada visão. Para ampliar a leitura do modelo, os diagramas podem ser ampliados ou reduzidos de acordo com o tamanho desejado.

5.7.4. Agente de Busca

Outra facilidade proporcionada pela VisAr3D é um agente de busca que permite procurar documentos ou elementos de modelagem por palavras-chave ou filtros. O agente de busca analisa informações sobre os modelos e sua documentação; e o filtro analisa as condições especificadas pelo usuário. O resultado pode ser uma lista de *links* ou uma visualização gráfica através de cores ou ícones. Este subsistema permite de forma rápida e fácil encontrar e acessar todos os tipos de informação associados ao modelo de software em estudo.

Em qualquer modelo de UML, especialmente nos grandes, pode ser difícil encontrar uma informação desejada. Este problema tem duas causas principais: a quantidade de informação pode ser muito grande e esta informação pode estar distribuída entre vários diagramas.

Como mencionado anteriormente, a VisAr3D dá uma visão geral dos diagramas que descrevem o modelo. Através dos resultados acessados pela busca através de objetos gráficos coloridos, o usuário pode identificar rapidamente seus diagramas correspondentes, mesmo à distância, onde a informação desejada está localizada.

5.7.5. Ponto de Vista

O recurso Ponto de vista é usado para definir as posições para a visualização do mundo virtual; proporcionam visitas guiadas no formato de animação. Ela facilita a navegação do usuário através de pontos de interesse pré-definidos. O professor, ou o aluno, pode escolher (ou definir) uma posição no diagrama como um novo ponto de vista. Ele será capaz de mover-se, rapidamente, entre esses pontos, mesmo quando eles estão longe no diagrama.

O aluno e o professor podem utilizar os pontos de vista interessantes disponíveis, olhando para eles, a partir de perspectivas diferentes. Podem também, criar novos pontos de vista.

Este recurso ajuda a preservar o usuário do “mapa mental”, possibilitando a visão de todos os elementos relacionados em um mesmo campo visual e facilitando a evocação de conhecimento. Métodos rápidos de navegação são essenciais para que o usuário aprenda como as partes se relacionam entre si. Sem isso, torna-se muito difícil para o usuário desenvolver um mapa mental do modelo (FRANCK *et al.*, 1995).

5.7.6. Visões

Segundo SHAW (2000), projetos de software práticos e úteis não acontecem por acaso. Requerem habilidades por parte do aluno. Ela aponta deficiências no ensino e sugere soluções. Em consonância com este trabalho, a VisAr3D apresenta um conjunto de visões que permitem diferentes formas de visualização. Ela possibilita a navegação através de todos os diagramas do sistema em estudo, no qual diferentes aspectos ou perspectivas do sistema podem ser focados independentemente.

A abordagem permite aos usuários explorarem as diversas visões, sem forçá-los a seguir um caminho específico. A intenção é incentivar a uma maior criatividade e, ainda, a realizar observações interessantes. Eles podem encontrar erros, analisá-los e aprender com eles, e ainda, compartilhá-los com os colegas (através da Visão de Anotação). A VisAr3D pretende que os estudantes recuperem o gosto por dominar o conhecimento de forma agradável. A ideia é trazer um maior dinamismo à sala de aula, resultando em uma participação mais significativa dos estudantes.

Algumas dessas visões são: Visão dos Relacionamentos com Outros Diagramas, Visão dos Relacionamentos com Outros Tipos de Diagramas, Visão de Pacote, Visão de Métrica, Visão de Atributos/Operações, Visão do Autor, Visão de Documentação, Visão de Anotação, Visão de Exercícios e Visão de Comportamento.

5.7.6.1. Visão dos Relacionamentos com Outros Diagramas

Uma classe em um modelo pode ser apresentada em vários diagramas. É muitas vezes difícil e tedioso descobrir quais as dependências entre classes, ou em que lugares elas estão dentro do sistema como um todo. Esta relação não é explicitamente apresentada nos diagramas. Para entender completamente este elemento de modelagem, pode ser necessário explorar seus atributos, métodos, e relacionamentos com outros elementos que podem ser encontrados em diagramas diferentes. Nesta visão, o diagrama sinaliza através de objetos virtuais coloridos (setas coloridas) os relacionamentos externos a este diagrama em cada elemento de modelagem, por exemplo, as classes deste diagrama que pertencem a mais de um diagrama exibirá links coloridos que farão o acesso a esta mesma classe num outro diagrama (a Figura 5.7 mostra a tela da VisAr3D com a Visão dos Relacionamentos com Outros Diagramas, simulando os links coloridos). O elemento de classe é centralizado na tela à medida que o usuário muda de um diagrama para outro.

É bem sabido que a compreensão de um modelo de sistema é determinada, principalmente, ao possibilitar o estudo de um modelo, uma parte por vez (PARNAS, 1972). À medida que a parte do modelo, que é relevante para o propósito do usuário, é mostrada, o tempo do processo de aprendizagem fica bastante reduzido (HAMMOUDA *et al.*, 2004).

5.7.6.2. Visão dos Relacionamentos com Outros Tipos de Diagramas (Figura 5.8)

Esta visão destaca os elementos de modelagem que pertencem a outros tipos de diagramas utilizando *links* coloridos associados a ele. Por exemplo, a classe “Rule” pode ser explorada e estudada num diagrama de sequência. Este é um acesso a um ponto de vista (*viewpoint*) especial. Neste item do menu, o usuário visualiza objetos virtuais coloridos (ícones coloridos) sobre os elementos de modelagem que apontam para outros tipos de diagramas, onde aquele objeto foi modelado. Ao clicar neste *link*, o usuário fará o acesso a esta mesma classe num outro diagrama. O seu ponto de vista é alterado, através de uma animação entre estes dois pontos no espaço 3D.

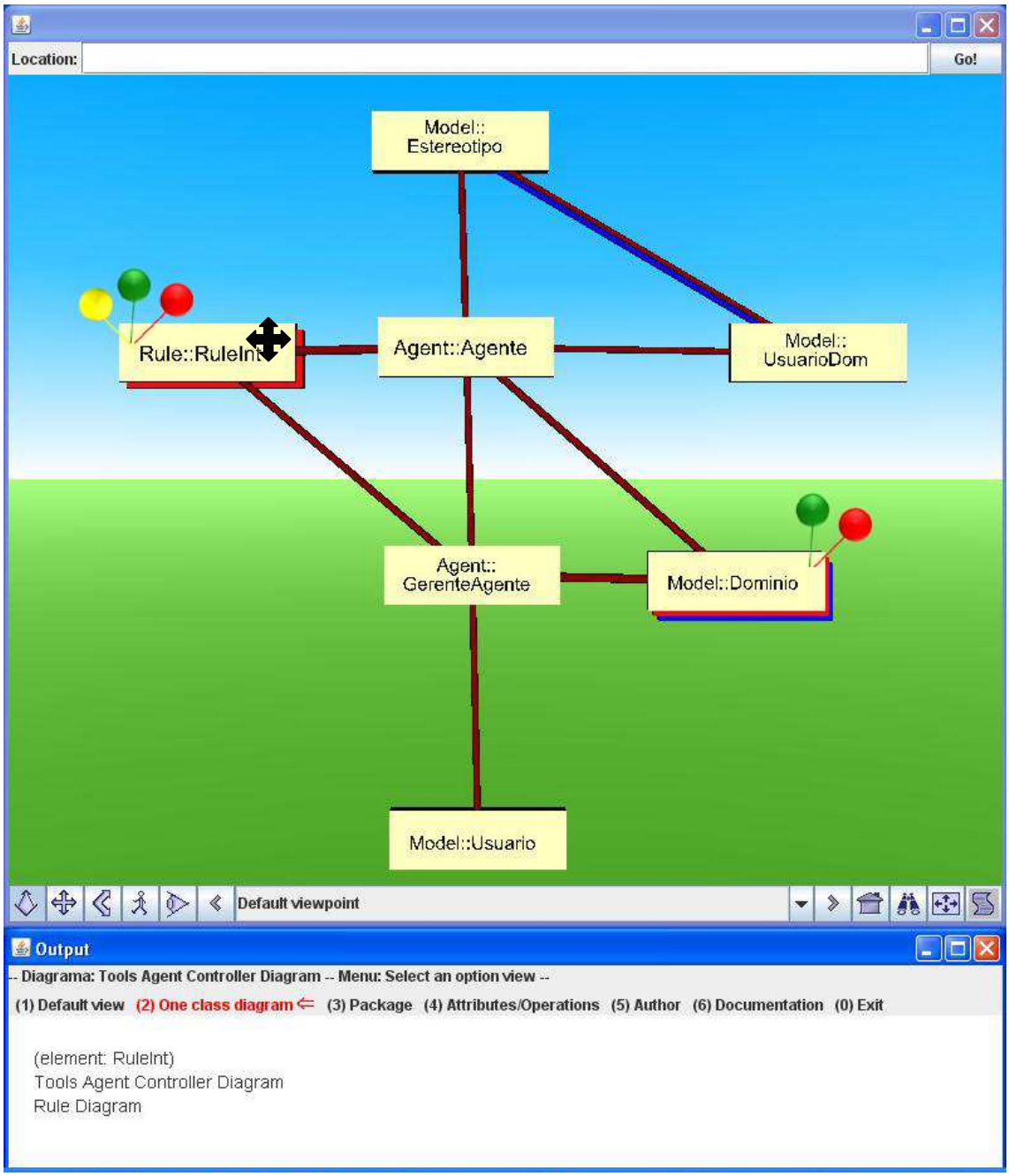


Figura 5.7: Visão dos Relacionamentos com Outros Diagramas

5.7.6.3. Visão de Pacote

Ao escolher esta visão no menu, objetos virtuais em cores são exibidos, indicando a correspondência das classes aos seus pacotes. Num universo com muitos elementos, as cores ajudam a percepção do usuário para identificação dos pacotes daquele diagrama. Ao clicar num ícone, pode-se acessar o diagrama de pacote correspondente. Isto acontece somente quando o diagrama de pacote está disponível (ou seja, se este diagrama foi modelado). Nesta visão, o usuário pode acessar este

nível de abstração, com um clique. A Figura 5.9 mostra a tela da VisAr3D com a Visão de Pacotes, simulando os links coloridos.

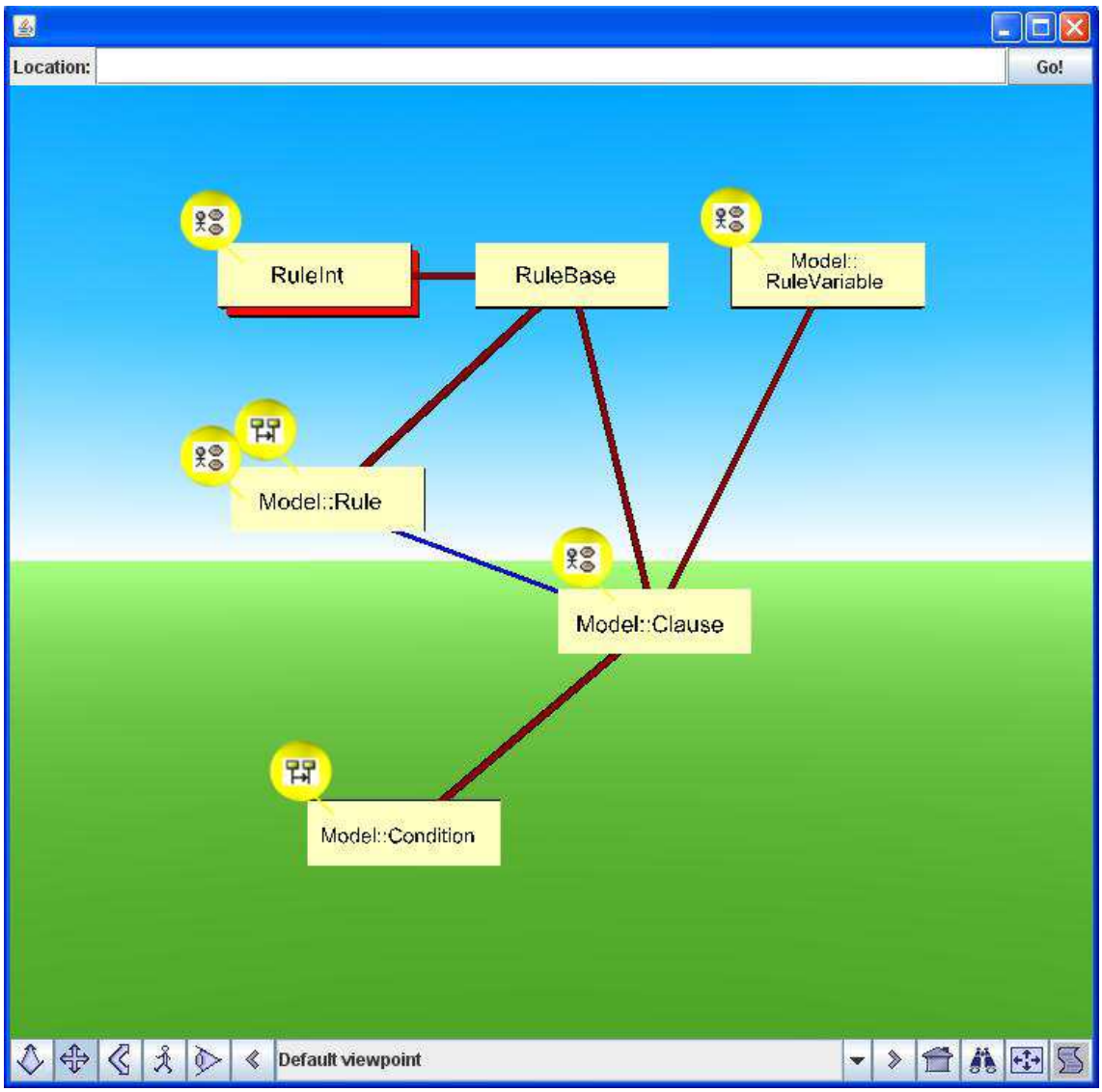


Figura 5.8: Visão dos Relacionamentos com Outros Tipos de Diagramas (simulação)

5.7.6.4. Visão de Métrica

A VisAr3D combina o layout existente dos diagramas de classe UML com a visualização de métricas. À medida que a aplicação de métricas para um modelo UML pode resultar em uma enorme quantidade de dados, e esses dados são normalmente apresentados em tabelas, é um trabalho difícil para um engenheiro de software, ou um estudante, fazer o mapeamento entre os valores das métricas de uma tabela e as classes de um diagrama UML, manualmente. A VisAr3D apoia esta atividade através da integração da visualização do modelo e das métricas, que foram extraídas do arquivo XMI, usando cor, tamanho e forma para visualizar valores. A Figura 5.10

mostra um exemplo de quatro diferentes métricas visualizadas sobre um diagrama de classes. Este estudo reduz a quantidade de métricas disponíveis a um conjunto de complexidade gerenciável que podem ser utilizadas como indicadores para a compreensão prévia. Este item do menu pode ou não estar disponível.

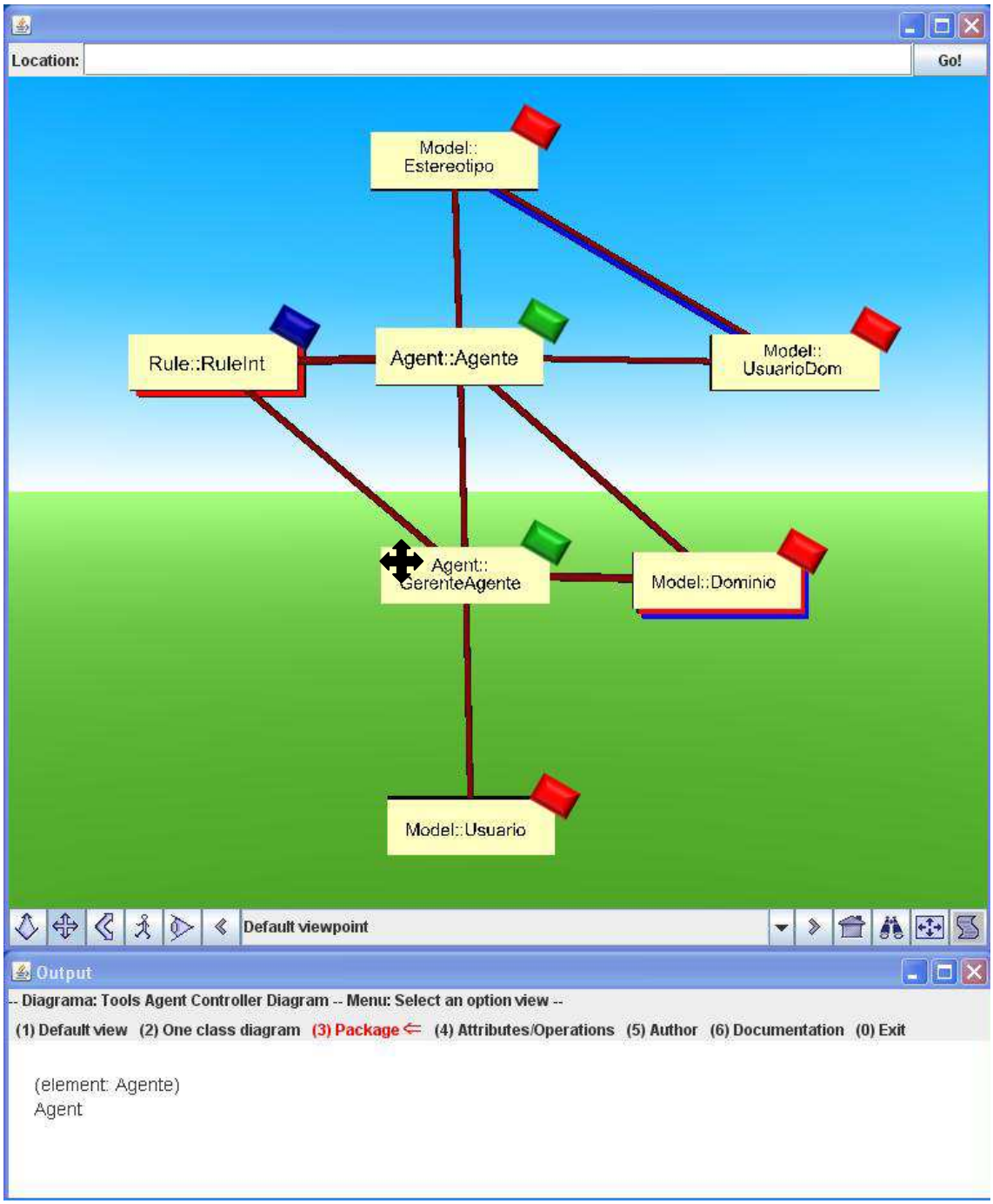


Figura 5.9: Visão de Pacote

5.7.6.5. Visão de Atributos/Operações

Os atributos e operações das classes de um diagrama só aparecem quando solicitadas pelo usuário na Visão de Atributos/Operações, para evitar a poluição de informações de grandes diagramas. Num sistema complexo, as classes possuem muitos atributos e operações que desarrumam o diagrama e comprometem sua leitura. Esta visão é essencial para obter clareza dos diagramas. O usuário passa o cursor sobre o elemento de modelagem e estas informações são exibidas em um tamanho legível numa janela (Figura 5.11).

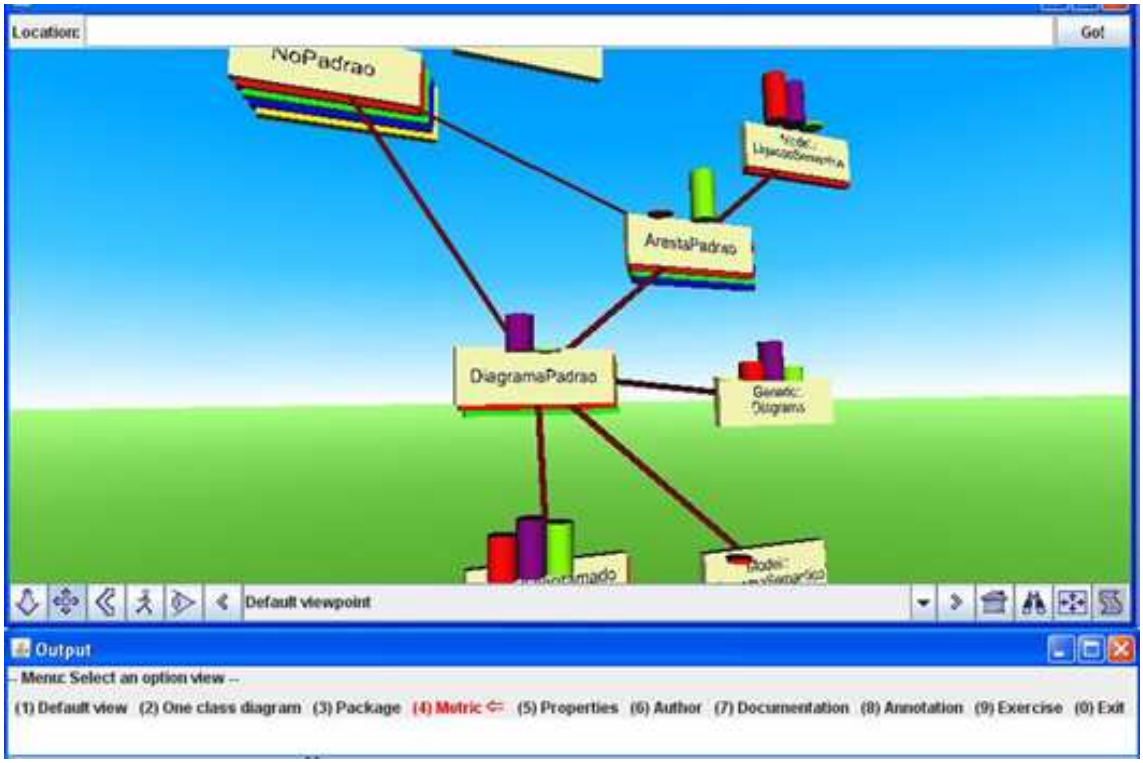


Figura 5.10: Visão de Métrica (simulação)

5.7.6.6. Visão do Autor (Figura 5.12)

A VisAr3D permite a visualização da autoria do modelo. À distância, num conjunto de muitas classes e relacionamentos, ícones coloridos indicam os autores dos elementos de modelagem. Na Visão do Autor, o nome do autor é apresentado quando o usuário move o cursor sobre os elementos de modelagem.

5.7.6.7. Visão de Documentação

O desenvolvimento de um projeto de arquitetura muitas vezes produz uma grande quantidade de documentação, bem como registros das decisões iniciais sobre o projeto para facilitar a comunicação entre os *stakeholders*. Esta informação é muito

importante para os estudantes de modelagem de software, especialmente para os novatos, para compreender todo o processo de criação e desenvolvimento do software. Durante a navegação através do modelo, os alunos terão acesso a esta informação sobreposta (profundidade na cor azul, conforme a Figura 5.13), à medida que se movem através do espaço do modelo. É fácil perceber sua correspondência com o seu conteúdo, pela sua identificação graficamente. À medida que o usuário optar por ler um documento, mais que um elemento de modelagem, associado a ele, é marcado. A intenção aqui é valorizar a boa documentação e incentivar os alunos a sempre documentar.

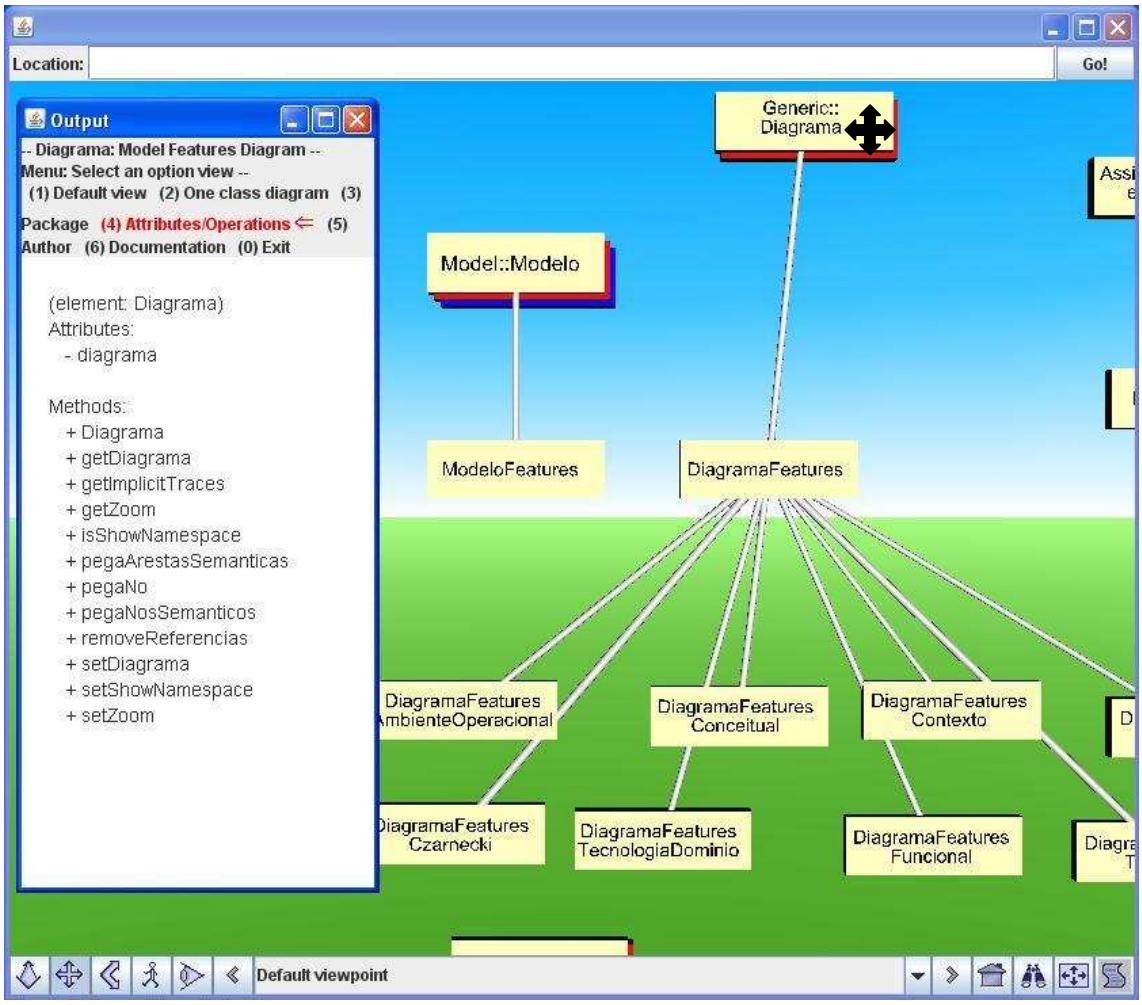


Figura 5.11: Visão de Atributos/Operações

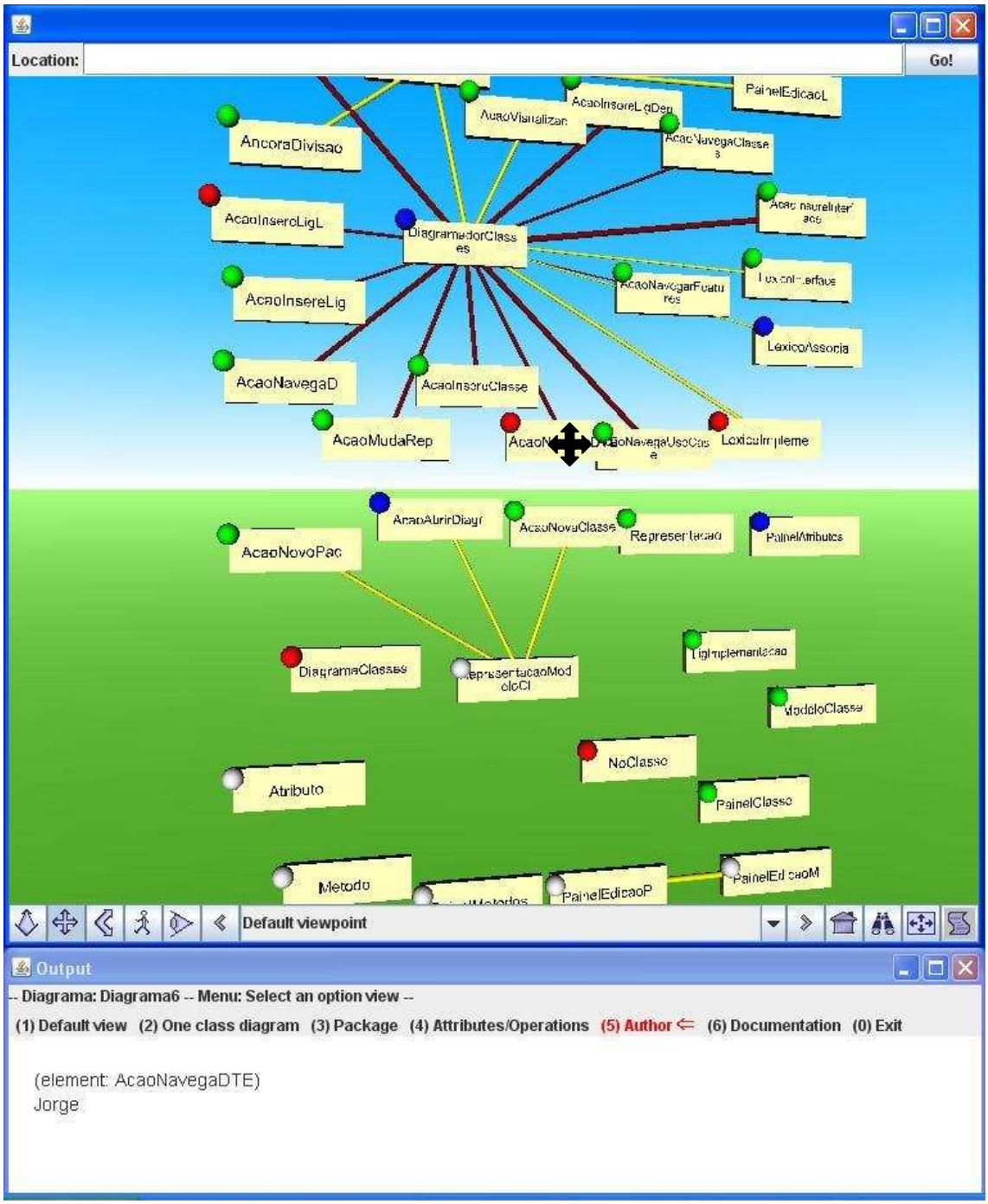


Figura 5.12: Visão do Autor

5.7.6.8. Visão de Anotação

A VisAr3D, como ferramenta pedagógica interativa, deve permitir a troca de mensagens e informações entre alunos e entre alunos e professores, tornando as aulas mais interessantes e favorecendo a construção do conhecimento dos alunos por meio da aprendizagem colaborativa.

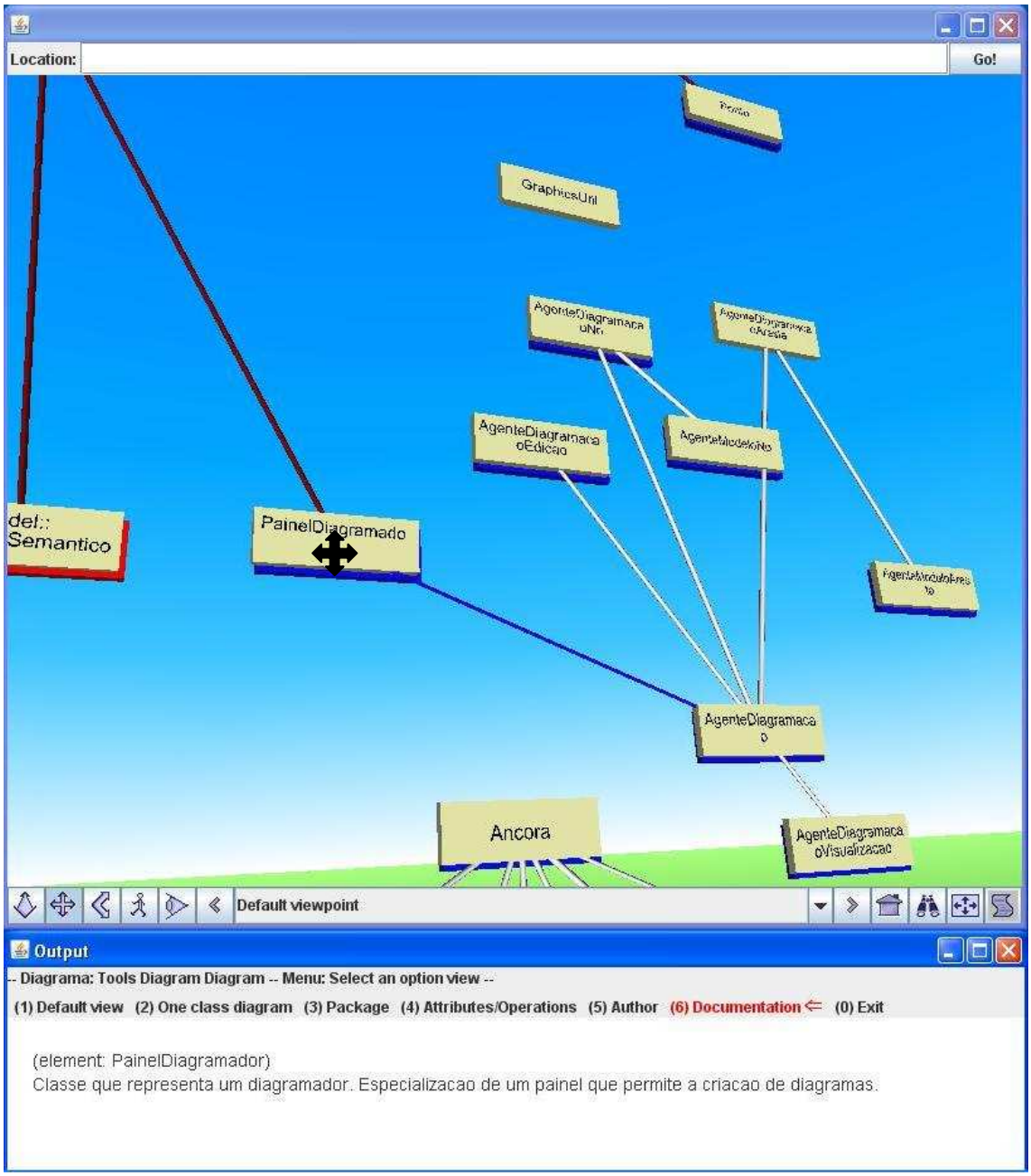


Figura 5.13: Visão de Documentação

Um dos pressupostos teóricos e diretrizes dos estudos de Piaget é fazer com que a interação social e a linguagem tenham um lugar proeminente na programação diária de ensino, estimulando a interação verbal entre os alunos e promovendo atividades de grupo que envolvam cooperação e troca de ideias (HAYDT, 2006).

As anotações são informações geradas dentro do espaço virtual. Servem para facilitar a compreensão dos modelos de software e, conseqüentemente, melhorar a comunicação entre os membros da equipe que constroem o sistema. Através de situações espontâneas e, às vezes, até lúdica, um estudante pode explorar, criar e,

especialmente, comunicar-se com outros alunos ou o professor para obter novas informações e para construir o seu próprio processo de aprendizagem, através do compartilhamento de descobertas, dúvidas ou mensagens simples, utilizando anotações.

Todo o histórico desta comunicação é armazenado no repositório da VisAr3D. Elas são valiosas para ajudar no entendimento de pontos polêmicos e na elaboração da documentação de toda modelagem. Este ambiente de troca de informações é propício para a incentivar a participação, interação e colaboração entre os alunos, não excluindo a interação face-a-face.

A abordagem possibilita a criação, alteração e exclusão das anotações durante a exploração.

5.7.6.9. Visão de Exercícios

Esta visão permite o professor interagir com os alunos através de exercícios que possibilitam melhorar o seu conhecimento sobre a teoria estudada. Essa visão é semelhante a da anotação e da documentação, mas é classificada como Visão de Exercício. Os alunos podem ser convidados a escolher entre alternativas de projeto, ser obrigados a fazer escolhas para atender às necessidades do cliente, ser solicitados a reutilizar componentes, a aderir aos padrões e, também, a corrigir elementos errados, ser encorajados a usar "Estudos de Caso", ou podem ser desafiados a refatorar uma parte do modelo. Esta é outra forma de informação gerada no espaço virtual. O professor pode criar situações para que o aluno, independentemente ou auxiliado por ele, aplique os conhecimentos adquiridos na solução de problemas diversos.

5.7.6.10. Visão de Comportamento

O comportamento é mais bem visualizado por meio de diagramas animados onde as mensagens são símbolos que passam do remetente para o receptor. Durante uma animação deste tipo, os símbolos de mensagens não processados são mostradas em cinza e os elementos envolvidos saltam para frente do plano do diagrama. A terceira dimensão pode ser empregada de forma útil para fornecer visualizações com mais semântica e animadas, como nos casos dos diagramas de sequência e de colaboração, por exemplo. A VisAr3D disponibiliza a visualização animada destes diagramas, permitindo ao aluno exercitar, utilizar o conhecimento que ele já tem ou buscar novas informações, enfatizando o raciocínio e a reflexão.

Outro ponto que é visto como um problema na UML é que cada tipo de diagrama é apresentado separadamente, isto resulta na ocultação da relação entre diferentes diagramas e elementos do modelo. A abordagem VisAr3D utiliza os benefícios da terceira dimensão, de interatividade e de animação, na visualização da combinação de aspectos estáticos e aspectos dinâmicos dos sistemas num único diagrama. A ideia é usar aspectos diferentes de um sistema no mesmo ponto de vista. Por exemplo, utilizar um diagrama de classes e acrescentar ou combinar a animação de um diagrama dinâmico, como o diagrama de sequência. Desta forma, utiliza-se um único diagrama mantendo as vantagens de ambos.

5.8. Protótipo Implementado

O principal objetivo do desenvolvimento do protótipo VisAr3D foi mostrar a viabilidade da tecnologia que se deseja desenvolver. Ele possui algumas limitações de interface e usabilidade, contudo ajudou a mostrar algo concreto e executável aos usuários, ou *stakeholders*. Ele foi projetado para ser um pequeno "exemplar" para testar algumas suposições importantes sobre a abordagem, quanto à funcionalidade, tecnologia e sua contribuição quanto ao ambiente 3D. Ele não tenta reproduzir o sistema final. Em vez disso, enfatiza o que este sistema fará, conforme a visão dos usuários.

Há um aumento pela demanda de Mundos Virtuais cada vez melhores, mais atraentes, mais eficazes e mais dinâmicos. Devido à exigência destes tipos de aplicações, seu desenvolvimento se torna complexo, demorado, caro e requer bastante experiência. Segundo DE TROYER *et al.* (2009), isso faz com que haja uma distância (*gap*) grande entre o domínio da aplicação e o nível em que o ambiente virtual precisa ser especificado, e torna a tradução dos conceitos do domínio da aplicação em conceitos de implementação uma tarefa muito difícil.

Embora haja uma variedade de ferramentas para construir objetos gráficos 3D animados através de simples funções, a animação, por exemplo, destes objetos ainda tem que ser programada usando uma linguagem de programação com comandos mais complexos. Esta forma de trabalhar não só requer pessoas qualificadas, como também aumenta o tempo de desenvolvimento e custo de construção deste mundo virtual (PELLENS *et al.*, 2008).

Para o desenvolvimento do protótipo da VisAr3D, foram desenvolvidas e testadas uma série de implementações, com diferentes linguagens de programação, assim como uma combinação de técnicas para escolher a mais adequada para a

criação da visualizações 3D de modelos UML. A ferramenta foi criada de uma forma incremental.

Esta Seção apresenta as funcionalidades do protótipo, organizadas de acordo com os módulos da visão geral da abordagem VisAr3D, apresentada na Seção 5.4. O termo “VisAr3D” será utilizado como referência ao protótipo, por motivo de simplificação.

5.8.1. Módulo Arquitetural

No Módulo Arquitetural, durante o desenvolvimento do protótipo, foi utilizada uma ferramenta comercial 2D para criar os modelos que seriam importados pela VisAr3D. Optou-se por utilizar uma ferramenta UML do mercado que suportasse UML 2.0 e importasse e exportasse arquivos XMI, para permitir o intercâmbio de modelos entre a ferramenta 2D e a VisAr3D.

A ferramentas UML comercial selecionada para criar e documentar os modelos foi a *Enterprise Architect* (EA) versão 7.0 (ENTERPRISE ARCHITECT, 2011) da Sparx Systems. Entre as ferramentas no mercado, esta foi a mais adequada por vários motivos: (i) a EA suporta UML 2.0; (ii) a EA inclui importação e exportação de arquivos XMI para permitir o intercâmbio de modelos entre ferramentas; e (iii) pelo seu baixo custo, ela é bastante utilizada, tanto no meio acadêmico (incluindo meu grupo de trabalho), quanto na indústria.

5.8.1.1. XMI (XMI, 2005)

XMI (*XML Metadata Interchange*) é um padrão da OMG (*Object Management Group*) que usa informações UML para criar documentos XML. O XMI não é uma extensão do XML, mas se baseia nele. Através do uso de XML, informações podem ser transferidas entre os programas de modelagem que tem por base a UML. Portanto, a possibilidade de utilização de arquivos XMI mantém a independência na utilização de uma ferramenta UML específica.

Para utilizar a VisAr3D, um arquivo XMI é importado depois de ter sido criado em uma ferramenta UML e diagramas na terceira dimensão são gerados, automaticamente, a partir dele. De posse das informações fornecidas pelo arquivo XMI (com aproximadamente, 16.737 linhas), extraídos através de comandos Java, a VisAr3D é capaz de:

1) Identificar todos os elementos de modelagem de cada diagrama criado e suas posições origem no modelo 2D.

2) Reconhecer todos os diferentes tipos de associações existentes, relacionados a cada classe, ou pacote, localizando-os em diferentes diagramas.
3) Reconhecer todos os atributos e operações de cada classe.
4) Reconhecer os pacotes, aos quais cada classe pertence.
5) Identificar os diferentes diagramas aos quais uma classe, ou pacote, pode pertencer.
6) Reconhecer a autoria e a documentação associadas aos elementos de modelagem.

Depois de interpretadas todas estas informações, a VisAr3D posiciona todos os diagramas no mundo virtual, acrescentando uma pequena espessura aos elementos e adicionando ícones e cores a eles.

Durante a utilização da VisAr3D, informações, como as métricas, devem ser adicionadas ao XMI (que foi exportado pela ferramenta *Enterprise Architect*). Neste caso, outra ferramenta, disponível no mercado, deve ser utilizada para fornecer estes indicadores, com seus respectivos valores.

5.8.2. Módulo Realidade Aumentada

Este Módulo tem como objetivo tornar os modelos exibidos pelo professor mais acessíveis. Ele aponta, na rede de computadores, para todos os arquivos do projeto que aquele diagrama pertence. E ainda localiza o ponto de vista exato da projeção. Ou seja, no ambiente tridimensional, o usuário visualiza este diagrama e todos os diagramas do projeto no mesmo ponto de vista da projeção exibida pelo professor. O professor utiliza o modelo 2D e, através dele, disponibiliza o acesso de todo o sistema modelado aos seus alunos. Este pode ser considerado como mais um recurso que reduz sua carga de aprendizado ao utlizar a ferramenta.

A implementação do Módulo Realidade Aumentada não foi concluída, apesar da tecnologia ter sido aprendida durante o desenvolvimento de um protótipo de Realidade Aumentada.

5.8.2.1. ARToolkit

Desenvolver aplicações de Realidade Aumentada (RA) ainda é uma tarefa desafiadora, até mesmo depois de alguns anos de pesquisa, avanço tecnológico, disponibilidade de produtos com custos acessíveis, e da construção de protótipos. Vários grupos de pesquisa têm desenvolvido uma série de aplicações para RA, que permite a criação de software para explorar as possíveis interfaces da RA. A biblioteca

ARToolkit é um exemplo marcante de recurso gratuito e livre, mas existem vários outros disponibilizados por pesquisadores e, mais recentemente, por empresas.

O ARToolKit foi, originalmente, desenvolvido para servir de apoio na concepção de interfaces colaborativas pelo Dr. Hirokazu Kato, na Universidade de Osaka. E, desde então, tem sido mantido pelo *Human Interface Technology Laboratory* (HIT Lab) da Universidade de Washington e pelo HIT Lab NZ da Universidade de Canterbury, em Christchurch. O ARToolKit é uma biblioteca de código aberto, escrita na linguagem C, para concepção de aplicações em RA. O pacote inclui bibliotecas de rastreamento e disponibiliza o código fonte completo, tornando possível o transporte do código para diversas plataformas, ou adaptá-lo para resolver as especificidades de suas aplicações. O rastreamento óptico oferecido pelo ARToolkit possibilita extrair, de forma rápida, a posição e orientação de padrões marcadores, apenas com o uso de um computador e uma *webcam* convencional.

O funcionamento do ARToolKit baseia-se no uso de marcadores (cartões de papelão com um símbolo impresso e uma moldura retangular), permitindo o uso de técnicas de visão computacional para calcular a posição da câmera e sua orientação em relação aos marcadores, de forma a fazer com que o sistema possa sobrepor objetos virtuais sobre os marcadores. A saída, portanto, nada mais é do que uma imagem formada de um ou mais objetos virtuais, animados ou estáticos, sobre o marcador.

5.8.2.2. Protótipo de RA

O protótipo de RA foi feito, utilizando a biblioteca ARToolkit. Ele foi desenvolvido com a finalidade de facilitar a aprendizagem do comportamento estrutural de modelos qualitativos no ensino básico de arquitetura e engenharia (RODRIGUES *et al.*, 2008).

Os objetos virtuais visualizados no protótipo foram implementados com C++ e OpengGL (OPENGL, 2011), uma API para criação de aplicações gráficas 2D e 3D.

Para finalizar o Módulo Realidade Aumentada da VisAr3D, serão utilizados os comandos de captura e reconhecimento do padrão gráfico e exibição de objetos virtuais idênticos ao protótipo. Poucas alterações ou comandos novos precisarão ser implementados.

5.8.3. Módulo Realidade Virtual

É no Módulo Realidade Virtual onde estão presentes as funcionalidades do protótipo do VisAr3D. Contudo foram implementados inicialmente dois outros

protótipos que demandaram um investimento significativo no estudo de linguagens, ferramentas e ambientes de desenvolvimento de software. O desenvolvimento dos dois primeiros protótipos de Realidade Virtual (RV) teve como objetivo servir como base para a exploração das tecnologias atualmente disponíveis e como fonte de motivação no desenvolvimento deste trabalho. Antes do desenvolvimento do protótipo VisAr3D foram desenvolvidos dois protótipos de RV que são descritos a seguir:

5.8.3.1. 1º protótipo de RV

Um protótipo de RV foi desenvolvido utilizando o mesmo tipo de aplicação do protótipo de RA, contudo voltado agora para a RV (RODRIGUES e WERNER, 2009a).

O VRML foi a linguagem de programação utilizada neste protótipo. A linguagem de programação VRML, abreviação de *Virtual Reality Modeling Language*, ou Linguagem para Modelagem em Realidade Virtual, surgiu da ideia de se levar Realidade Virtual para a Internet. Ela foi desenvolvida por Mark Pesce e Tony Parisi em 1994. A VRML impulsionou o desenvolvimento da RV no mundo, sendo uma das primeiras iniciativas bem sucedidas de software livre. Segundo STANEK (1996), a VRML é uma linguagem de criação de ambientes virtuais em três dimensões através de uma definição textual de objetos a serem desenhados por um *browser*. Ela permite ainda a interação entre o usuário e este ambiente. A interatividade se dá através do uso de funções básicas de navegação e também pela interferência do visitante que pode clicar sobre objetos que determinam ligações com outras cenas, ou páginas *web*, arquivos de texto, imagens, sons, vídeos; ou possuem comportamentos associados. Devido ao formato de arquivo, ela pode ser facilmente transportada entre diversas plataformas. Para a visualização e manipulação deste ambiente virtual, é necessária a utilização de um software de navegação para internet (*browser*) que possua um *plugin* de reconhecimento dos códigos VRML. Este plugin instalado no *browser* é o encarregado de interpretar o código e gerar o ambiente descrito por ele (HEIDRICH, 2003).

5.8.3.2. 2º protótipo de RV

Um segundo protótipo de RV foi desenvolvido voltado para a proposta desta tese. Ele foi utilizado como primeiro teste com a interface, movimentação no mundo virtual e funcionalidades. Ele foi implementado utilizando a linguagem sucessora da VRML, a X3D. A Figura 5.14 mostra uma tela deste protótipo e a Figura 5.15 mostra um trecho do código implementado com X3D.

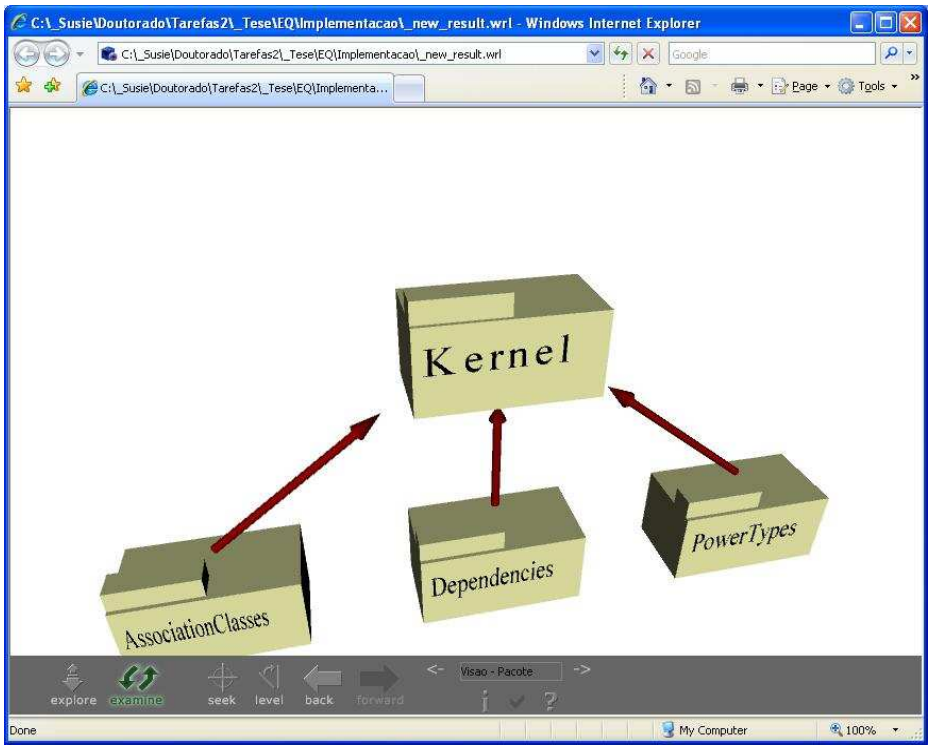


Figura 5.14: Tela do 2º protótipo de RV

```
<Viewpoint description="Visao geral 2" orientation="1
<Viewpoint description="Visao - Perto" orientation="1
<Viewpoint description="Visao - Pacote" position="0 0
<LOD range="20 49">
  <Group DEF="View3dModelAtCloseRange">
    <Transform translation="0 0 0">
      <Shape DEF="DEF_Classe">
        <Appearance>
          <Material DEF="CorClasse" diffuseColor="0.99
        </Appearance>
        <Box size="2 1 1"/>
      </Shape>
```

Figura 5.15: Trecho do código em X3D

5.8.3.3. X3D (X3D, 2011)

Para a implementação do ambiente virtual da VisAr3D, foi utilizada a linguagem sucessora do VRML, o X3D. O X3D (eXtensible 3D) é um padrão aberto do Consórcio Web3D para distribuir conteúdos de RV em 3D, em especial pela Internet. Esta linguagem é capaz de representar e comunicar cenas tridimensionais e objetos, desenvolvidos com a sintaxe XML. Seus objetos são descritos como formas geométricas, e seus comportamentos podem ser controlados a partir da cena, tanto

interna como externamente, pelo arquivo X3D através linguagens de programação ou de script.

O fato de ser uma linguagem baseada em XML permite ao X3D vantagens como melhor interoperabilidade entre aplicações.

O X3D fornece um rico conjunto de recursos de modelagem, projetado para a criação de mundos 3D, que também pode ser aplicado para a visualização de software, contudo o desenvolvimento de um mundo virtual X3D dinâmico não é uma tarefa fácil (PELLENS *et al.*, 2008). Com o desenvolvimento ativo da tecnologia dos browsers, a tecnologia do X3D está melhorando, rapidamente.

Além das razões acima citadas, o X3D foi escolhido para o desenvolvimento desta tese por ele ainda ser capaz de capturar um ambiente virtual completo, contendo 3D, 2D, animações, áudio, vídeo, interação com o usuário e simulações.

Para que a visualização dos arquivos X3D seja possível no *browser*, fez-se necessário a instalação de um *plugin* específico, o *Flux Player da Media Machines* (FLUX DEVELOPER WIKI, 2011).

5.8.3.4. Protótipo VisAr3D

O protótipo da ferramenta VisAr3D, atualmente, fornece uma visão estática de um modelo de sistema em grande escala. Ele foi desenvolvido usando Java, X3D, Xj3D (XJ3D, 2011) e XMI versão 2.1. A presente versão da VisAr3D roda apenas no ambiente Windows.

5.8.3.5. XJ3D e SAI

Neste protótipo foi utilizada a API disponível para manipulação do padrão X3D, o Xj3D. O Xj3D também é um padrão aberto do Consórcio Web3D, totalmente desenvolvido na linguagem Java. Ele oferece um conjunto de ferramentas totalmente desenvolvidas em Java, composto de um *browser* capaz de interpretar o padrão X3D e uma biblioteca Java para manipulação dinâmica do padrão, permitindo a construção de gráficos 3D. Para acesso e modificação destes conteúdos, através da comunicação entre o X3D e o Java, são utilizados os métodos da API chamada SAI (*Scene Access Interface*), também parte integrante da especificação X3D. A Figura 5.16 mostra um trecho do código do protótipo VisAr3D.

A Figura 5.13 mostra um exemplo da tela principal da VisAr3D. Ela é dividida em duas partes: um ambiente virtual onde são exibidos os diagramas e uma janela *output* que exibe as informações sobre os elementos de modelagem.

Cada elemento gráfico 3D criado no ambiente virtual é explicado a seguir:

(1) Classe: são objetos virtuais 3D retangulares na cor amarela, identificadas pelo seu nome. Os seus atributos e operações são exibidos somente quando solicitados pelo usuário. Ela pode apresentar outras camadas com uma pequena profundidade nas cores vermelha e azul, representando que esta classe pertence também a outro diagrama e que esta classe possui documentação associada a ela.

(2) Relacionamentos: os relacionamentos são tubos coloridos que ligam as classes. Nesta primeira versão do protótipo, os direcionamentos dos relacionamentos não foram desenhados, por motivos de decisão de interface. Na dúvida de como representar em 3D as setas dos relacionamentos, pois, em 3D, a seta de uma associação simples poderia se confundir com a da generalização, optou-se apenas em colorir os relacionamentos. Elas possuem cores diferentes indicando os seus diferentes tipos. Na cor vinho, o relacionamento representa uma associação simples; na cor branca, uma generalização; na cor amarela, uma realização; e na cor azul, uma agregação. Ao passar o mouse sobre a associação, mais informações são exibidas como o sentido do relacionamento, as classes envolvidas, o seu nome, o seu tipo e a sua cardinalidade.

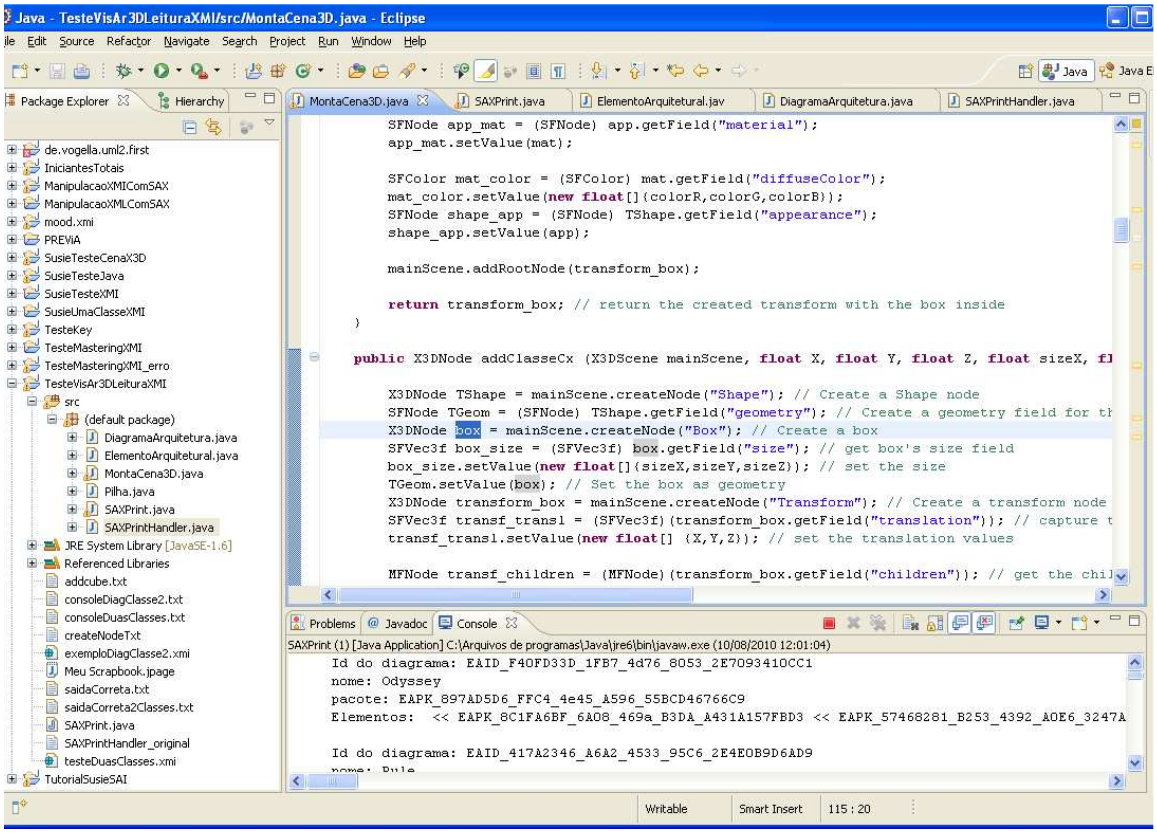


Figura 5.16: Código do Protótipo VisAr3D

(3) Em relação à movimentação no ambiente virtual, os botões de navegação na parte de baixo da janela, são os seguintes da esquerda para a direita:

i) O primeiro botão permite o afastamento e a aproximação do objeto 3D;

ii) O segundo botão move o objeto para a esquerda, para a direita, para cima e para baixo;

iii) O terceiro botão sobe e desce o plano;

iv) O quarto botão gira em torno do objeto;

v) O quinto botão permite a aproximação, o afastamento, o deslocamento para a esquerda e para a direita; e

vi) O botão no formato de uma casinha, restaura a visão inicial.

A janela *output* possui as seguintes informações:

(1) Nome do diagrama corrente: esta informação corresponde ao diagrama que está centralizado na tela.

(2) O menu que acessa as funcionalidades da VisAr3D pode ser acionado, tornando a janela output corrente e escolhendo o número que corresponde ao menu desejado. Eles são descritos a seguir:

i) O menu (1) apresenta a visão Default, onde o nome da classe ou relacionamento é exibido ao passar o mouse sobre o elemento de modelagem.

ii) O menu (2) apresenta a Visão dos Relacionamentos com Outros Diagramas, onde informações sobre as classes que pertencem a mais de um diagrama são exibidas.

iii) O menu (3) apresenta a Visão de Pacote, que exibe o nome do pacote a qual a classe selecionada pertence.

iv) O menu (4) apresenta a Visão de Atributos/Operações que exibe na janela de *output* os atributos e as operações quando o mouse está sobre o elemento de modelagem.

v) O menu (5) apresenta a Visão do Autor, que exibe esferas coloridas sobre os elementos virtuais em cores diferentes que representam diferentes autores.

vi) O menu (6) apresenta a Visão de Documentação, que exibe na janela output a documentação associada àquele elemento de modelagem.

5.8.3.6. Limitações do protótipo

Para o desenvolvimento do protótipo, decidiu-se implementar um conjunto mínimo de funcionalidades que não comprometessem a sua utilização e a exploração por parte dos usuários. No entanto, algumas funcionalidades ficaram comprometidas devido a problemas técnicos. As principais limitações da VisAr3D são:

1) Janela *output*: inicialmente, optou-se por utilizar uma segunda janela para o posicionamento do menu e para a exibição das informações solicitadas junto aos diagramas. Esta simplificação contradiz a ideia de utilizar somente uma cena para a apresentação das informações sobre os modelos.

2) *Browser*: o *browser* utilizado foi o *browser* do XJ3D, que não obteve um desempenho muito bom junto a programação em Java. A ideia é pesquisar a possibilidade de mudar o *browser*, mesmo ainda trabalhando com o XJ3D.

3) Menu: o menu no protótipo fica na janela *output*, sendo acessado através do teclado.

4) Navegabilidade: o tempo de resposta é grande na navegação dos diagramas. Este problema pode ser solucionado com a alteração do *browser*, ou com através de uma próxima versão do mesmo. Ao utilizar o *browser* do XJ3D externamente, sem a programação Java, o seu desempenho fica bem melhor. Portanto, o problema de navegabilidade pode ser relacionado a um erro (*bug*) com fácil solução.

5) Falta de animações: a VisAr3D, por enquanto não apresenta animações.

6) Simplificação do modelo UML: a falta de direcionamento nos relacionamentos foge um pouco da notação UML. Este problema pode ser rapidamente solucionado após decisões de interface.

5.9. Considerações Finais

Pelas suas características ligadas especificamente à Realidade Virtual (RV) e a Realidade Aumentada (RA), a abordagem VisAr3D (Visualização de Arquitetura de Software em 3D) possui em si, potencialidades que, na área da Educação, podem transformá-la num poderoso instrumento a serviço da Academia, que procura a mudança e a evolução no ensino de Engenharia de Software em geral, e de Modelagem de Sistemas, em especial. Estas tecnologias reúnem em si várias especificidades e atributos que podem ajudar nas múltiplas situações e contextos de pesquisa e aprendizagem.

Com a RV e a RA, a assimilação do conhecimento pode se tornar muito mais simples. Imersos no mundo virtual, os alunos participam de uma viagem exploratória que suscita a curiosidade, o interesse, o desafio e, conseqüentemente, a motivação, fundamental em qualquer contexto educativo. Por esta razão, a sua adaptação é mais rápida e há, portanto, uma maior probabilidade de um maior engajamento, por parte dos alunos, na aula.

Ao interagir com este mundo envolvente, o estudante pode obter dele, em tempo real, a percepção adequada, a possibilidade de análise numa nova perspectiva e a compreensão das relações mais complexas dentre um número grande de elementos de modelagem exibidos.

Deve-se ressaltar que durante todo o processo de utilização da VisAr3D, o papel do professor continua a ser fundamental. Ele é visto como um orientador, que trabalha conjuntamente com os seus alunos e os auxilia, aprendendo com eles, simultaneamente.

Uma característica importante da ferramenta, também, é a geração automática de modelos 3D, a partir de arquivos XMI. Estes diagramas são capazes de fornecer uma semântica mais rica do que seu correspondente em 2D.

Este capítulo apresentou a abordagem proposta, baseada na caracterização do problema e numa lista de requisitos. Uma solução foi apresentada para atendê-los, utilizando o potencial que a combinação das tecnologias de RV e RA tem para oferecer como suporte à educação. O protótipo desenvolvido implementou parte das características da abordagem, contudo ajudou a mostrar a viabilidade da tecnologia.

Em resumo, apostando na crescente demanda pela análise visual do software, a VisAr3D se propõe a apoiar o desenvolvimento e a participação dos alunos em projetos complexos, reduzir a distância entre a teoria e a prática e apoiar a dinâmica em sala de aula, utilizando recursos que incluem a visualização 3D como diferencial facilitador e motivador para os alunos ao aprenderem Modelagem de Sistemas.

No próximo capítulo, será apresentado o estudo experimental realizado para avaliar a viabilidade do apoio oferecido pelo visualizador UML 3D a alunos da disciplina de Modelagem de Sistemas, bem como a contribuição da inserção da terceira dimensão.

Capítulo 6 - Avaliação da Abordagem

6.1. Introdução

Há evidências de que é possível perceber e compreender melhor os sistemas de software cada vez mais complexos, se eles são exibidos como objetos gráficos em um espaço tridimensional (WARE *et al.*, 1993).

Segundo MAFRA e TRAVASSOS (2006), a engenharia de software experimental (ou engenharia de software baseada em evidências) permite a caracterização de uma tecnologia em uso, sendo possível determinar, com níveis razoáveis de segurança, o que funciona e o que não funciona, e sob quais circunstâncias.

Portanto, um estudo de viabilidade foi planejado e conduzido para atingir um nível adequado de evidência a respeito das tecnologias utilizadas no contexto desta tese. A avaliação da abordagem, realizada em julho de 2011, foi feita através do protótipo VisAr3D, que apesar de implementar parte de suas funcionalidades, ajudou a analisar algumas suposições importantes sobre a abordagem, quanto à funcionalidade, tecnologia e sua contribuição quanto ao ambiente 3D. Este capítulo apresenta os detalhes deste estudo.

Este capítulo é organizado da seguinte forma: nesta seção (Seção 6.1), foi apresentada a Introdução ao capítulo. A Seção 6.2 define os objetivos do estudo e a Seção 6.3 descreve o Planejamento do Estudo Experimental. Na Seção 6.4, é apresentada a Execução do Estudo, e na Seção 6.5, a Análise e Interpretação dos Resultados. O capítulo finaliza com a Seção 6.6, que apresenta as Considerações Finais.

6.2. Definição dos Objetivos

6.2.1. Objetivo Global

O propósito deste estudo é verificar a capacidade da abordagem VisAr3D (através do protótipo de mesmo nome) de contribuir para a compreensão de modelos UML em sistemas com muitos elementos de modelagem.

6.2.2. Objetivo do Estudo

Seguindo a abordagem GQM – *Goal/Question/Metric* (BASILI *et al.,* 1994), o objetivo do estudo pode ser descrito como:

Analisar	o uso do protótipo VisAr3D comparado ao uso da ferramenta *Enterprise Architect*
Com o propósito de	Caracterizar
Com respeito a	precisão, cobertura, tempo e percepção do usuário no uso e adoção de novas tecnologias
Do ponto de vista do	Pesquisador
No contexto da	execução de tarefas por alunos, semelhantes às aplicadas na disciplina de Modelagem de Sistemas, utilizando um sistema com muitos elementos de modelagem.

6.3. Planejamento

6.3.1. Definição das Hipóteses

A exploração da modelagem de um sistema de software com muitos elementos de modelagem é uma tarefa muito complexa. Geralmente, sistemas como estes são de difícil entendimento; lidar com o sistema como um todo é impossível, e para entender os seus detalhes, é preciso dividi-lo em partes; informações importantes muitas vezes ficam escondidas entre dados irrelevantes e a sua recuperação fica prejudicada. A ideia do estudo experimental foi investigar o apoio oferecido pelo visualizador de diagrama de classes UML 3D construído aos alunos da disciplina de Modelagem e Projeto de Sistemas, enquanto os mesmos executam tarefas utilizando este tipo de sistema. Para fins deste trabalho, um sistema grande, ou sistema complexo, é definido como qualquer sistema de software composto por um número grande de elementos com muitas interações. Para isso, foram utilizadas durante o estudo a comparação entre as ferramentas 2D (*Enterprise Architect*) e a 3D (protótipo VisaAr3D).

Este estudo foi utilizado para testar as seguintes hipóteses:

H0 – Utilizando o protótipo VisAr3D, os alunos resolvem as tarefas de modelagem com a mesma precisão e eficácia do que utilizando a ferramenta 2D, *Enterprise Architect*.

H1 – Utilizando o protótipo VisAr3D, os alunos resolvem as tarefas de modelagem com mais precisão e eficácia do que utilizando a ferramenta 2D, *Enterprise Architect*.

A variável independente do estudo foi a ferramenta utilizada (protótipo VisAr3D e *Enterprise Architect*). E as variáveis dependentes foram precisão, cobertura e tempo.

Contudo, este estudo, também, ajudou a observar os seguintes pontos:

1. A terceira dimensão contribui para apoiar a compreensão de modelos UML em sistemas com muitos elementos?
2. O ambiente na terceira dimensão desperta o interesse dos alunos em relação ao ambiente 2D?
3. A terceira dimensão dá suporte maior à prática de ensino em projetos com muitos elementos em relação ao ambiente 2D?

6.3.2. Descrição da Instrumentação

6.3.2.1. Seleção do Contexto

Como sistema a ser analisado durante o experimento foi utilizado parte da modelagem do sistema de software "*Kernel* do Odyssey", que contém até o momento 175 classes, 15 pacotes e 12 diagramas. A modelagem ainda se encontra em um estágio não concluído, portanto, é possível encontrar classes sem atributos ou operações, ou relacionamentos que ainda não foram criados e elementos de modelagem sem documentação. O objetivo foi criar uma situação real de um sistema com muitos elementos de modelagem, que não implicasse no prejuízo da resolução das tarefas pelos participantes.

O Odyssey, originalmente, é um projeto que visa explorar técnicas e ferramentas que apoiem a reutilização de software, permitindo a evolução de um ferramental de apoio. O Odyssey (WERNER *et al.*, 1999) é um ambiente de reutilização baseado em modelos de domínio, provendo tecnologias de apoio à Engenharia de Domínio (ED), Linha de Produtos (LP) e Desenvolvimento Baseado em Componentes (DBC). Ele serve como um arcabouço onde modelos conceituais e arquiteturas de software são especificados para domínios de aplicações específicos. É um sistema com cerca de 17.9 MB de código fonte em 690 arquivos Java, em 97 pacotes.

6.3.2.2. Ferramentas Utilizadas

Durante o estudo foram comparadas duas ferramentas, uma 3D, que é o protótipo de apoio à abordagem VisAr3D (que possui o mesmo nome) e uma 2D, *Enterprise Architect*.

A *Enterprise Architect* (EA) versão 7.0 (ENTERPRISE ARCHITECT, 2011) é uma ferramenta comercial de modelagem visual UML da Sparx Systems. Entre as

ferramentas no mercado, esta é adequada para o propósito deste estudo. A EA suporta UML 2.0 e inclui importação e exportação de arquivos XMI para permitir o intercâmbio de modelos entre ferramentas. Pela sua capacidade e, principalmente, pelo seu baixo custo, ela é bastante utilizada, inclusive no meio acadêmico (33% dos participantes do estudo já tinham utilizado a EA). É bastante recomendada pelos seus usuários, segundo a sua relação custo-benefício.

Conforme foi apresentado no Capítulo 5, o protótipo VisAr3D é o ferramental de apoio da abordagem de mesmo nome, que implementa parte das características propostas nesta abordagem e que, mesmo com algumas limitações de interface e usabilidade, cumpre o seu principal objetivo que é mostrar a viabilidade do que se deseja desenvolver.

A ideia deste estudo foi comparar a ferramenta comercial EA com o protótipo acadêmico VisAr3D, portanto, na verdade, pretendeu-se mostrar o potencial percebido deste último pelos participantes.

6.3.2.3. Estudo Piloto

Para a preparação do experimento, primeiramente, foi executado um estudo piloto em abril de 2011. Conforme sugerido por MAFRA e TRAVASSOS (2006), o planejamento e os instrumentos utilizados no estudo foram revisados por dois pesquisadores que não possuem interesse direto nos resultados do mesmo, visando minimizar a presença de viés. Estes pesquisadores têm experiência prévia no ensino de modelagem de sistemas. Um deles possui, ainda, experiência prática em experimentação. Anteriormente, ambos já conheciam e utilizaram a ferramenta EA como ferramenta UML. Estes pesquisadores contribuíram, suficientemente, para servir de base para explorar as dificuldades das tarefas e o tempo necessário para a sua solução, bem como ajudaram na reformulação dos formulários utilizados no estudo. Ambos executaram as mesmas tarefas, porém em ordem distintas.

6.3.2.4. Seleção dos Indivíduos

Como participantes para o estudo foram selecionados, por conveniência, alunos de pós-graduação e graduação, com experiência prévia em modelagem UML. Foi proposto aos participantes um questionário com o objetivo de caracterizar sua formação do ponto de vista acadêmico, sua experiência profissional, incluindo a experiência no ensino da disciplina de Modelagem de Sistemas de Software, entre outros, para analisar os dados e reduzir o viés. Considerou-se que estes indivíduos

possuíam disponibilidade para participar do estudo. Este questionário está disponível no Apêndice B (Caracterização do Participante).

6.3.2.5. Materiais Utilizados para a Experimentação

Neste estudo, os indivíduos participaram como alunos da disciplina de Modelagem de Sistemas e responderam algumas questões relacionadas a um sistema de software que possui muitos elementos de modelagem (cerca de 175 classes).

Os participantes foram distribuídos, aleatoriamente, em dois grupos de trabalho onde utilizavam primeiramente a Configuração A e, em seguida, a Configuração B. E em outro grupo que utilizava, primeiramente, a Configuração B e, em seguida, a Configuração A. Cada grupo fez apenas uma passagem do experimento de modo a evitar os efeitos da aprendizagem.

Configuração A: Utilização da ferramenta EA como visualizador de diagramas UML para resolução de seis tarefas propostas. A Figura 6.1 mostra uma tela da EA utilizada durante o experimento.

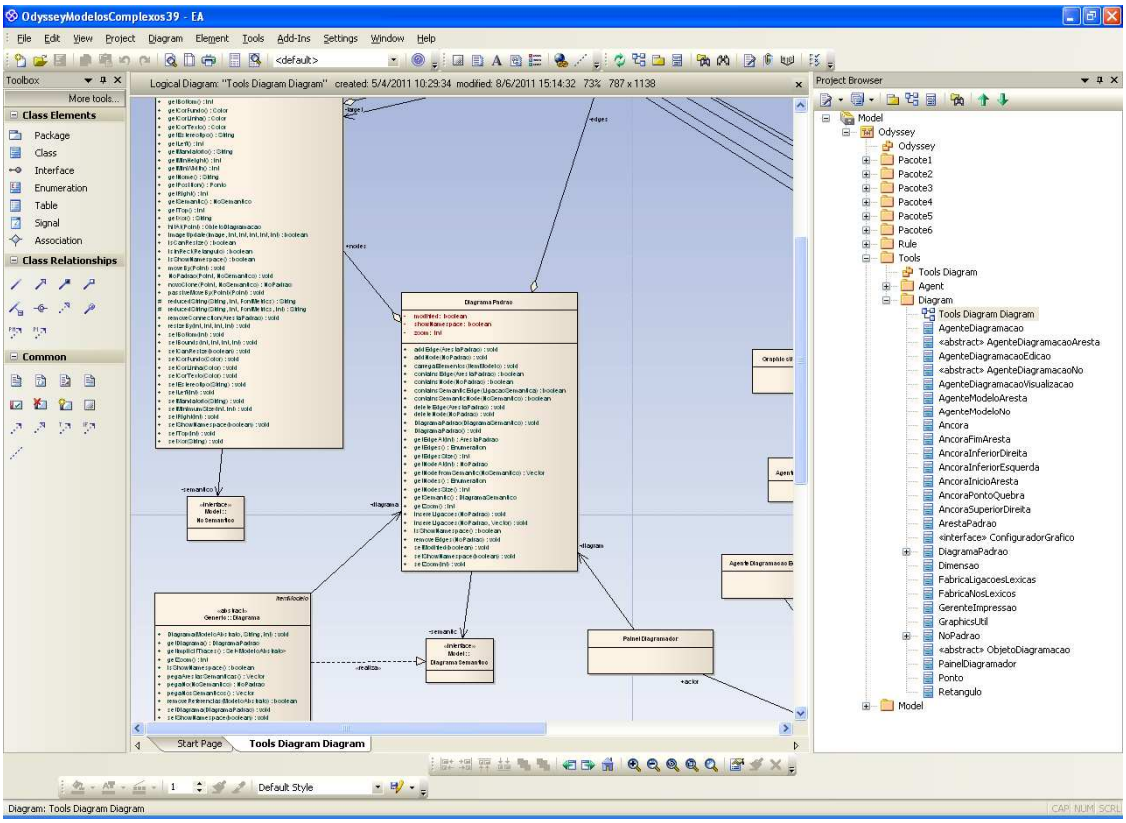


Figura 6.1: Tela da EA

Configuração B: Utilização do protótipo VisAr3D como visualizador de diagramas UML para resolução de seis tarefas propostas (semelhantes às tarefas da

configuração A). A Figura 6.2 mostra uma tela do VisAr3D utilizada durante o experimento.

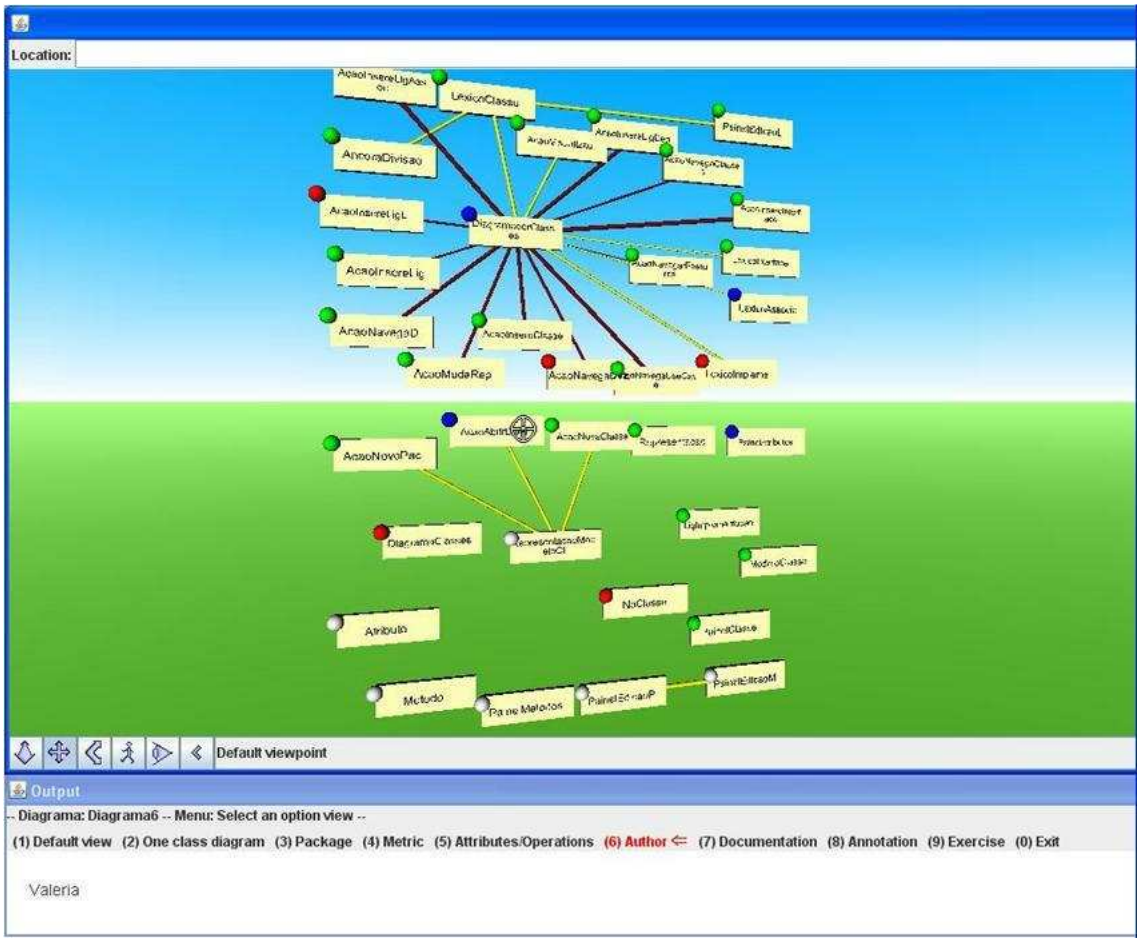


Figura 6.2: Tela do VisAr3D

Todo o material necessário para o estudo foi instalado em um computador disponível, incluindo dois vídeos de treinamento sobre a utilização do protótipo VisAr3D e sobre a ferramenta EA. Os participantes ficaram à vontade para fazerem perguntas de esclarecimento, enquanto acompanhados pelo pesquisador responsável, que fazia anotações. Foi, também, disponibilizado um guia impresso com as principais informações do vídeo de treinamento, que podia ser utilizado durante o experimento.

6.3.2.6. Tarefas Experimentais

Ambos os grupos A e B foram convidados a resolver 12 tarefas como alunos da disciplina de Modelagem de Sistemas (ver Apêndice C com as tarefas propostas). As tarefas representaram a resolução de exercícios no contexto desta disciplina, utilizando para isto a análise de diagramas de classes com muitos elementos de modelagem do "*Kernel* do Odyssey".

As tarefas trataram dos conteúdos da seguinte forma: tornando a identificação das classes de um diagrama, uma tarefa simples, para o entendimento de um sistema; mostrando a classe, não só dentro de um diagrama, mas dentro do sistema como um todo, analisando relacionamentos e herança; utilizando exemplos de análise de operações, como polimorfismo; verificando a autoria destas classes, ou seja, a participação da equipe naquele diagrama; explorando a solução dada, em forma de modelagem, a partir de um código Java etc.

Baseadas no trabalho de KNODEL *et al.* (2006), as tarefas elaboradas foram divididas em três níveis de dificuldades diferentes: Tarefas de Filtragem, Tarefas Básicas e Tarefas de Assimilação.

Tarefas de Filtragem: estas tarefas são tão fáceis que todos devem ser capazes de resolvê-las. Se há pessoas que não são capazes de resolver este tipo de tarefas, as mesmas são excluídas do experimento. Neste caso, provavelmente, há problemas na compreensão da tarefa, no uso das ferramentas ou do experimento, ou algum outro motivo. As tarefas 1 e 7 pertencem a esta categoria.

Tarefas Básicas: estas são tarefas que podem ser resolvidas por extração de fatos a partir da visualização. As tarefas 2, 4, 5, 8, 10 e 11 estão incluídas nesta categoria.

Tarefas de Assimilação: este grupo de tarefas é considerado mais difícil, exigindo do participante um maior raciocínio e entendimento para interpretar a informação solicitada. Desta forma, as tarefas 3, 6, 9 e 12 fazem parte desta categoria.

A seguir são apresentados os objetivos de cada tarefa proposta durante o experimento (Apêndice C):

Questões 1 e 7: tornar a identificação e reconhecimento das classes de um diagrama, uma tarefa simples, para o entendimento de um sistema grande.

Questões 2 e 8: identificar uma classe, não somente dentro de um diagrama, mas dentro do sistema como um todo.

Questões 3 e 9: utilizar exemplos de análise de operações, como polimorfismo, que, geralmente, passam despercebidas em sistemas com muitos elementos de modelagem.

Questões 4 e 10: verificar se num universo com muitas classes é mais intuitivo ver, graficamente, os autores das classes (ou melhor, a participação da equipe naquele diagrama).

Questões 5 e 11: explorar a completude da documentação do sistema.

Questões 6 e 12: explorar a correspondência entre o código Java e a modelagem do sistema, bem como identificar o elemento de modelagem em diagramas complexos.

6.3.2.7. Análise Quantitativa

Neste estudo, a análise quantitativa deve apresentar as métricas de precisão, cobertura e tempo, adquiridas durante a resolução das tarefas aplicadas.

Neste estudo, precisão é um indicador de exatidão, que se refere ao número de respostas válidas, com relação ao número de respostas do participante (calcula as respostas corretas em relação ao número de respostas do participante). Seu valor é fornecido pela seguinte fórmula:

$$precisão = \frac{número\ de\ respostas\ válidas}{número\ respostas}$$

Cobertura refere-se a um indicador de eficácia e completude, e é calculado com o número de respostas válidas, com relação ao número de respostas corretas esperadas (ou seja, o número total de respostas do gabarito). Seu valor é fornecido pela seguinte fórmula:

$$cobertura = \frac{número\ de\ respostas\ válidas}{número\ respostas\ do\ gabarito}$$

Não é correto comparar o tempo médio gasto na resolução de cada exercício, devido à característica de cada um: em alguns casos, exercícios complexos levavam a respostas curtas e exercícios fáceis levavam muito tempo para serem respondidos. Portanto, o indicador tempo é calculado como tempo médio gasto para a resolução de todos os exercícios utilizando uma ferramenta em particular.

De forma a apoiar a análise estatística destes indicadores, no contexto deste estudo, foi utilizado o software JMP 9.0 (JMP, 2011) com nível de significância igual a 0,05 (p-value = 5%).

Antes de usar os testes estatísticos, foi feita uma análise de *outliers* para cada métrica, por meio de gráficos *box-plot*. Depois disso, caso necessário, *outliers* foram identificados, justificados e removidos.

Como a quantidade de valores da amostra analisada foi menor que 50 (foi utilizado o total de 18 valores, correspondente ao número de participantes), inicialmente, foi utilizado o teste Shapiro-Wilk para identificar a normalidade das variáveis. Garantindo a normalidade da distribuição, foi utilizado o método de Levene.

Pressuposto de normalidade e uma vez que as variâncias são iguais, pode-se proceder com a análise de comparação das médias das duas amostras, gerando um novo teste, o Teste T.

Caso os dados não possuam distribuição normal, um método não paramétrico é utilizado, o teste de Wilcoxon. Estes métodos visam determinar se a diferença entre as médias dos valores das métricas nos diferentes grupos é estatisticamente significante.

6.4. Execução do Estudo

6.4.1. Procedimentos Experimentais

Um computador foi preparado para o experimento. Dois espaços de trabalho foram preparados, um para a ferramenta EA e outro para o VisAr3D, contendo todos os dados necessários. Cada sessão do estudo utilizou um único participante, durou cerca de 2 horas e foi conduzida na seguinte ordem:

1) Inicialmente, cada participante foi informado sobre o experimento através do Formulário de Consentimento (Apêndice A), tomando conhecimento sobre o seu objetivo, formato e termo de confidencialidade.

2) A partir da concordância em participar do experimento, o participante preencheu o Formulário de Caracterização (Apêndice B). Este questionário avaliou o seu nível de conhecimento e experiência. Estas informações garantiram que os mesmos estavam aptos a executar o estudo e também serviram para interpretar os resultados obtidos por cada um dos participantes. O preenchimento dos formulários iniciais levou em torno de 13 minutos.

3) Foi exibido um vídeo[1] de treinamento com uma breve explicação das principais funcionalidades de cada ferramenta e os elementos gráficos disponíveis. O vídeo do VisAr3D utilizou cerca de 7 minutos e o vídeo da EA utilizou cerca de 3 minutos.

4) Antes de começar a resolver as tarefas com uma ferramenta, cada participante teve alguns instantes para explorá-la com o objetivo de familiarizar-se um pouco mais. Um guia impresso também foi disponibilizado para sanar algumas dúvidas durante o experimento.

5) Ao final do treinamento, foi entregue um formulário com todas as tarefas que deveriam ser resolvidas, 6 tarefas com a ferramenta EA e 6 tarefas com o protótipo

[1] http://lab3d.coppe.ufrj.br/index.php/projetos/70-visar3d-arquitetura-de-software-em-3d-video.html

VisAr3D. A resolução das tarefas levava em média 1h10. Um modelo do formulário de Tarefas está disponível no Apêndice C.

6) Para cada tarefa, foi solicitado, ainda, que o participante a avaliasse segundo o seguinte critério na escala de Likert: muito fácil, fácil, normal, difícil e muito difícil. Em seguida, a pesquisadora anotava o tempo gasto para sua resolução.

7) No final do tempo previsto para as tarefas, os participantes responderam a um Questionário de Avaliação (Apêndice D), abrangendo questões de percepção do participante: quanto à qualidade das tarefas, qualidade dos resultados, questões sobre a própria experiência (tempo suficiente, adequação das tarefas) e comparação entre as duas ferramentas. O preenchimento deste questionário durou entre de 12 e 20 minutos.

8) Finalmente, o material resultante de cada participante foi recolhido.

6.5. Análise e Interpretação dos Resultados

6.5.1. Perfil dos Participantes

Conforme dito anteriormente, os participantes do estudo são alunos de pós-graduação e graduação da UFRJ, e foram selecionados por conveniência. Não houve nenhum tipo de compensação para os mesmos. No total, 18 indivíduos participaram do estudo, entre eles, 7 doutorandos, 5 mestrandos e 6 graduandos.

A Tabela 6.1 apresenta a distribuição das duas configurações disponíveis em relação ao número de participantes.

Tabela 6.1: Número de participantes por grupo

	Configuração A			**Configuração B**		
Grupo de pessoas	**Doutorandos**	**Mestrandos**	**Graduandos**	**Doutorandos**	**Mestrandos**	**Graduandos**
Qtde.	4	2	3	3	3	3

A Tabela 6.2 apresenta o grau de experiência dos participantes em Modelagem de Sistemas de Informação, Orientação a Objetos, Java e Padrões de Projeto, de acordo com a escala de cinco pontos. 28% dos participantes possuem experiência de ensino da disciplina de Modelagem de Sistemas de Software, atuando como monitor, instrutor ou professor. Enquanto 44,5% possuem experiência com modelagem UML trabalhando na indústria. (3 participantes trabalharam 1 ano, 4 participantes

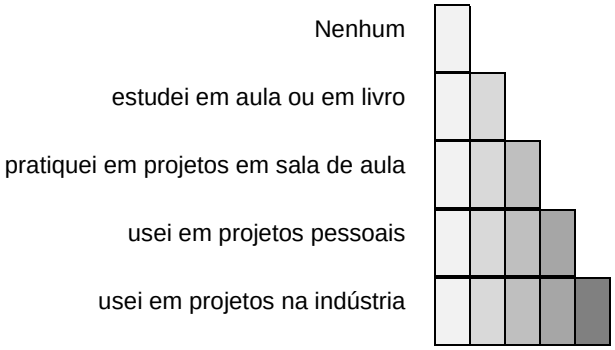


Tabela 6.2: Grau de experiência dos participantes

Código do Participante	Modelagem de Sistemas de Informação	Orientação a Objetos	Java	Padrões de Projeto
I131				
I132				
I133				
I134				
I135				
I136				
I137				
I138				
I139				
I461				
I462				
I463				
I464				
I465				
I466				
I467				
I468				
I469				

trabalharam 3 anos e 1 trabalhou 5 anos na indústria). Portanto, a grande maioria contribuiu para este estudo com uma visão de aluno e de profissional da indústria.

Das ferramentas UML mais utilizadas pelos participantes, destacam-se a *Jude* com 44%, a *Rational Rose* com 35%, a *Enterprise Architect* com 28% e a *StarUML* com 17%. A EA, portanto, é a terceira ferramenta mais utilizada.

As respostas dos participantes quanto à maior quantidade de classes de um sistema modelado por eles, está apresentada na Tabela 6.3. Ela mostra que 50% dos participantes tiveram experiência com sistemas com mais de 50 classes, um índice significativo que contribui para o resultado da avaliação. No entanto, somente 2 do total de 18 participantes (ou seja, 11%) utilizaram um sistema com mais de 200 classes dentro de sala de aula. Segundo o relato dos próprios participantes, a vantagem em se utilizar este tipo de sistema em sala de aula é oferecer ao aprendiz a oportunidade de trabalhar com um problema mais próximo da realidade, onde mais situações do cotidiano podem ser exploradas e várias condições de modelagem podem ser exemplificadas. Dessa forma, são apresentados casos compatíveis com o que ele, provavelmente, vai lidar no mercado de trabalho.

Quanto ao uso de sistemas com muitos elementos em sala de aula, as desvantagens percebidas pelos participantes foram expressas da seguinte forma: “no momento do aprendizado, a complexidade do modelo pode influenciar negativamente no entendimento dos diagramas”; “a visualização do sistema pode ficar prejudicada”; “dificulta o ensino de conceitos mais básicos”; e “não há tempo suficiente”.

Tabela 6.3: Respostas dos participantes quanto ao maior número de classes modeladas

Participantes	Número de classes
39%	Até 20 classes
11%	De 21 a 50 classes
17%	de 51 a 100 classes
22%	de 101 a 200 classes
11%	mais de 200 classes

6.5.2. Remoção de Valores Extremos (*Outliers*)

Outliers ou valores extremos são valores observados que estão muito distantes dos demais valores. Estes dados podem representar erros no conjunto de valores observados e usualmente são removidos deste conjunto antes de se aplicar as técnicas de inferência estatística. Eles podem ocorrer por vários motivos e só podem ser removidos se identificadas as suas origens.

As Tabelas 6.4, 6.5 e 6.6 mostram a média das métricas Precisão, Cobertura e Tempo por participante, respectivamente, na primeira e segunda etapa da resolução de tarefas. Estes valores foram calculados segundo as fórmulas descritas na Seção 6.3.2.7. Nas Tabelas, as células em cinza indicam a utilização do protótipo VisAr3D e em branco, a utilização da ferramenta EA.

A análise de *outliers* para as métricas de precisão, cobertura e tempo (min) é feita através de gráficos *box-plot* exibidos na Figura 6.3. Conforme os gráficos, foi decidida a remoção do participante I132 tanto ao utilizar a ferramenta EA, quanto o protótipo VisAr3D. A razão da exclusão deste participante, ao utilizar a ferramenta EA, foi devido a um erro ao preencher o formulário de resposta. O participante se confundiu e respondeu uma questão na posição errada, deixando a tarefa 5 sem resposta. E a razão da exclusão deste mesmo participante, ao utilizar o protótipo VisAr3D, foi que ele gastou um tempo excessivo (aproximadamente, 3 vezes maior que o tempo médio dos outros participantes) para responder a questão 11, ao fazer observações e comentários antes do fim da resolução do exercício.

Outro critério para a remoção de *outliers* foi aplicado da seguinte forma: para cada variável foi calculado (observar a Figura 6.3) o valor mínimo e máximo que esta deveria ter. O valor mínimo foi obtido, subtraindo da média, três vezes o valor do desvio padrão, e o valor máximo foi obtido, adicionando à média, três vezes o valor do desvio padrão. Foram considerados *outliers* os valores menores que o valor mínimo e os valores maiores que o valor máximo.

Tabela 6.4: Média da Precisão por participante

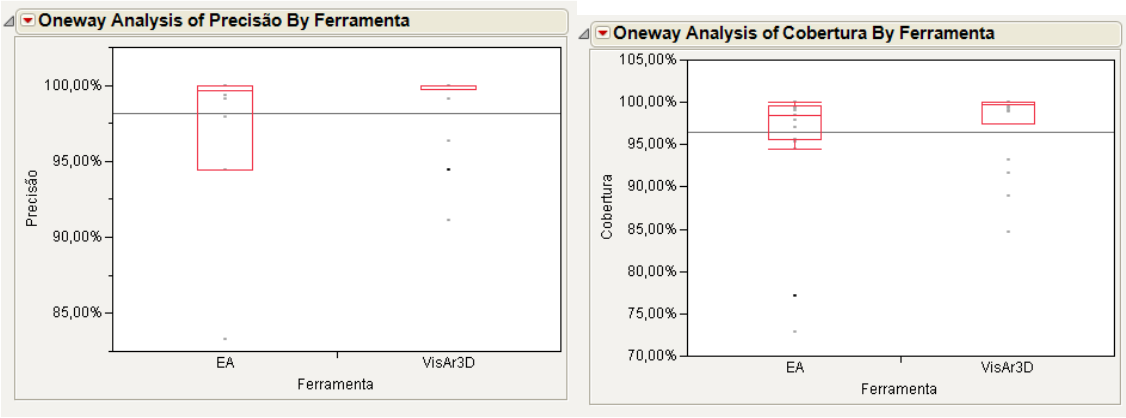
Participante	1ª etapa	2ª etapa
I131	100,00%	99,07%
I132	83,33%	94,44%
I133	100,00%	100,00%
I134	100,00%	100,00%
I135	100,00%	100,00%
I136	100,00%	100,00%
I137	97,92%	100,00%
I138	100,00%	100,00%
I139	100,00%	100,00%
I461	100,00%	94,44%
I462	100,00%	94,44%
I463	100,00%	100,00%
I464	100,00%	94,44%
I465	96,30%	94,44%
I466	100,00%	99,07%
I467	100,00%	94,44%
I468	91,15%	99,31%
I469	100,00%	100,00%

Tabela 6.5: Média da Cobertura por participante

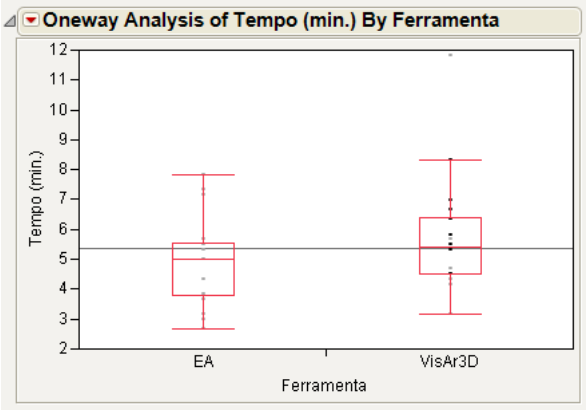
Participante	1ª etapa	2ª etapa
I131	100,00%	100,00%
I132	72,85%	88,98%
I133	100,00%	100,00%
I134	95,30%	99,02%
I135	98,39%	99,02%
I136	99,46%	93,18%
I137	97,85%	100,00%
I138	98,39%	100,00%
I139	77,06%	99,28%
I461	100,00%	99,02%
I462	100,00%	99,02%
I463	100,00%	99,28%
I464	98,92%	94,44%
I465	100,00%	95,65%
I466	99,46%	100,00%
I467	84,72%	97,06%
I468	91,67%	97,06%
I469	100,00%	100,00%

Tabela 6.6: Média do Tempo (min) por participante

Participante	1ª etapa	2ª etapa
I131	5,00	5,67
I132	7,83	11,83
I133	5,50	4,17
I134	5,33	6,33
I135	5,00	4,33
I136	7,33	4,50
I137	5,67	4,67
I138	5,50	4,67
I139	3,83	3,17
I461	5,83	3,83
I462	5,50	2,67
I463	6,67	5,33
I464	8,33	5,00
I465	7,00	4,33
I466	5,33	3,17
I467	6,33	7,17
I468	4,50	3,67
I469	5,33	3,00



(a) **(b)**



(c)

Figura 6.3: *Box-plots* das métricas precisão (a), cobertura (b) e tempo (c) antes da eliminação do *outlier*

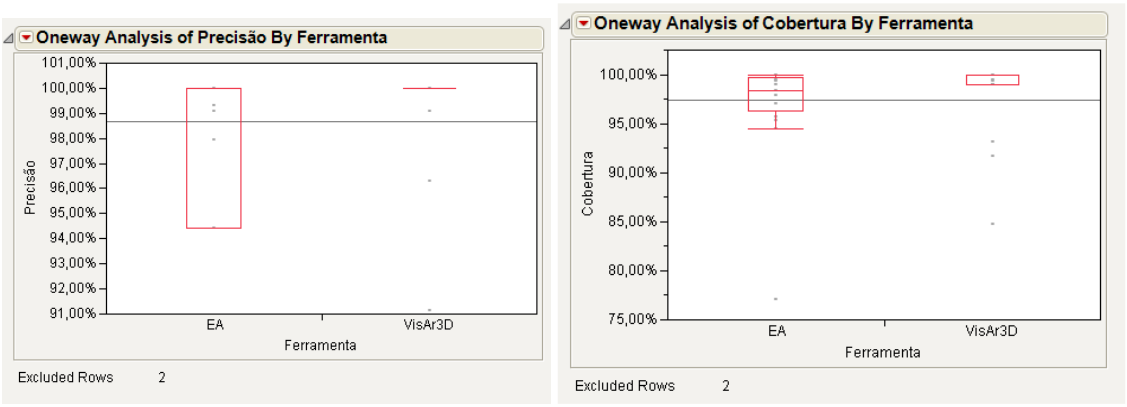
Após a decisão de remoção dos *outliers*, os gráficos *box-plot* foram alterados e exibidos na Figura 6.4. A média geral dos resultados dos participantes por ferramenta, das métricas Precisão, Cobertura e Tempo, com seus respectivos desvios padrão, antes da remoção dos *outliers* é apresentada na Tabela 6.7 e depois da remoção dos *outlliers* é apresentada na Tabela 6.8.

Tabela 6.7: Média e Desvio Padrão dos resultados dos participantes para cada ferramenta antes da remoção dos *outliers*

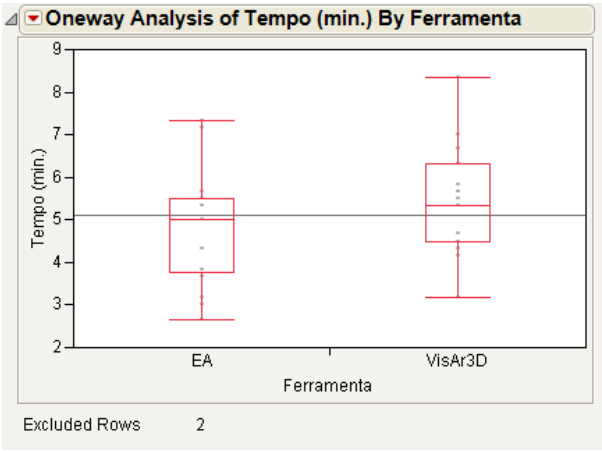
	EA		VisAr3D	
Métrica	**Média**	**Desvio padrão**	**Média**	**Desvio padrão**
Precisão	97,31%	4,33%	98,63%	2,67%
Cobertura	95,60%	7,73%	97,46%	4,59%
Tempo	4,95	1,48	5,79	1,94

Tabela 6.8: Média e Desvio Padrão dos resultados dos participantes para cada ferramenta depois da remoção dos *outliers*

	EA		VisAr3D	
Métrica	**Média**	**Desvio padrão**	**Média**	**Desvio padrão**
Precisão	98,15%	2,52%	98,88%	2,53%
Cobertura	96,94%	5,41%	97,96%	4,20%
Tempo	4,78	1,33	5,43	1,25



(a) **(b)**



(c)

Figura 6.4: *Box-plots* das métricas precisão (a), cobertura (b) e tempo (c) depois da eliminação dos *outliers*

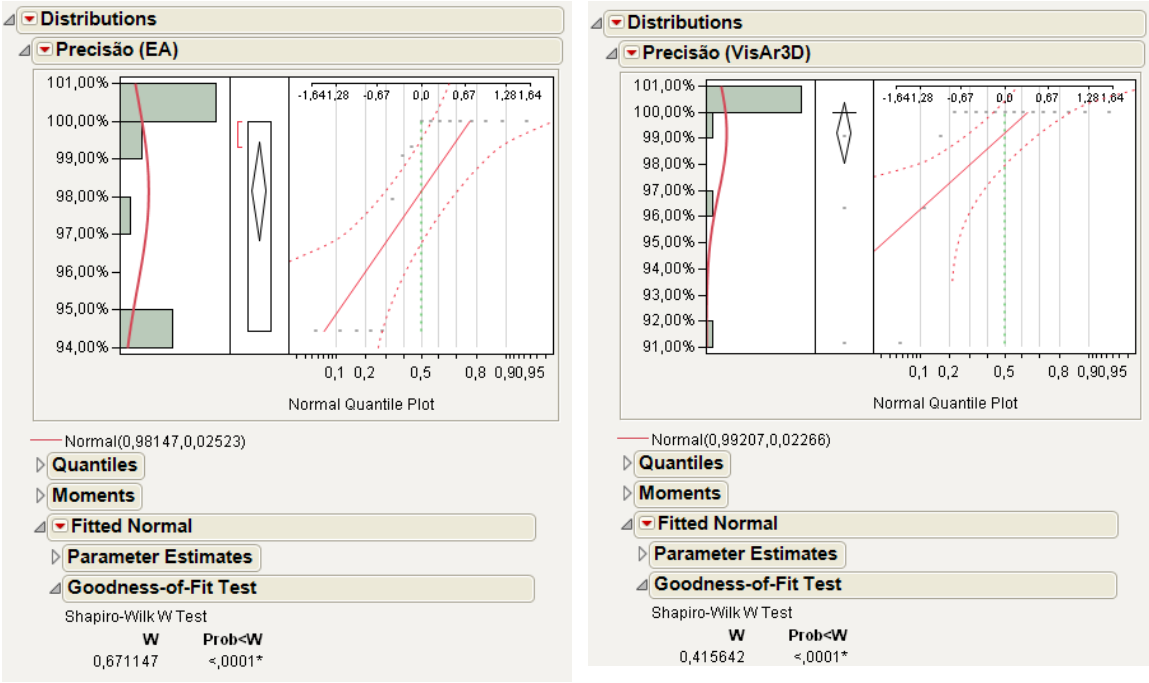
6.5.3. Análise para a Variável Precisão

Através da análise do Teste de Shapiro-Wilk para a variável Precisão (Figura 6.5), observa-se que ambas as amostras possuem o valor de significância (Prob<W) inferior a 0,05 (0,0001 nos dois casos). Desta forma, a distribuição das amostras para a variável Precisão não é normal. Neste caso, deve-se utilizar um método não paramétrico.

Pela Figura 6.6, percebe-se ao executar o Teste Wilcoxon, que a significância (Prob >|Z| = 0,2022) é superior a 0,05 e, desta forma, não existem indícios para rejeitar

a hipótese nula, concluindo-se que as médias são iguais a um nível de significância de 5%. Por estas análises efetuadas, pode-se concluir que “estatisticamente não existe diferença significativa em relação à variável Precisão na utilização das ferramentas VisAr3D e EA”.

O teste foi realizado, também, para *alpha-value* igual a 10% e o resultado final foi o mesmo.



(a) **(b)**

Figura 6.5: Teste Shapiro-Wilk para a variável Precisão utilizando as ferramentas EA (a) e VisAr3D (b)

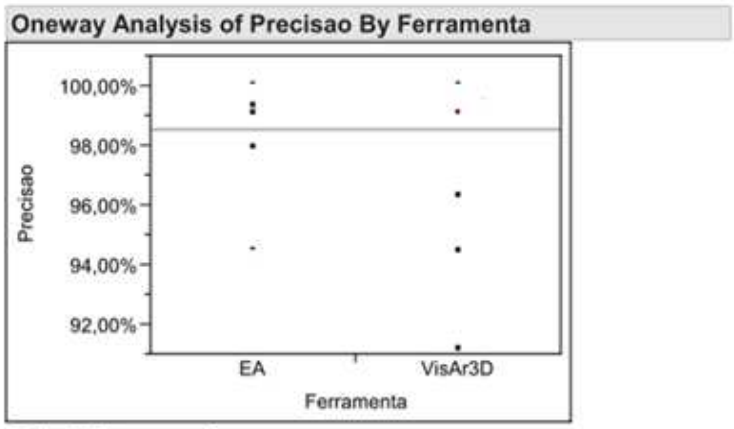
6.5.4. Análise para a Variável Cobertura

Através da análise do Teste de Shapiro-Wilk para a variável Cobertura (Figura 6.7), observa-se que ambas as amostras possuem o valor de significância (Prob<W) inferior a 0,05 (0,0001 nos dois casos). Desta forma, a distribuição das amostras para a variável Cobertura não é normal. Neste caso, deve-se utilizar um método não paramétrico.

Pela Figura 6.8, percebe-se ao executar o Teste Wilcoxon, que a significância (Prob >|Z| = 0,1026) é superior a 0,05 e, desta forma, não existem indícios para rejeitar a hipótese nula, concluindo-se que as médias são iguais a um nível de significância de 5%. Por estas análises efetuadas, pode-se concluir que “estatisticamente não existe

diferença significativa em relação à variável Cobertura na utilização das ferramentas VisAr3D e EA".

O teste foi realizado, também, para *alpha-value* igual a 10% e o resultado final foi o mesmo.



Excluded Rows 2

Wilcoxon / Kruskal-Wallis Tests (Rank Sums)

Level	Count	Score Sum	Expected Score	Score Mean	(Mean-Mean0)/Std0
EA	17	265,500	297,500	15,6176	-1,275
VisAr3D	17	329,500	297,500	19,3824	1,275

2-Sample Test, Normal Approximation

S	Z	Prob>|Z|
329,5	1,27519	0,2022

1-way Test, ChiSquare Approximation

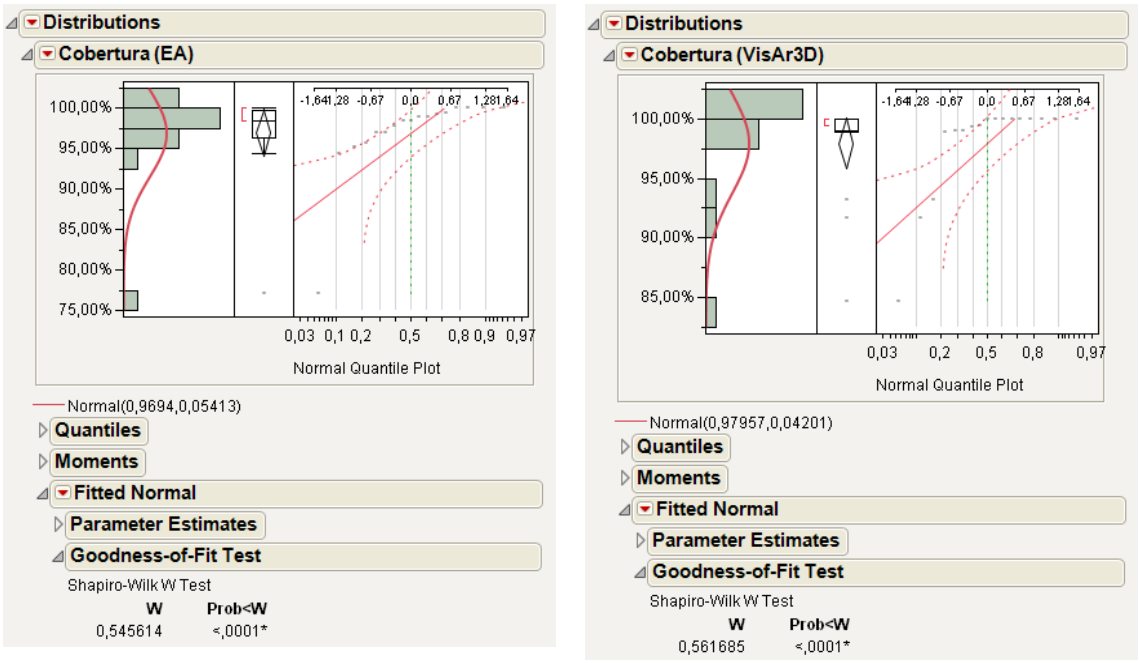
ChiSquare	DF	Prob>ChiSq
1,6781	1	0,1952

Figura 6.6: Teste Wilcoxon para a variável Precisão

6.5.5. Análise para a Variável Tempo

Através da análise do Teste de Shapiro-Wilk para a variável Tempo (Figura 6.9), observa-se que ambas as amostras possuem o valor de significância (Prob<W) superior a 0,05 (0,4138 para a ferramenta EA e 0,8391 para o protótipo VisAr3D). Desta forma, a distribuição das amostras para a variável Tempo é normal.

Pela linha do Teste de Levene para Igualdade de Variâncias observada na Figura 6.10, verifica-se que as amostras possuem variâncias iguais, uma vez que a significância (p-Value = 0,7274) é maior que 0,05, não havendo indícios para rejeitar a hipótese nula. Logo, as variâncias são iguais.



(a) **(b)**

Figura 6.7: Teste Shapiro-Wilk para a variável Cobertura utilizando as ferramentas EA (a) e VisAr3D (b)

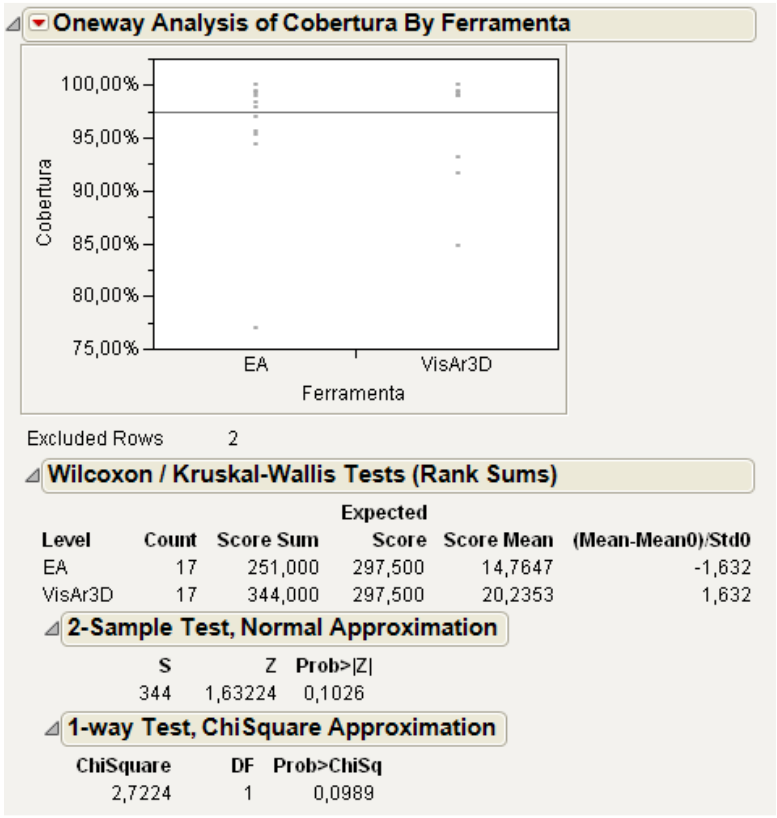
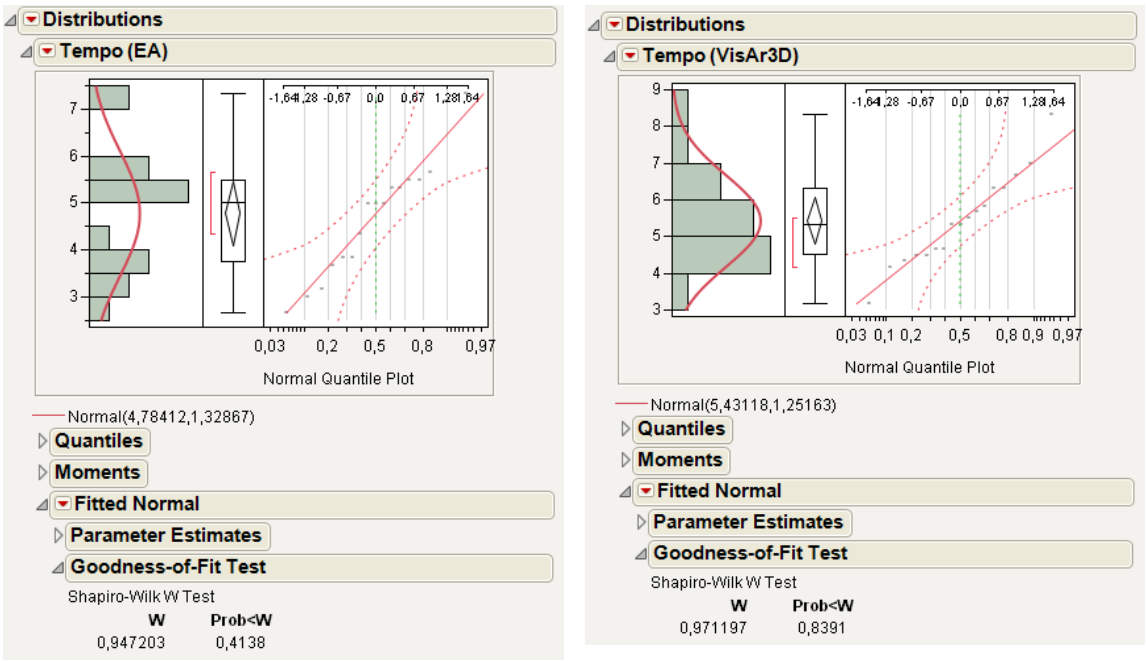


Figura 6.8: Teste Wilcoxon para a variável Cobertura



(a) **(b)**

Figura 6.9: Teste Shapiro-Wilk para a variável Tempo utilizando as ferramentas EA (a) e VisAr3D (b)

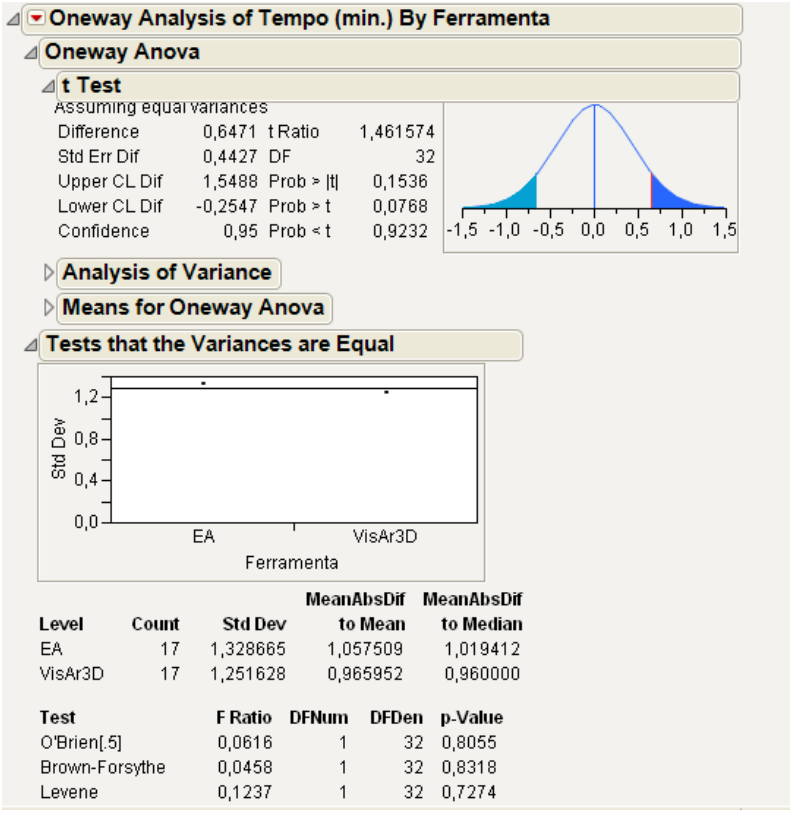


Figura 6.10: Teste Levene e Teste T para a variável Tempo

Percebe-se na linha do Teste T, a significância (Prob >|t| = 0,1536) é superior a 0,05 e, desta forma, não existem indícios para rejeitar a hipótese nula, concluindo-se que as médias são iguais a um nível de significância de 5%.

Por estas análises efetuadas, pode-se concluir que “estatisticamente não existe diferença significativa em relação à variável Tempo na utilização das ferramentas VisAr3D e EA”.

O teste foi realizado, também, para alpha-value igual a 10% e o resultado final foi o mesmo.

6.5.6. Análise Quantitativa

As tarefas do estudo foram elaboradas de tal forma que pudessem ser resolvidas pelas duas ferramentas, contudo o objetivo principal era observar como cada participante desempenhava cada uma delas e quais eram as suas necessidades e expectativas, utilizando cada ferramenta, 2D e 3D. Por isso, esperava-se que os índices de precisão e cobertura não fossem muito diferentes, como foi confirmado pela análise dos dados.

Os participantes conseguiram atingir os seus objetivos e conseguiram resolver as tarefas, com os indicadores de precisão e cobertura (ou eficácia), em média, próximos nas duas ferramentas, conforme mostra a Tabela 6.8 (98,15%, utilizando a ferramenta EA, contra 98,88%, utilizando o protótipo VisAr3D, de precisão, e 96,94% utilizando a ferramenta EA, contra 97,96% utilizando o protótipo VisAr3D, de cobertura). Apesar de obter uma pequena vantagem para o VisAr3D, segundo a análise estatística, não existe diferença significativa em relação às variáveis Precisão e Cobertura na utilização das ferramentas EA e VisAr3D.

Não é muito útil comparar o tempo gasto médio na resolução de cada exercício, devido à característica de cada um: em alguns casos, exercícios complexos levavam a respostas curtas e exercícios fáceis levavam muito tempo para serem respondidos. Contudo, num contexto geral, gastou-se mais tempo, em média, para resolver as tarefas utilizando a ferramenta 3D comparada à ferramenta 2D, como mostra a Tabela 6.8 (5,43 minutos contra 4,78 minutos); segundo os participantes, isto pode ser explicado pelo fato deles estarem mais familiarizados com padrões 2D. Foi observado, também, que se gastou mais tempo na primeira etapa das tarefas, analisando o efeito da aprendizagem durante a execução do estudo. Da mesma forma que as outras variáveis, estatisticamente, não existe diferença significativa em relação a variável Tempo na utilização das ferramentas EA e VisAr3D.

Assim, com este estudo experimental são apresentadas evidências que usando o protótipo VisAr3D, os alunos resolvem as tarefas de modelagem com valores de precisão e eficácia bem próximos, mas ainda melhores, do que utilizando a ferramenta 2D, *Enterprise Architect*. Tratando-se da fragilidade de um protótipo, com uma nova proposta de navegação e interação, com problemas de *performance* e onde apenas parte da abordagem foi implementada, ele ainda apresentou uma pequena vantagem. Deve-se lembrar ainda que este é um protótipo acadêmico que foi comparado a uma ferramenta comercial bastante aceita e utilizada no mercado por muitos anos, desenvolvida por uma equipe profissional grande. A análise quantitativa mostra um indício positivo que a ferramenta 3D está no caminho certo. A análise qualitativa, apresentada na Seção 6.5.7, também reforça esta afirmativa.

As Figuras 6.11 e 6.12 apresentam a quantidade de avaliações, segundo o critério na escala Likert de “muito fácil, fácil, normal, difícil e muito difícil” para cada tarefa oferecida. Uma tarefa fácil de ser realizada, que exigia um gasto de energia para resolvê-la (esforço braçal), era avaliada, muitas vezes pelos participantes, como normal. Ou seja, a avaliação se tornava mais rigorosa. Isto foi notado nas tarefas 1 e 7 que são tarefas muito fáceis, onde se pede que sejam listadas todas as classes presentes no diagrama. Estas foram consideradas fáceis e normais, segundo este critério. As Figuras 6.11 e 6.12 mostram que, na tarefa 1, o protótipo VisAr3D foi avaliado como um pouco mais fácil que a ferramenta 2D, no entanto na tarefa 7, semelhante àquela, a ferramenta 2D foi considerada mais fácil. Isto se deve, talvez, pela complexidade do modelo e a dificuldade de manipulação do mouse no ambiente tridimensional.

Nas tarefas 2 e 8, percebe-se uma vantagem na avaliação do protótipo VisAr3D por causa da funcionalidade provida por este que facilitava o obtenção da resposta da tarefa.

As figuras mostram uma aprovação da utilização de objetos virtuais exibidos na visão Autor do protótipo VisAr3d nas tarefas 4 e 10. O mesmo se repete nas tarefas 5 e 11, em relação à utilização da camada na cor azul adicionada às classes para indicar a presença de documentação, facilitando a identificação da informação pelo participante.

As tarefas mais difíceis, as de número 6 e 12, foram também avaliadas pelos participantes com uma vantagem do protótipo 3D em relação à ferramenta 2D.

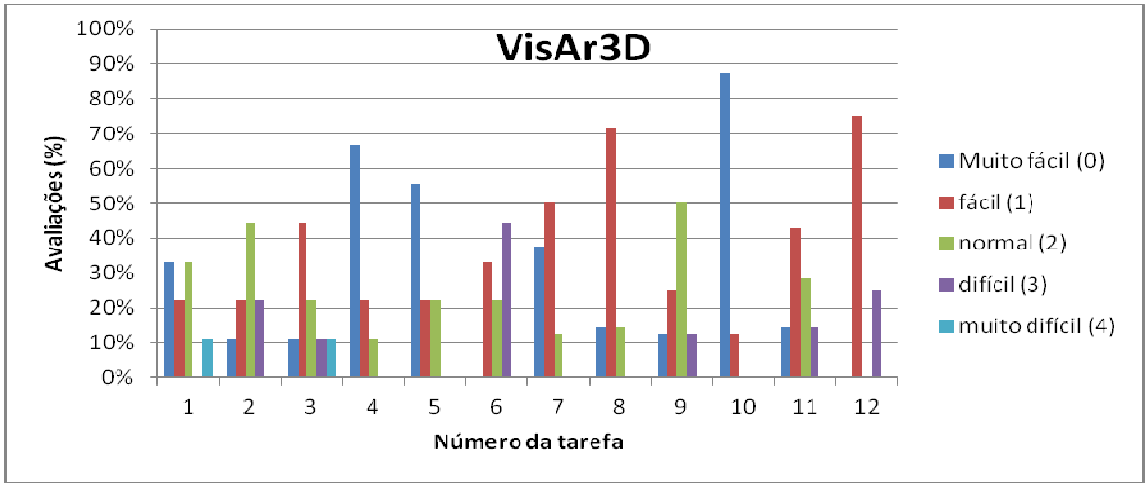


Figura 6.11 – Avaliação dos participantes em relação a cada tarefa utilizando o protótipo VisAr3D

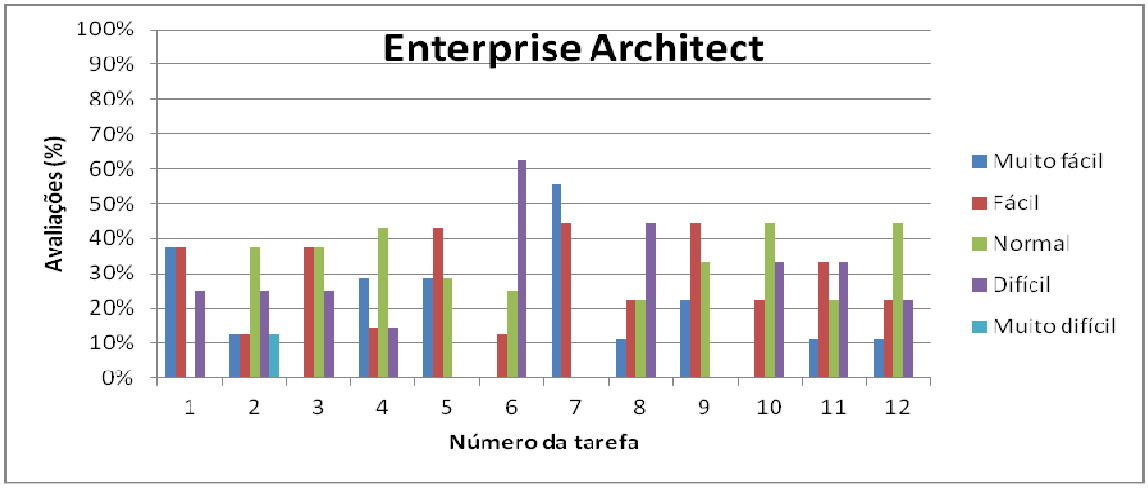


Figura 6.12 – Avaliação dos participantes em relação a cada tarefa utilizando a ferramenta *Enterprise Architect*

As tarefas 4, 5, 10 e 11 seguiram o mesmo critério quando resolvidas utilizando a ferramenta *Enterprise Architect*, que não apresenta uma funcionalidade que facilita a resolução do exercício naquele diagrama. Estas funcionalidades no VisAr3D fizeram com que estas tarefas fossem muito bem avaliadas pelos participantes. Pelas Figuras 6.11 e 6.12, pode-se concluir, também, que mesmo na resolução das tarefas mais difíceis (tarefas 6 e 12), os participantes avaliaram melhor a ferramenta 3D.

A Tabela 6.9 resume a melhor contribuição, comparando a visualização 2D e a visualização 3D, na opinião dos participantes, segundo alguns tópicos por ferramenta. Esta Tabela apresenta as respostas dos participantes no Questionário de Avaliação (Apêndice D). As suas respostas eram ilustradas com comentários úteis e relevantes para uma análise qualitativa do estudo. O participante escolheu qual ferramenta, na

sua percepção, melhor contribuiu segundo os tópicos que correspondem às linhas da Tabela. A coluna “Não se aplica” foi considerada, pelos participantes, como dúvida entre as duas ferramentas e em outras situações como “nenhuma das duas opções”. Na maioria dos casos, cada escolha vinha acompanhada de um comentário. E foi observado que na maioria dos tópicos o VisAr3D foi apontado como a melhor escolha.

Os participantes perceberam, segundo esta Tabela, e registraram conforme seus comentários, a importância e a contribuição de obter informações sobre a autoria e a documentação no ambiente 3D, utilizando recursos de cores e formas. Assim, nota-se a aceitação destes recursos e sugere-se sua exploração para futuras implementações. Os participantes comentaram a Visão do Autor (na tabela com 94% de aceitação do 3D contra 0% do 2D) dizendo: “Muito bom o recurso utilizado com as cores diferenciando os autores com uma rápida percepção a quantidade de trabalho realizado e participação no diagrama” e “Para projetos realmente grandes, percebe-se mais facilidade na gestão de projetos, por exemplo, a identificação dos autores foi muito fácil no modelo 3D”.

Tabela 6.9: Melhor contribuição por ferramenta segundo alguns tópicos

	Ferramenta 2D	Ferramenta 3D	Não se aplica*
Apoio à Compreensão UML em sistemas grandes	22%	56%	22%
Apoio à exploração de modelos	33%	56%	11%
Apoio à resolução de tarefas	17%	61%	22%
Redução da complexidade	17%	61%	22%
Diminuição da poluição visual	6%	83%	11%
Facilidade na leitura de detalhes da informação	22%	44%	33%
Ambiente mais intuitivo	28%	44%	28%
Facilidade de obtenção de informações sobre autoria	0%	94%	6%
Facilidade de obtenção de informações sobre relacionamento	44%	33%	22%
Facilidade de obtenção de informações sobre classe em mais de um diagrama	6%	83%	11%
Facilidade de obtenção de informações sobre documentação	0%	94%	6%
Suporte à prática de ensino	11%	83%	6%

(*) “Não se aplica” foi considerado pelos participantes com dúvida entre uma ferramenta e outra, e como nenhuma das duas opções.

A Visão de documentação foi aprovada com 94% para ferramenta 3D contra 0% para a 2D, mostrando a aceitação dos participantes quanto a inserção de cores como indicação de informação adicional aos diagramas. Sobre a Visão de Documentação, eles acrescentaram: “A ferramenta 3D acaba dando uma melhor visão do diagrama como um todo e através do menu de documentação fica fácil verificar quais classes possuem documentação”, “A ferramenta 3D tem uma facilidade desenvolvida para

percepção rápida desta informação", "Mais fácil pela cor presente em cada classe", "Novamente, o esquema de cores acelerou bastante o processo".

A questão mais difícil (identificada, inclusive na opinião dos participantes) foi a de número 6, que explorava a solução dada, em forma de modelagem, a partir de um código Java. Ela foi resolvida através do VisAr3D com tempo, em média, de 7,44 minutos e através da ferramenta 2D com o tempo, em média, de 7,0 minutos, ou seja, em média o VisAr3D levou somente 6% mais tempo que a EA. Analisando este índice e a Tabela 6.9, pode-se concluir que os tópicos "Redução da complexidade", "Diminuição da poluição visual" e "Facilidade na leitura de detalhes da informação" contribuíram para auxiliar a resolução da tarefa.

Para a resolução das tarefas 2 e 8, o participante deveria analisar a presença de uma classe em mais de um diagrama. A contribuição do protótipo VisAr3D foi muito significativa (83%).

Segundo a Tabela, a ferramenta 2D se destaca melhor que a 3D somente no item "Facilidade de obtenção de informações sobre relacionamento" (44% para a ferramenta 2D, contra 33% para a ferramenta 3D), isto deve-se a falta de direcionamento dos relacionamentos no VisAr3D, citada como aspecto negativo entre os participantes. Algumas questões exigiam destreza do participante ao utilizar o mouse para se movimentar no ambiente virtual, o que prejudicou bastante o VisAr3D durante a avaliação.

Durante a avaliação, incluindo a avaliação piloto, 7 participantes possuíam experiência de ensino da disciplina de Modelagem de Sistemas de Software como professor, instrutor ou monitor. 86% responderam que preferiam a ferramenta 3D comparada à 2D em relação ao "suporte à prática de ensino em projetos com muitos elementos de modelagem", enquanto somente um participante (14%) respondia "não se aplica". Suas escolhas e comentários foram os seguintes: participante com 5 anos como professor: "3D. É muito difícil trabalhar com diagramas 2D com muitos elementos"; participante com 1,5 ano como professor: "3D. O 3D diminui drasticamente a complexidade do sistema modelado"; participante com 6 meses como monitor: "3D. Mais para explorar novas informações do que as usualmente utilizadas. Exemplos são os autores e a documentação"; participante com 1 ano como instrutor: "3D. No modelo 3D, a visualização dos artefatos é mais coesa e mais integrada"; participante com 2 anos como professor: "3D. Mais interessante e dispertaria mais interesse"; participante com 3 meses como professor: "não se aplica"; participante com 6 meses como monitor: "3D. Mais fácil de visualizar, tanto por cores e

pelo ambiente 3D em si, com maior área de trabalho e maior clareza de relacionamentos em geral".

Em resumo, a Tabela 6.9 mostra que a ferramenta 3D apresenta evidências positivas em comparação à ferramenta 2D na resolução de tarefas no contexto da disciplina de Modelagem de Sistemas. E ainda responde, auxiliada pelas respostas dos participantes ao Questionário de Avaliação, aos principais questionamentos do estudo, concluindo que: (1) a terceira dimensão contribuiu para apoiar a compreensão de modelos UML em sistemas com muitos elementos; (2) o ambiente na terceira dimensão desperta o interesse dos alunos em relação ao ambiente 2D; e (3) a terceira dimensão deu um suporte maior à prática de ensino em projetos com muitos elementos em relação ao ambiente 2D.

Embora o estudo possa apresentar viés devido à proximidade da pesquisadora com alguns dos participantes, a tabela apresenta indícios significativos quantos aos seus resultados. Deve-se considerar, ainda, o longo tempo de duração do experimento e que outras pessoas foram convidadas (o convite para a participação foi enviado através de uma lista de discussão e através de convites pessoais de alguns professores) e muitos destes não aceitaram participar.

6.5.6.1. Análise por Agrupamento

Esta Seção apresenta uma análise quantitativa do experimento após o agrupamento dos participantes por nível de experiência, segundo a sua formação acadêmica: aluno de doutorado, aluno de mestrado e aluno de graduação. A Tabela 6.10 mostra os indicadores Precisão, Cobertura e Tempo, segundo esta análise. Ela apresenta a média e o desvio padrão de cada indicador para cada grupo de participantes: doutorado, mestrado e graduação, ao utilizar o protótipo VisAr3D e a ferramenta EA. Por estas análises efetuadas, concluiu-se que "estatisticamente não existe diferença significativa em relação às variáveis Precisão, Cobertura e Tempo na utilização das ferramentas VisAr3D e EA", utilizando *alpha-value* igual a 5% e 10%.

Tabela 6.10: Média e Desvio Padrão dos resultados dos grupos de participantes para cada ferramenta

		Doutorado		Mestrado		Graduação	
		Média	**Desvio Padrão**	**Média**	**Desvio Padrão**	**Média**	**Desvio Padrão**
Precisão	**EA**	97,99%	2,77%	97,78%	3,05%	98,61%	2,20%
	VisAr3D	99,23%	1,49%	100%	0%	97,60%	3,86%
Cobertura	**EA**	98,75%	1,64%	97,61%	2,54%	94,57%	8,64%
	VisAr3D	98,61%	2,69%	99,59%	0,57%	95,95%	6,40%
Tempo (min)	**EA**	4,58	1,65	5,00	0,68	4,81	1,57
	VisAr3D	5,39	0,96	6,27	1,50	4,79	1,04

A análise desta tabela deve ser realizada comparando os resultados de um mesmo grupo de participantes. Resumindo a tabela, a relação entre a ferramenta 2D e o protótipo VisAR3D, para os alunos de:

1. Doutorado → precisão: aumentou, cobertura: igual, tempo: aumentou
2. Mestrado → precisão: aumentou, cobertura: aumentou, tempo: aumentou
3. Graduação → precisão: diminuiu, cobertura: aumentou, tempo: igual

Pela sua maior capacidade de abstração, a tabela mostra que o aluno de doutorado apresenta cobertura igual nas duas ferramentas, ou seja, ele tem o mesmo desempenho nas duas ferramentas, independente da tecnologia utilizada. E o aluno acerta mais na ferramenta 3D (precisão). Ele demora mais, provavelmente, devido ao tempo de aprendizado para resolver os exercícios numa nova perspectiva.

O aluno de mestrado tem a precisão e a cobertura maior no VisAr3D. O tempo maior deve-se a questão do aprendizado de uma nova tecnologia.

O aluno de graduação tem a precisão menor e a cobertura maior, com tempo igual. Foi observado que alguns alunos de graduação, devido à sua imaturidade, faziam as tarefas rapidamente e, por distração, só acertavam metade da questão.

Foi analisado, também, os resultados dos participantes depois da exclusão dos exercícios 1 e 7, muito fáceis e, portanto, considerados *outliers*. A Tabela 6.11mostra os indicadores Precisão, Cobertura e Tempo, segundo esta análise. Ela apresenta a média e o desvio padrão de cada indicador para cada grupo de participantes: doutorado, mestrado e graduação, ao utilizar o protótipo VisAr3D e a ferramenta EA. Por estas análises efetuadas, concluiu-se que “estatisticamente não existe diferença

significativa em relação às variáveis Precisão e Cobertura na utilização das ferramentas VisAr3D e EA", utilizando *alpha-value* igual a 5% e 10%.

No entanto, concluiu-se, também, que "estatisticamente existe diferença significativa em relação à variável Tempo na utilização das ferramentas VisAr3D e EA", utilizando *alpha-value* igual a 10% no grupo de alunos de mestrado. O tempo gasto utilizando a ferramenta 3D foi maior, mas o comportamento da variável tempo é mais uniforme, observado pelo menor valor do desvio padrão.

Tabela 6.11: Média e Desvio Padrão dos resultados dos grupos de participantes para cada ferramenta depois de excluídos os exercícios 1 e 7

		Doutorado		Mestrado		Graduação	
		Média	**Desvio Padrão**	**Média**	**Desvio Padrão**	**Média**	**Desvio Padrão**
Precisão	**EA**	97,59%	3,33%	97,32%	3,65%	98,47%	2,71%
	VisAr3D	99,08%	1,78%	100%	0%	97,22%	4,43%
Cobertura	**EA**	99,80%	0,48%	97,43%	3,08%	96,60%	5,16%
	VisAr3D	99,02%	1,89%	99,76%	0,52%	95,28%	7,78%
Tempo (min)	**EA**	4,47	1,59	4,52	0,61	4,47	1,32
	VisAr3D	4,50	1,12	5,68	1,51	4,03	0,87

Resumindo a tabela, a relação entre a ferramenta 2D e o protótipo VisAR3D, para os alunos de:

1. Doutorado → precisão: aumentou, cobertura: igual, tempo: igual
2. Mestrado → precisão: aumentou, cobertura: aumentou, tempo: aumentou
3. Graduação → precisão: diminuiu, cobertura: diminuiu, tempo: igual

O aluno de doutorado continua apresentando uma precisão melhor na ferramenta 3D, com cobertura igual, ou seja, tendo o mesmo desempenho independente da ferramenta. Contudo, o tempo de utilização fica o mesmo, indicando que não necessitou de tempo de aprendizado, conforme análise da Tabela 6.10. O tempo elevado, naquela tabela foi devido às tarefas 1 e 7.

O aluno de mestrado tem a precisão e a cobertura maior no VisAr3D. O tempo maior deve-se a questão do aprendizado de uma nova tecnologia.

O aluno de graduação teve uma queda também na cobertura, mantendo o tempo igual. Ou seja, o perfil do aluno de graduação deve ser melhor avaliado antes dele utilizar a ferramenta 3D.

6.5.7. Análise Qualitativa

Nas questões 1 e 7, os participantes perdiam um tempo listando o nome das classes de um diagrama. No VisAr3D, eles deveriam se movimentar no espaço 3D e anotar os nomes, ou passar o mouse sobre as classes e ler os seus nomes na janela “output”. Na EA, foi observado, que muitos participantes optaram por listar as classes se baseando numa lista do “Project Browser” disponível à direita da tela. Mesmo os mais experientes, consideravam esta lista como contendo as classes daquele diagrama, que é um erro. Contudo, conclui-se que uma lista de todos os elementos de um diagrama pode ser útil, mesmo no espaço 3D.

Nas questões 2 e 8, os participantes tinham que percorrer mais de um diagrama para responder sobre a herança de uma classe. Foi observado, que os participantes, acostumados a lidar com projetos pequenos, não atentavam sobre a importância deste recurso ao compreender um sistema.

Nas questões 3 e 9, os participantes reconheciam os casos de polimorfismo num diagrama complexo, com muitos atributos e operações. Foi observado que o conceito foi mais bem compreendido num exemplo com muitos elementos. E que eles aprendem a teoria, mas na prática têm dificuldades em identificar os casos de polimorfismo. Durante o experimento, a princípio, eles ficavam inseguros, mas conseguiam responder as questões corretamente nas duas ferramentas.

Nas questões 4 e 10, na ferramenta EA, os participantes tinham que ver as propriedades de cada elemento, enquanto que no protótipo VisAr3D, esferas coloridas identificavam mais rapidamente a quantidade e o nome dos autores, ou melhor, daqueles que participaram da confecção daquele diagrama.

Nas questões 5 e 11, os participantes, na EA tinham que acessar as propriedades de cada classe para saber sobre a sua documentação, enquanto no VisAr3D, para identificar as classes com documentação, bastavam observar a cor azul da sua camada de profundidade. Segundo um participante: “O esquema de cores acelerou bastante o processo”.

Nas questões 6 e 12, foi surpresa observar que, tanto na EA quanto no VisAr3D, os participantes, como alunos, esqueciam, muitas vezes que a correspondência entre o código Java e a modelagem podia ser diferente de atributos. Além de atributo interno a classe, a resposta poderia ser uma associação simples, uma agregação ou uma composição.

Alguns participantes sentiram dificuldades em comparar ambientes de visualização tão diferentes, segundo eles: “fica injusto”, já que uma ferramenta é comercial e a outra um protótipo acadêmico. No entanto, o protótipo do VisAr3D foi muito bem avaliado, mesmo sendo um ambiente novo, com uma nova proposta de navegação e exploração, e com a sua limitação inerente de protótipo. A ferramenta de visualização 2D tradicional utilizada, como ferramenta comercial, adota padrões do Windows já bastante conhecidos, tornando-se um ambiente confortável de navegação, mesmo para quem não a conhecia.

Entre os pontos positivos do protótipo VisAr3D, citados pelos participantes, destacam-se: “Informações, sobre os autores, disponibilizadas no diagrama através de elementos gráficos, não poluem a tela”; “A profundidade das classes, que informam a presença de documentação e que a classe está presente em mais de um diagrama”; “Possui bastante detalhamento de informações contidas no modelo numa representação visual confortável ao usuário”; “Fácil para encontrar informações, poucos cliques para chegar nas informações necessárias, navegação intuitiva”; “Facilidade e objetividade na obtenção de informações”; “Clareza dos diagramas, nova forma de interação com diagrama UML (atrativa), facilidade dos comandos (basta passar o *mouse*)”; “A forma de ver o modelo é mais enriquecida. Percebe-se a potencialidade de exibir mais de um diagrama em uma mesma visão (perspectiva), certamente facilita o aprendizado”.

Entre os aspectos negativos, citados pelos participantes, destacam-se: “Os mecanismos de navegabilidade da ferramenta precisam ser melhorados.”; “Resolução do diagrama quando diminui o zoom”; “Falta de direcionamento nos relacionamentos” e “A janela que mostra a informação deveria pertencer à mesma janela de visualização dos diagramas”. Estas questões podem ter prejudicado na avaliação do VisAr3D, no entanto não atrapalharam na resolução das tarefas, segundo relatos no Questionário de Avaliação.

Foi observado que os alunos de graduação foram mais exigentes quanto à *performance* do protótipo, sendo rígidos na avaliação. E que muito deles se confundiram ou esqueceram os diferentes tipos de relacionamentos, tanto usando a ferramenta 2D, quanto usando a 3D.

O objetivo principal de um estudo de viabilidade, segundo MAFRA *et al.* (2006), não é encontrar uma resposta definitiva, mas criar um corpo de conhecimento sobre a aplicação da tecnologia. Nesse sentido, é possível ao pesquisador avaliar se a

aplicação da tecnologia é viável, ou seja, se atende aos objetivos inicialmente definidos, de forma a justificar ou não a continuação da pesquisa.

Assim, o resultado desse estudo de viabilidade foi positivo, mostrando que o protótipo VisAr3D atende ao seu objetivo geral de apoiar a compreensão de modelos UML em sistemas com muitos elementos de modelagem.

6.5.8. Ameaças à Validade

Algumas ameaças a validade (WOHLIN *et al.*, 2000), relacionadas a este estudo, podem ser resumidas como:

- Como mencionado na Seção 6.3.2.4. (Seleção dos indivíduos), o estudo propõe-se utilizar indivíduos que possuem experiência prévia com modelagem UML. Assim, assume-se que eles são representativos para a população dos engenheiros de software. E, ainda, através do formulário de Caracterização do Participante, não é possível confirmar que as informações fornecidas estejam corretas.
- O estudo não foi executado em um único dia por todos os participantes, isto pode ter influenciado os resultados. Contudo, pela característica do estudo, a pesquisadora precisava observar como cada participante desempenhava cada tarefa e quais eram as suas necessidades e expectativas ao utilizar cada ferramenta (2D e 3D), então não poderia ser conduzido de outra forma.
- Medida de tempo: não é possível informar que a medição de tempo seja precisa.
- Efeito do treinamento: tentou-se evitar este risco, utilizando os mesmos vídeos de treinamento para todos os participantes.
- Efeito de aprendizado: tentou-se minimizar o efeito de aprendizado fazendo com que os participantes resolvessem as mesmas tarefas, utilizando as duas ferramentas em ordens alternadas.
- Embora os participantes tenham sido alertados e instruídos durante o estudo, não é possível garantir que os mesmos não tenham comunicado alguma informação ou resultado a outro participante antes do fim do estudo.
- Sabe-se que os participantes não possuem a mesma habilidade para resolução das tarefas, para tentar minimizar este efeito, os participantes foram alocados aos grupos de forma aleatória.

- Apesar do tamanho da amostra ser limitado do ponto de vista estatístico, é considerado razoável em experimentos em engenharia de software.
- O visualizador 2D *Enterprise Architect* utiliza uma interface gráfica semelhante a outras ferramentas UML 2D, e isto não garante que o tempo medido e o desempenho obtido com outras ferramentas UML 2D tenha a mesma magnitude.
- A proximidade entre os participantes e a pesquisadora. Todos os participantes pertencem à mesma instituição da pesquisadora.
- A pesquisadora que aplicou o experimento é a mesma que desenvolveu o protótipo.
- O pequeno número de exercícios utilizados. Foi optado por utilizar seis exercícios para cada ferramenta (2D e 3D), com o objetivo de não cansar e desmotivar o participante durante a resolução das tarefas.
- O critério utilizado ao elaborar as tarefas em três níveis diferentes de dificuldade pode não corresponder à percepção do participante quanto às dificuldades das tarefas.
- Alguns participantes já conheciam a ferramenta EA e nenhum conhecia o protótipo VisAr3D. E, ainda, a ferramenta 2D utiliza o padrão Windows, padrão este conhecido por todos os participantes.

6.6. Considerações Finais

Este capítulo apresentou a avaliação da abordagem VisAr3D (através do protótipo de mesmo nome), visando analisar os resultados de sua utilização na resolução de tarefas dentro da disciplina de Modelagem de Sistemas, comparado ao uso da ferramenta *Enterprise Architect*.

A partir da condução desta avaliação, foi possível observar alguns pontos fracos da abordagem, bem como identificar oportunidades de melhoria e trabalhos futuros. A avaliação também forneceu evidências positivas de que a abordagem é capaz de apoiar na compreensão de modelos UML em sistemas com muitos elementos de modelagem. E ainda vislumbrar a contribuição da inserção da terceira dimensão nesta compreensão, no interesse dos alunos e como suporte à prática de ensino.

Capítulo 7 - Conclusões

7.1. Epílogo

Ao longo dos anos, a comunidade acadêmica tem se mobilizado para atender a demanda por desenvolvimento de software cada vez mais sofisticado e complexo. Dentro da Engenharia de Software, a Arquitetura de Software, como uma disciplina importante no desenvolvimento de software, torna-se fundamental. A UML, como linguagem de modelagem de sistemas, pode ser empregada para a representação de arquiteturas. Apesar de seu baixo formalismo, possui a vantagem de representar um padrão acadêmico e industrial para a modelagem de software OO. A abordagem VisAr3D procura trazer uma nova dimensão à prática da disciplina de Modelagem de Software, enfrentando este desafio e propondo uma solução diferenciada para o apoio à compreensão de diagramas UML em sistemas com muitos elementos de modelagem, combinando recursos de tecnologias emergentes de visualização 3D, como a Realidade Virtual e a Realidade Aumentada. Ela visa apoiar o processo de ensino e aprendizagem dos futuros arquitetos de software através da comunicação visual, de estruturas e técnicas que facilitam lidar com a complexidade e através de uma experiência mais atraente para o usuário.

A avaliação da abordagem conduzida durante a pesquisa mostrou, através de evidências positivas, de que a abordagem é capaz de apoiar a compreensão de modelos UML em sistemas com muitos elementos de modelagem e ainda despertar o interesse dos alunos ao utilizar a terceira dimensão.

A Seção 7.2 destaca as Contribuições desta pesquisa. A Seção 7.3 apresenta suas Limitações. E a Seção 7.4 descreve sugestões de Trabalhos Futuros.

7.2. Contribuições

Esta tese apresentou os resultados de um trabalho de pesquisa que visa apoiar a compreensão de modelos UML com muitos elementos de modelagem utilizando a Realidade Virtual e Realidade Aumentada, para atender as necessidades dos alunos e do professor. As principais contribuições do trabalho de pesquisa como um todo são:

1. Especificação de um novo ambiente de ensino-aprendizagem para a disciplina de Modelagem de Sistemas, baseado em tecnologias 3D. Conforme o resultado da segunda *quasi* Revisão Sistemática, não foi

encontrada na literatura outra abordagem com esta mesma proposta. Este ambiente apresenta características importantes que o permite:

a) Apoiar a compreensão de modelos UML, possibilitando a comunicação entre os envolvidos e a utilização de sistemas complexos. Estas contribuições, que estão em sintonia com os benefícios da Arquitetura de Software, com exceção da comunicação entre os envolvidos, foram levadas em consideração na avaliação da abordagem.
b) Apoiar à prática da disciplina de Modelagem de Sistemas. Esta contribuição, aliada à utilização de sistemas complexos, teve como principal objetivo atender às necessidades da academia e da indústria.
c) Prover um ambiente dinâmico, intuitivo e moderno para exploração, que tenta despertar o interesse do aluno. Estas contribuições, destaques positivos observados no estudo experimental, desafiam e mobilizam os alunos.
d) Fortalecer os diagramas UML, adicionando informações relevantes sobre o projeto, organizando-as e disponibilizando-as aos usuários de forma sobreposta ao diagrama.
e) Analisar os diagramas em diferentes tipos de detalhes e abstrações. A abordagem incentiva para isso a exploração dos diagramas no espaço 3D.
f) Apoiar tanto o aluno como o professor, sem impor uma curva de aprendizagem como um pré-requisito, fornecendo suporte ao seu público-alvo.
g) Investir no trabalho em equipe, no desempenho de funções especializadas em rodízio, na discussão em grupo e no apoio do instrutor: Estas foram necessidade do ensino de Arquitetura de Software detectada na primeira *quasi* Revisão Sistemática realizada, que mobilizam competências em situações diferenciadas.

2. Desenvolvimento do protótipo VisAr3D, que apesar de implementar apenas parte das características da abordagem, ajudou a mostrar a viabilidade da tecnologia.
3. Planejamento e execução do estudo experimental, que permitiu a análise de suposições importantes sobre a abordagem.
4. Revisão da literatura técnica que para tal foi definido um protocolo, baseado nos conceitos da revisão sistemática, que pode servir de base para futuras investigações sobre esse tema.

De maneira geral, pode-se afirmar que os objetivos foram atingidos, uma vez que, através da avaliação realizada, a maioria das características do ambiente citadas acima foram atendidas, justificando sua contribuição. Contudo, duas dessas características são contribuições que a abordagem oferece mas não foram, ainda, implementadas, e portanto não avaliadas. Elas compõem requisitos fundamentais do ambiente de ensino-aprendizagem para a disciplina Modelagem de Sistemas, ampliando a capacidade de percepção da abordagem VisAr3D. São elas: a comunicação entre os envolvidos (item 1a) e o investimento no trabalho em equipe, na discussão em grupo e no apoio do instrutor (item 1g).

7.3. Limitações

A partir de uma análise crítica sobre a abordagem proposta e sua implementação, puderam ser identificadas algumas limitações. Dentre elas, as que se relacionam com decisões tomadas durante o desenvolvimento da abordagem são listadas nesta Seção.

A abordagem, que tem no seu título o apoio à compreensão de arquitetura de software, se restringe atualmente à compreensão de modelos UML.

A abordagem é bastante ambiciosa e tem apenas uma pequena parte implementada como protótipo. Este protótipo é restrito pelo principal motivo de ter sido desenvolvido por uma pessoa e não por uma equipe multidisciplinar, como um ambiente 3D de apoio ao ensino e aprendizagem deste tipo normalmente requer.

Outras limitações que pertencem ao protótipo desenvolvido são em relação ao seu desempenho e usabilidade, que os participantes deixaram claro durante a avaliação. São questões importantes e necessárias para um ambiente de ensino efetivo. E, ainda, o fato do protótipo só visualizar diagramas estáticos é uma outra limitação atual do trabalho. Os diagramas dinâmicos e a combinação de diagramas estáticos e dinâmicos seriam contribuições importantes que necessitam um arcabouço tecnológico mais estável e mais completo para poderem ser construídos.

Há também limitações com relação a avaliação realizada. Pode-se citar o tamanho da amostra, que é considerado ainda pequeno do ponto de vista estatístico embora aceitáveis para estudos em Engenharia de Software, além das ameaças à validade identificadas.

7.4. Trabalhos Futuros

A realização desse trabalho de pesquisa levou à especificação de um ambiente de ensino-aprendizagem para a disciplina de Modelagem de Sistemas. Essa infraestrutura abre novas perspectivas de pesquisa que podem ser exploradas em trabalhos futuros. Alguns desses trabalhos são sugeridos a seguir:

1) Ampliar a abordagem para apoiar de fato o ensino da disciplina Arquitetura de Software, incorporando o conceito de estilos arquiteturais, permitindo um gerenciamento de riscos e oportunidades e focando na qualidade de software em diferentes níveis de abstração.

2) Incluir a visualização e o rastro dos requisitos do sistema na visualização 3D. De forma a representar as decisões e os requisitos derivados a partir do entendimento do problema, em diferentes níveis de detalhes.

3) Incluir a visualização do código do sistema associado ao elemento de modelagem correspondente. Ao permitir o rastro das dependências entre as partes do código, pode-se avaliar o custo de alterações propostas.

4) Apoiar a prática e o ensino de modelagem de sistemas ao ampliar trabalhos desenvolvidos pelo grupo de Reutilização da COPPE/UFRJ como o PREViA (OLIVEIRA, 2011) e a abordagem ROOSC (MOURA, 2009), desenvolvida no ambiente Odyssey. Isso envolve utilizar a abordagem PREVia, importando modelos para favorecer a percepção e a compreensão na comparação dos objetos comuns ou diferenças entre eles. Ou ainda, mostrar no ambiente 3D a diferença entre duas versões de diagramas. E, ainda, utilizar a abordagem ROOSC, permitindo a visualização 3D dos diagramas, para favorecer o entendimento das métricas, o entendimento das reestruturações sugeridas por esta abordagem e das combinações de aplicações destas reestruturações, dando suporte à tomada de decisão do aluno.

5) Investir num ambiente mais colaborativo online, onde muitos usuários possam compartilhar o mesmo ponto de vista, remotamente, e num ambiente 3D. O objetivo principal é contribuir para a introdução e discussão de novas situações de ensino. A ideia tem por base estimular a criação de grupos virtuais e permitir que interajam, de forma síncrona, guiados por uma base de informação comum, em que se torne possível competir e colaborar de forma a realizar tarefas em conjunto.

6) Organizar, automaticamente, os modelos UML no ambiente 3D. Quando os diagramas são importados, principalmente, os diagramas de classes, eles são exibidos sem os atributos e as operações. Na disposição no espaço 3D, as caixinhas das

classes ficam distantes das outras classes e ligadas por um relacionamento extenso. Isto acontece porque, segundo a proposta da VisAr3D, a exibição destes diagramas deve ser semelhante aos modelos em 2D. A ideia para um trabalho futuro seria utilizar algoritmos para organizar estes modelos no espaço tridimensional sem descaracterizar o diagrama e torná-los diferentes dos modelos 2D correspondentes.

7) Oferecer oportunidades para a utilização de outros dispositivos interativos, diferenciados do padrão (*mouse* e teclado), como os óculos, as luvas (*data-gloves*) ou gestos, permitindo novas experiências aos alunos e ao professor. Está em fase de desenvolvimento, associado a um projeto final de curso do aluno Ariel Schvartz, um protótipo de interação para interface computacional que será acoplado ao protótipo VisAr3D. Este protótipo utiliza o dispositivo da Microsoft, chamado Kinect[1] (KINECT, 2011) para controlar os comandos de interação com o mundo virtual através de multigestos. Com gestos simples, os usuários podem substituir a interação feita com o *mouse* ou teclado no Módulo de Realidade Virtual. Através de sensor usado na captação dos seus movimentos, ele poderá explorar o ambiente, movimentando-se no espaço 3D e acessando os menus.

8) Permitir a utilização da estereocopia, que é uma técnica relacionada com a percepção espacial que pode ser utilizada no apoio ao ensino da disciplina de Modelagem de Sistemas. A possibilidade de proporcionar a imersão aumenta a sensação de realismo e, por isso, pode trazer motivação ou um ganho didático aos estudantes, que pode ser averiguado através de uma avaliação da utilização deste recurso.

9) Realizar novos estudos experimentais e aplicá-los numa situação real de ensino. Após a ampliação da abordagem para apoiar o professor a introduzir procedimentos de ensino como: aprendizagem por descoberta, método de solução de problemas, jogo, trabalho em grupo e estudo de caso, um estudo experimental pode ser conduzido em sala de aula, fortalecendo as evidências obtidas e indicando direções para o aprimoramento da abordagem.

[1] O Kinect é uma tecnologia para jogos sensível ao movimento, uma combinação altamente inovadora de câmeras, microfones e software que transforma seu corpo em um controle de videogame lançado pela Microsoft em 2010. Ele é capaz de permitir aos jogadores interagir com os jogos eletrônicos sem a necessidade de ter em mãos um controle ou *joystick*, inovando no campo da jogabilidade.

Referências Bibliográficas

ALFERT, K., FRONK, A., 2000, “3-dimensional visualization of Java class relations”, In: *Proceedings of the The Fifth World Conference on Integrated Design & Process Technology*, pp. 91-106 Dallas, Texas, Jun.

AL-TAHAT, K. S., SEMBOK, T. M. T., IDRIS, S. B., 2001, “Using design patterns in the development of a planner-based courseware system”, In: *Proceedings of the IEEE Region 10 International Conference on Electrical and Electronic Technology*, pp. 873–876, Bangi, Malaysia, Aug.

AZUMA, R. T., BAILLOT, Y., BEHRINGER, R. *et al.*, 2001, “Recent Advances in Augmented Reality”, *IEEE Computer Graphics and Applications*, v. 21, n. 6 (Nov.), pp. 34-47.

BAKER, A., NAVARRO, E. O., VAN DER HOEK, A., 2005, "An experimental card game for teaching software engineering processes", *Journal of Systems and Software*, v. 75, n. 1-2 (Feb.), Software Engineering Education and Training, pp. 3-16.

BAR-YAM, M., RHOADES, K., SWEENEY, L. B. *et al.* , 2004, “Changes in the teaching and learning process in a complex education system, Necsi Research projects”, disponível em: <http://www.necsi.edu/research/management/education/teachandlearn.html>, acesso em: 16 nov. 2011.

BASILI, V.; CALDIERA, G.; ROMBACH, H., 1994, "Goal Question Metric Paradigm", Encyclopedia of Software Engineering, v. 1, n. edited by John J. Marciniak, John Wiley & Sons, pp. 528-532.

BASS, L., CLEMENTES, P. KAZMAN, R., 2003, *Software Architecture in Practice*, Second Edition, Boston, Pearson Education Inc.

BELL, J., FOGLERL, H. S., 1995, “The Investigation and Application of Virtual Reality as an Educational Tool”, In: *Proceedings of the American Society for Engineering Education Annual Conference*, pp. 1718-1728, Anheim, USA, Jun.

BERGHAMMER, R., FRONK, A., 2003, “Applying Relational Algebra in 3D Graphical Software Design”, In: Proceedings of the Seventh International Seminar on Relational Methods in Computer Science (RelMiCS '03), R. Berghammer, B. Möller, and G. Struth, eds., pp. 62-74, Bad Malente, Germany, May.

BIOLCHINI, J., MIAN, P. G., NATALI, A. C. C. *et al.*, 2005, *Systematic Review in Software Engineering*, In: Technical Report ES 679/05, COPPE/UFRJ, Rio de Janeiro.

BOER, R. C., FARENHORST, R, VAN VLIET, H., 2009, “A Community of Learners Approach to Software Architecture Education”, In: *Proceedings of the 22nd Conference on Software Engineering Education and Training (CSEET '09)*, pp. 190-197, Hyderabad, India, Feb.

BOOCH, G., 1994, *Object-Oriented Analysis and Design with Applications*, Second Edition, Redwood City Addison Wesley.

BOOCH, G., RUMBAUGH, J., JACOBSON, I., 1998, *The Unified Modeling Language User Guide*, First Edition, Addison Wesley.

BUCCI, P., LONG, T. J., WEIDE, B. W., 1998, “Teaching software architecture principles in CS1/CS2”. In: *Proceedings of the Third International Workshop on Software Architecture*, pp. 9-12, Orlando, USA, Nov.

CARDOSO, A., LAMOUNIER, E. A., 2008, “Aplicações na Educação e Treinamento”, In: *Realidade Virtual e Aumentada - Uma Abordagem Tecnológica*, v. 1, Editora SBC - Sociedade Brasileira de Computação, Porto Alegre, pp. 343-357.

CHENOWETH, S., ARDIS, M., DUGAS, C., 2007, “Adapting cooperative learning to teach software architecture in multiple role-teams”, In: *Proceedings of the ASEE Annual Conference and Exposition*, pp. 1-12, Honolulu, USA, Jun.

CONN, R., 2002, “Developing software engineers at the C-130J software factory”, *IEEE Software*, v. 19, n. 5 (Oct.), pp 25-29.

COPLIEN, J. O., SCHMIDT, D. C., 1995, *Pattern Languages of Program Design*, Addison-Wesley Publishing, Reading.

CREIGHTON, O., SINGER, M., 2008, “Who leads our future leaders? on the rising relevance of social competence in software development”, In: *Proceedings of the International Conference on Software Engineering*, pp. 23-26, Leipzig, Germany, May.

DE LUCENA, V.F., BRITO, A., GOHNER, P., 2006, "A Germany-Brazil Experience Report on Teaching Software Engineering for Electrical Engineering Undergraduate Students", In: *Proceedings of the 19th Conference on Software Engineering Education & Training (CSEET'06)*, pp. 69-76, Turtle Bay, USA, Apr.

DE TROYER, O., BILLE, W., KLEINERMANN, F., 2009, “Defining the semantics of conceptual modeling concepts for 3D complex objects in virtual reality”, *Journal on Data Semantics XIV*, vol. 5880 of Lecture Notes in Computer Science, Springer Berlin / Heidelberg, pp. 1-36.

DENZLER, C., GRUNTZ, D., 2008, “Design patterns: Between programming and software design”, In: *Proceedings of the 30th International Conference on Software Engineering*, pp. 801-804, Leipzig, Germany, May.

DIEHL, S.; 2007, Software Visualization: Visualizing the Structure, Behaviour, and Evolution of Software, Springer, 185p.

DIKEL, D., KANE, D., ORNBURN, S. *et al.*, 1997, “Applying software product-line architecture”, *Computer*, v. 30, n. 8 (Aug.) pp. 49-55.

DONGSUN, K., SUNTAE, K., SEOKHWAN, K. *et al.*, 2008, “Software engineering education toolkit for embedded software architecture design methodology using robotic systems”, In: *Proceeding of the 15th Asia-Pacific Software Engineering Conference*, pp. 317-324, Beijing, China, Dec.

DUGERDIL, PH., ALAM, S., 2008, “Execution Trace Visualization in a 3D Space”, In: *Proceeding of the 5th IEEE International Conference on Information Technology : New Generations (ITNG 2008)*, pp. 38-43, Las Vegas, Apr.

DWYER, T., 2001, "Three dimensional UML using force directed layout", In:: *Proceedings of the APVis '01 Proceedings of the 2001 Asia-Pacific symposium on Information Visualization”,* pp. 77-85, Sydney, Australia, Dec.

ENTERPRISE ARCHITECT, Disponível em: <http://www.sparxsystems.com.au>, Acesso em: 16 nov. 2011.

FEIJS, L., JONG, R.D., 1998, “3D Visualization of Software Architectures”, *Communication of the ACM*, v. 41, n. 12 (Dec.), pp. 73-78.

FLUX DEVELOPER WIKI, Disponível em: <http://wiki.developer.flux.com>, Acesso em: 16 nov. 2011.

FRANCK, G., SARDESAI, M., WARE, C., 1995, “Layout and structuring object oriented software in three dimensions”, In: Proceedings of the 1995 conference of the Centre for Advanced Studies on Collaborative research (CASCON '95), pp. 22-22, Toronto, Ontario, Canada, Nov.

GARLAN, D, PERRY, D., 1995, “Introduction to the special issue on software architecture”. *IEEE Transactions on Software Engineering*, v. 21, n. 4 (Apr.), pp. 269-274.

GARLAN, D., SHAW, M., 1994, *An Introduction to Software Architecture*, In: Technical report CS-94-166, Carnegie Mellon University.

GAST, H., 2008, “Patterns and traceability in teaching software architecture”, In: *Proceedings of the 6th International Conference on Principles and Practice of Programming in Java (PPPJ 2008)*, pp. 23-31, Modena, Italy, Sep.

GEF3D, “GEF3D Eclipse Project Website”, <http://www.eclipse.org/gef3d/>, Acesso em: 16 nov. 2011.

GERSHON, N. D., 1994, From perception to visualization, In Scientific Visualization: Advances and Challenges, Academic Pres.

GIL, J., KENT, S., 1998, "Three dimensional software modelling", In: *Proceedings of the 20th international conference on Software Engineering*, pp. 105-114, Kyoto, Japan, Apr.

GRISHAM, P. S., HAWTHORNE, M. J., PERRY, D. E., 2007, “Architecture and design intent: An experience report”, In: *Proceedings of the Second Workshop on Sharing and Reusing architectural Knowledge Architecture, Rationale, and Design Intent (SHARK-ADI'07)*, pp. 12-18, Minneapolis, USA, May.

HAMMOUDA, I., GULDOGAN, O., KOSKIMIES, K. *et al.*, 2004, “Tool-Supported Customization of UML Class Diagrams for Learning Complex Systems”, In: Proceedings of the 12th International Workshop on Program Comprehension (IWPC 2004), pp. 24-33, Bari, Italy, Jun.

HAYDT, R.C.H., 2006, *Curso de didática geral*, 8ª ed. São Paulo, Editora Ática.

HAZZAN, O., KRAMER, J., 2007, "Abstraction in Computer Science & Software Engineering: A Pedagogical Perspective", *Frontier Journal*, v. 4, n. 1 (Jan.), Kluwer Academic Publisher Editorial, pp. 6-14.

HEIDRICH, F.E., CLARO, A., PEREIRA, A., 2003, “Simulação Computacional de Iluminação Natural através de Ambientes em VRML”, In: Anais do VII Congresso Ibero-americano de Gráfica Digital (Sigradi 2003), pp. 117-120, Rosario, Nov.

HILBURN, T. B., TOWHIDNEJAD, M., 2007, “A Case for Software Engineering”, In: *Proceedings of the 20th Conference on Software Engineering Education and Training (CSEET’2007)*, pp. 107-114, Dublin, Ireland, Jul.

HOFMEISTER, C., NORD, R. L., SONI, D., 2005, “Global Analysis: Moving from software requirements specification to structural views of the software architecture”, *IEE Proceedings Software*, v. 152, n. 4 (Aug.), pp. 187-197.

HUANG, S., DISTANTE, D., 2006, “On Practice-Oriented Software Engineering Education” In: *Proceedings of the 19th Conference on Software Engineering Education and Training Workshops (CSEETW'06)*, pp. 15-20, North Shore Oahu, USA, Apr.

HUOTARI, J., 2004, “Integrating uml views with visual cues”, In: *Second International Conference on Coordinated and Multiple Views in Exploratory Visualization*, IEEE Computer Society,pp. 84-92, London, England, Jul.

IRANI, P., WARE, C., 2000, "Diagrams based on structural object perception", in *Proceedings of the working conference on Advanced Visual Interfaces*, pp. 61-67, Palermo, Italy, May.

ITOH, T., YAMAGUCHI, Y., IKEHATA, Y. *et al.*, 2004, “Hierarchical Data Visualization Using a Fast Rectangle-Packing Algorithm”, *IEEE Transactions on Visualization and Computer Graphics*, vol. 10, n. 3 (May), pp. 302-313.

JABREF versão 2.4.2, Disponível em: <http://jabref.sourceforge.net/index.php>, Acesso em: 04 mai. 2009.

JABREF versão 2.6, Disponível em: <http://jabref.sourceforge.net/index.php>, Acesso em: 01 dez. 2011.

JACOBSON, M. and Working Group 2 Collaborators, Complex Systems and Education: Cognitive, Learning, and Pedagogical Perspectives, 2003, Disponível em: <http://www.necsi.edu/events/cxedk16/cxedk16_2.html>, Acesso em: 01 dez. 2011.

JENNINGS, N.R., 2001, “An agent-based approach for building complex software systems”, *Communications of the ACM*. v. 44, n. 4, pp. 35-41.

JMP, Disponível em: http://www.jmp.com, Acesso em: 03 Out 2011.

KARAM, O., QIAN, K., DIAZ-HERRERA, J., 2004, “A model for SWE course “Software Architecture and Design”, In: *34th ASEE/IEEE Frontiers in Education Conference*, pp. 1-5, Savannah, GA, Oct.

KATO, H., “ARToolkit”, Human Interface Technology Laboratory, University of Washington, Disponível em: <http://www.hitl.washington.edu/artoolkit>, Acesso em: 16 nov. 2011.

KAZMAN, R., ABOWD, G., BASS, L. *et al.*, 1996, “Scenario-based analysis of software architecture”, *IEEE Software*, v. 13, n. 6 (Nov.), pp. 47-55.

KINECT, Disponível em: <http://www.xbox.com/en-US/kinect>, Acesso em: 16 nov. 2011.

KIRNER, C., TORI, R., 2006, “Fundamentos de Realidade Aumentada”. In: *Fundamentos e Tecnologia de Realidade Virtual e Aumentada*, v. 1, Editora SBC - Sociedade Brasileira de Computação, Porto Alegre, pp. 22-38.

KNODEL, J.; MUTHIG, D.; NAAB, M., 2006, "Understanding Software Architectures by Visualization - An Experiment with Graphical Elements". In: Proceedings of the 13th Working Conference on Reverse Engineering (WCRE '06), pp. 39-50, Benevento, Italy.

KRAMER, J., 2007, “Is Abstraction the Key to Computing?”, *Communication of ACM*, v. 50, n. 4. (Apr.), pp. 36-42.

LAGO, P., VAN VLIET, H., 2005, “Teaching a Course on Software Architecture”, In: *Proceedings of the 18th Conference on Software Engineering Education and Training (CSEE&T2005)*, pp. 35-42, Ottawa, Canada, Apr.

LANGE, C.F.J., WIJNS, M.A.M., CHAUDRON, M.R.V., 2007, "A Visualization Framework for Task-Oriented Modeling Using UML", In: *Proceedings of the 40th Hawaii International Conference on System Sciences*, pp. 289ª, Big Island, Hawaii, USA, Jan.

LEE, A. H., 2003, “A manageable web software architecture: Searching for simplicity”, *ACM SIGCSE Bulletin*, v. 35, n. 1 (Jan.), pp. 229-233.

LIBÂNEO, J. C., 1990, *Didática*, São Paulo , Editora Cortez.

LOPES, L. F. B., 2005, “O Estudo e a Implementação de Interfaces para Utilização em Sistemas de Realidade Aumentada”, Dissertação de M.Sc, Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha, Marília, SP, Brasil.

LUCKESI, C.C., 2005, *Avaliação da Aprendizagem Escolar*, 17ª ed. São Paulo, Cortez Editora.

MAENNISTOE, T., SAVOLAINEN, J., MYLLÄRNIEMI, V., 2008, “Teaching software architecture design”, In: *Proceedings of the 7th IEEE/IFIP Working Conference on Software Architecture (WICSA 2008)*, pp. 117-124, Vancouver, Canada, Feb.

MAFRA, S. N., TRAVASSOS, G. H., 2006, *Estudos Primários e Secundários apoiando a busca por Evidência em Engenharia de Software*, In: Relatório Técnico ES-687/06, COPPE/UFRJ, Rio de Janeiro.

MAFRA, S., BARCELOS, R., TRAVASSOS, G. H., 2006. "Aplicando uma Metodologia Baseada em Evidência na Definição de Novas Tecnologias de Software". In: Proceedings of the 20th Brazilian Symposium on Software Engineering (SBES 2006), v. 1, pp. 239 – 254, Florianopolis. October.

MCGREGOR, J. D., BACHMAN, F., BASS, L. *et al.*, 2007, “Using an architecture reasoning tool to teach software architecture”, In: *Proceedings of the 20th Conference on Software Engineering Education & Training*, pp. 275-282, Dublin, Ireland, Jul.

MCINTOSH, P., 2009, “X3D-UML : user-centred design, implementation and evaluation of 3D UML using X3D”, PhD. Thesis, RMIT University.

MCINTOSH, P., HAMILTON, M., VAN SCHYNDEL, R., 2005, “X3D-UML: enabling advanced UML visualization through X3D”, In: Proceedings *of the 10th International Conference on 3D Web Technology (Web3D 2005)*, pp. 135-142, Bangor, U.K, Mar.

MCINTOSH, P., HAMILTON, M., VAN SCHYNDEL, R., 2008, “X3D-UML: 3D UML State Machine Diagrams”, In: *Proceedings of the 11th international conference on Model Driven Engineering Languages and Systems*, pp. 264-279, Toulouse, France, Springer-Verlag, Oct.

MENDES, A., 2002, *Arquitetura de Software: Desenvolvimento Orientado para Arquitetura*, Rio de Janeiro, Editora Campus.

MEYER, B., 2001, “Software engineering in the academy”, *Computer*, v. 34, n. 5 (May), pp. 28-35.

MILGRAM, P., KISHINO, F., 1994, “A Taxonomy of Mixed Reality Visual Displays”, *IEICE Transactions on Information Systems*, v. E77-D, n. 12 (Dec.), pp. 1321-1329.

MOHRENSCHILDT, M. V., PETERS, D. K., 1998, "Draw-Bot: A project for teaching software engineering", In: *Proceedings of the 28th Annual Frontiers in Education Conference*, pp. 1022-1027, Tempe, USA, Jun.

MOURA, A.M.M., 2009, “ROOSC: Uma Abordagem de Reengenharia de Sistemas Orientados a Objetos para Componentes Baseada em Métricas”, Dissertação de Mestrado em Engenharia de Sistemas e Computação, Universidade Federal do Rio de Janeiro.

NAVEDA, J. F., 1999, “Teaching architectural design in an undergraduate software engineering curriculum”, In: *Proceedings of the 29th Annual Frontiers in Education Conference (FIE '99)*, pp. 12B1/1-12B1/4, San Juan, Puerto Rico, Nov.

OLIVEIRA, M.S., 2011, “PREViA: Uma Abordagem para a Visualização da Evolução de Modelos de Software”, Dissertação de Mestrado em Engenharia de Sistemas e Computação, Universidade Federal do Rio de Janeiro, Brasil.

OPENGL; Disponível em: <http://www.opengl.org/>, Acesso em: 16 nov. 2011.

PAI, M., MCCULLOCH, M., GORMAN, J.D., *et al.*, 2004, "Systematic reviews and meta-analyses: an illustrated, step-by-step guide", *The National Medical Journal of India*, v. 17, n. 2, pp. 86-95.

PARNAS, D.L., 1972, “On the criteria to be used in decomposing systems into modules”, *Communications of the ACM*, v.15, n. 12 (Dec.), pp. 1053-1058.

PAVLOVIC, V.I., SHARMA, R., HUANG, T.S., 1997, “Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 19, n. 7 (Jul.), pp. 677-695.

PELLENS, B., DE TROYER, O., KLEINERMANN, F., 2008, “CoDePA: A conceptual design pattern approach to model behavior for X3D worlds”, 2008, In: Proceeding of the 13th International Conference on 3D Web Technology, Los Angeles, California, USA, pp. 91-99, Aug.

PERRY, D. E., WOLF, A. L., 1992, “Foundations for the Study of Software Architecture”, *ACM SigSoft Software Engineering Notes*, v. 17, n. 4 (Oct.), pp. 40-52.

PIEKARSKI, W., THOMAS, B., 2002, “ARQuake: The Outdoor Augmented Reality Gaming System”, *Communications of the ACM*, v. 45, n. 1 (Jan.), pp. 36-38.

PILGRIM J.V., DUSKE, K., 2008, "Gef3D: a framework for two-, two-and-a-half-, and three-dimensional graphical editors", In: *Proceedings of the 4th ACM symposium on Software visualization*, pp. 95-104, Ammersee, Germany: ACM, Sep.

PILGRIM, J.V., DUSKE, K., MCINTOSH, P., 2009, "Eclipse GEF3D: Bringing 3D to Existing 2D Editors", *Journal of Information Visualisation*, v. 8, n. 2, pp. 107-119, summer.

POUR, G., 2006, "Software Engineering for Pervasive Computing: An Outlook for Educational Reform," In: *Proceedings of the 5th IEEE/ACIS International Conference on Computer and Information Science and 1st IEEE/ACIS International Workshop on Component-Based Software Engineering - Software Architecture and Reuse (ICIS-COMSAR'06)*, pp. 168-173, Honolulu, USA, Jul.

PREECE, J., ROGERS, Y., SHARP, H., 2005, *Design de Interação: além da interação homem-computador*, Porto Alegre, Ed. Bookman.

PRENSKY, M., 2001, “Digital Natives, Digital Immigrants”, *On The Horizon*, NCB University Press, vol. 9, n. 5 (Oct,), pp. 1-6.

RADFELDER, O., GOGOLLA, M., 2000, “On Better Understanding UML Diagrams through Interactive Three-Dimensional Visualization and Animation”, In: *Proceedings of the Working Conference Advanced Visual Interfaces (AVI '00),* pp. 292-295, Palermo, Italy, May.

RODRIGUES, C.S.C., 2010, “VisAr3D: An Approach to Software Architecture Teaching Based on Virtual and Augmented Reality”. In: Proceedings of the 32nd International Conference on Software Engineering - Doctoral Symposium, pp. 351-352, Cape Town, South Africa, May.

RODRIGUES, C.S.C., RODRIGUES, P.F.N., WERNER, C.M.L., 2008, “An Application of Augmented Reality in Architectural Education for Understanding Structural Behavior through Models”, In: *Proceedings of the X Symposium on Virtual and Augmented Reality (SVR2008)*, pp. 163-166, João Pessoa, USA, May.

RODRIGUES, C.S.C., WERNER, C.M.L., 2008, Realidade Aumentada e Engenharia de Software, In: Relatório técnico 1/2008, COPPE/UFRJ, Rio de Janeiro.

RODRIGUES, C.S.C., WERNER, C.M.L., 2009a, “VisAr3D: Uma abordagem baseada em Realidade Aumentada para o Ensino de Arquitetura de Software”, In: *Proceedings of the XII Conferencia Iberoamericana de Ingeniería de Requisitos Ambientes de Software (IDEAS 2009)*, pp. 361-372, Medellín, Abr.

RODRIGUES, C.S.C., WERNER, C.M.L., 2009b, “Software Architecture Teaching: A Systematic Review”, In: Proceedings of the *9th World Conference on Computers in Education (WCCE 2009)*, pp. 1-10, Bento Gonçalves. Porto Alegre, Jul.

RODRIGUES, C.S.C., WERNER, C.M.L., 2009c, Uma Revisão Sistemática sobre as Iniciativas Realizadas no Ensino de Arquitetura de Software, In: *Relatório técnico ES-728/09*, COPPE/UFRJ, Rio de Janeiro.

RODRIGUES, C.S.C., WERNER, C.M.L., 2011, “Making the Comprehension of Software Architecture Attractive”, In: *24th IEEE-CS Conference on Software Engineering Education and Training (CSEE&T 2011)*, Honolulu, pp. 416-420.

SANATNAMA, H., BRAHIMI, F., 2010, “Graph Drawing Algorithms: Using in Software Tools”, Journal of Applied Sciences, v. 10, n. 17, pp. 1894-1901, Jul.

SCHAUER, R., KELLER, R. K., 1998, “Pattern visualization for software comprehension”, In: *Proceedings of the 6th International Workshop on Program Comprehension*, pp. 4-12, Ischia, Italy, Jun.

SECOND LIFE, “Second Life: Your World. Your Imagination”, Disponível em: <http://secondlife.com>, Acesso em: 14 dez 2011.

SEI – Software Engineering Institute, Disponível em: <http://www.sei.cmu.edu/>, Acesso em: 01 dez. 2011.

SHAW, M., 2000, “Software Engineering Education: A Roadmap”. In: Proceedings of the 22nd Conference on the Future of Software Engineering (ICSE), pp. 371-380, Limerick, Ireland, Jun.

SHAW, M., CLEMENTS, P., 2006, “The golden age of software architecture”, *IEEE Software*, v. 23, n. 2, pp 31-39.

SHAW, M., GARLAN D., 1996, *Software Architecture: Perspectives on an Emerging Discipline*, Prentice Hall, Inc.

SHNEIDERMAN, B., 1996, “The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations”, In: *Proceedings of the 1996 IEEE Conference on Visual Languages*, pp. 336-343, Boulder, CO, Set.

SIMON, H.A., 1996, *The Sciences of the Artificial*. MIT Press.

SISCOUTTO, R. A., RAPOSO, A. B., TORI, R. *et al.*, 2006, “Estereoscopia”, In: *Fundamentos e Tecnologia de Realidade Virtual e Aumentada*, v. 1, Editora SBC - Sociedade Brasileira de Computação, Porto Alegre, pp. 221-245.

STANEK, W. R., 1996, *HTML, Java, CGI, VRML, SGML Web Publishing*, Sams.net Publishing.

SVAHNBERG, M., MAERTENSSON, F., 2007, “Six years of evaluating software architectures in student projects,” *Journal of Systems and Software*, v. 80, n. 11 (Nov.), pp. 1893-1901.

TEYSEYRE, A.R., CAMPO, M.R., 2009, "An Overview of 3D Software Visualization", *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, n. 1 (Jan.), pp. 87-105.

THADEN, U., STEIMANN, F., 2003, "Animated UML as a 3d-illustration for teaching OOP", In: *ECOOP 2003 - Object-Oriented Programming. Proceedings of the 17th European Conference*, pp. 1-6, Darmstadt, Germany, Jul.

THOMPSON, J. B., EDWARDS, H., 2009, "Preparing Graduate Students for Industry and Life Long Learning: - A Project Based Approach", In: Education and Technology for a Better World: *9th World Conference on Computers in Education (WCCE 2009)*, pp. 292-30, New York: Springer, Jul.

TNMC, The New Media Consortium, "The Horizon Report 2011", Disponível em: <http://net.educause.edu/ir/library/pdf/HR2011.pdf>, Acesso em: 02 dez. 2011.

TORI, R., 2010, *Educação sem distância: as tecnologias interativas na redução de distâncias em ensino e aprendizagem*, Editora Senac São Paulo.

TORI, R., KIRNER, C., SISCOUTO, R., 2006, *Fundamentos e Tecnologia de Realidade Virtual e Aumentada*, v. 1, Editora SBC - Sociedade Brasileira de Computação, Porto Alegre.

TRAVASSOS, G.H., SANTOS, P.M., MIAN, P.G. *et al.*, 2008, "An Environment to Support Large Scale Experimentation in Software Engineering", In: *13th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS 2008)*, pp. 193-202, Belfast, Northern Ireland, Mar.

TREHARNE, H., 2002, "Supplementing a UML Development Process with B", In: *Proceedings of FME 2002:Formal Methods - Getting IT Right, International Symposium of Formal Methods Europe*, pp. 568-586, Copenhagen, Denmark, Jul.

VALLIESWARAN, V., MENEZES, B., 2007, "ArchKriti: A software architecture based design and evaluation tool suite", In: *Proceedings of the Fourth International Conference on Information Technology-New Generations (ITNG '07)*, pp. 701-706, Las Vegas, USA, Apr.

VARMA, V., GARG, K., 2005, "Case studies: The potential teaching instruments for software engineering education", In: *Proceedings of the Fifth International Conference on Quality Software (QSIC 2005)*, pp. 279-284, Melbourne, Australia, Sep.

WAGNER, D., PINTARIC, T., LEDERMANN, F. *et al.*, 2005, “Towards Massively Multi-User Augmented Reality on Handheld Devices”, In: *Proceedings of the 3dn International Conference on Pervasive Computing*, pp. 208-219, Munich, Germany, May.

WANG, A. I., ARISHOLM, E., JACCHERI, L., 2007, “Educational approach to an experiment in a software architecture course”, In: *Proceedings of the 20th Conference on Software Engineering Education & Training*, pp. 291-298, Dublin, Ireland, Jul.

WANG, A. I., SOERENSEN, C. F., 2006, “Writing as a tool for learning software engineering”, In: *Proceedings of the 19th Conference on Software Engineering Education & Training*, pp. 35-42, Turtle Bay, USA, Apr.

WANG, A. I., STAELHANE, T., 2005, “Using post mortem analysis to evaluate software architecture student projects”, In: *Proceedings of the 18th Conference on In Software Engineering Education and Training*, pp. 43-50, Ottawa, Canada, Apr.

WANG, W. L.; SCANNELL, D., 2005, “An architecture-based software reliability modeling tool and its support for teaching”, In: *Proceedings of the Frontiers in Education Conference (FIE)*, pp. T4C-15, Indianapolis, USA, Oct.

WARE, C.; HUI, D. FRANCK, G., 1993, “Visualizing object oriented software in three dimensions”, In CASCON '93: Proceedings of the 1993 conference of the Centre for Advanced Studies on Collaborative research, Toronto, Canadá, October, pp. 612-620.

WERNER, C., MATTOSO, M., BRAGA, R. *et al.*, 1999, “Odyssey: Infra-estrutura de Reutilização baseada em Modelos de Domínio”, Caderno de Ferramentas do XIII Simpósio Brasileiro de Engenharia de Software (XIII SBES), pp.17-20, Florianópolis, Brasil, Out.

WOHLIN, C.; RUNESON, P.; HÖST, M. *et al.*, 2000, Experimentation in Software Engineering: An Introduction. 1 ed. Norwell, USA, Kluwer Academic Publishers.

X3D, Web3D, Disponível em: <http://www.web3d.org/>, Acesso em: 16 nov. 2011.

Xj3D, Disponível em: <http://www.xj3d.org>, Acesso em: 16 nov. 2011.

XMI, OMG, XML Metadata Interchange (XMI) Specification. Version 2.0, Object Management Group, 2005.

YOUNG, P., MUNRO, M., 1998, “Visualising Software in Virtual Reality”, In: *Proceedings of the IEEE the International Workshop on Program Comprehension*, pp19-26, Ischia, Italy, Jun.

Apêndice A

Formulário de Consentimento

VISAR3D

Este estudo tem como objetivo avaliar a abordagem VisAr3D, considerando seu apoio à compreensão de diagramas de classes UML, utilizando visualização 3D. Para isso, será utilizado um protótipo, que já implementa parte das características da abordagem.

IDADE

Eu declaro ter mais de 18 anos de idade e concordar em participar de um estudo experimental conduzido por Claudia Susie Camargo Rodrigues na COPPE/UFRJ.

PROCEDIMENTO

Este estudo ocorrerá em uma única sessão, que incluirá um treinamento sobre a abordagem VisAr3D. Eu entendo que, uma vez o experimento tenha terminado, os trabalhos que desenvolvi serão estudados visando entender a eficiência dos procedimentos e as técnicas que me foram ensinadas.

CONFIDENCIALIDADE

Toda informação coletada neste estudo é confidencial, e meu nome não será divulgado. Da mesma forma, me comprometo a não comunicar os meus resultados enquanto não terminar o estudo, bem como manter sigilo das técnicas e documentos apresentados e que fazem parte do experimento.

BENEFÍCIOS, LIBERDADE DE DESISTÊNCIA

Eu entendo que os benefícios que receberei deste estudo são limitados ao aprendizado do material que é distribuído e ensinado. Eu entendo que sou livre para realizar perguntas a qualquer momento ou solicitar que qualquer informação relacionada a minha pessoa não seja incluída no estudo. Eu entendo que participo de livre e espontânea vontade com o único intuito de contribuir para o avanço e desenvolvimento de técnicas e processos para a Engenharia de Software.

PESQUISADOR RESPONSÁVEL

Claudia Susie Camargo Rodrigues
Programa de Engenharia de Sistemas e Computação - COPPE/UFRJ

PROFESSOR RESPONSÁVEL

Profa. Cláudia M.L. Werner
Programa de Engenharia de Sistemas e Computação - COPPE/UFRJ

Nome (em letra de forma):_____

Assinatura: _____ **Data**: _____

Apêndice B

Caracterização do Participante

Código do Participante: _____ **Data**: _____

Esta fase da pesquisa tem como objetivo obter informações sobre a sua experiência acadêmica e profissional.

Por favor, NÃO inclua nenhum detalhe que poderá identificá-lo.

Perfil do participante

1) Formação Acadêmica:

( ) Doutorado

( ) Doutorado em Andamento

( ) Mestrado

( ) Mestrado em Andamento

( ) Graduação

( ) Graduação em Andamento

( ) Outro: _____

Ano de ingresso: _____ Ano de conclusão (de previsão): _____

2) Você possui experiência de ensino da disciplina de Modelagem de Sistemas de Software?
( ) Sim, como professor
( ) Sim, como instrutor
( ) Sim, como monitor
( ) Sim, como _____
( ) Não
Se sim, quantos anos de experiência? _____

3) Qual é sua experiência com modelagem UML? (se necessário, marque mais de um item)

( ) já li material sobre modelagem UML.

( ) já participei de um curso sobre modelagem UML.

( ) nunca fiz uma modelagem UML.

( ) já fiz modelagem UML para uso próprio.

( ) já fiz modelagem UML como parte de uma equipe, relacionada a um curso.

( ) já fiz modelagem UML como parte de uma equipe, na indústria.

4) Por favor, explique sua resposta. Inclua o número de semestres ou número de anos de experiência relevante em modelagem UML.

(Ex: “Eu trabalhei por 2 anos fazendo modelagem UML na indústria”)

_____

_____

_____

5) Na escala dos 5 pontos abaixo:

0 = nenhum
1 = estudei em aula ou em livro
2 = pratiquei em projetos em sala de aula
3 = usei em projetos pessoais
4 = usei em projetos na indústria

Marque **uma** opção, indicando o grau de sua experiência em:

Modelagem de sistemas de informação	0	1	2	3	4
Orientação a Objetos	0	1	2	3	4
Java	0	1	2	3	4
Padrões de Projeto	0	1	2	3	4

6) Qual ferramenta UML utiliza (ou utilizou) para modelar Diagramas de Classes?

_____

_____

_____

7) Indique a ferramenta UML mais utilizada e cite alguns pontos positivos da mesma.

_____

_____

_____

8) Cite alguns pontos negativos desta ferramenta UML

_____

_____

_____

9) Em relação ao maior sistema modelado por você:

a) Indique a ferramenta UML utilizada: _____

b) Indique a quantidade aproximada de classes do sistema modelado:
( ) até 20 ( ) de 21 a 50 ( ) de 51 a 100 ( ) de 101 a 200 ( ) mais de 200

c) Houve dificuldades em modelar este sistema? Se sim, por favor, descreva-as.

_____

_____

_____

10) Em relação à sua experiência dentro de sala de aula:

a) Dentro de sala de aula, você já utilizou um sistema com mais de 50 classes? Se sim, com que frequência sistemas desta escala são utilizados?

b) No seu ponto de vista, descreva as vantagens e desvantagens em ensinar/estudar utilizando um sistema com um número grande de classes.

Apêndice C

Tarefas

Código do Participante: _____ **Data**: _____

Instruções

Este estudo será acompanhado por meio de anotações feitas pelo pesquisador. Sempre que possível, verbalize seus pensamentos, para que o experimentador possa melhor avaliar os resultados obtidos. Pergunte e comente tudo que achar necessário.

Contextualização

Você está participando deste estudo como um aluno da disciplina de Modelagem de Sistemas, e utilizará as ferramentas EA (*Enterprise Architect*) e VisAr3D para responder algumas questões relacionadas a um sistema que possui muitos elementos de modelagem.

O sistema de software "*Kernel* do Odyssey" contém cerca de 140 classes. Sua modelagem ainda não foi concluída. Ele é composto, atualmente, pelos diagramas Diagrama1, Diagrama2, Diagrama3, Diagrama4, Diagrama5, Diagrama6, *Models Generic Diagram*, *Model Diagram*, *Model Features Diagram*, *Rule Diagram*, *Tools Agent Controller Diagram*, *Tools Diagram* e *Tools Diagram Diagram*.

Para responder as perguntas de 1 a 6, utilize a ferramenta EA (*Enterprise Architect*).

1) Quais classes compõem o diagrama *Tools Diagram Diagram* pertencente ao Sistema "*Kernel* do Odyssey"? Escreva o nome de cada uma delas.

_____

_____

_____

_____

Esta tarefa foi ❍ **muito fácil** ❍ **fácil** ❍**normal** ❍**difícil** ❍**muito difícil**

2) A classe “ItemModelo” presente no diagrama *Model Diagram* é um elemento de modelagem presente em mais de um diagrama. Liste todas as suas subclasses e superclasses (um nível, ou seja, filhas e mães apenas), dentro de todo o sistema (isto é, elas podem estar em outros diagramas).

Esta tarefa foi ❍ **muito fácil** ❍ **fácil** ❍**normal** ❍**difícil** ❍**muito difícil**

3) Polimorfismo é a propriedade que indica que uma operação pode, apesar de ter o mesmo nome, executar ações diferentes. Como exemplo, os códigos Java a seguir:

```
public class Soma extends OperacaoMatematica {
   public double calcular(double x, double y) {
      return x+y;
   }
}

public class Subtracao extends OperacaoMatematica {
   public double calcular(double x, double y) {
      return x-y;
   }
}
```

Destaque dois (2) exemplos de polimorfismo no diagrama *Model Generic Diagram*.

_____

_____

_____

Esta tarefa foi ❍ **muito fácil** ❍ **fácil** ❍**normal** ❍**difícil** ❍**muito difícil**

4) O professor passou um exercício prático de modelagem para seus alunos e designou você para atuar como Gerente de Projeto. Uma de suas tarefas é fazer relatórios de acompanhamento de trabalho regularmente. Examine o diagrama “Diagrama6” e liste os autores das classes.

_____

_____

_____

Esta tarefa foi ❍ **muito fácil** ❍ **fácil** ❍**normal** ❍**difícil** ❍**muito difícil**

5) Como Gerente de Projeto do seu grupo, você deve acompanhar o trabalho de documentação que está sendo feito no diagrama *Model Diagram*. Indique a quantidade de classes que ainda faltam ser documentadas.

_____

_____

_____

Esta tarefa foi ❍ **muito fácil** ❍ **fácil** ❍**normal** ❍**difícil** ❍**muito difícil**

6) No diagrama *Tools Diagram Diagram*, descubra qual classe possui o código Java abaixo. Indique, em seguida, qual foi a solução dada em forma de modelagem para (*1) e (*2), dentre as seguintes possibilidades: a) associação simples; b) agregação; c) composição; d) atributo interno à classe.

```
public class X {
	private DiagramaSemantico semantic; (*1)
```

```
        private ArestaPadrao edges = new ArestaPadrao(); (*2)
}
```

Esta tarefa foi ❍ **muito fácil** ❍ **fácil** ❍**normal** ❍**difícil** ❍**muito difícil**

Para responder as perguntas de 7 a 12, utilize o protótipo VisAr3D.

7) Escreva o nome das classes que compõem o diagrama *Model Diagram* pertencente ao Sistema “*Kernel* do Odyssey”?

Esta tarefa foi ❍ **muito fácil** ❍ **fácil** ❍**normal** ❍**difícil** ❍**muito difícil**

8) A classe “Diagrama” presente no diagrama *Tools Diagram Diagram* é um elemento de modelagem, presente em mais de um diagrama. Liste todas as subclasses (um nível, ou seja, filhas apenas) da classe “Diagrama”, dentro de todo o sistema (isto é, elas podem estar em outros diagramas).

_____

_____

_____

Esta tarefa foi ❍ **muito fácil** ❍ **fácil** ❍**normal** ❍**difícil** ❍**muito difícil**

9) Os códigos Java a seguir apresentam exemplos de Polimorfismo:

```
public class Soma extends OperacaoMatematica {
   public double calcular(double x, double y) {
      return x+y;
   }
}

public class Subtracao extends OperacaoMatematica {
   public double calcular(double x, double y) {
      return x-y;
   }
}
```

Destaque dois (2) exemplos de polimorfismo no diagrama *Model Generic Diagram*.

_____

_____

_____

Esta tarefa foi ❍ **muito fácil** ❍ **fácil** ❍**normal** ❍**difícil** ❍**muito difícil**

10) Num exercício prático de modelagem, sua tarefa, como Gerente de Projeto, é listar os autores das classes do diagrama *Model Features Diagram*.

_____

_____

_____

Esta tarefa foi ❍ **muito fácil** ❍ **fácil** ❍**normal** ❍**difícil** ❍**muito difícil**

11) Indique a quantidade de classes que já foram documentadas no diagrama *Tools Diagram Diagram*.

_____

_____

_____

Esta tarefa foi ❍ **muito fácil** ❍ **fácil** ❍**normal** ❍**difícil** ❍**muito difícil**

12) No diagrama *Tools Diagram Diagram*, descubra qual classe possui o código Java abaixo. Indique, em seguida, qual foi a solução dada em forma de modelagem para (*1) e (*2), dentre as seguintes possibilidades: a) associação simples; b) agregação; c) composição; d) atributo interno à classe.

```
public class X {
	private NoSemantico semantico; (*1)
	public ArestaPadrao connections = new ArestaPadrao();
(*2)
}
```

_____

_____

_____

Esta tarefa foi ❍ **muito fácil** ❍ **fácil** ❍**normal** ❍**difícil** ❍**muito difícil**

Apêndice D

Questionário de Avaliação

Código do Participante: _____ **Data**: _____

Realização da tarefa

1) Você conseguiu efetivamente realizar todas as tarefas propostas? Comente, se necessário.

( ) Sim ( ) Não ( ) Parcialmente

_____

_____

_____

2) Você ficou satisfeito com o resultado final das tarefas? Comente, se necessário.

( ) Sim ( ) Não ( ) Parcialmente

_____

_____

_____

Treinamento

3) Você considera que o treinamento aplicado para o uso das ferramentas e para a realização das tarefas foi suficiente? O que poderia ser acrescentado/modificado?

( ) Sim ( ) Não ( ) Parcialmente

_____

_____

_____

4) Você considera que existiu alguma dificuldade na interpretação das informações apresentadas sobre a ferramenta? Se sim, por favor, explique.

( ) Sim ( ) Não ( ) Parcialmente

_____

_____

_____

Comparação entre visualização 2D e visualização 3D

A seguir serão apresentados alguns tópicos que servem para comparar a visualização 2D com a visualização 3D. Escolha o tipo de visualização que você considera que melhor contribuiu. Em seguida, faça um comentário a respeito, se necessário.

5) Apoio à compreensão de modelos UML em sistemas com muitos elementos?

( ) 2D ( ) 3D ( ) Não se aplica

_____

_____

6) Apoio à exploração dos modelos?

( ) 2D ( ) 3D ( ) Não se aplica

7) Apoio à resolução das tarefas?

( ) 2D ( ) 3D ( ) Não se aplica

8) Redução da complexidade da visualização?

( ) 2D ( ) 3D ( ) Não se aplica

9) Diminuição da poluição visual, favorecendo a clareza dos diagramas?

( ) 2D ( ) 3D ( ) Não se aplica

10) Facilidade na leitura de detalhes da informação?

( ) 2D ( ) 3D ( ) Não se aplica

11) Disponibilização de ambiente mais intuitivo?

( ) 2D ( ) 3D ( ) Não se aplica

_____

12) Facilidade de obtenção de informações sobre Autoria?

( ) 2D ( ) 3D ( ) Não se aplica

_____

_____

_____

13) Facilidade de obtenção de informações sobre Relacionamentos?

( ) 2D ( ) 3D ( ) Não se aplica

_____

_____

_____

14) Facilidade de obtenção de informações sobre Classes em mais de um diagrama?

( ) 2D ( ) 3D ( ) Não se aplica

_____

_____

_____

15) Facilidade de obtenção de informações sobre a Documentação de uma classe?

( ) 2D ( ) 3D ( ) Não se aplica

_____

_____

_____

16) Suporte à prática de ensino em projetos com muitos elementos de modelagem?

( ) 2D ( ) 3D ( ) Não se aplica

_____

_____

_____

Uso do Protótipo VisAr3D

17) Marque as funções do menu do protótipo VisAr3D que foram mais úteis na execução das tarefas. Comente, se necessário.

( ) Visão *default* ( ) Classe em outros diagramas ( ) Pacote ( ) Atributos/Operações ( ) Autor ( ) Documentação

_____

_____

18) Liste os aspectos positivos do protótipo VisAr3D.

19) Liste os aspectos negativos do protótipo VisAr3D.

20) Por favor, adicione qualquer outro comentário desejado aqui.

Obrigada por sua colaboração!