DISSERTAÇÃO DE MESTRADO

UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

End-User Programming como Apoio ao Desenvolvimento de Sistemas com Realidade Virtual

Fernando Cesar Balbino

ORIENTADORA: Profa. Dra. Junia Coutinho A. Silva
CO-ORIENTADORA: Profa. Dra. Rosângela Ap. D. Penteado

São Carlos
Julho/2003

End-User Programming como Apoio ao Desenvolvimento de Sistemas com Realidade Virtual

Fernando Cesar Balbino

Ficha catalográfica elaborada pelo DePT da Biblioteca Comunitária/UFSCar

B172eu

Balbino, Fernando Cesar.
End-user programming como apoio ao desenvolvimento de sistemas com realidade virtual / Fernando Cesar Balbino. -- São Carlos : UFSCar, 2004.
81 p.

Dissertação (Mestrado) -- Universidade Federal de São Carlos, 2003.

1. Interação homem - máquina. 2. Realidade virtual. 3. Projeto de interfaces. 4. Programação por usuários finais. I. Título.

CDD: 004.019(20ª)

Em relação a todos os atos de iniciativa e de criação, existe uma verdade fundamental cujo desconhecimento mata inúmeras idéias e planos esplêndidos: a de que no momento em que nos comprometemos, a Providência move-se também.
Toda uma corrente de acontecimentos brota da decisão, fazendo surgir a nosso favor toda a sorte de incidentes, encontros e assistência material que nenhum homem sonharia que viesse em sua direção.
O que quer que você possa fazer, ou sonha que possa fazer, faça.
Coragem contém genialidade, poder e magia.
Comece agora.

Johann Wolfgang von ***Goethe***

AGRADECIMENTOS

Se alguém me pedisse para destacar as páginas mais importantes desta dissertação, não hesitaria em dizer que são estas, as páginas em que tento reproduzir, com a maior fidelidade possível, o nome de todas as pessoas que contribuíram, direta ou indiretamente, para a realização deste trabalho. Obviamente, ao longo de todos os capítulos, estão os resultados de uma importante etapa de minha vida e que não deixam de ser tesouros valiosos. Mas é possível que um dia eu não me recorde tão claramente de tudo o que escrevi, das idéias que defendi, dos detalhes da pesquisa que realizei. Entretanto, as pessoas a quem agradeço nestas linhas certamente serão lembradas nitidamente durante toda a minha vida.

Em primeiro lugar, agradeço a Deus, nosso Pai Maior, a Inteligência Suprema, o Princípio de todas as coisas. Agradeço pelo esplendor de minha vida, pelo amparo concedido nos dias de preocupação e tristeza, pelo apoio durante as dificuldades, pelos sorrisos das pessoas com quem convivo e que sempre trazem alegrias para a minha alma.

Meu agradecimento especial à minha orientadora Junia Coutinho Anacleto Silva e à minha co-orientadora Rosângela Aparecida Dellosso Penteado, por terem acreditado em meu potencial para a realização deste trabalho. Sem vocês, eu não teria sequer iniciado. Com vocês, pude conclui-lo com muita satisfação e bem-estar, porque acima de tudo, tivemos uma relação de amizade, e não simplesmente de aluno e professor.

A meus queridos pais, José e Maria Luiza, pelo incentivo constante nas pequenas e grandes realizações de minha vida. Por nunca terem tomado uma decisão que coubesse a mim, mas por terem me ensinado a refletir sobre as conseqüências de cada decisão antes que eu mesmo optasse pelo caminho mais adequado. Obrigado, meu pai e minha mãe, por tantas renúncias em prol de meu bem-estar, de minha formação acadêmica e profissional.

À minha amada esposa Luciana, que desde os dias de nossos primeiros encontros, vem abrilhantando a minha vida com a intensa luz de seu sorriso e acariciando a minha alma com a doçura do seu olhar. Obrigado pela paciência nos dias em que estive distante (uma distância nem sempre física), tendo que compartilhar o meu tempo entre os diversos compromissos, entre eles o desenvolvimento deste trabalho, dedicando a você o carinho insuficiente. Apesar dos dias difíceis, você nunca deixou de dizer “eu te amo”. Eu também amo você, pelo seu companheirismo, pela sua dedicação, pela sua beleza e pelo seu amor!

Ao meu querido irmão Leonardo, com quem não pude viver muito a infância, porque quando ele queria brincar, eu já começava a me preparar para as coisas “mais sérias” da vida. Apesar disso, somos grandes amigos. Muito obrigado por ter sido o *designer* ao longo deste trabalho, desbravando os mistérios da modelagem gráfica 3D para produzir os objetos usados nos exemplos e no estudo de caso apresentados nesta dissertação.

Aos meus sogros, Fonseca e D. Lúcia, e aos meus cunhados, Vagner e Aline, por terem cuidado de minha querida Luciana nos dias em que ela não estava bem e que eu precisava me ausentar. A presença de vocês sempre foi muito importante.

Aos meus tios, Gilberto e Luzia, que sempre me acolheram com muito carinho em sua casa, nos primeiros meses em que iniciei minha trajetória em São Carlos.

À minha querida avó e madrinha, D. Matilde, por me acompanhar, todas as semanas, do portão da casa de meus pais até a esquina do quarteirão, para me desejar uma boa viagem até São Carlos e dar-me a sua bênção.

Aos professores de graduação Roberta Spolon Ulson e Marcos Antônio Cavenaghi, que forneceram as cartas de recomendação para o processo de seleção do Mestrado, confiando no meu comprometimento para a realização deste empreendimento.

À querida e eterna amiga Ariane Scarelli, que esteve em contato com esses professores para levar o meu pedido em relação às cartas de recomendação. Só mesmo você poderia ter encaminhado pessoalmente um pedido tão importante.

Aos professores Sérgio Roberto Pereira da Silva, Simone Diniz Junqueira Barbosa e Cecília Kremer Vieira da Cunha, que sempre responderam com muito atenção aos meus e-mails, enviando-me os materiais que contribuíram consideravelmente para a minha pesquisa e colocando-se sempre à disposição para o auxílio necessário.

A todos os funcionários do Departamento de Computação da UFSCar, em especial a Maria Cristina Carreira Trevelin e Mirian Cristina Thomé, que sempre me atenderam com muita atenção e nunca deixaram de orientar nas questões administrativas e burocráticas.

A todos os amigos que me acompanharam desde os primeiros dias do Mestrado, especialmente a Val Fontanette, amiga com quem sempre tive muita afinidade e que aprendi a admirar principalmente pelo empenho na busca de seus ideais.

Aos amigos de república, Eduardo Santana de Almeida, a quem dedico muita admiração, por ter se destacado como um pesquisador de alta qualidade, Edson da Silva Guimarães, Vinícius Cardoso Garcia, Cláudio Haruo Yamamoto, Alessandro Rodrigues, Raphael Marcílio de Souza Neto e Darley Rosa Peres.

A José da Silva, homem simples e íntegro, admirável pela dedicação ao trabalho e pela força inabalável na construção de seus sonhos. Muito obrigado por ter dedicado seu pouco tempo de descanso na realização do estudo de caso apresentado neste trabalho. Conforme prometi, divulgo aqui a locadora de vídeos Star Vídeo, na cidade de Guararapes, interior de São Paulo, fruto de seus esforços para a realização de um sonho.

Finalmente, não poderia deixar de agradecer à companhia do querido Toy, nosso "primogênito" (meu e da Luciana), o cãozinho *poodle* que todos os dias se deitava à porta de meu escritório ou num cantinho próximo à mesa do computador, enquanto eu escrevia linhas de código de programas, pesquisava ou redigia as linhas desta dissertação. E quando ele percebia que o computador estava sendo desligado, rapidamente corria em direção aos brinquedos, chamando-me para brincar. Agora teremos mais tempo!

A exemplo das palavras de Goethe, impressas na página que antecede estes agradecimentos, todos vocês foram encontros e assistências de incalculável grandeza, pequenos milagres enviados pela Providência para que este trabalho pudesse ser desenvolvido. A cada um de vocês, o meu mais sincero e profundo agradecimento, na forma de um abraço, um sorriso e uma oração. A cada um de vocês, o meu coração.

RESUMO

Ao longo dos anos a indústria de software vem aumentando as funcionalidades das aplicações, com o intuito de atender a um número maior de usuários com diferentes necessidades e diversos perfis. Essa tentativa gera inúmeros problemas, como a sobrecarga de funções, dificultando a usabilidade do software e oferecendo opções que podem nunca ser utilizadas. Por outro lado, o usuário pode precisar de funções que não foram previstas pelo projetista da aplicação ou que precisam ser implementadas devido a novos requisitos. Nesse caso, usuários finais podem ser beneficiados caso tenham condições de estender as funcionalidades de um software ou configurá-lo segundo as suas necessidades. Por isso, as pesquisas na área de programação por usuários finais (EUP) têm a finalidade de propor soluções para tornar a programação de computadores uma tarefa mais fácil e acessível, permitindo que usuários finais estendam suas aplicações. As técnicas de EUP, no entanto, ainda não têm sido exploradas em ferramentas CASE (*Computer-Aided Software Engineering*).

Este trabalho apresenta o GaCIV (Gabaritos Configuráveis para elaboração de Interfaces com realidade Virtual), uma ferramenta CASE que apóia a construção de interfaces com Realidade Virtual para aplicações de diferentes domínios. A nova versão da ferramenta foi adaptada para suportar a programação por usuários finais (EUP), ou seja, permitir que o projeto de interfaces seja realizado com a participação direta do usuário final. Assim, o usuário pode ser beneficiado em dois principais aspectos: a) tem a chance de realizar uma avaliação contínua da usabilidade, ao longo de toda a criação das interfaces; b) como ele mesmo organiza as opções de menu através da distribuição dos objetos no ambiente virtual, é mais provável que a interação com a aplicação seja facilitada.

Conforme será apresentado ao longo deste trabalho, a ferramenta GaCIV traz importantes contribuições para: a) a interação humano-computador, principalmente através do uso da Realidade Virtual; b) para a Engenharia de Software, ao oferecer a possibilidade de se realizar a reengenharia de interfaces de sistemas em funcionamento (legados ou não) e; c) para a programação por usuários finais, uma área de pesquisa em expansão, ao oferecer suporte para o projeto de interfaces com a participação do usuário final.

ABSTRACT

Over the years, the software industry has steadily increased software application functionalities with the idea of meeting the requirements of a large number of users with different needs and profiles. This attempt has generated innumerable problems, such as function overloads, complicating software usability and offering features that may never be used. On the other hand, the user may need functions that have not been foreseen by the applications designer or that must be implemented in response to new requirements. In that case, the end user will benefit if he can extend the functionalities of a software program or configure it according to his needs. For this reason, research in the area of EUP focuses on proposing solutions to render computer programming an easier and more accessible task, enabling the end user to extend his software's applications. EUP techniques, however, have so far not been exploited in CASE (Computer-Aided Software Engineering) tools.

This project discusses the new version of the GaCIV (Configurable Templates for the Development of Virtual Reality Interfaces), a tool for building interfaces with Virtual Reality for various domain applications. The tool's new version was adapted to aid end-user programming (EUP), i.e., to allow the design of interfaces to be carried out with the direct participation of the end user. Thus, the user benefits from two standpoints: a) he has the chance to make a continuous evaluation of the usability throughout the entire creation of the interfaces; b) because he himself organizes the menu options by distributing the objects in the virtual environment, interaction with the application is more likely to be facilitated.

As will be discussed throughout this project, the GaCIV tool makes important contributions to: a) human-computer interaction, mainly through the use of Virtual Reality; b) to Software Engineering, by offering the possibility of performing interface reengineering of working systems (legated and non-legated); and c) to end-user programming, an expanding research area, by offering support for the construction of interfaces with end user participation.

Sumário

Lista de Figuras

Lista de Tabelas

1. Introdução

O objetivo deste trabalho é promover o envolvimento do usuário final no projeto de interfaces com Realidade Virtual (RV), através de uma ferramenta CASE (*Computer-Aided Software Engineering*) apoiada por técnicas de programação por usuários finais (EUP – *End-User Programming*). Dessa forma, pretende-se obter uma integração efetiva do usuário final na construção de interfaces para suas aplicações e, como conseqüência, garantir uma maior qualidade na usabilidade dessas aplicações. Aliás, a avaliação da usabilidade ao longo de todo o processo de desenvolvimento de softwares é uma tendência destacada por Madsen (1999). No caso da proposta deste trabalho, o usuário final poderá avaliar a usabilidade da interface à medida em que ele próprio a desenvolve, de acordo com as suas necessidades e o seu conhecimento quanto ao domínio de aplicação em que está inserido. Uma outra atividade que também passa a ser beneficiada é a de levantamento de requisitos. Aliás, essa é uma das fases mais importantes no desenvolvimento de softwares [Brooks'1987 *apud* Drake *et al*'1997], mas a grande parte das falhas encontradas durante os testes ou a operação desses softwares é resultado de um entendimento incompleto ou da interpretação errada dos requisitos [Drake *et al*'1997]. Drake *et al* (1997) ainda ressaltam que esse é um problema conseqüente da falta de conhecimento do domínio em questão. Essa afirmação reforça a importância de se envolver ativamente o usuário final no processo de desenvolvimento, pois é ele quem detém o conhecimento do domínio da aplicação e é, por isso mesmo, a fonte de aquisição dos requisitos para o sistema. Assim, ao se permitir que o projeto de interfaces seja um processo iterativo com a participação do usuário final, também é oferecida a oportunidade de se realizar uma verificação contínua quanto ao atendimento desses requisitos. A qualidade do software tende a estar mais em conformidade com as expectativas do usuário final, porque a avaliação dos resultados parciais de um projeto pode resultar em mudanças para a melhoria desse projeto, e assim sucessivamente.

Para validar o trabalho, uma nova versão da ferramenta CASE GaCIV [Silva'1999] é apresentada e os aspectos que a caracterizam como uma aplicação para EUP são discutidos. O objetivo central dessa ferramenta é apoiar o projeto de interfaces com RV, ocultando as dificuldades inerentes ao uso dessa tecnologia para facilitar o trabalho do projetista de interfaces e, agora, tornar possível o envolvimento efetivo do usuário final. Por

isto, no contexto deste trabalho, entende-se "programação" como o desenvolvimento de interfaces com RV para aplicações de diferentes domínios através da manipulação e configuração de objetos tridimensionais, em ciclos iterativos que permitem o refinamento gradativo e contínuo dessas interfaces. Embora este conceito de programação não envolva o uso de uma linguagem de programação convencional e a codificação de um programa, ele está de acordo com um dos sentidos empregados por Cypher (1993), ou seja, a configuração de uma aplicação por meio da seleção de valores e escolha de opções em um quadro de parâmetros.

Apesar da importância da nova versão da ferramenta como meio de validação da proposta deste trabalho, é importante ressaltar que os conceitos que a sustentam são, de fato, os mais relevantes. Em outras palavras, esses conceitos, tais como o de gabaritos configuráveis e o uso da RV apoiados por técnicas de EUP para possibilitar o projeto de interfaces por usuários finais, são independentes de uma ferramenta específica implementada em uma determinada linguagem de programação. O objetivo maior é contribuir com um estudo que analisa a viabilidade e algumas das vantagens do envolvimento do usuário final no processo de desenvolvimento de software e como as técnicas da EUP e os recursos de RV podem auxiliar esse envolvimento, promovendo inclusive melhorias na qualidade de todo o processo e do produto de software.

Entre os diversos trabalhos já realizados na área de EUP, como os destacados em [Cypher'1993] e [Lieberman'2001], é possível perceber que as diferentes e importantes contribuições, especificamente em relação à técnica de programação por demonstração (PBD – *Programming by Demonstration*), não exploram a extensão da potencialidade de ferramentas CASE para o usuário final. É certo que as técnicas de configuração de parâmetros, gravação de macros e linguagens de *script* têm sido muito mais exploradas comercialmente, como no pacote de aplicações Microsoft® Office 2000/2002, mas, ainda assim, essas aplicações não têm englobado ferramentas CASE. Portanto, destaca-se aqui uma das motivações para a realização deste trabalho: oferecer uma pesquisa que explora as técnicas da EUP integradas a ferramentas CASE. Embora seja explorada somente a técnica de configuração de parâmetros neste trabalho, espera-se abrir o caminho para que outras técnicas também sejam exploradas em trabalhos futuros, considerando sempre o envolvimento do usuário final na equipe de desenvolvimento de sistemas.

A partir do que foi comentado até esse ponto, é possível que a seguinte questão tenha sido instigada: o que se propõe neste trabalho é que o usuário final execute as atividades pertinentes a engenheiros de software e projetistas de interface? A essa pergunta, cabe a

seguinte resposta: não. O que se deseja efetivamente é que o usuário final tenha um papel participativo, o mais atuante possível, em todas as etapas do processo de desenvolvimento de software em que essa atuação seja cabível. Mesmo porque o usuário final pode oferecer uma certa disponibilidade para colaborar no desenvolvimento de sistemas, mas dificilmente terá a disposição ou o tempo necessário para executar atividades estranhas[1] àquelas que são as principais em sua área de atuação profissional. É por essa razão que neste trabalho também se destaca a RV, um fator que pode ser um diferencial na aproximação do usuário final no projeto de interfaces. É possível notar, portanto, que três atores[2] se destacam no desenvolvimento de interfaces com RV usando a ferramenta GaCIV: 1) o usuário final, de quem se deseja obter uma participação ativa no projeto de interfaces; 2) o projetista de interfaces, que detém conhecimentos e habilidades referentes ao projeto de interfaces e; 3) o engenheiro de software, cuja tarefa principal, no contexto deste trabalho, é desenvolver os programas aplicativos do sistema. Também faz parte da proposta da ferramenta GaCIV apoiar a integração desses atores e a comunicação entre eles, ou seja: o projetista de interfaces pode desempenhar o papel de orientador e facilitador da tarefa do usuário final no desenvolvimento da interface, enquanto o engenheiro de software disponibiliza os aplicativos do sistema, isto é, os programas com as funcionalidades do sistema. Um quarto ator – o *designer* gráfico – também pode integrar a equipe de desenvolvimento de interfaces com RV na função de modelador de objetos tridimensionais, com o uso de softwares específicos para esta tarefa. Embora este ator seja citado algumas vezes no texto, é interessante esclarecer que sua função pode ser executada pelo próprio projetista de interfaces.

O desejo de explorar o uso dos recursos de RV integrados à EUP também se destaca, por duas razões, como motivação para este trabalho. A primeira razão é a de que a RV não tem sido explorada nas ferramentas que se inserem no contexto da EUP, com exceção de ToonTalk® [Kahn'2001, ToonTalk'2003] e Alice [Alice'2003, Conway'1997, Conway *et al*'2000], as duas únicas encontradas com essa característica durante as pesquisas realizadas ao longo deste trabalho. A segunda razão está fundamentada no objetivo de se analisar as possíveis vantagens que a RV pode proporcionar ao desenvolvimento de interfaces, principalmente em relação à usabilidade.

Apesar de a RV estar sendo constatada como uma promissora plataforma para a construção de sistemas em diferentes domínios, não se tem visto um amplo desenvolvimento

[1] Neste texto, a expressão "atividades estranhas" refere-se às atividades específicas de engenheiros de software, tais como a modelagem e a programação de sistemas.
[2] Neste texto, a palavra "atores" refere-se às pessoas envolvidas no desenvolvimento de interfaces com RV.

e uso de aplicações de RV na prática [Tanriverdi *et al*'2001]. Essa pequena proliferação pode ser atribuída parcialmente aos desafios para se construir sistemas com RV [Astheimer *et al*'1999, Brooks'1999]. Com o objetivo de propor uma solução para esse problema, a estrutura da ferramenta GaCIV permite ocultar o desafio inerente ao uso dos recursos de RV, sendo que toda a complexidade de processamento é executada de maneira transparente. Essa é, inclusive, uma característica de extrema importância no contexto deste trabalho, já que a meta é oferecer as condições necessárias para que o usuário final se sinta o mais confortável possível no uso desses recursos.

Finalmente, espera-se que o objetivo de prover um ambiente para possibilitar a integração efetiva do usuário final no projeto de interfaces com RV, usando uma abordagem de prototipação e considerando uma avaliação contínua da usabilidade, seja um passo em direção à integração efetiva do usuário final nas diferentes etapas do processo de desenvolvimento. Para isso, a EUP deve exercer um papel fundamental.

É interessante ressaltar que ao longo deste trabalho, quando a palavra "usuário" for usada, deve ser entendida como sinônimo de "usuário final". Além disso, os "sistemas com RV" devem ser compreendidos, aqui, como todo sistema cuja interface utiliza os recursos de RV. Assim, a ferramenta GaCIV pode ser entendida como uma CASE para o projeto de sistemas com RV, ou seja, ela fornece meios para a construção de interfaces com RV a partir das quais o usuário pode interagir com um determinado sistema.

Este trabalho está organizado da seguinte maneira: no capítulo 2 são apresentados os principais conceitos de EUP e de RV, a proximidade entre essas duas áreas de pesquisa que se destacam neste trabalho e uma análise sobre o uso da EUP como apoio à extensão das potencialidades de ferramentas CASE. No capítulo 3 é discutido o desenvolvimento da proposta deste trabalho, através da apresentação da ferramenta CASE GaCIV, seus conceitos, estrutura e as características que a inserem no contexto da EUP. No capítulo 4 são apresentados um exemplo de aplicação da ferramenta e também um estudo de caso, em que é discutida a participação de um usuário final no projeto de interfaces com RV para uma aplicação no domínio de locadoras de vídeo (o questionário aplicado ao usuário durante o experimento é apresentado no Apêndice A). Finalmente, no capítulo 5, são apresentadas as conclusões finais do trabalho e apresentadas algumas propostas para trabalhos futuros.

2. EUP e RV: dois pilares de apoio ao envolvimento do usuário final no projeto de interfaces

2.1. Considerações iniciais

Neste capítulo são apresentados alguns conceitos das áreas de pesquisa utilizadas para a realização deste trabalho. Destacam-se a programação por usuários finais (*End-User Programming* ou EUP) e a Realidade Virtual (RV).

Na seção 2.2 são apresentados os princípios de EUP, como propósitos, técnicas e a quem se destina. Na seção 2.3 são apresentados os principais conceitos de RV, como uma introdução necessária à compreensão das possíveis vantagens dessa tecnologia no contexto de EUP. A proximidade entre essas duas áreas de pesquisa é discutida na subseção 2.3.1. Na seção 2.4, é apresentada uma análise sobre o uso da EUP como apoio à extensão das potencialidades de ferramentas CASE, no intuito de envolver o usuário final em atividades de desenvolvimento de sistemas. Na seção 2.5, são apresentadas, brevemente, as principais características da API (*Application Programming Interface*) Java 3D, uma extensão da linguagem Java que oferece os recursos para a programação de aplicações com RV. Também são discutidos alguns aspectos que influenciaram o desenvolvimento da nova versão da ferramenta GaCIV. Finalmente, na seção 2.6, são apresentadas as considerações finais referentes aos tópicos abordados neste capítulo.

2.2. Princípios da programação por usuários finais (EUP)

Ao se referir à motivação que tem sido base para a evolução da programação por demonstração (*Programming by Demonstration*), uma das técnicas de EUP, Cypher (1993) a descreve da seguinte maneira:

> *"(...) if a user knows how to perform a task on the computer, that should be sufficient to create a program to perform the task. It should not be necessary to learn a programming language like C or BASIC. Instead, the user should be able to instruct the computer to*

"Watch what I do", and the computer should create the program that corresponds to the user's actions."

Nesta afirmação, é possível encontrar uma referência ao cerne da EUP: permitir que um usuário final, sem conhecimentos específicos e/ou inclinação para aprender as técnicas e ferramentas de desenvolvimento de software e as habilidades de um programador, possa criar seus próprios programas [Goodell'2002]. Em outras palavras, a EUP é a área de pesquisa que se dispõe a permitir que os próprios usuários finais configurem ou estendam as funcionalidades das aplicações, através de mecanismos de extensão e uma interação intuitiva com o computador [Barbosa'1999, Cunha'2001, da Silva'2001]. Para isso, o usuário não precisa ser um profissional experiente em informática, nem ter as habilidades de um programador ou conhecimentos específicos em linguagens de programação.

No contexto da EUP, os usuários finais são pessoas que [Cypher'1993, Baron *et al*'2001]:

- usam aplicações no computador como parte de sua vida diária ou no trabalho, mas não são programadores ou especialistas em computação;
- podem ser usuários experientes com computadores, mas não têm experiência com linguagens de programação convencionais;
- programam ocasionalmente, mas essa não é a parte mais importante da atividade que exercem.

Para possibilitar que os usuários finais configurem ou estendam aplicações, algumas técnicas vêm sendo exploradas e aperfeiçoadas com o intuito de facilitar essas atividades. Na subseção a seguir, essas técnicas são descritas sucintamente, dando-se maior atenção aos paradigmas de programação apresentados por da Silva (2001), a fim de oferecer informações suficientes para uma discussão posterior sobre a adaptação da ferramenta GaCIV ao contexto da EUP. Uma discussão mais detalhada de cada técnica pode ser encontrada em [Barbosa'1999, Cunha'2001, Cypher'1993, da Silva'2001].

2.2.1. Técnicas de EUP e a classificação de paradigmas

Cypher (1993) divide as técnicas para programação por usuários finais em quatro categorias: **configuração de parâmetros** (*preferences*), **gravação de macros**, **linguagens de**

script (*scripting languages*) e **programação por demonstração** (*Programming by Demonstration* ou PBD). As três primeiras são as mais utilizadas comercialmente.

Um exemplo da técnica de configuração de parâmetros é a caixa de diálogo "Opções" do aplicativo Microsoft® Word 2000/2002, que pode ser aberta através da opção de menu "Ferramentas". O usuário pode configurar alguns comportamentos ou a aparência da aplicação ao escolher as alternativas que considerar mais convenientes entre as diferentes opções de combinações possíveis de parâmetros.

A técnica de gravação de macros, também disponível em diversas aplicações comerciais, inclusive nas aplicações do pacote Microsoft® Office 2000/2002, permite que os usuários ativem um gravador de instruções que registrará a seqüência de ações executadas no computador, e que pode ser reproduzida a qualquer momento. A maior vantagem dessa técnica é a familiaridade do usuário com a linguagem usada na especificação da extensão da aplicação, que é a própria linguagem de interface da aplicação [da Silva'2001]. Por outro lado, esta técnica tem a desvantagem de ser muito literal. Isto acontece porque as aplicações que a disponibilizam gravam a seqüência de ações literalmente, ou seja, replicam exatamente os eventos acionados pelo usuário e mantêm constantes os valores atribuídos às variáveis de contexto da macro [Barbosa'1999, Cunha'2001, da Silva'2001].

As linguagens de *script*, por sua vez, são linguagens de programação textuais disponibilizadas ao usuário. Um de seus problemas é o esforço do usuário para o aprendizado destas linguagens; por mais fácil que elas sejam, ele terá que aprender sua sintaxe e suas convenções para poder utilizá-las. Além disso, nas aplicações que oferecem linguagens de *script* ou de programação para os usuários finais, quando o usuário deseja fazer uma extensão, geralmente ocorre uma mudança drástica na interface da aplicação porque, na maioria das vezes, é oferecido a ele um editor de programa, textual, que pode desorientá-lo [Barbosa'1999].

A programação por demonstração, também conhecida como programação por exemplo, é o resultado de uma melhor elaboração das idéias de gravação de macros. A maior diferença é que, à medida que o usuário instrui a aplicação através de demonstrações (uma seqüência de operações), são criados programas generalizados, que podem conter inclusive iterações e condicionais. Portanto, esta técnica propõe uma solução ao maior problema da técnica de gravação de macros: o de ser muito literal. O grande desafio para se projetar aplicações com a técnica de PBD é embutir nelas um mecanismo de inferência, que seja capaz de capturar as intenções implícitas nas ações do usuário. Daí surge um dos problemas dessa técnica, que atribui ao usuário a função de exemplificar sua intenção através de diversos

exemplos, numa tentativa de esgotar todas as situações que possam ocorrer. Esses exemplos ampliam a possibilidade dos mecanismos de inferência produzir bons resultados, ou seja, extensões da aplicação que se assemelham àquelas pretendidas pelo usuário [Barbosa'1999].

Essas técnicas são agrupadas por da Silva (2001) em três diferentes paradigmas de EUP, conforme o tipo de linguagem empregada pelos mecanismos que os implementam:

- **Paradigma de Programação Paramétrica**, que compreende a técnica de configuração de parâmetros;
- **Paradigma de Programação Imitativa**, que compreende as técnicas de gravação de macros e programação por demonstração; e
- **Paradigma de Programação Descritiva**, que compreende a técnica de linguagens de *script*.

Essa classificação é importante porque serve como referência aos esforços de adaptação da ferramenta GaCIV para a programação por usuários finais. Destaca-se o Paradigma Paramétrico, dentro do qual se enquadra a ferramenta GaCIV em sua atual versão. Nesse paradigma, devem ser disponibilizados para o usuário um conjunto básico de módulos e uma forma de configurar esses módulos. Dois níveis possíveis de atuação do usuário na criação de uma extensão à aplicação através deste paradigma podem ser oferecidos: no primeiro, o usuário estará limitado a selecionar alternativas e a mudar valores para os parâmetros disponibilizados na aplicação e, dessa forma, ele não estará verdadeiramente criando uma extensão à aplicação, mas somente configurando-a; no segundo nível, o usuário pode criar novos itens léxicos para ser apresentados na interface da aplicação, por meio da seleção, ativação e agrupamento de um conjunto de parâmetros e sua posterior associação a um nome [da Silva'2001]. Para exemplificar este nível, pode-se citar a criação de estilos no Microsoft® Word 2000/2002, em que é possível agrupar um conjunto de valores para parâmetros (como o alinhamento do texto e o estilo de fonte) que determinará o estilo de um parágrafo ou linha de texto [da Silva'2001].

Quanto aos mecanismos de implementação utilizados neste paradigma, da Silva (2001) ressalta os três mais adotados:

- A **configuração de preferências,** em que o usuário pode ativar ou desativar opções em um conjunto de parâmetros contextualizados, alterando o comportamento global da aplicação. A caixa de diálogo "Opções", no menu "Ferramentas" do Microsoft® Word 2000/2002, exemplifica este mecanismo.

- A **personalização da aparência** da interface, que possibilita ao usuário realizar mudanças nos elementos visuais, ou léxicos (barras de ferramentas e menus) presentes na interface da aplicação, através da alteração da composição e/ou posição desses elementos;
- O **uso de moldes**, que permite que o usuário agrupe um ou mais conjuntos de parâmetros, dê uma denominação ao agrupamento e o empregue para personalizar o leiaute de um documento (como na criação de estilos no Microsoft® Word 2000/2002). Este mecanismo está no limiar entre o paradigma de programação paramétrico e o imitativo, porque possibilita a criação de novos itens léxicos (nomes com novos significados) na aplicação, conforme os mecanismos da programação imitativa.

Duas observações importantes podem ser feitas em relação a esses mecanismos, de acordo com da Silva (2001):

a) a configuração de preferências apresenta dificuldades de usabilidade, porque os usuários normalmente não conseguem estimar o efeito final do conjunto de escolhas de parâmetros sobre o comportamento da aplicação. Isso é resultado do fato de os usuários não apresentarem um conhecimento suficiente do modelo de usabilidade da aplicação;

b) a personalização da aparência da interface pode apresentar um problema que se refere ao grau de liberdade permitido nas alterações locais, ou seja, algumas aplicações permitem que itens léxicos presentes na interface sejam alterados por outros itens quaisquer. No Microsoft® Word 2000/2002, por exemplo, os nomes e imagens dos itens de menu podem ser alterados. Alterações aleatórias podem causar a destruição do significado original atribuído a esses elementos da interface pelo projetista da aplicação. Isso significa que o próprio usuário pode se confundir ao interagir com a aplicação, já que pode ter atribuído nomes não significativos a elementos da interface.

Embora muitas questões estejam relacionadas também aos paradigmas imitativo e descritivo, elas não são discutidas aqui por não pertencerem ao escopo deste trabalho. No entanto, a fim de organizar um resumo dos paradigmas e as respectivas técnicas de EUP a partir da análise dos trabalhos de Barbosa (1999), Cunha (2001), Cypher (1993) e da Silva (2001), é apresentada a Tabela 2-1:

Tabela 2-1. Paradigmas de EUP

Paradigma	Técnica	Características	Vantagens	Desvantagens
Paramétrico	Configuração de parâmetros	- ativação/desativação de parâmetros predefinidos - mudança dos valores para os parâmetros disponíveis - seleção, ativação e agrupamento de um conjunto de parâmetros e sua associação a um nome (criação de estilos)	- personalização da interface da aplicação - criação de estilos (personalizações) que podem ser compartilhados entre usuários	- as configurações são restritas às situações previstas pelo projetista da aplicação - poucos parâmetros podem resultar em um baixo potencial de extensão - muitos parâmetros podem dificultar a previsão dos resultados
Imitativo	Gravação de macros	- armazenamento do histórico de uma seqüência de ações - reprodução da seqüência gravada	- familiaridade do usuário com a linguagem de extensão (interação com a linguagem de interface) - automatização de seqüências de ações repetitivas	- gravação literal da seqüência de ações - não possibilita a criação de condicionais - algumas implementações permitem iterações, mas não a especificação direta de iterações controladas
Imitativo	PBD	- elaboração da idéia de gravação de macros - criação de um programa generalizado a partir de um conjunto de exemplos de tarefas - uso de mecanismos de inferência (técnicas de Inteligência Artificial)	- familiaridade do usuário com a linguagem de extensão (interação com a linguagem de interface) - programas generalizados podem conter iterações e condicionais	- pode ser necessário demonstrar diversos exemplos para enumerar todas as situações possíveis - dificuldade no uso de exemplos negativos - o usuário pode ter que inferir que exemplos são necessários para demonstrar satisfatoriamente sua intenção
Descritivo	Linguagem de *script*	- uso de uma linguagem de programação para a extensão - uso de um ambiente de programação e um interpretador para a linguagem de extensão	- possibilidades amplas de extensão, por empregar uma linguagem de programação completa	- o usuário não tem a ajuda do software para criar uma extensão - necessidade de aprendizado da sintaxe da linguagem de programação usada - mudança da linguagem de interface para um editor de programas textual

2.3. Princípios de Realidade Virtual (RV)

Pimentel e Teixeira [Pimentel *et al*'1995] definem a Realidade Virtual como o uso de alta tecnologia para convencer o usuário de que ele está em outra realidade, sentindo-se dentro da informação e tocando-a com suas mãos. Essa sensação é possível graças a uma das

características da RV – a **imersão**, que se refere à sensação que o usuário pode ter de estar dentro do ambiente.

De acordo com o grau de imersão, os sistemas de RV podem ser classificados em: imersivos, não-imersivos e semi-imersivos [Vince'1995]. Os sistemas imersivos são caracterizados por ocultarem o mundo real, substituindo-o por imagens geradas por computador que reagem à posição e movimentos da cabeça do usuário [Vince'1995, Bowman *et al*'2001]. Isso é possível, por exemplo, através do uso de capacetes de visualização (*Head-Mounted Display* ou HMD). Nos sistemas não-imersivos, o mundo real não é ocultado e o usuário não perde a noção de seu ambiente natural, já que o mundo virtual é visualizado através de dispositivos de visualização como monitores de vídeo tradicionais [Vince'1995, Assis'2001]. Além disso, o usuário também precisa utilizar algum dispositivo de entrada para se movimentar no mundo virtual [Jacobson'1994 *apud* Machado'1997], geralmente *mouse* e/ou teclado. As principais vantagens dos sistemas não-imersivos que podem ser destacadas em relação aos sistemas imersivos são: facilidade de uso, menor custo e, conseqüentemente, a possibilidade de uso dos recursos de RV por um público maior de usuários [Assis'2001]. Finalmente, os sistemas semi-imersivos permitem ao usuário visualizar tanto o mundo físico real quanto o mundo virtual [Bowman *et al*'2001]. As imagens virtuais são sobrepostas à visão do mundo real, configurando a técnica conhecida como Realidade Aumentada (*Augmented Reality*) [Vince'1995, Assis'2001].

Outro fator essencial no contexto de RV é a computação gráfica 3D. Considerando-se que o estímulo da visão é uma das principais características da RV, é necessário que as imagens virtuais geradas estejam o mais próximo possível das imagens do mundo real [Machado'1997]. Por isso, a computação gráfica passa a ter um papel fundamental na criação das imagens em ambientes virtuais, constituindo uma importante base tecnológica para o desenvolvimento da RV.

Apesar dos potenciais benefícios que se pode obter com a RV, principalmente na elaboração de ambientes computacionais mais realísticos, ainda existem grandes desafios na construção desse tipo de aplicações [Astheimer *et al*'1999, Brooks'1999], particularmente em relação às interfaces 3D, bem mais complexas quando comparadas às interfaces gráficas convencionais [Tanriverdi *et al*'2001]. Para simplificar a complexidade do projeto de interfaces com RV, a ferramenta GaCIV propõe o uso de gabaritos configuráveis, ou seja, modelos semiprontos que facilitam e agilizam a construção dessas interfaces. Dessa forma, é necessário apenas que um *designer* modele e disponibilize os objetos 3D. A partir daí, a

ferramenta oferecerá todos os serviços necessários para que os ambientes sejam projetados facilmente, compondo os gabaritos e interfaces com RV.

Apresentadas as definições essenciais de RV, é possível perceber a proximidade dessa área de pesquisa com a EUP, conforme análise apresentada na próxima subseção.

2.3.1. A RV e a EUP: em direção aos mesmos objetivos

A interface com RV envolve um ambiente tridimensional altamente interativo, isto é, um cenário dinâmico armazenado em computador e exibido, em tempo real, através de técnicas de computação gráfica [Pinho'2000], com o qual o usuário pode interagir usando seus sentidos, particularmente os movimentos naturais tridimensionais do corpo.

O principal esforço da RV é representar o ambiente computacional através da transferência do conhecimento intuitivo que o usuário tem do mundo real para o mundo virtual, um fator muito interessante para tornar a comunicação humano-computador mais natural. Conseqüentemente, beneficia-se a usabilidade da aplicação, já que o usuário, ao encontrar um ambiente que reflete a realidade, pode se situar melhor durante as atividades de interação e navegação. Aliás, de acordo com Lakoff *et al* (1980) *apud* Barbosa (1999), Lakoff (1987) *apud* Barbosa (1999) e Repenning *et al* (2001), analogias são poderosos mecanismos cognitivos usados pelas pessoas para construir novos conhecimentos a partir de outros conhecimentos adquiridos e compreendidos.

Tais características das interfaces com RV podem ser relacionadas ao principal objetivo da EUP: tornar a programação uma atividade mais simples e acessível para o usuário, através de técnicas que o permitam comunicar-se de uma maneira mais natural com o computador. Portanto, pode-se afirmar que, tanto no contexto da RV quanto no de EUP, o objetivo é permitir que o usuário se sinta confortável no ambiente computacional, familiarizando-se mais facilmente com os meios de interação com o computador e a linguagem de comunicação com a máquina, de acordo com os princípios de IHC (Interação Humano-Computador).

Assim, é possível perceber a proximidade da RV e da EUP: ambas têm como objetivo facilitar a interação do usuário com o ambiente computacional, permitindo-o explorar as potencialidades do computador de forma mais proveitosa. Uma das contribuições deste trabalho é justamente o de prover um ambiente para a construção de interfaces com RV a partir do qual se torna possível o estudo da viabilidade do uso da RV no apoio à programação

por usuários finais. Essa contribuição se torna um pouco mais expressiva quando se nota que as pesquisas em torno da EUP não têm explorado o uso da RV, razão pela qual se pode afirmar que ainda há muito a se investigar em relação aos benefícios que se pode obter no uso da RV em conjunto com a EUP. Entre os diversos trabalhos compilados por Cypher (1993) e Lieberman (2001), somente um utiliza a RV: a ferramenta ToonTalk® [Kahn'2001, ToonTalk'2003], cujo objetivo é estimular o raciocínio lógico do público infantil de forma mais divertida, através da animação tridimensional, com elementos gráficos que são analogias às abstrações computacionais [Kahn'2001]. Embora não seja classificada explicitamente pelos autores como uma aplicação para EUP, a ferramenta Alice [Alice'2003, Conway'1997, Conway *et al*'2000] também pode ser inserida neste contexto, já que o objetivo também é permitir que usuários sem experiência em programação gráfica 3D possam criar programas com RV.

Mas é importante ressaltar que nem sempre a RV pode garantir um alto nível de usabilidade a uma aplicação [Balbino *et al*'2002]. Uma análise da ferramenta ToonTalk® mostrou que, apesar do ambiente virtual animado com analogias às abstrações computacionais, a programação não é uma atividade tão simples e a interação com a ferramenta não é muito intuitiva. Tais características parecem ser efeito do próprio propósito da ferramenta que, segundo Kahn (2001), é facilitar o ensino da programação, o que naturalmente exige maior esforço do usuário para adequar-se à utilização dos objetos análogos aos processos computacionais e estruturas de programação. Portanto, é válido afirmar que a colaboração da RV para um menor ou maior grau de usabilidade dependerá, também, do domínio da aplicação [Balbino *et al*'2002]. Embora ainda não se tenha feito uma análise formal, junto a um grupo de usuários, das questões de usabilidade da ferramenta GaCIV e das interfaces com ela geradas, é possível afirmar que a organização das funções de um sistema em um ambiente virtual pode trazer consideráveis benefícios, já que o usuário, ao interagir com esse ambiente, pode encontrar um modelo do ambiente real com o qual ele está habituado. Dessa forma, se para encontrar a ficha de um cliente de um estabelecimento comercial o usuário se dirige até um armário de aço, agirá da mesma forma no ambiente virtual que reflete a realidade do estabelecimento e, ao clicar sobre um objeto análogo a um armário de aço, a aplicação para cadastrar ou consultar as fichas dos clientes será executada.

2.4. Uma nova perspectiva para a EUP

São muitos os trabalhos que trouxeram e continuam trazendo importantes contribuições para a evolução da programação por usuários finais. Conforme já foi comentado, as técnicas de configuração de parâmetros, gravação de macros e linguagens de *script* são as mais usadas comercialmente, podendo ser encontradas, por exemplo, nos aplicativos do pacote Microsoft® Office 2000/2002. Já a técnica de programação por demonstração ainda tem sido explorada mais intensamente em trabalhos acadêmicos, como se pode notar na compilação de alguns desses trabalhos elaborada por Cypher (1993) e Lieberman (2001). Os aplicativos que se destacam comercialmente com o uso da programação por demonstração, apresentados na subseção 3.2, "Ferramentas para EUP e suas principais aplicações", do próximo capítulo, são ambientes direcionados especialmente para o público infantil, visando ao estímulo do raciocínio através da construção de jogos e simulações em que condicionais e iterações podem ser especificadas através de demonstrações, sem que haja a necessidade de intervenção de uma linguagem de programação convencional. Na maior parte do tempo, durante a programação, a interação com o ambiente é realizada através da manipulação direta de elementos visuais disponíveis na interface.

Tais observações permitem afirmar que as possíveis facilidades que a EUP pode oferecer nos mais diversos contextos da informática ainda têm sido restritas a aplicações de domínio mais comum, como para a edição de textos, elaboração de planilhas eletrônicas, estimulação do raciocínio lógico através da programação de jogos, entre outras.

No entanto, a EUP pode ser pensada como um conjunto potencial de técnicas através do qual é possível prover meios para o envolvimento do usuário no processo de desenvolvimento de softwares. Não é defendida aqui a possibilidade de se delegar ao usuário as atividades pertinentes a um engenheiro de software, mas a de permitir que ele seja um membro ativo na equipe de desenvolvimento de sistemas, principalmente na etapa de levantamento de requisitos e na avaliação da usabilidade do sistema. De acordo com Madsen (1999), essa é a uma tendência cada vez mais emergente: o trabalho de aplicação e avaliação de usabilidade tem se tornado uma atividade cooperativa, envolvendo grupos de usabilidade, usuários e a equipe de desenvolvimento. Com isso, usuários têm participado ativamente no processo de desenvolvimento e contribuído para o projeto, principalmente nos aspectos de usabilidade.

Neste trabalho, a EUP estende a potencialidade da ferramenta GaCIV, apresentada no próximo capítulo, principalmente ao prover um ambiente que oferece condições para o

envolvimento do usuário numa importante etapa do processo de desenvolvimento de software: o projeto de interfaces. No contexto deste trabalho, o envolvimento do usuário se torna ainda mais interessante ao se considerar que a proposta é a de se projetar interfaces com RV. Isso porque os projetistas normalmente se preocupam tanto com dificuldades técnicas normalmente encontradas na construção de interfaces 3D, que acabam dedicando atenção insuficiente para as questões que envolvem a interação do usuário com o ambiente virtual [Kaur *et al*'1998]. Nesse caso, a participação direta do usuário no projeto de interfaces 3D traz uma importante contribuição para se obter um grau mais adequado de usabilidade, já que a construção e a organização do ambiente virtual passam a ser realizadas pelo próprio usuário, cabendo a intervenção do projetista quando necessário.

Conforme afirmação na subseção 2.3.1, "A RV e a EUP: em direção aos mesmos objetivos", a RV envolve principalmente o esforço de se representar o ambiente computacional da maneira mais semelhante possível ao mundo real, promovendo, assim, benefícios à comunicação humano-computador. Conseqüentemente, o que se deseja é que a RV contribua de forma significativa no grau de interatividade embutida nos sistemas computacionais. Isso exige não só esforços no aperfeiçoamento das técnicas de RV, mas também, e mais importante, uma maior compreensão dos requisitos para a especificação da interação humano-computador no projeto desses sistemas. Certamente, o envolvimento do usuário no projeto resultará no estudo e na compreensão mais eficientes dos requisitos, ou na percepção de eventuais falhas durante a etapa de estudo de requisitos, através da análise contínua da usabilidade pelo próprio usuário.

De acordo com as idéias expostas e com a proposta principal deste trabalho, é possível perceber que a EUP também pode trazer significativas contribuições quando suportada por ferramentas CASE, no intuito de envolver o usuário nas diferentes etapas do processo de desenvolvimento de software.

2.5. A linguagem de programação Java

Java é uma linguagem orientada a objetos de propósito geral e projetada para ser simples. Uma característica dessa linguagem que tem merecido muito destaque é a sua independência de plataforma. Cabe salientar, no entanto, que embora seja mais fácil escrever programas portáveis em Java do que em outras linguagens de programação, há diferenças

entre compiladores, interpretadores e principalmente entre computadores, que podem dificultar a portabilidade [Deitel *et al*'2002].

A versão da ferramenta GaCIV apresentada neste trabalho foi desenvolvida com a linguagem de programação Java. Uma das razões que motivou a escolha da linguagem foi a possibilidade de se criar uma ferramenta independente de plataforma. Nesta primeira versão, no entanto, não houve uma preocupação necessária para que se pudesse garantir a sua portabilidade, conforme as orientações citadas anteriormente. Isso se justifica pelo fato de que o propósito desta primeira versão foi adaptar a ferramenta ao contexto da EUP e, por isso, o foco do esforço não se deu em detalhes tecnológicos.

Além disso, a escolha da linguagem de programação Java também foi motivada porque ela dispõe de uma extensão formada por uma hierarquia de classes que permitiu disponibilizar os recursos de RV na ferramenta GaCIV.

De qualquer forma, é importante destacar que a proposta do ambiente GaCIV é independente de linguagens de programação específicas. Quaisquer linguagens de programação podem validar a proposta, desde que ofereçam os recursos necessários para a implementação das características de RV.

2.5.1. A API Java 3D

A API (*Application Programming Interface*) Java 3D é uma interface criada para o desenvolvimento de aplicações gráficas tridimensionais em Java [Bicho *et al*'2002], ou seja, aplicações que exibam e interajam com gráficos tridimensionais [Java 3D'2002]. Java 3D também suporta o conceito "*Write Once, Run Anywhere*"[3] [Cheung'2003], permitindo que programadores de aplicações gráficas tridimensionais também explorem a característica de portabilidade de Java.

A característica de orientação a objetos, por sua vez, oferece uma abordagem de alto nível à programação e possibilita que o programador se dedique mais à criação da aplicação do que aos problemas de mais baixo nível pertinentes à programação 3D [Bicho *et al*'2002]. Nesse sentido, a programação se torna mais simples e acessível a programadores que não estejam familiarizados com os detalhes necessários à implementação de operações 3D. Em outras palavras, a API Java 3D deixa transparente ao programador os detalhes de

[3] Expressão que se refere à portabilidade dos programas escritos na linguagem de programação Java: "Escreva uma vez, execute em qualquer lugar".

baixo nível, enquanto ele se concentra na estruturação das cenas tridimensionais. Tarefas que exigiriam um grande esforço de programação, como a renderização[4], são realizadas automaticamente, através da chamada a funções de bibliotecas gráficas de mais baixo nível, como a OpenGL e a Direct 3D [Bicho *et al*'2002, Java 3D'2002]. OpenGL, por exemplo, é uma interface de software para se trabalhar com gráficos a nível de hardware[5] e que não tem comandos de alto-nível para descrever objetos 3D complexos [Cheung'2003]. Ao invés disso, ela disponibiliza um conjunto de primitivas geométricas – pontos, linhas e polígonos – que podem construir modelos mais complexos quando combinadas entre si [Cheung'2003].

As classes Java 3D também podem ser usadas no desenvolvimento de aplicativos ou *applets* 3D. Em relação ao desenvolvimento de *applets*, pode-se destacar, de acordo com Bicho *et al* (2002), que essa possibilidade faz da API Java 3D uma solução fortemente viável para a disponibilização de aplicações gráficas 3D na Web. Isso torna possível a adaptação da ferramenta GaCIV, numa próxima versão, para utilização na Web e para a construção de interfaces que também possam ser exibidas por *browsers*.

Mas a API Java 3D também conta com algumas desvantagens. Dentre elas, pode-se enumerar duas: a primeira se refere ao desempenho das aplicações, que pode ser baixo. Aliás, esse é um efeito produzido por uma característica da própria linguagem Java: ela é interpretada, e os interpretadores rodam lentamente comparados aos códigos de máquina integralmente compilados [Deitel *et al*'2002]. No entanto, a ferramenta GaCIV tem mostrado um desempenho suficientemente razoável em suas funções gráficas, tendo superado as expectativas. A segunda desvantagem está relacionada a erros[6] que freqüentemente acontecem nos *drivers* de DirectX e de OpenGL. Isso pode exigir um esforço do programador para solucionar os problemas que surgem e prejudicam a aplicação.

Apesar da facilidade de uso da API Java 3D destacadas anteriormente, pode-se afirmar que, assim como qualquer outra tecnologia com as que não se está familiarizado, a programação 3D com Java não deixa de ser uma tarefa árdua. De qualquer forma, deve-se enfatizar que o encapsulamento de operações primitivas certamente beneficia o programador, porque lida com instruções de alto nível.

---

[4] Renderização é a obtenção da imagem a partir do modelo. É neste processo que se adiciona, por exemplo, sombreamento, cores e iluminação à cena [Bicho *et al*'2002].
[5] *graphics hardware*.
[6] *bugs*

2.6. Considerações finais

Através de linguagens e técnicas de níveis de facilidade e flexibilidade diferentes, a EUP propõe a integração do usuário no mundo da programação de computadores, seja em tarefas simples como a configuração de uma aplicação, seja proporcionando-lhe condições de estender as funcionalidades de aplicações e, portanto, podendo ele mesmo propor soluções às suas novas necessidades. O fato de os usuários serem os próprios fornecedores dos requisitos durante o ciclo de desenvolvimento de um sistema reforça a proposta da EUP, já que eles normalmente visualizam mais claramente o melhor meio para a execução de suas atividades. São eles, afinal, que as executam diariamente. Portanto, por conhecerem bem os requisitos do domínio em que estão inseridos, podem expressar mais precisamente as suas necessidades na criação de um programa. Isso justifica os esforços para se oferecer uma ferramenta CASE em que o usuário pode participar ativamente do projeto de interfaces com RV, realizando, inclusive, uma contínua avaliação da usabilidade.

Destaca-se aqui que a ferramenta GaCIV é um protótipo, em constante evolução. Esse protótipo permite a validação da proposta deste trabalho, que é explorar as potencialidades de EUP no apoio ao desenvolvimento de sistemas com RV através de ferramentas CASE, promovendo o envolvimento efetivo do usuário nesse processo.

No próximo capítulo, é apresentada a nova versão da ferramenta adaptada ao contexto da EUP.

3. A ferramenta CASE GaCIV

3.1. Considerações Iniciais

Ferramentas CASE (*Computer-Aided Software Engineering* – engenharia de software apoiada por computador) apóiam profissionais de engenharia de software em toda atividade associada com o processo de software [Pressman'2002], oferecendo o apoio automatizado a métodos de desenvolvimento [Mian *et al*'2001]. A ferramenta GaCIV é uma CASE de apoio ao projeto e desenvolvimento de interfaces com RV para aplicações de diferentes domínios. A nova versão da ferramenta foi adaptada para suportar a programação por usuários finais (EUP), ou seja, permitir que o projeto de interfaces seja realizado com a participação direta do usuário. Suas características e potencialidades são apresentadas neste capítulo.

Na seção 3.2 são apresentadas algumas ferramentas que se destacam como aplicações para a EUP. Na seção 3.3, é apresentada uma introdução aos princípios da ferramenta GaCIV para, em seguida, ser apresentada a sua estrutura, na subseção 3.3.1. Na subseção 3.3.2, é apresentada uma análise sobre as características que classificam a nova versão da ferramenta GaCIV no contexto da EUP, conforme o Paradigma Paramétrico, proposto por da Silva (2001). Na subseção 3.3.3, é apresentada uma discussão sobre a possibilidade de se estender as potencialidades da ferramenta GaCIV para que ela possa suportar mecanismos que a classifiquem no Paradigma Imitativo [da Silva'2001]. Finalmente, na seção 3.4, são apresentadas as considerações finais deste capítulo.

3.2. Ferramentas para EUP e suas principais aplicações

Nesta seção são apresentadas, sucintamente, três ferramentas para EUP. O objetivo principal é mostrar que os domínios de aplicação que essas ferramentas se propõem a atender ainda não têm sido aqueles que podem apoiar a Engenharia de Software no sentido de oferecer novas possibilidades às diferentes etapas do processo de desenvolvimento de software. A escolha dessas ferramentas foi motivada pelo fato de serem aplicações de uso comercial e, portanto, destacam-se por estar difundindo a EUP que, até há pouco tempo,

estava restrita às pesquisas acadêmicas. Isso, segundo Lieberman (2001), é um sinal do início da aceitação principalmente da programação por demonstração (PBD).

Stagecast® Creator© [Smith *et al*'2001, Stagecast Creator'2003] e ToonTalk© [Kahn'2001, ToonTalk'2003] destacam-se no contexto infantil. O principal objetivo dessas ferramentas é estimular a criatividade e o raciocínio das crianças de forma divertida.

A versão original de Stagecast® Creator© era conhecida como KidSim [Cypher *et al*'1995]. Creator© é um ambiente de programação em que crianças podem criar suas próprias histórias interativas, jogos e simulações. A programação é realizada visualmente com demonstrações através de regras *before-after*. As regras são equivalentes a subrotinas em linguagens de programação convencionais. A diferença é que as regras são criadas através da manipulação direta, ou seja, usando o mouse é possível determinar as ações dos personagens, tais como andar e saltar obstáculos. Além das regras de programação, algumas características dos personagens que aparecem nas histórias e simulações também podem ser manipuladas. É possível, por exemplo, criar comportamentos abstratos para eles, tais como definir que mostrem cansaço à medida que o tempo passa.

A ferramenta ToonTalk© começou com a idéia de que talvez a animação e a tecnologia de jogos de computador poderiam tornar o aprendizado da programação mais fácil e divertido [Kahn'2001]. Portanto, foi construída com base na idéia de programação animada com RV. Todo o seu sistema computacional é uma cidade. O programador é um personagem no mundo virtual animado, em que a programação de abstrações é substituída por analogias tangíveis: uma estrutura de dados, por exemplo, é uma caixa cujos buracos podem ser preenchidos com números ou blocos de texto, com outras caixas, com robôs, etc. Um processo é representado por uma casa, dentro da qual funções de um programa são executadas por robôs treinados através de demonstrações. De acordo com Kahn (2001), ToonTalk© tem sido usada com muito sucesso por crianças na construção de jogos e programas para a manipulação de palavras e para cálculos matemáticos.

A terceira ferramenta que se destaca no contexto de EUP é AgentSheets© [AgentSheets'2001, AgentSheets'2003, Repenning *et al*'2001], que permite a criação de jogos e simulações. AgentSheets© combina PBD com regras que podem ser escritas graficamente. Um número qualquer de regras de reescrita gráfica (*graphical rewrite rules* – GRR) pode ser agregado para criar comportamentos completos para os agentes. Os agentes são objetos que podem ser programados pelo usuário para reagir a cliques do mouse e entradas do teclado, ter sua aparência alterada, calcular fórmulas, entre outras funções. A programação em

AgentSheets© começa com a criação dos agentes. Em seguida, para definir o comportamento de um agente, o usuário abre as janelas *Conditions* e *Actions* e define as condições que devem ser atendidas para que um agente execute uma determinada ação. Isso é feito por manipulação direta, ou seja, para cada comportamento de um agente, o usuário estabelece uma regra através da combinação de ações e condições disponíveis na ferramenta, selecionando-as e arrastando-as diretamente com o mouse.

Com mecanismos de programação semelhantes ao de AgentSheets©, Alice [Alice'2003, Conway'1997, Conway *et al*'2000] é uma ferramenta que também merece destaque no contexto de EUP, embora não seja explicitamente classificada dessa forma pelos autores. O objetivo central de Alice também é oferecer um ambiente com RV que faça da programação uma atividade mais simples e agradável a estudantes. Assim, ao invés de exigir a digitação correta de uma série de comandos conforme a sintaxe de uma determinada linguagem de programação, a programação é realizada através do arrastar-e-soltar (*drag-and-drop*) em uma interface de manipulação direta [Alice'2003].

É interessante ressaltar que, em todas essas ferramentas, a programação é realizada com o uso de elementos visuais, eliminando a necessidade do usuário ter que aprender a sintaxe de uma linguagem de programação convencional.

Do estudo das ferramentas apresentadas nesta seção e de muitos outros trabalhos compilados por Cypher (1993) e Lieberman (2001), é possível perceber que as técnicas de EUP ainda não têm sido exploradas no apoio a ferramentas CASE. Neste trabalho, verificou-se a possibilidade de se estender a potencialidade dessas ferramentas permitindo um envolvimento do usuário nas atividades a que elas se propõem. Dessa forma, todo o processo de desenvolvimento de software pode ser beneficiado com a participação ativa do usuário nas diferentes etapas do projeto – na engenharia de requisitos, por exemplo – bem como no auxílio à elaboração da documentação do software em produção e, principalmente, na avaliação da usabilidade ao longo de todo o processo.

A breve apresentação dos principais aspectos dessas ferramentas também tem um outro propósito. A ferramenta GaCIV, na atual versão apresentada a seguir, pode ser classificada como pertencente ao Paradigma Paramétrico definido por da Silva (2001), conforme os mecanismos que ela disponibiliza para a programação por usuários finais. Uma outra pesquisa [Albertin'2002], também em desenvolvimento no Departamento de Computação da UFSCar, tem contribuído para que, em trabalhos futuros, a ferramenta disponibilize mecanismos que tornem possível a programação por demonstração e, portanto, possa ser classificada no Paradigma Imitativo. Por isso, algumas características das

ferramentas apresentadas nesta seção serão úteis na discussão apresentada ao final deste capítulo, com o intuito de se fazer uma pequena introdução a essa possibilidade emergente.

3.3. Princípios da ferramenta CASE GaCIV

A ferramenta GaCIV (Gabaritos Configuráveis para elaboração de Interfaces com realidade Virtual) [Silva'1999] tem como objetivo o desenvolvimento de interfaces com RV para aplicações de diferentes domínios, de forma rápida e simples, utilizando o conceito de RV não-imersiva. Nos sistemas não-imersivos, o mundo real não é ocultado e o usuário não perde a noção de seu ambiente natural, já que o mundo virtual é visualizado através do monitor de vídeo tradicional. A característica de não-imersão é usada para que as interfaces geradas com a ferramenta possam ser utilizadas em qualquer microcomputador com os dispositivos de interação convencionais, como mouse e teclado, tornando-as acessíveis a um número muito maior de usuários [Robertson *et al*'1993 *apud* Silva'1999]. Por esta razão, a não-imersão ajusta a ferramenta ao propósito da EUP, ou seja, disponibilizar os recursos computacionais a uma grande população de usuários. Além disso, o GaCIV pode evoluir para se adaptar a outras formas de interação [Assis'2001], inclusive para utilização na Web.

A ferramenta tem evoluído desde a sua primeira versão. O desafio de prover um ambiente em que o próprio usuário pudesse criar as interfaces para suas aplicações foi a grande motivação, entre outras, para o desenvolvimento da nova versão apresentada neste trabalho, cujo objetivo principal é a de inserir a ferramenta no contexto da EUP, conforme proposta apresentada em [Balbino *et al*'2002]. A nova versão foi desenvolvida com a linguagem de programação Java™. A API (*Application Programming Interface*) Java 3D é utilizada para a aplicação dos recursos de RV.

A seguir, é descrita a estrutura da ferramenta GaCIV, conforme ilustração na Figura 3-1.

3.3.1. A estrutura da ferramenta GaCIV

A ferramenta GaCIV é composta por dois programas: o InterBuilder, ferramenta responsável pela construção das interfaces com RV, e o InterViewer, um visualizador dessas interfaces.

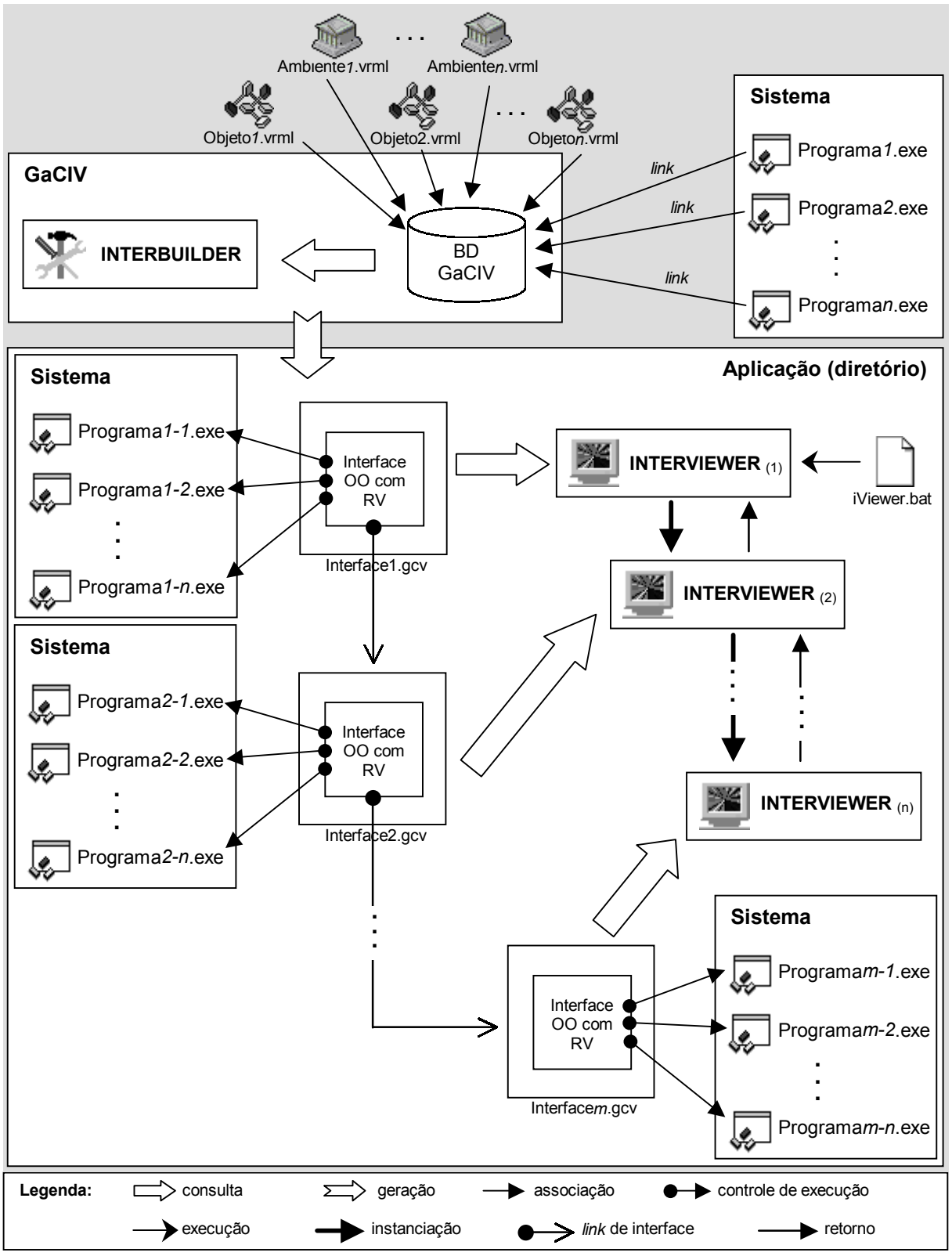


Figura 3-1. A estrutura da ferramenta GaCIV.

O InterBuilder oferece recursos de RV de forma acessível e prática, permitindo a manipulação de quatro componentes (no sentido mais amplo da palavra) básicos: objetos, ambientes, gabaritos e interfaces. Esses componentes são relacionados da seguinte maneira:

um gabarito é formado por um ambiente e um conjunto finito de objetos, enquanto que uma interface é composta a partir de um gabarito. Através da seleção e manipulação direta de ambientes e objetos, os gabaritos configuráveis podem ser construídos, servindo como modelos semiprontos que podem ser vistos como facilitadores do reuso de componentes, agilizando o desenvolvimento de interfaces para aplicações que se insiram no mesmo domínio [Assis *et al*'2000]. Por isso, é possível criar um gabarito para cada domínio de aplicação distinto como, por exemplo, locadora de vídeo, hospital, escritório, farmácia, etc. Após a criação de um gabarito configurável para um determinado domínio de aplicação, tem-se um modelo básico para o projeto de interfaces de aplicações que se enquadram no respectivo domínio, podendo ser editado para a inclusão de novos requisitos ou alteração de alguns aspectos do ambiente e dos objetos. Essa possibilidade de reuso, que explora a idéia de repositórios (chamados de gabaritos), representando domínios distintos de aplicações e contendo objetos representativos desses domínios, que ficam à disposição do usuário para geração de suas interfaces, é uma potencialidade intrínseca ao GaCIV que se destaca como um aspecto relevante no contexto da programação por usuários finais. Se a intenção da EUP é permitir que o usuário tenha facilidade para configurar ou estender uma aplicação, principalmente porque tais atividades normalmente não fazem parte de sua atividade principal, então o reuso se torna um fator ainda mais interessante porque agiliza o trabalho do usuário ao oferecer-lhe um modelo inicial para a programação. Neste trabalho, cujo objetivo é integrar o usuário no projeto de interfaces com RV, a possibilidade de reuso através dos gabaritos configuráveis oferece um diferencial para a ferramenta GaCIV, porque agiliza a participação do usuário, evitando que o tempo e o esforço necessários para essa tarefa não excedam os limites que o usuário se dispõe a oferecer.

O modo como o InterBuilder está estruturado possibilita que o desenvolvimento da interface seja dissociado do desenvolvimento da aplicação, garantindo, dessa forma, a independência de diálogo. Com essa característica, a ferramenta GaCIV pode ser usada tanto para a construção de uma interface para novos sistemas (engenharia avante) como para apoio à reengenharia de interfaces de sistemas legados [Soares'2002]. Além disso, o GaCIV pode ser visto como ferramenta de prototipação rápida, gerando protótipos que podem apoiar o levantamento de requisitos antes de iniciar o projeto do sistema em questão.

Na construção de uma interface, o usuário deve primeiramente selecionar o gabarito correspondente ao modelo que deseja, gabarito esse composto durante a etapa de levantamento de requisitos. Feita a seleção, o respectivo ambiente vinculado ao gabarito será visualizado e a construção da interface poderá ser realizada através da escolha dos objetos

disponíveis naquele gabarito. A presença de um objeto no gabarito não obriga o seu uso no projeto da interface, assim como um mesmo objeto pode ser criado mais de uma vez em uma mesma interface, atribuindo a cada um deles nomes diferentes.

Para integrar o subsistema de interface com o de aplicação, formando um sistema interativo, o InterBuilder oferece a opção de se atribuir um *link* de aplicação a cada objeto da interface. Um *link* de aplicação é uma âncora para uma aplicação, isto é, uma interligação de um objeto da interface e um arquivo executável. Assim, ao se clicar sobre um objeto da interface virtual do sistema, a aplicação associada ao objeto será executada. Também é possível atribuir um *link* de interface para objetos de outra interface, formando árvores de menu.

A construção de árvores de menu é resultado de outra proposta deste trabalho. Com essa função, novas possibilidades são oferecidas ao usuário para a construção de interfaces ainda mais completas. Assim, o usuário pode associar um objeto a um programa executável ou a um outro conjunto de objetos, que compõem uma outra interface. Ao executar a interface e clicar sobre este objeto, o usuário é direcionado a outro ambiente com as mesmas características de navegação do primeiro. A partir daí, um clique sobre um novo objeto executa uma aplicação ou conduz a outro ambiente virtual, e assim sucessivamente.

Com as árvores de menu, as interfaces podem ser organizadas mais adequadamente, permitindo uma interação mais fácil, principalmente no caso de sistemas compostos por um número expressivo de funcionalidades. Ao criar as interfaces para um sistema farmacêutico, por exemplo, o usuário pode criar o primeiro ambiente com os seguintes objetos: um balcão de vendas, que executa a aplicação de registro de vendas de medicamentos e outros produtos; um arquivo de aço, que executa a aplicação para cadastro de novos clientes; duas prateleiras, associadas a uma aplicação para consulta de disponibilidade de um determinado produto na farmácia e os respectivos preços; e uma porta com a palavra "Administração". Este último objeto pode ser associado a um outro ambiente que, ao ser exibido, mostra os novos objetos associados a outras aplicações, tal como um aparelho de fax relacionado a uma aplicação de correio eletrônico, através da qual são feitos os pedidos de reposição de estoque via e-mail, e outros objetos relacionados a um escritório de administração.

Além do exemplo da farmácia, pode-se citar um outro, referente à construção da interface com RV de um sistema para escritório. Supondo que ainda não existe um gabarito que atenda aos requisitos de aplicações deste domínio, a primeira tarefa será criá-lo. O gabarito é composto, então, por um ambiente semelhante a uma sala de escritório e por alguns

objetos comuns a esse ambiente, como mesa, cadeira, máquina de datilografia, computador, calculadora, lixeira, aparelho de fax, telefone, armário, entre outros. Durante a criação do gabarito, os objetos podem ser previamente configurados quanto à posição, rotação e orientação no ambiente. A partir daí, todas as interfaces construídas para sistemas de escritório podem reusar os objetos desse gabarito, com alterações na disposição desses objetos no ambiente, além da possibilidade de omissão ou duplicação de alguns deles. A mesa e a cadeira podem ser objetos meramente decorativos. Já a máquina de datilografia pode ser associada a um editor de textos, a calculadora a uma planilha eletrônica e o arquivo à aplicação responsável pelo armazenamento de informações sobre clientes.

Nota-se, portanto, uma considerável flexibilidade na configuração das interfaces e na manipulação de gabaritos configuráveis, que podem ser alterados conforme as necessidades e requisitos das aplicações de um determinado domínio. Essa característica também faz do GaCIV uma ferramenta de prototipação, permitindo o refinamento iterativo a partir de mudanças fáceis e rápidas durante todo o projeto de interfaces.

A partir do momento em que a interface estiver pronta e salva, a aplicação pode ser gerada. A aplicação é um diretório com o mesmo nome dado à interface, quando salva. A estrutura do diretório da aplicação tem a seguinte composição, conforme a Figura 3-1:

- O arquivo *iViewer.bat*, com a linha de comando para a execução da ferramenta InterViewer;
- A ferramenta InterViewer, semelhante a um *browser*, que possibilita a visualização de interfaces virtuais e a execução das aplicações associadas aos objetos que compõem a interface;
- Um ou mais arquivos textos com extensão *gcv*. Cada arquivo contém todas as especificações de uma interface orientada a objetos (OO) com RV, incluindo o ambiente e os objetos escolhidos, bem como as configurações de cada um desses objetos, isto é, posição, orientação, rotação, escala e o *link* para um arquivo executável ou para uma outra interface. Diz-se que a interface é OO por ser composta por um conjunto de objetos 3D. Todas essas informações são consultadas pelo InterViewer para a apresentação da interface ao usuário. O número de arquivos *gcv* depende da quantidade de interfaces diferentes que compõem a árvore de menus; em outras palavras, para cada interface que faz parte da árvore de menus, existirá um arquivo *gcv* correspondente, com todas as especificações necessárias. O nome do arquivo é o mesmo da interface;

- Os diretórios com os arquivos executáveis – as aplicações – e os arquivos referentes aos ambientes e objetos que compõem as interfaces.

A ferramenta InterViewer, ao consultar um arquivo *gcv* para carregar e exibir uma interface, verifica cada objeto para distinguir se ele está associado a um *link* de aplicação ou um *link* de interface. Dessa forma, quando o usuário clica sobre um objeto que contém um *link* de aplicação, o InterViewer executa o programa executável. Se o objeto clicado estiver associado a um *link* de interface, duas ações são executadas: a) é verificado o nome da interface referenciada; b) uma nova instância da ferramenta InterViewer é criada e o respectivo arquivo *gcv* é consultado para que o *browser* carregue a interface. Os objetos que compõem a nova interface exibida podem conter tanto *links* de aplicação como *links* de interface, e assim sucessivamente, gerando-se uma árvore de menu. A Figura 3-1 mostra a execução de uma árvore de menus. É possível perceber que, na primeira interface, definida pelo arquivo *Interface1.gcv*, existe um *link* de interface que aciona a exibição da interface definida no arquivo *Interface2.gcv*. Obviamente, a primeira interface poderia ter mais de um objeto associado a *links* de interface, ou nenhum. A partir da interface definida no arquivo *Interface2.gcv*, também é possível acionar outras interfaces, através de algum *link* de interface associado a um ou mais objetos 3D. À medida que os *links* de interface são executados e as interfaces exibidas sucessivamente, as novas instâncias das janelas do *browser* (o InterViewer) sobrepõem as anteriores. O usuário fica impossibilitado de retornar ao ambiente anterior sem que feche aquele exibido atualmente. Portanto, ao fechar a última interface exibida, o retorno à interface anterior é imediato. Na Figura 3-1, o objetivo principal é o de ilustrar o conceito dessa nova função e, por isso, limita-se a exibir somente três interfaces. No entanto, a profundidade da árvore de menus é ilimitada, e será definida de acordo com a complexidade do sistema e as necessidades do usuário. Uma possível limitação que se pode enfrentar refere-se aos recursos computacionais em que a aplicação é executada. A memória, por exemplo, pode ser insuficiente para a execução de várias instâncias do InterViewer, à medida que são criadas para exibir as interfaces que compõem a árvore de menu. No entanto, esse é um problema que dependerá do grau de profundidade da árvore e da quantidade de objetos que compõem uma interface.

A seguir, de uma maneira mais estruturada, são descritas as principais etapas para construção de interfaces através da ferramenta GaCIV. Uma apresentação mais ilustrada de suas funcionalidades será apresentada no próximo capítulo. As etapas são as seguintes:

a. Durante o levantamento e análise dos requisitos do sistema, o usuário se reúne com a equipe de desenvolvimento para definição dos principais objetos virtuais que deveriam compor a interface para aquele domínio de aplicação, de acordo com as funcionalidades definidas para o sistema [Assis *et al*'2000];

b. Se o ambiente e os objetos tridimensionais necessários ainda não estão disponíveis na ferramenta, são então criados por um *designer* gráfico e cadastrados no banco de dados do GaCIV, conforme ilustra a Figura 3-1. É interessante citar que a nova versão da ferramenta aceita objetos de diferentes formatos, como **.wrl* (VRML), **.obj* (Wavefront® Maya [Wavefront'2003]), entre outros. A Figura 3-2 mostra um objeto que representa um computador sendo cadastrado;

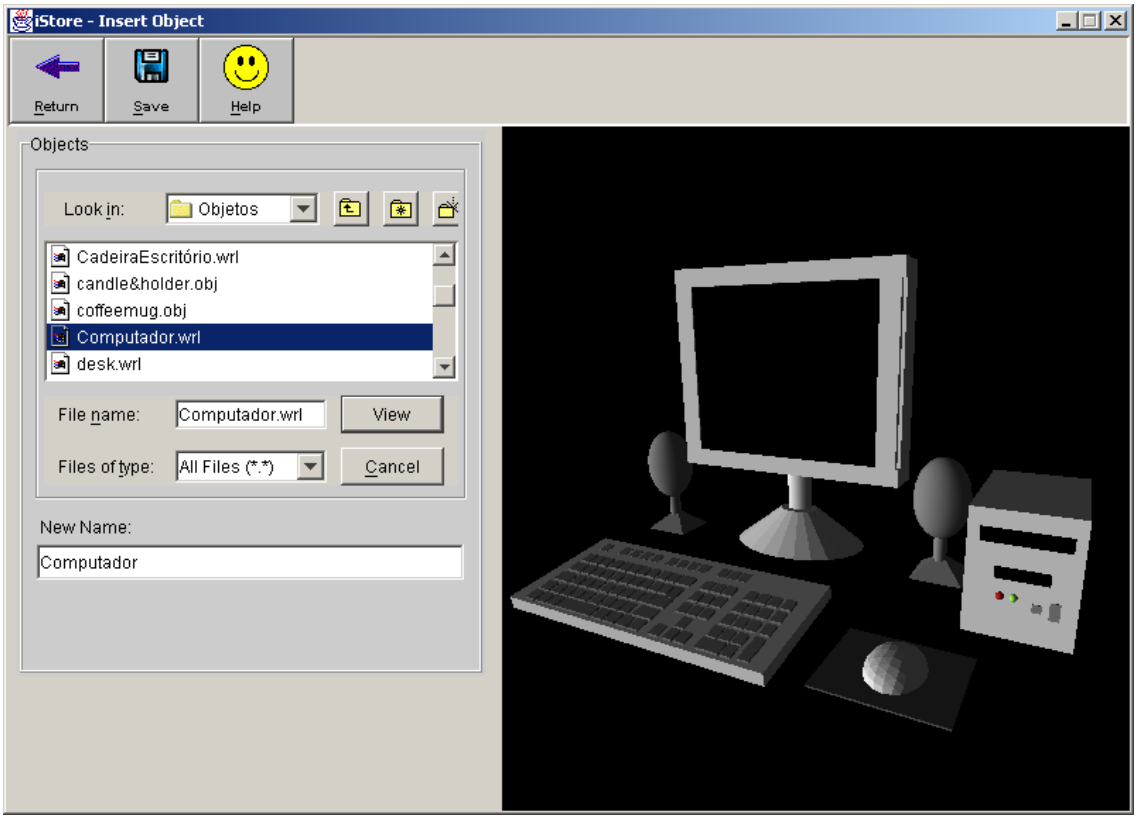


Figura 3-2. A inserção de um objeto no banco de dados do GaCIV.

c. Cadastrados os objetos, o InterBuilder pode ser usado para criação do gabarito, através da escolha de um ambiente e a composição da aparência do ambiente, através da seleção e posicionamento dos objetos. Neste caso, considera-se que o gabarito ainda não existe para o domínio da aplicação em desenvolvimento. Na Figura 3-3, um gabarito é criado para o domínio de locadora de vídeo. À medida que os objetos são selecionados e posicionados no ambiente para compor o gabarito, o InterBuilder

exibe a aparência do ambiente, possibilitando um acompanhamento de toda a criação do modelo por parte do usuário, que valida os requisitos levantados;

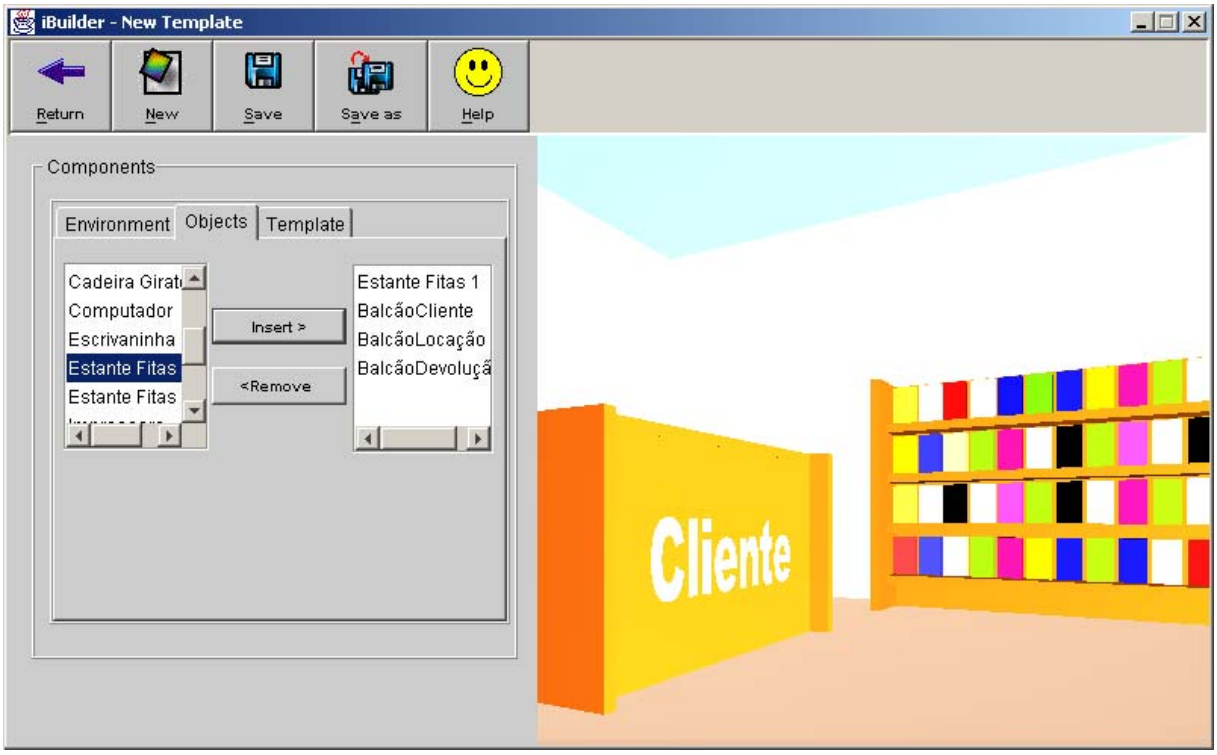


Figura 3-3. A criação de um gabarito.

d. Após a composição do gabarito configurável, que se torna disponível para ser utilizado em qualquer projeto de um sistema interativo que tenha características semelhantes à sua proposta [Assis'2001], a interface é criada e as aplicações são associadas aos objetos correspondentes. Na Figura 3-4, é mostrada uma interface sendo construída a partir do gabarito criado para o domínio de locadoras de vídeo (conforme ilustra a Figura 3-3). Os únicos objetos disponíveis para a construção da interface são os que foram escolhidos durante a criação do gabarito correspondente. A posição original dos objetos escolhidos para compor a interface é a mesma definida no gabarito, mas pode ser alterada. A Figura 3-5 mostra uma ampliação de parte da tela de criação de interfaces. Em A, é exibida uma lista com todos os objetos disponíveis no gabarito selecionado para a criação de interfaces; em B, é exibida uma lista dos objetos escolhidos para compor a nova interface. Em C, uma *combobox* lista todos os *links* de aplicação ou de interface, conforme a opção feita no painel "*Show only*". No exemplo, o objeto "Clientes" foi associado ao *link* "CadastroCliente", referente a uma aplicação para controle do cadastro de clientes.

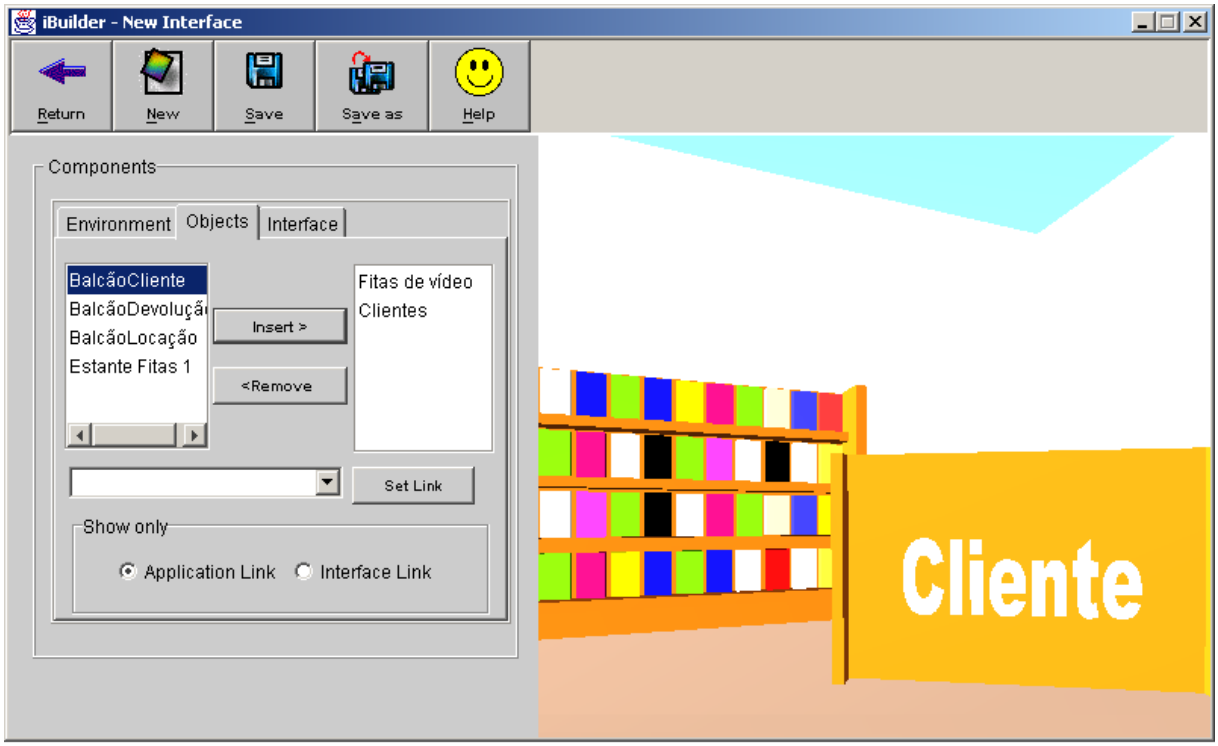


Figura 3-4. A criação de uma interface a partir de um gabarito disponível no banco de dados da ferramenta GaCIV.

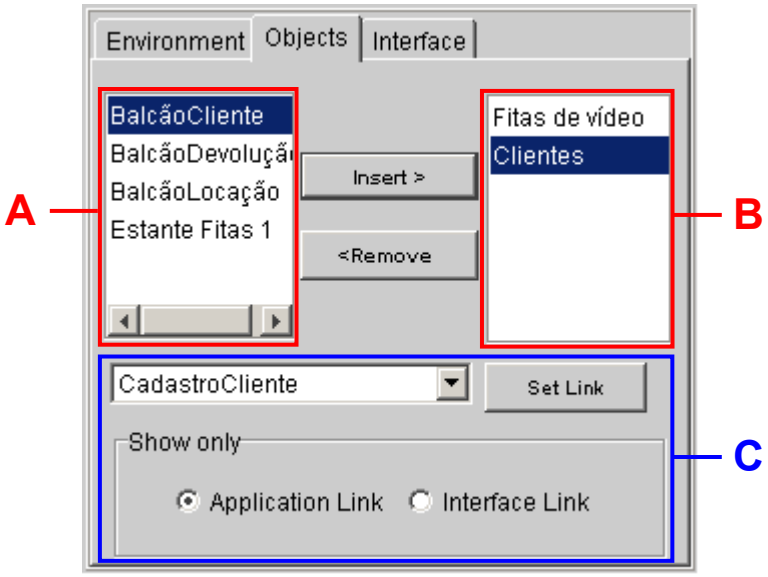


Figura 3-5. Associação dos objetos de uma interface a *links* de aplicação ou *links* de interface.

Finalmente, o pacote da aplicação pode ser gerado e transportado para a máquina do usuário.

Três observações importantes devem ser feitas. A primeira corresponde ao item (c), etapa em que o gabarito configurável é criado. Não é necessário que todos os objetos definidos durante a fase de análise de requisitos estejam disponibilizados. A composição do

gabarito pode ser gradativa, à medida que os objetos são criados e disponibilizados no banco de dados. A segunda observação corresponde ao item (d), etapa de criação da interface em que as aplicações poderão ainda estar em desenvolvimento. No entanto, conforme já explicado, o GaCIV oferece flexibilidade para a construção gradativa de interfaces. A terceira observação é conseqüência das anteriores: a flexibilidade proporcionada pela ferramenta permite que as interfaces sejam construídas e refinadas gradualmente, através da prototipação. Por isso, todas as etapas podem ser executadas em um ciclo iterativo, com uma avaliação contínua da usabilidade, envolvendo a equipe de desenvolvimento e o usuário.

3.3.2. Processo de desenvolvimento de interfaces com RV usando a ferramenta GaCIV

No processo de desenvolvimento de interfaces com RV com a ferramenta GaCIV é considerada uma equipe composta pelos seguintes atores: usuário, projetista de interfaces, designer gráfico e o engenheiro de software. Conforme descrito na introdução deste trabalho, o objetivo da EUP é justamente apoiar a integração efetiva do usuário nesse processo de desenvolvimento, especificamente na construção das interfaces com RV. A participação do usuário tem início na fase de levantamento de requisitos, de acordo com a descrição das etapas para a construção de interfaces apresentada na subseção 3.3.1, "A estrutura da ferramenta GaCIV". Nessa fase, não só os requisitos funcionais são analisados, mas também os requisitos de interface, ou seja, os objetos virtuais que devem compor a interface para aquele domínio de aplicação são identificados. O designer gráfico disponibiliza, então, os objetos tridimensionais identificados pelo usuário, através de softwares de modelagem de objetos 3D. Na construção do gabarito, a próxima etapa do processo, a participação do usuário pode ser direta ou indireta, isto é, ele mesmo pode construí-lo, ou pode apenas acompanhar e orientar o projetista de interfaces nesta tarefa. Nesta fase, a presença do usuário é muito importante, pois o gabarito deve ser composto pelos objetos virtuais necessários à construção de interfaces para o domínio de aplicações em questão. Por isso o usuário, como fonte de aquisição de requisitos, é fundamental para que o gabarito seja composto de acordo com as reais necessidades do sistema. Além disso, a fase de construção do gabarito pode ser útil para que a falta de objetos necessários para aquele domínio de aplicações seja mais facilmente identificada, além de outras possíveis alterações nos requisitos de interface. Já na construção da interface, é esperada uma participação direta do usuário, ou seja, ele mesmo deve construí-la usando a ferramenta GaCIV e sua percepção da realidade em que está inserido. Esta

percepção pode trazer benefícios à usabilidade das interfaces, porque são construídas segundo as expectativas e visão do próprio usuário. De qualquer forma, espera-se que o projetista de interfaces acompanhe o trabalho do usuário ao longo de todo o processo de construção de interfaces, fornecendo orientações e esclarecendo dúvidas. Além disso, o engenheiro de software também tem uma participação constante nesse processo, já que ele é o fornecedor dos programas aplicativos do sistema.

Nota-se, portanto, que a comunicação entre os atores envolvidos no processo de desenvolvimento de interfaces com RV usando a ferramenta GaCIV é constante: o usuário compartilha sua experiência em relação ao domínio considerado para o levantamento de requisitos, a definição do gabarito e a construção das interfaces. O projetista de interfaces acompanha o trabalho do usuário, esclarece dúvidas e o orienta na construção de interfaces, a fim de que tão somente a percepção pessoal do usuário não tenha impactos negativos na usabilidade das interfaces e prejuízos na interação com o sistema. Isso não significa que o usuário seja incapaz de construir interfaces com RV adequadas às suas aplicações, mas quando se espera que o projetista de interfaces interceda no projeto, deseja-se que ele colabore com sua experiência para ajustar as interfaces construídas aos princípios de interação humano-computador, principalmente nos projetos em que árvores de menu são utilizadas, atribuindo maior complexidade às interfaces. O designer gráfico, função que pode ser do próprio projetista de interfaces, modela os objetos 3D identificados na fase de estudos de requisitos. O engenheiro de software tem presença constante no processo de desenvolvimento de softwares e, no projeto de interfaces com RV usando a ferramenta GaCIV, tem a responsabilidade de garantir que os requisitos funcionais sejam atendidos e disponibilizados como links de aplicação no banco de dados do GaCIV.

Na Figura 3-6 é apresentado o modelo de processo seqüencial linear, adaptado do modelo original em cascata proposto por Royce (1970) *apud* Pressman (2002). Na Tabela 3-1 é apresentado um detalhamento das atividades em cada fase proposta neste modelo de processo de software para exemplificar o projeto de interfaces com RV com a ferramenta GaCIV. A tabela está dividida em três colunas: a primeira especifica a fase do modelo de processo; a segunda detalha as atividades que compõem a respectiva fase e; a terceira especifica os artefatos produzidos em cada fase. São especificados os artefatos relacionados ao escopo de Engenharia de Software e aqueles relacionados à interação humano-computador, neste caso, os artefatos referentes ao desenvolvimento de interfaces com RV usando a ferramenta GaCIV. O modelo de processo apresentado, no entanto, é apenas um exemplo e não limita o uso da ferramenta GaCIV unicamente ao seu escopo.

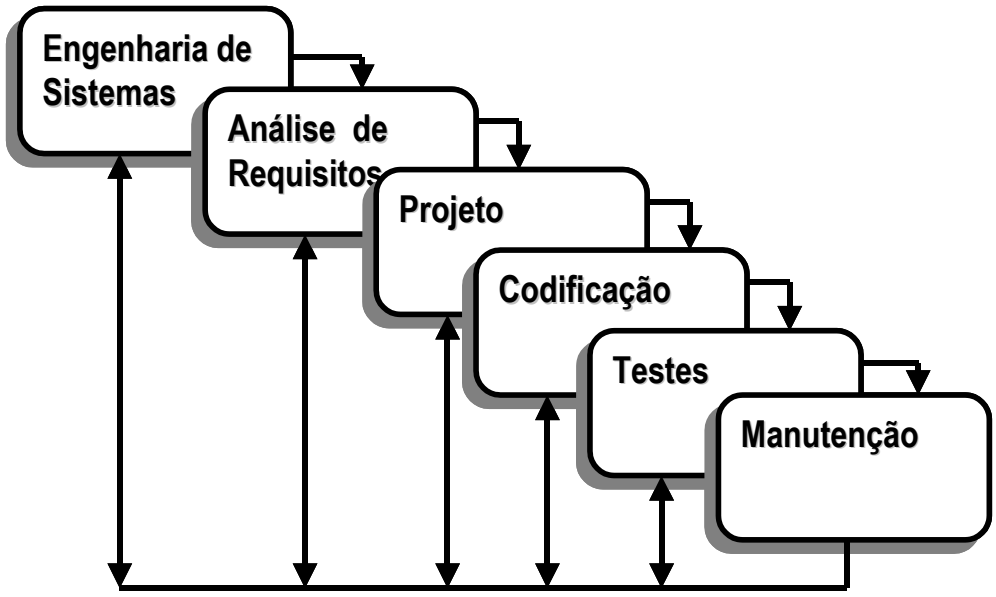


Figura 3-1. Modelo de processo seqüencial linear.

Tabela 3-1. Exemplo de projeto de interfaces com RV inserido em um modelo de processo de software

Fases do ciclo de vida	Tarefas	Artefatos de Engenharia de Software / IHC
Engenharia de sistemas	Discutir as opções de sistema *stand alone* e baseado na Web com cliente	Lista de opções / gabarito
	Discutir sistema *stand alone* ou baseado na Web	Descrição da opção escolhida / gabarito e 1.º protótipo da interface com RV
Análise de requisitos	Elicitar requisitos com cliente	Entrevistas transcritas / gabarito e protótipo refinado
	Construir documento de requisitos	Documento de requisitos / gabarito completo e protótipo refinado
	Rever requisitos com cliente	Documento inspecionado / protótipo refinado
	Criar versão preliminar de manual de usuário	Manual preliminar / protótipo refinado
	Rever versão preliminar de manual de usuário	Manual revisado / protótipo refinado
	Construir casos de teste	Lista de casos de teste / protótipo refinado
Projeto	Projetar o protótipo do sistema	Projeto lógico do sistema / protótipo refinado
	Projetar formato de relatórios	Projeto de formato de relatórios / protótipo refinado
Codificação	Desenvolver o protótipo do sistema	Protótipo implementado / *links* de aplicação disponíveis
	Implementar a seção de buscas e relatórios	Seção implementada / *links* de aplicação disponíveis
	Rever o protótipo com o cliente	Relatório de revisão de protótipo / aplicações associadas à interface com RV
	Implementar a versão final	Versão implementada / interface com RV completa e pacote de aplicação gerado
Testes	Testar a versão final do sistema	Relatório de testes
	Treinar clientes	Relatório de treinamento
	Entregar documentação	Documentação liberada
Manutenção	Atender solicitação do cliente	Relatório de atendimento

De acordo com a Tabela 3-1, a ferramenta GaCIV pode auxiliar em parte a fase de levantamento de requisitos. Na primeira fase – Engenharia de sistemas – são discutidas as opções pelas quais o usuário (ou cliente) pode decidir: o desenvolvimento de um sistema stand alone ou baseado na Web. Nesta fase pode-se utilizar um gabarito já existente para a elaboração de um primeiro protótipo da interface com RV a fim de auxiliar o usuário na escolha das opções apresentadas. No caso da inexistência do gabarito para o domínio de aplicações em questão, pode-se construir uma primeira versão para que seja possível elaborar o primeiro protótipo da interface com RV.

Na segunda fase do modelo de processo – Análise de requisitos, quando os requisitos são levantados e analisados criteriosamente, o gabarito, já existente ou inicialmente criado na fase anterior, é refinado para explicitar os requisitos de interface, conforme os requisitos funcionais, tendo em vista que normalmente cada programa será associado a um objeto virtual da interface. Paralelamente, o protótipo da interface com RV também é refinado. Nesse ponto, é interessante destacar que o protótipo da interface com RV pode auxiliar no levantamento de requisitos porque é um recurso visual em que o usuário, o projetista de interfaces e o engenheiro de software podem visualizar mais claramente os requisitos e se, para cada objeto disponível na interface, foi especificado o respectivo programa, e vice-versa. Assim, à medida que o documento de requisitos é elaborado, também é criado um protótipo gradativamente mais refinado da interface com RV, apoiando visualmente a validação de requisitos funcionais e de interface. O refinamento do protótipo da interface com RV prossegue na fase seguinte – Projeto, em que o principal objetivo é projetar o protótipo do sistema em desenvolvimento.

Na quarta fase – Codificação, os protótipos do sistema são implementados e disponibilizados na ferramenta GaCIV como links de aplicação. Isso permite a associação de objetos 3D, que compõem a interface com RV, com os respectivos programas. Dessa forma, obtém-se a aplicação completa e o pacote de aplicação pode ser gerado e instalado no(s) computador(es) do usuário. Com o pacote de aplicação gerado e instalado, a fase de testes pode ser realizada para identificar possíveis problemas remanescentes no sistema e para treinar demais usuários.

O modelo seqüencial linear, ilustrado na Figura 3-6, prevê um *feedback* contínuo de uma fase a outra anterior, a fim de que o sistema seja gradativamente refinado. Por isso, a ferramenta CASE GaCIV se adequa a esse modelo de processo por uma de suas potencialidades – a prototipação, destacada na subseção 3.3.1, “A estrutura da ferramenta GaCIV”. Assim, a possibilidade de desenvolvimento de protótipos evolutivos destaca-se

como apoio à atividade de design de interfaces com RV com a ferramenta GaCIV. Além disso, a fase de manutenção é beneficiada pela facilidade oferecida para o acomodamento de alterações nas interfaces com RV provindas de alterações nos requisitos do sistema solicitadas pelo próprio usuário ou necessárias pela presença de falhas ao longo de todo o ciclo de vida do sistema.

3.3.3. Classificação da ferramenta GaCIV no contexto da EUP

Após a apresentação das características e potencialidades da ferramenta GaCIV, é possível identificar os pontos que a inserem no contexto da EUP. Essa potencialidade embutida na ferramenta GaCIV a transformou numa CASE de apoio ao projeto de interfaces que proporciona o envolvimento do usuário, através de sua participação direta na construção das interfaces para suas aplicações. O envolvimento do usuário ao longo de todo o processo de desenvolvimento de software tem sido destacado positivamente por diferentes autores [Drake *et al*'1997, Leonard'2002, Nielsen'1993, Noyes *et al*'1995a, Noyes *et al*'1995b, Wiklund'1994], principalmente porque conduz a produtos e sistemas projetados mais adequadamente, conforme a perspectiva do usuário [Noyes *et al*'1995b]. Nielsen (1993) comenta que, ao se promover o envolvimento do usuário no projeto de interfaces, não é razoável esperar que ele traga idéias prontas de projeto. Apesar disso, continua o autor, os usuários são muito bons para reagir a projetos concretizados de que eles não gostam ou que, segundo sua visão, não irão funcionar na prática. Nesse ponto, é importante destacar que a nova versão da ferramenta GaCIV, dadas as suas características, tem o objetivo de fornecer as condições necessárias para que o usuário seja, de fato, um projetista de interfaces. Conforme comentado na seção 2.3, "Princípios de Realidade Virtual (RV)" deste trabalho, a forma como a ferramenta disponibiliza os recursos para o projeto de interfaces com RV facilita essa tarefa de tal maneira que o próprio usuário possa se sentir suficientemente confortável para o seu desenvolvimento.

A manipulação direta é uma das características da ferramenta GaCIV que merecem destaque no contexto da EUP como apoio ao desenvolvimento de sistemas com RV. Os gabaritos configuráveis e as interfaces são construídos através da seleção de objetos em uma lista, que são inseridos no ambiente virtual. Uma vez inseridos no ambiente, os objetos podem ser arrastados para uma determinada posição e movimentados para determinar a sua rotação e orientação.

De acordo com Shneiderman (2001), manipulação direta é um termo que descreve o mundo visual de ação em muitas interfaces gráficas, tendo como princípios:

- A representação visual de objetos e ações;
- A substituição da digitação pelo apontar (*pointing*) e arrastar (*dragging*);
- A possibilidade de realização de ações rápidas, incrementais e reversíveis;
- Uma resposta (*feedback*) imediata e contínua aos usuários.

A ferramenta GaCIV suporta os princípios da manipulação direta enumeradas por Shneiderman, porque os objetos virtuais são visualizados pelo usuário e podem ser apontados e arrastados com o mouse, sendo que as ações de posicionamento, rotação e orientação são também visuais. Com isso, o usuário acompanha passo a passo a construção da interface, dada a resposta imediata possibilitada pela visualização contínua da aparência do ambiente virtual. A associação dos objetos a módulos executáveis ou a outras interfaces também é realizada através da seleção em uma lista de opções e, portanto, com o uso do mouse. Essas características permitem que as ações sejam rápidas e incrementais, à medida que os objetos são inseridos e configurados, e podem ser revertidas segundo a necessidade do usuário em retornar um objeto a uma posição anterior ou removê-lo, por exemplo.

Enquanto a manipulação direta é uma característica que oferece benefícios ao usuário, conforme explicitado anteriormente, outros mecanismos devem ser destacados pelo apoio à classificação do GaCIV como uma ferramenta que se insere no paradigma de programação paramétrico proposto por da Silva [da Silva'2001].

Em relação ao mecanismo de **configuração de preferências**, o usuário tem a capacidade de ativar (incluir) ou desativar (excluir) objetos previamente selecionados como possíveis opções na composição de um ambiente virtual. A atribuição de nomes aos objetos, diferenciando-os durante a construção de interfaces, também é vista como a definição de valor de um parâmetro (no caso, o nome de um objeto). Ainda como configuração de preferências, destaca-se a associação de um objeto a um programa executável ou a uma outra interface, pela atribuição de um *link*, outra propriedade dos objetos que pode ser configurada.

Barbosa (1999) afirma que as aplicações extensíveis que usam a técnica de configuração de parâmetros podem ser analisadas de acordo com o número de parâmetros que o usuário pode configurar e o número de combinações obtidas. Assim, poucos parâmetros e combinações resultam em um baixo potencial de extensão da aplicação, enquanto muitos parâmetros reduzem as chances de o usuário prever com sucesso qual será a configuração resultante da combinação desses parâmetros, uma dificuldade também destacada por da Silva

(2001), conforme foi comentado na subseção 2.2.1, “Técnicas de EUP e a classificação de paradigmas”, deste trabalho. No caso da criação de interfaces na ferramenta GaCIV, o número de parâmetros disponíveis para configuração é bastante reduzido, pois compreende a escolha do ambiente e dos objetos, a definição da posição, rotação, orientação e nome desses objetos e a associação de cada um deles a *links* de aplicação ou de interface. Essa característica, no entanto, é um ponto forte da ferramenta, por dois motivos: 1) os parâmetros de configuração disponíveis, embora sejam poucos, são suficientes para que interfaces completas com RV sejam construídas e, portanto, não há prejuízos quanto ao potencial da ferramenta em relação ao seu objetivo principal; 2) essa limitação evita que um número excessivo de parâmetros reduza a chance de o usuário realizar sua tarefa satisfatoriamente segundo as suas necessidades. Quanto à dificuldade de o usuário prever o efeito final de um conjunto de escolha de parâmetros sobre a aplicação, a ferramenta GaCIV oferece uma solução que também é resultado de suas próprias características. A definição dos parâmetros de posicionamento e rotação dos objetos dentro do ambiente é realizada através da manipulação direta, e o efeito final dessa configuração nada mais é do que a aparência visual do ambiente, que o usuário visualiza durante toda a construção da interface. Essa visualização oferece, portanto, um *feedback* constante ao usuário e ao projetista, possibilitando um acompanhamento contínuo dos resultados obtidos. Já a definição de *links* associados aos objetos de uma interface é um tipo de parâmetro que dificilmente dificultará a compreensão do usuário em relação ao efeito final do projeto da interface em construção. Isso porque esse tipo de configuração é uma tarefa bastante simples, já que a associação de um *link* de aplicação a um objeto implicará, como efeito final, na execução da respectiva aplicação, e a associação de um *link* de interface a um objeto implicará na visualização da respectiva interface. Um possível problema que pode ocorrer está relacionado ao cadastro de *links*, ou seja: se no momento em que se cadastra um *link* de aplicação, atribui-se a ele um nome que não corresponde claramente ao respectivo programa, o usuário pode confundir-se no momento de realizar a associação com um objeto 3D. Ou, ainda, ele pode não encontrar o *link* desejado. O mesmo pode acontecer com *links* de interface, como resultado do cadastramento de interfaces com nomes confusos. Como conseqüência, aplicações podem acabar sendo cadastradas mais de uma vez e uma mesma interface pode ser criada novamente sem necessidade. De qualquer forma, esse problema poderá ser adequadamente dirimido ao se ter o usuário integrado e atuante no projeto de interfaces, já que ele auxiliará na definição dos termos adequados referentes ao domínio da aplicação em questão.

O mecanismo de **personalização da aparência da interface** é uma atividade intrínseca à construção da interface. Tendo o gabarito como modelo, o usuário define a aparência do ambiente virtual, inserindo objetos e configurando as suas propriedades. Nesse mecanismo, é importante destacar que a interação humano-computador será beneficiada conforme o grau de similaridade entre o ambiente virtual e o ambiente real. Por exemplo: se em um escritório de contabilidade os funcionários estão habituados a guardar as fichas de cadastro dos clientes em um armário de aço, então se torna interessante que um dos objetos que compõem o ambiente virtual em construção represente um armário de aço e seja associado à aplicação de cadastro de clientes. Assim, o usuário absorverá mais facilmente o modelo de usabilidade da aplicação em desenvolvimento. Essa prática reforça a questão da força da analogia como um mecanismo cognitivo de aquisição e construção de novos conhecimentos [Lakoff *et al*'1980 *apud* Barbosa'1999, Lakoff'1987 *apud* Barbosa'1999, Repenning *et al*'2001].

No caso do mecanismo de **uso de moldes**, a ferramenta GaCIV é adequada porque permite a construção de modelos de interface – os gabaritos configuráveis – que podem ser comparados ao agrupamento de um conjunto de parâmetros como, por exemplo, o ambiente vinculado ao modelo, os objetos que compõem esse modelo e suas respectivas propriedades de posicionamento, rotação e orientação. Estes modelos, que são disponibilizados no banco de dados do GaCIV, servem como uma configuração inicial das interfaces que podem ser geradas para um tipo específico de tarefa, ou seja, um domínio particular de aplicação.

3.3.4. Em direção ao Paradigma Imitativo

Um outro trabalho tem sido desenvolvido para que as interfaces com RV criadas com o GaCIV sejam orientadas a componentes [Albertin'2002]. A partir de alguns resultados já obtidos na implementação dessa nova versão da ferramenta, de discussões e análises informais, um importante avanço é o de embutir na ferramenta características que a classificarão no Paradigma Imitativo. Em outras palavras, o GaCIV poderá suportar a criação de componentes (objetos virtuais com comportamento) pela programação por usuários finais. Nesse caso, o usuário poderá configurar e estender os comportamentos dos componentes multimídia que compõem os gabaritos e interfaces. É importante ressaltar que os componentes multimídia representam, numa definição sucinta, os objetos virtuais atualmente manipulados

no GaCIV, mas com comportamentos associados a eles. Comportamentos ou funcionalidades são métodos que controlam a maneira como os componentes multimídia interagem com usuários e incluem controle de dispositivos de entrada/saída, transformações geométricas, animações, tratamento de colisão, reações a eventos externos, etc.

Nas ferramentas AgentSheets© e Alice, apresentadas na seção 3.2, "Ferramentas para EUP e suas principais aplicações", destaca-se um mecanismo muito interessante de programação por manipulação direta. Dentre as duas, as características de Alice são as que mais se aproximam das características da ferramenta GaCIV, porque dispõe de um ambiente para programação de cenas compostas por objetos tridimensionais e, portanto, também usa recursos de RV. As cenas são construídas através da escolha de objetos 3D disponibilizados na ferramenta e com comportamentos predefinidos. Um coelho, por exemplo, tem comportamentos distintos para as diferentes partes do corpo, como movimentar a cabeça para os lados direito e esquerdo, movimentar os braços para frente e para trás, movimentar-se para uma determinada direção, entre outros. Os novos comportamentos podem, assim, ser criados a partir da combinação dos que já existem, inclusive com o uso de condicionais e iterações também disponíveis para a programação. Se é desejado que um coelho pule, mas esse comportamento não existe, é possível que ele seja criado através da composição dos seguintes comportamentos:

a. O coelho se movimenta verticalmente, para cima, numa altura que é determinada através da configuração do parâmetro correspondente;

b. O coelho se movimenta horizontalmente, para a frente, numa distância que é determinada através da configuração do parâmetro correspondente. Nesse caso, o movimento é realizado acima do solo, em razão do movimento anterior;

c. O coelho se movimenta verticalmente, para baixo, de acordo com a mesma altura determinada no passo (a).

É possível determinar que esses três movimentos sejam executados num ciclo iterativo, até que uma condição de parada seja satisfeita.

Dessa breve análise da ferramenta Alice, pode-se afirmar que ela é um referencial para a implementação dos mecanismos necessários à programação dos comportamentos de componentes no GaCIV. Ao lado de Alice, destaca-se também AgentSheets©, que se baseia nos mesmos mecanismos, diferenciando-se apenas por não utilizar recursos de RV.

Na ferramenta GaCIV, um determinado objeto 3D é associado a uma classe que implementa os seus comportamentos, compondo assim os componentes multimídia. Dessa

forma, métodos que implementam uma resposta a um clique do mouse, que alteram a geometria de um objeto quando o ponteiro do mouse é movimentado sobre ele, entre outros, podem ser disponibilizados como elementos gráficos que, uma vez combinados e configurados, dão origem a novos comportamentos que podem ser cadastrados e reutilizados para outros objetos.

Essa, no entanto, é uma proposta emergente para trabalhos futuros e que surge como um grande desafio. Um dos problemas que se poderá enfrentar é como limitar a associação inadequada de comportamentos a objetos. Se um comportamento é responsável pelo movimento de abertura de uma porta, como ele funcionará se for associado a uma janela que também deve ser aberta, mas de maneira ligeiramente diferente? É possível que, através da configuração de parâmetros, possam ser determinados os detalhes do movimento. Entretanto, a única finalidade desta subseção é abrir a discussão para uma nova proposta para a ferramenta GaCIV e, por isso, não tem a intenção de explorar o assunto minuciosamente ou de garantir que a implementação desses novos mecanismos é de fato possível ou viável. Conforme já foi comentado, as idéias aqui apresentadas são o resultado de observações e discussões informais, não contando com a fundamentação teórica necessária. De qualquer forma, é um desafio que merece atenção.

3.4. Considerações finais

A classificação de uma ferramenta CASE no contexto da EUP traz para essa área de pesquisa uma nova perspectiva que poderá propiciar, ainda, muitos benefícios não só à interação humano-computador, mas também à Engenharia de Software, como apoio às diversas etapas do processo de desenvolvimento de sistemas através do envolvimento do usuário.

Nota-se que as técnicas de EUP associadas aos recursos de RV trazem interessantes benefícios no projeto de interfaces. Destaca-se a característica de não-imersão, que promove o uso de recursos de RV por um número consideravelmente maior de usuários, porque não precisam dispor de equipamentos caros e sofisticados para a interação com o ambiente virtual. Dessa forma, todos aqueles que têm um computador pessoal, têm igualmente a oportunidade de utilizar a ferramenta GaCIV e usufruir suas potencialidades na construção de interfaces com RV.

No próximo capítulo, é apresentado um estudo de caso, que ilustra o funcionamento da ferramenta, permitindo uma visualização mais clara do uso de todos os conceitos discutidos neste capítulo.

4. Construindo interfaces com a ferramenta GaCIV

4.1. Considerações Iniciais

Neste capítulo são apresentados um exemplo de projeto de uma interface para escritório, que mostra claramente todas as atividades necessárias para a construção de interfaces desde a definição dos gabaritos, e um estudo de caso, onde é projetada a interface para uma aplicação do domínio de locadora de vídeo por um usuário.

Na seção 4.2 é apresentado o exemplo de aplicação da ferramenta GaCIV no projeto de uma interface para escritório. Na seção 4.3, é apresentado e discutido o estudo de caso, em que um usuário teve um envolvimento ativo. Na seção 4.4, são apresentadas as considerações finais deste capítulo.

4.2. Um exemplo de aplicação

Conforme descrito na subseção 3.3.1, "A estrutura da ferramenta GaCIV", do capítulo anterior, o primeiro passo é definir, durante a etapa de levantamento de requisitos, quais objetos irão compor a interface da aplicação em desenvolvimento. Nesta etapa, o usuário tem uma importante participação, já que ele é o principal fornecedor dos requisitos do sistema. Caso seja verificado que alguns desses objetos (ou todos eles) não estão disponíveis, devem ser modelados por um *designer* gráfico. O GaCIV não tem um modelador de objetos 3D e, portanto, eles devem ser modelados com o auxílio de softwares para essa finalidade.

Quando é executada, a ferramenta GaCIV apresenta, inicialmente, a caixa de diálogo em que o modo de acesso deve ser definido. Existem dois modos de acesso, conforme mostra a Figura 4-1:

a. *Complete Mode*, que permite o acesso a todas as funções (para projetistas de interface e engenheiros de software);

b. *End-User Mode*, um modo de acesso mais restrito, próprio para usuários, e que habilita o acesso somente às funções de construção de gabaritos, de interfaces e a geração do pacote de aplicação.

O acesso completo só difere porque habilita as opções de cadastro de ambientes, objetos e *links* de aplicação, conforme mostra a Figura 4-2. Essas opções são reservadas para o engenheiro de software e/ou projetista de interface, que disponibilizam todos os elementos necessários para a elaboração das interfaces.

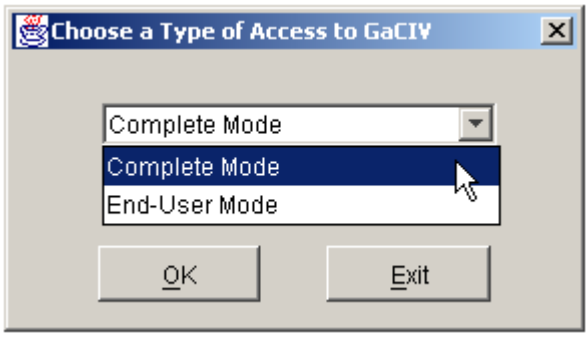


Figura 4-1. Modos de acesso da ferramenta GaCIV.

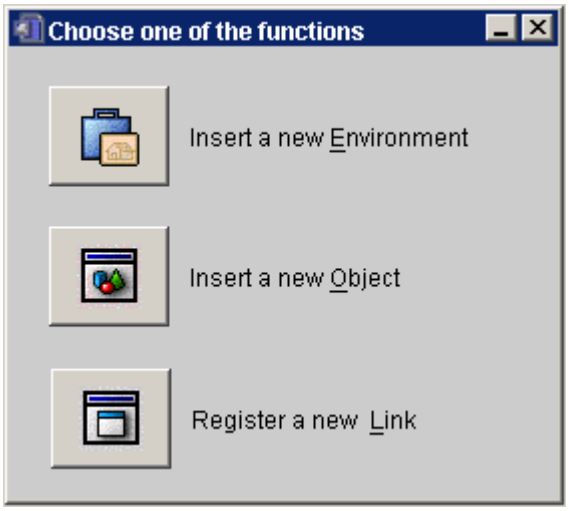


Figura 4-2. Opções para cadastramento de ambientes, objetos e *links* de aplicação.

A Figura 4-3 mostra um ambiente sendo cadastrado. Para isso, um arquivo é selecionado e o botão "*View*" é pressionado. Assim, o ambiente é carregado e exibido no *browser*. Caso se deseje configurar uma determinada visão do ambiente como padrão, basta que a visão seja definida e, em seguida, que o botão "*Set View Point*" seja pressionado. A visão determina a forma como o ambiente será visualizado sempre que for utilizado para a construção de um gabarito. Finalmente, um nome é atribuído ao ambiente para que ele seja registrado no banco de dados. O registro dos objetos é muito semelhante ao dos ambientes, conforme mostra a Figura 4-4, em que um armário de aço é cadastrado.

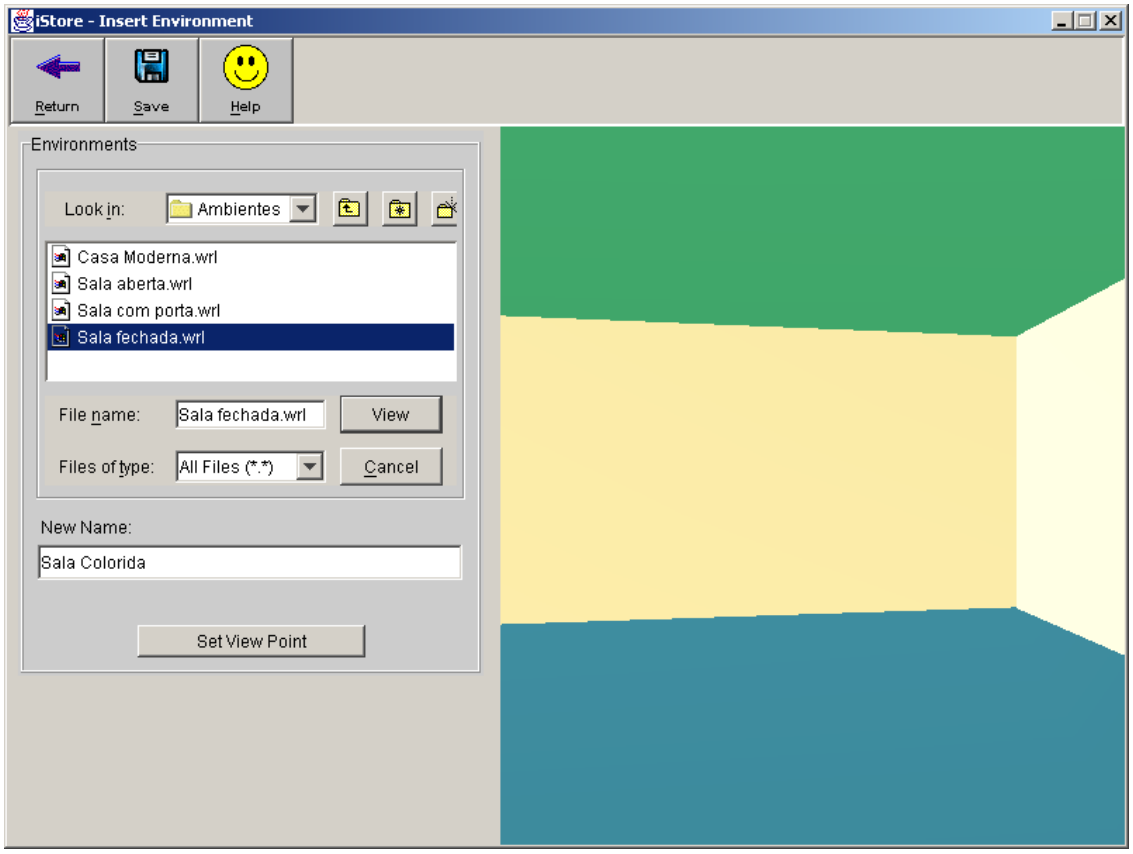


Figura 4-3. Um ambiente sendo cadastrado.

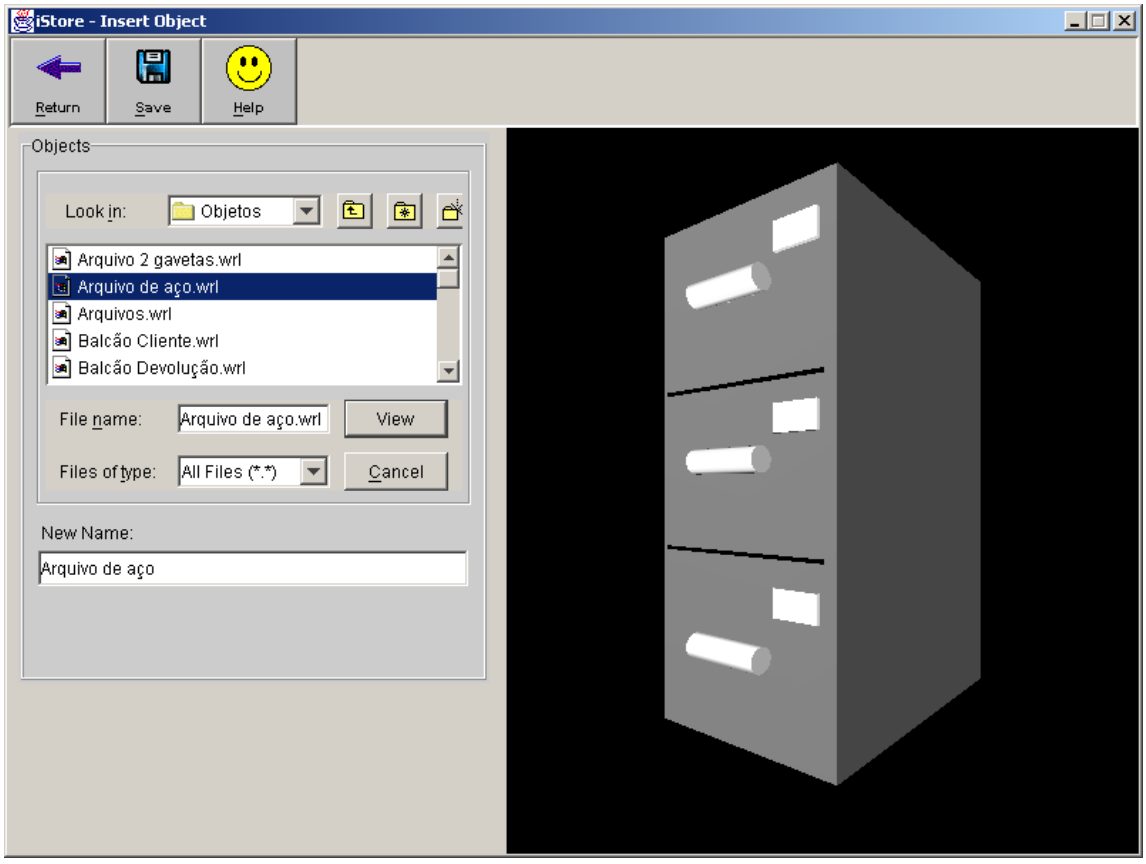


Figura 4-4. Um objeto sendo cadastrado.

O registro de *links* de aplicação também é uma atividade simples. Inicialmente, um programa executável é selecionado. Assim como nas opções de registro de ambientes e objetos, um novo nome é atribuído ao *link* para que seja identificado mais facilmente durante a construção das interfaces. Uma descrição também pode ser inserida junto com o *link*, a fim de esclarecer mais detalhadamente as funções da respectiva aplicação. A Figura 4-5 mostra um programa de livro-caixa sendo cadastrado.

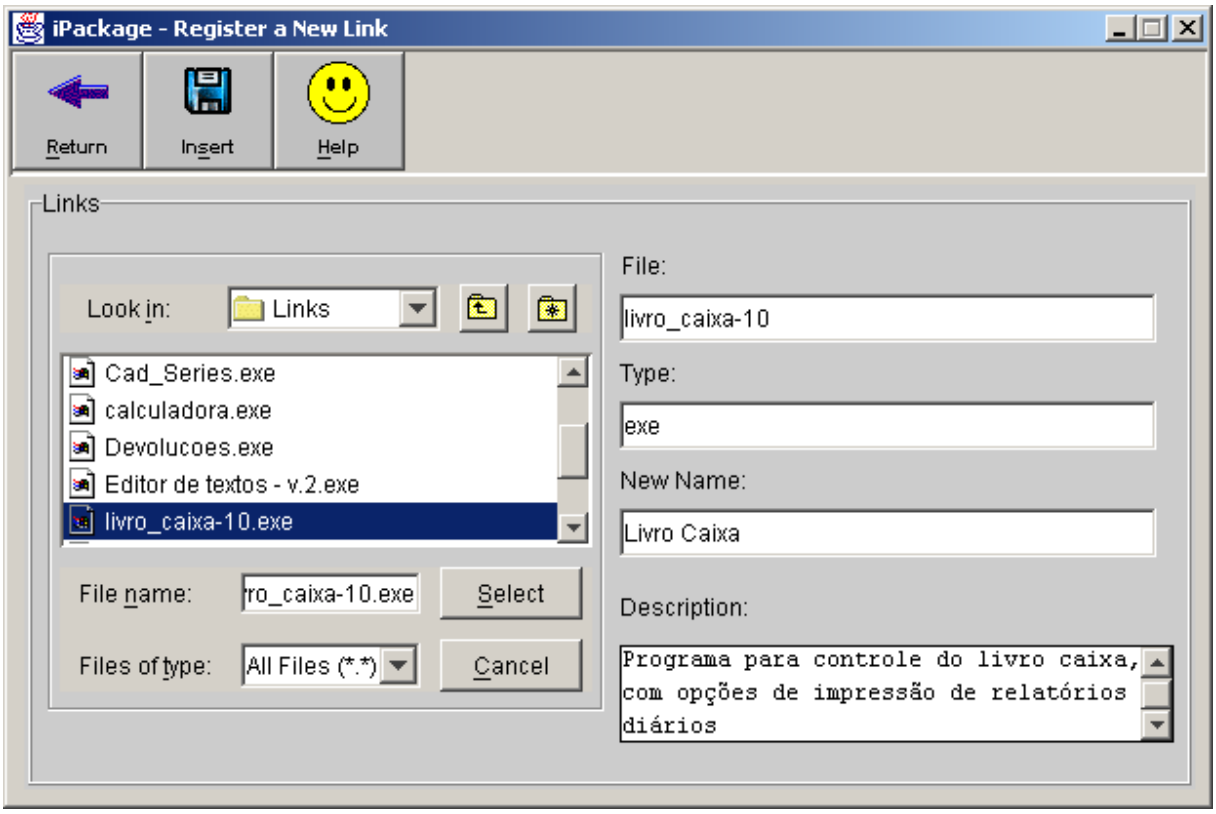


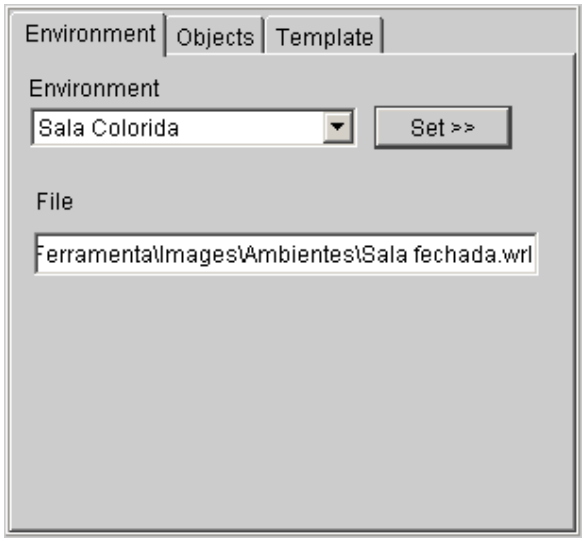
Figura 4-5. Um *link* de aplicação sendo cadastrado.

A opção para se atribuir um novo nome a ambientes, objetos e *links* tem a finalidade de proporcionar o registro desses componentes com nomes que sejam entendidos mais facilmente pelos usuários, conforme o vocabulário com que estejam habituados dentro de um determinado domínio.

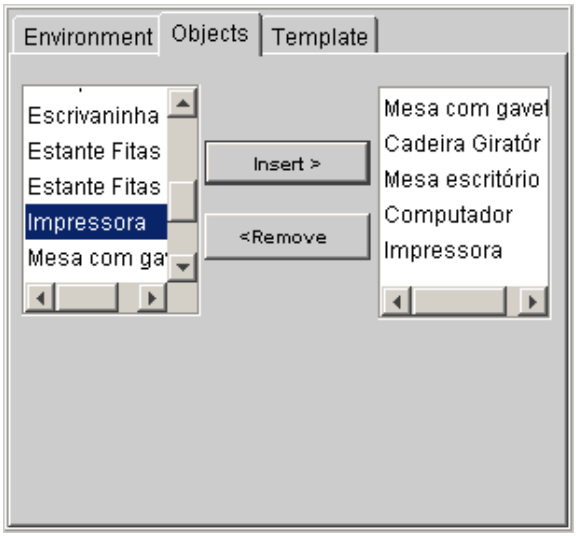
Uma vez que ambientes, objetos e *links* de aplicação estejam cadastrados, é possível iniciar a construção do gabarito para um determinado domínio.

A Figura 4-6 mostra uma ampliação de parte da tela para construção de gabaritos, exibindo, assim, todas as etapas necessárias para a construção do gabarito para o domínio de aplicações para escritório. Na etapa (a), é escolhido um ambiente que represente a sala de um escritório. A escolha é feita entre os diversos ambientes cadastrados e listados na *combobox*.

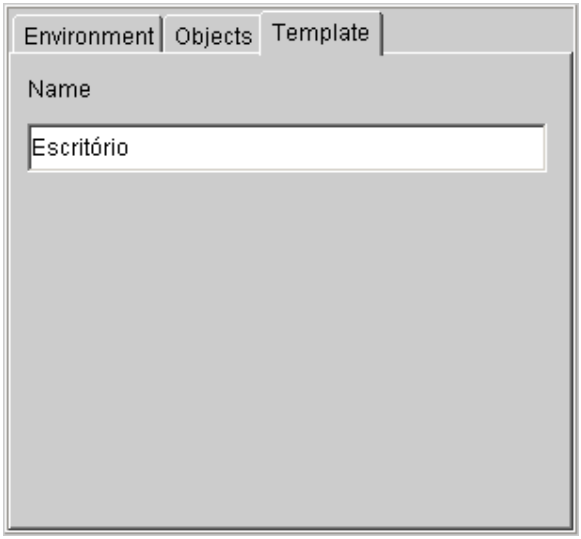
Na etapa (b), são escolhidos todos os objetos que compõem o ambiente. A lista à esquerda exibe todos os objetos cadastrados. Para inserir os objetos em um gabarito, basta que ele seja selecionado nessa lista e, em seguida, que o botão "*Insert*" seja pressionado. Os objetos escolhidos para compor o gabarito são exibidos na lista à direita e, ao mesmo tempo, podem ser visualizados no *browser* da ferramenta. À medida que os objetos são inseridos no ambiente, o usuário pode alterar a sua posição, rotação e orientação. Essa configuração é feita através da manipulação direta dos objetos e define os parâmetros que serão assumidos no instante da construção de interfaces. Caso o usuário decida retirar algum objeto do gabarito, basta que ele selecione o respectivo objeto na lista à direita e pressione o botão *Remove*. Quando o gabarito estiver pronto, um nome deve ser atribuído a ele, conforme mostra a etapa (c) da Figura 4-6.



(a) a escolha de um ambiente



(b) a escolha dos objetos



(c) a atribuição de um nome para o gabarito

Figura 4-6. As etapas para a construção de um gabarito.

A distribuição das etapas necessárias para a construção de um gabarito em painéis diferentes tem a finalidade de facilitar a interação do usuário com o sistema. Isso porque a apresentação gradativa dessas etapas atende a dois princípios destacados por Hix *et al* (1993) que enriquecem o nível da qualidade de interação de uma aplicação: a) conseguir a atenção do usuário ponderadamente e; b) organizar a tela para gerenciar a complexidade. Em relação ao princípio (a), a escolha de implementação apresentada na Figura 4-6 requer a atenção do usuário, em momentos distintos, para as três fases necessárias para a construção do gabarito: escolha do ambiente, escolha dos objetos e, finalmente, a atribuição de um nome ao gabarito. Em relação do princípio (b), a distribuição ordenada das opções evita que elas sejam apresentadas de um única vez e, por isso, tende a diminuir a possibilidade de o usuário se confundir na realização das tarefas.

A Figura 4-7 mostra o gabarito construído. Conforme comentado na subseção 3.3.1, "A estrutura da ferramenta GaCIV", do capítulo anterior, o gabarito pode ser alterado posteriormente, de acordo com as necessidades ou mudanças nos requisitos.

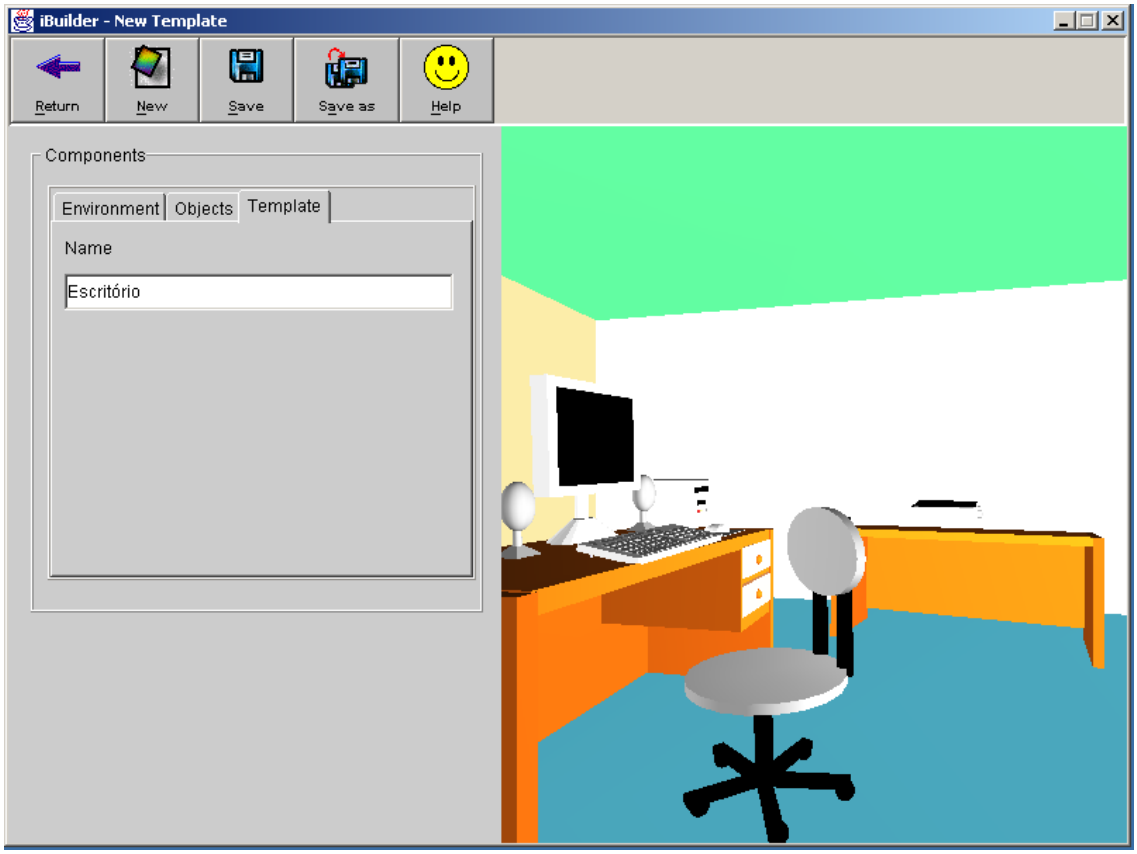


Figura 4-7. O gabarito para a construção de interfaces no domínio de escritórios.

Para construir uma interface, o usuário deve escolher, primeiramente, o gabarito que será o modelo para o projeto. A escolha é feita em uma caixa de diálogo, conforme mostra a Figura 4-8. Após a escolha, é aberta a tela para criação da nova interface. O ambiente é carregado automaticamente, e os objetos que compõem aquele gabarito são listados para que o usuário possa construir a interface.

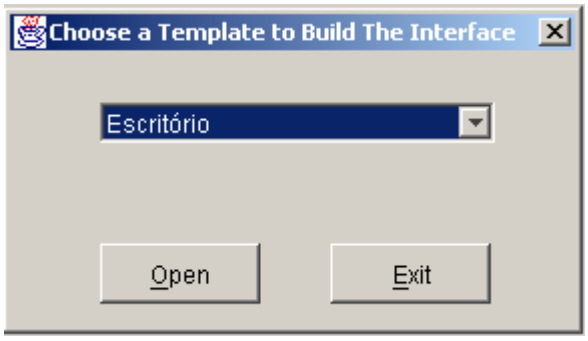


Figura 4-8. Escolha do gabarito a partir do qual será construída uma interface.

A Figura 4-9 mostra a interface construída a partir do gabarito criado anteriormente. É importante ressaltar que a aparência da tela para a construção de interfaces é muito semelhante à tela de construção de gabaritos. Isso atende um outro princípio destacado por Hix *et al* (1993) para se obter mais qualidade no nível de interação de uma aplicação: a aparência de uma tela deve mudar o mínimo possível de uma para outra, facilitando assim a interação humano-computador, pois o usuário se habitua mais facilmente à disposição de botões, opções, etc, e aos caminhos de interação da aplicação. Na Figura 4-9, a impressora está sendo associada a um programa de edição de textos. O computador foi associado a um programa de livro-caixa e todos os outros objetos são apenas decorativos. Após a construção da interface e a associação dos objetos aos respectivos *links* de aplicação, a interface pode ser salva. Toda interface, quando é salva, também é cadastrada automaticamente como um *link* de interface. Isso é feito por dois motivos: 1) toda interface é candidata a participar de diferentes níveis em uma árvore de menu; 2) o registro automático das interfaces como *links* de interface evita que uma nova funcionalidade deva ser disponibilizada na ferramenta.

Para finalizar, o pacote de aplicação pode ser gerado facilmente. Basta que a opção "*Create an Application*" seja ativada. A geração é automática, conforme descrito na subseção 3.3.1, "A estrutura da ferramenta GaCIV", do capítulo anterior. A partir daí, o arquivo *iViewer.bat* pode ser executado, acionando o InterViewer e exibindo a aplicação resultante, conforme é mostrado na Figura 4-10. A mesma figura mostra o editor de textos em execução, após um clique sobre a impressora.

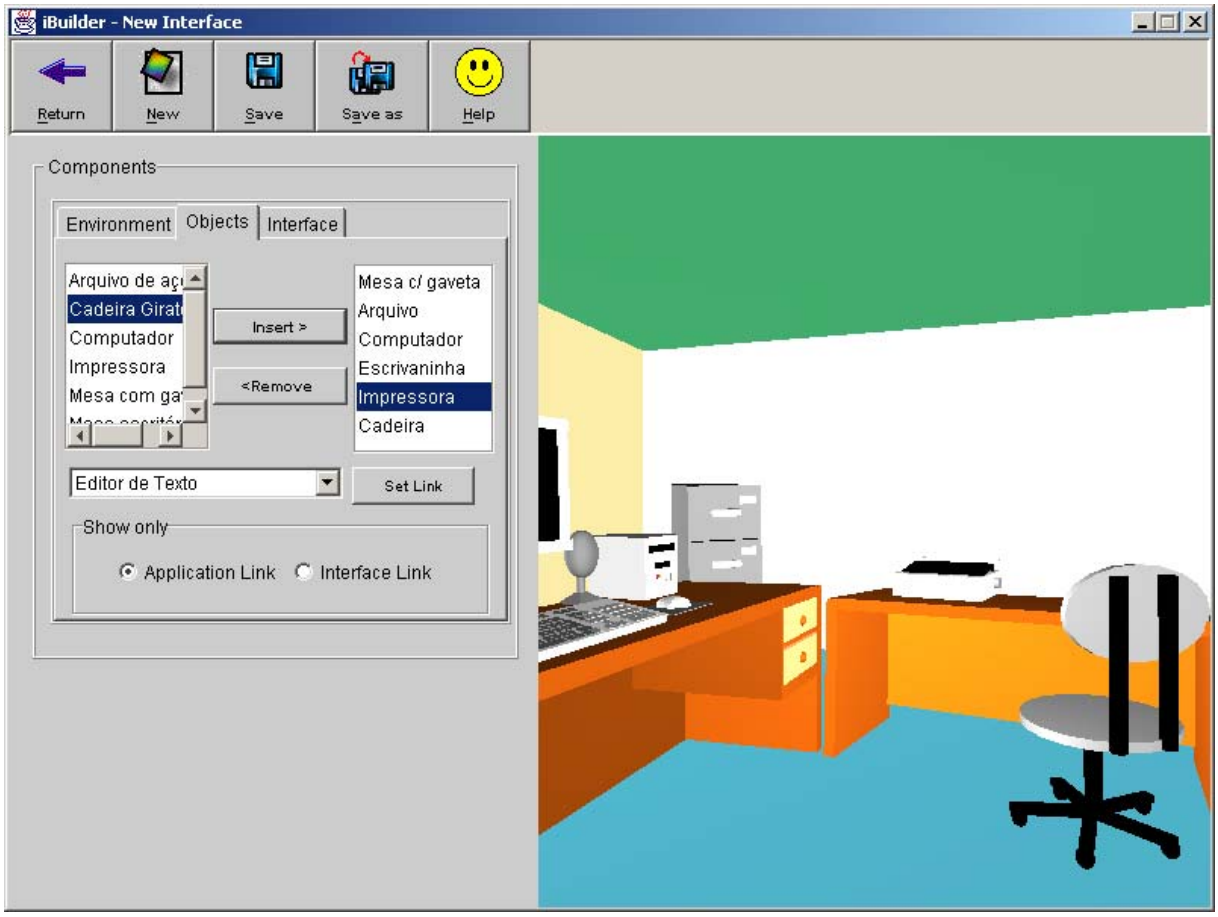


Figura 4-9. A interface para um sistema de escritório.

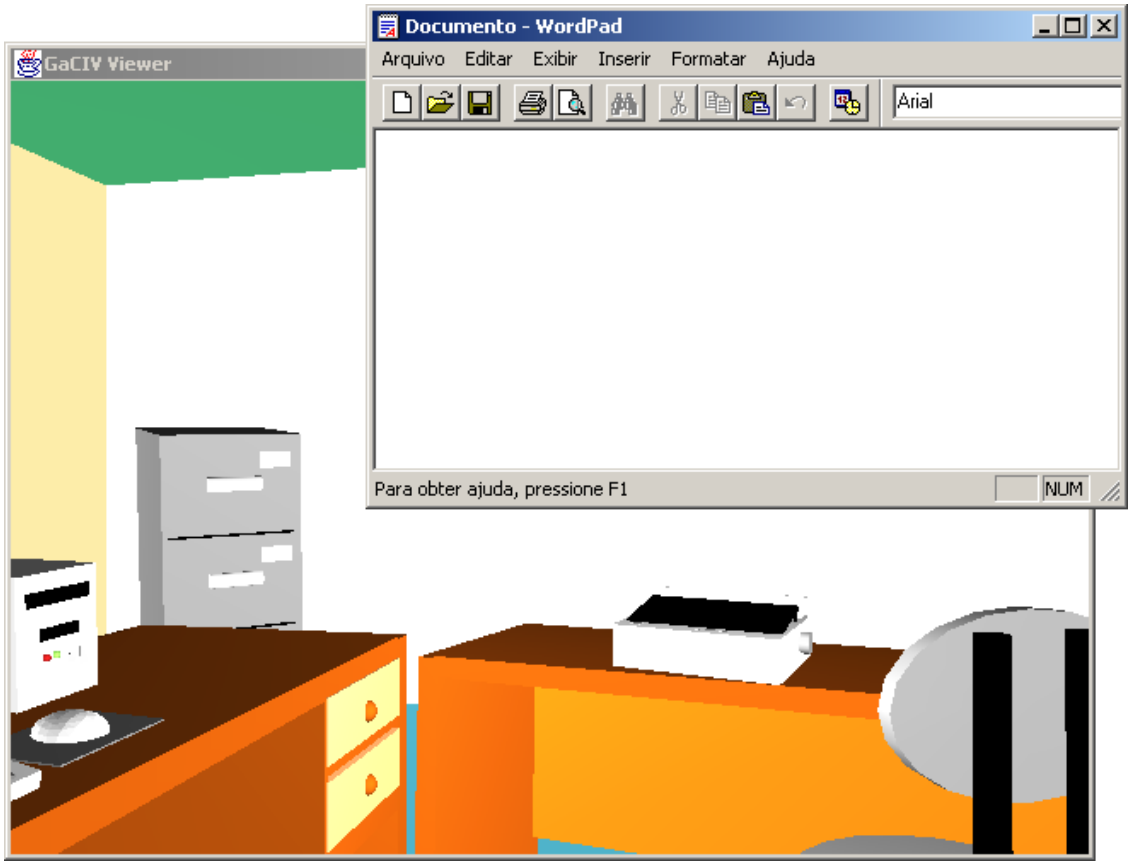


Figura 4-10. A execução do InterViewer e a exibição da interface construída para escritório.

4.3. Estudo de caso

O estudo de caso discutido nesta seção foi realizado em duas etapas e teve a participação de um proprietário de uma locadora de vídeo. Sua experiência com computadores é pequena, sendo que ele basicamente usa o computador para trabalhar com o sistema da locadora.

Antes que o estudo de caso fosse realizado com o uso da ferramenta GaCIV, o usuário foi entrevistado para que os programas e objetos necessários para essa finalidade pudessem ser disponibilizados. Os programas que seriam cadastrados como *links* de aplicações foram desenvolvidos principalmente com base no sistema dessa locadora, considerado pelo usuário um sistema adequado às suas necessidades. De acordo com a funcionalidade, foram desenvolvidos os programas para: cadastro de clientes, cadastro de fitas de vídeo, cadastro de gêneros, cadastro de séries, registro de locações e devoluções. Alguns objetos para esse domínio já estavam disponíveis, como um balcão de atendimento ao cliente, um balcão para realização das locações e outro para as devoluções, além de uma estante com capas coloridas de filmes, representando as estantes consultadas pelos clientes. Outros foram criados a partir da entrevista com o usuário, à medida que ele descrevia o funcionamento da locadora de vídeo, e também a partir da observação da organização das instalações do estabelecimento. Assim, foram criadas uma estante com capas pretas, representando as fitas de vídeo, uma caixa de sapatos com a inscrição “LOC”, representando a caixa onde são depositados os recibos de locação, entre outros. A tentativa de se disponibilizar antecipadamente os possíveis objetos (além daqueles indicados pelo próprio usuário) a partir de uma dedução daquilo que o usuário poderia querer usar é justificada pelo fato de se tentar evitar um desvio do principal objetivo do estudo de caso: analisar a interação do usuário com a ferramenta GaCIV na construção de interfaces e avaliar os possíveis benefícios do envolvimento de usuários nessa etapa de desenvolvimento de software. Por isso, houve a preocupação de evitar que a falta de objetos, não relacionados durante o levantamento de requisitos, impossibilitasse a continuidade do estudo de caso, por não haver a presença de um *designer* gráfico experiente que pudesse disponibilizá-los enquanto o usuário executasse outras atividades.

Sabe-se que, durante a fase de levantamento de requisitos, dificilmente se consegue especificar todas as necessidades de um sistema. Isso requer a escolha de um modelo de processo que considere uma abordagem iterativa, ou seja, um ciclo de desenvolvimento em que o software é refinado gradativamente através de iterações ao longo

das diferentes etapas do processo. Nesse ponto, o GaCIV destaca-se no auxílio ao levantamento de requisitos, conforme foi comentado na subseção 3.3.1, "A estrutura da ferramenta GaCIV", deste trabalho, e conforme se pode verificar a seguir, na descrição e análise do estudo de caso. Isso é possível porque o usuário, à medida que visualiza o resultado da especificação do sistema através do projeto de interfaces com RV, também visualiza mais facilmente as novas necessidades e consegue analisar se os requisitos requerem alterações e melhorias.

4.3.1. A primeira etapa do estudo de caso

Inicialmente, foi dado um treinamento ao usuário quanto ao uso da ferramenta GaCIV. Primeiro, uma demonstração de suas funcionalidades através da construção de um gabarito e a respectiva interface colaborou para que o usuário pudesse compreender a finalidade da ferramenta. Obviamente, o exemplo demonstrado não foi para o domínio de locadoras de vídeo, para que ele não fosse influenciado. Em seguida, para poder se familiarizar com a ferramenta, o usuário a usou durante algum tempo (cerca de 45 minutos) sob orientação, até que se sentisse suficientemente confortável com as funcionalidades.

Após o treinamento, foi sugerida a construção do gabarito para o domínio de uma locadora de vídeo. O usuário preferiu usar a ferramenta GaCIV somente na construção da interface e, por essa razão, a seu pedido, o gabarito foi construído sob a sua orientação e sem a sua participação direta. Nesse ponto, o envolvimento do usuário na construção do gabarito foi muito interessante. O posicionamento de alguns objetos foi alterado, conforme sua orientação. Além disso, o usuário se baseou em sua experiência para inserir no gabarito os principais objetos necessários à construção de uma locadora de vídeo. À medida que o gabarito era construído, sua validação era realizada durante todo o tempo pelo usuário, em relação à aparência do modelo e se todos os objetos necessários eram inseridos. A Figura 4-1 mostra uma visão parcial do gabarito construído, em que é possível visualizar uma estante com as capas de filmes com a face para a frente, uma estante com as capas de filmes viradas de lado, duas mesas, um computador, uma impressora e uma caixa de sapato, onde são colocados os recibos de locação. O usuário solicitou que fossem inseridas as duas estantes com as capas de filmes porque, em determinadas locadoras de vídeo, algumas estantes têm as capas realmente viradas de lado, para que caiba uma quantidade maior delas. Normalmente, isso só é feito com os filmes mais antigos, ressaltou o usuário.

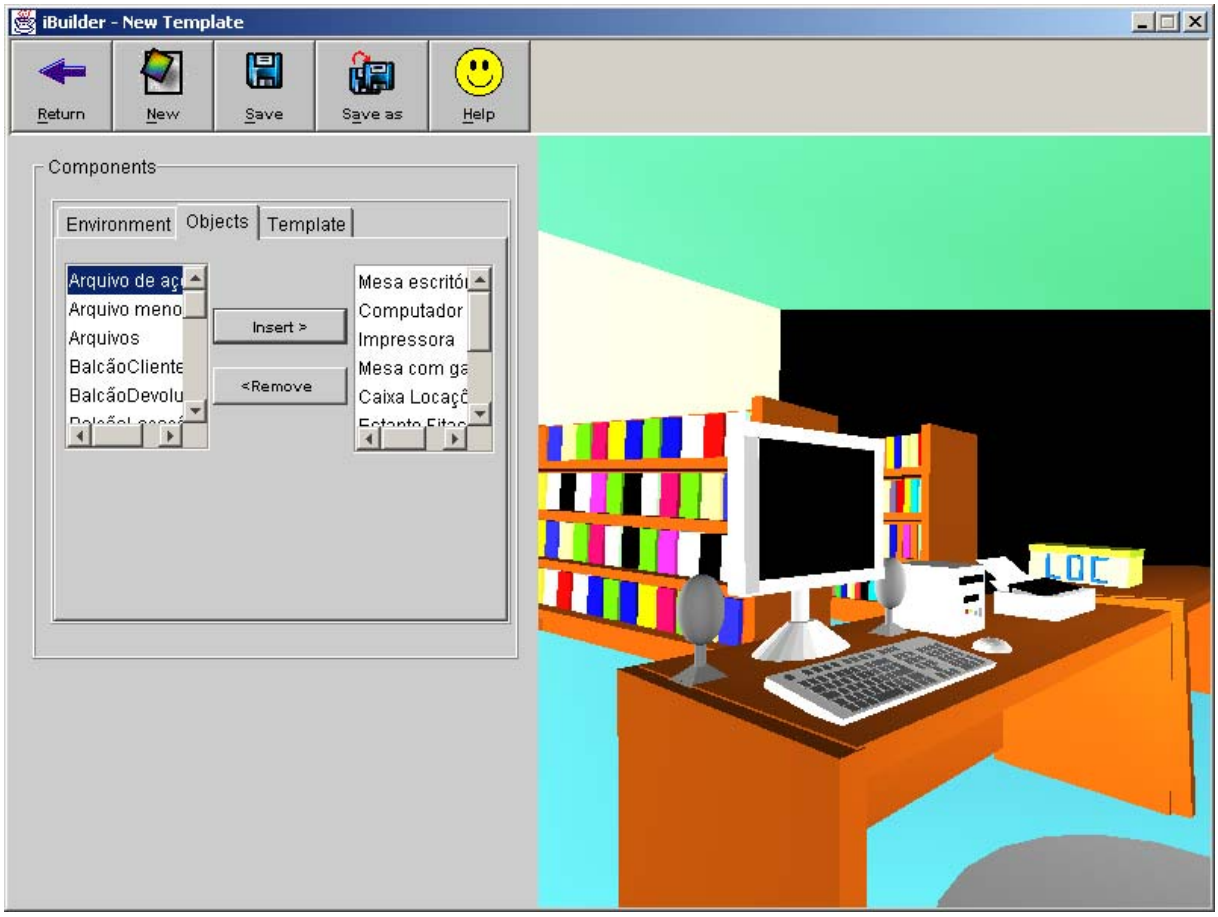


Figura 4-1. Visão parcial do gabarito para locadoras de vídeo.

Terminada a construção do gabarito, de acordo com os requisitos fornecidos pelo usuário, a construção da interface teve início. Dessa vez, o usuário teve uma participação direta, já que ele mesmo é quem construiu a interface. É interessante destacar que o projeto se baseou na organização que o usuário planejou para o novo prédio que está construindo como sede de sua locadora de vídeo e em algumas características do prédio atual como, por exemplo, a quantidade de estantes com as capas dos filmes.

A Figura 4-2 mostra uma visão parcial da interface construída pelo usuário. Cinco estantes com as capas dos filmes com a face para a frente foram inseridas na interface. O usuário não usou a estante com as capas de filmes viradas de lado porque essa é uma situação que não acontece em sua locadora de vídeo. As duas mesas também foram inseridas, uma com o computador e a impressora e a outra apenas com a caixa de sapatos. Essa é a organização que ele deseja para o novo prédio. Além disso, atrás das mesas de atendimento, o usuário inseriu uma outra estante, desta vez para representar, de fato, as fitas de vídeo. Isso porque, em uma locadora, o cliente escolhe o filme que deseja, olhando as capas vazias dos filmes

distribuídas pelas diversas estantes. As fitas propriamente ditas ficam em caixas pretas, entregues aos clientes no momento da locação.

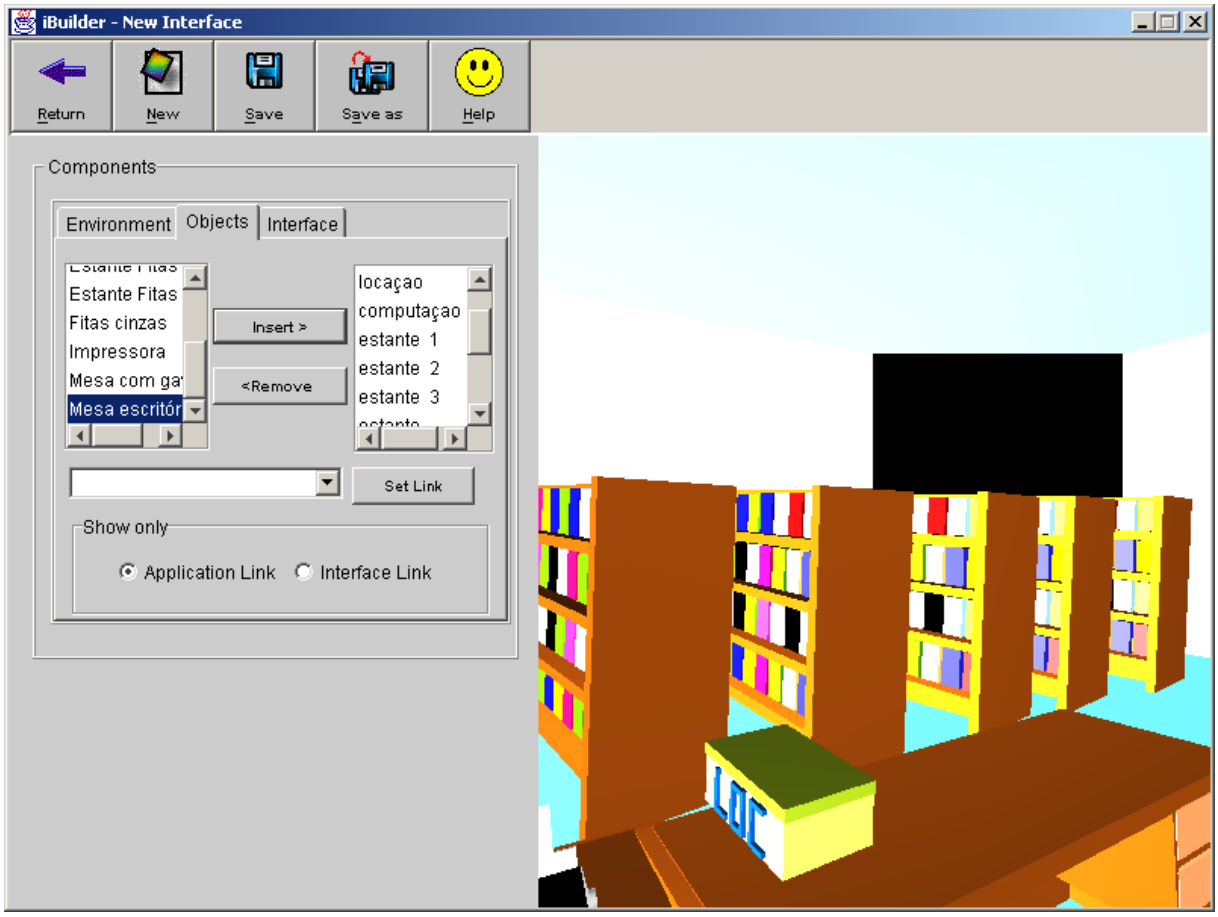


Figura 4-2. A interface construída pelo usuário, com base nas características atuais de sua locadora de vídeo e no projeto de seu novo prédio.

A Figura 4-3 mostra uma outra visão da interface em que é possível visualizar a estante com as fitas. O usuário optou pelo uso de apenas uma estante, porque pretende, no novo prédio, construir estantes embutidas na parede. Por essa razão, considerou que uma única estante seria suficiente para representar o que deseja, já que o gabarito não dispunha de um objeto que representasse uma estante embutida. Na mesma figura, também é possível visualizar um arquivo de aço para guardar as fichas dos clientes.

Depois de construir a interface e organizar os objetos nas posições ideais, o usuário associou os objetos aos respectivos *links* de aplicação. Foi muito interessante perceber a preocupação que ele tinha em associar adequadamente os objetos às aplicações, de tal forma que fosse fácil e intuitiva a localização de cada funcionalidade. A primeira associação foi entre cada uma das estantes com as capas dos filmes com o *link* para o programa de registro das locações e devoluções. Em seguida, foi feita a associação entre a estante com as fitas nas

caixas pretas, que pode ser visualizada na Figura 4-3, com o programa para o cadastro das fitas de vídeo. Os programas para cadastro dos gêneros e das séries (estas últimas determinam o preço de locação de cada fita) eram independentes. O usuário considerou que não havia nenhum objeto que pudesse representar intuitivamente esses programas. Por isso, foi sugerido que, a partir da tela de cadastro das fitas, fosse possível abrir as telas de cadastro de gêneros e séries. Então, dois botões foram inseridos no programa de cadastro de fitas, a partir dos quais é possível executar os outros dois programas. Finalmente, o programa de cadastro de clientes foi associado ao arquivo de aço. Todos os demais objetos ficaram apenas como decoração.

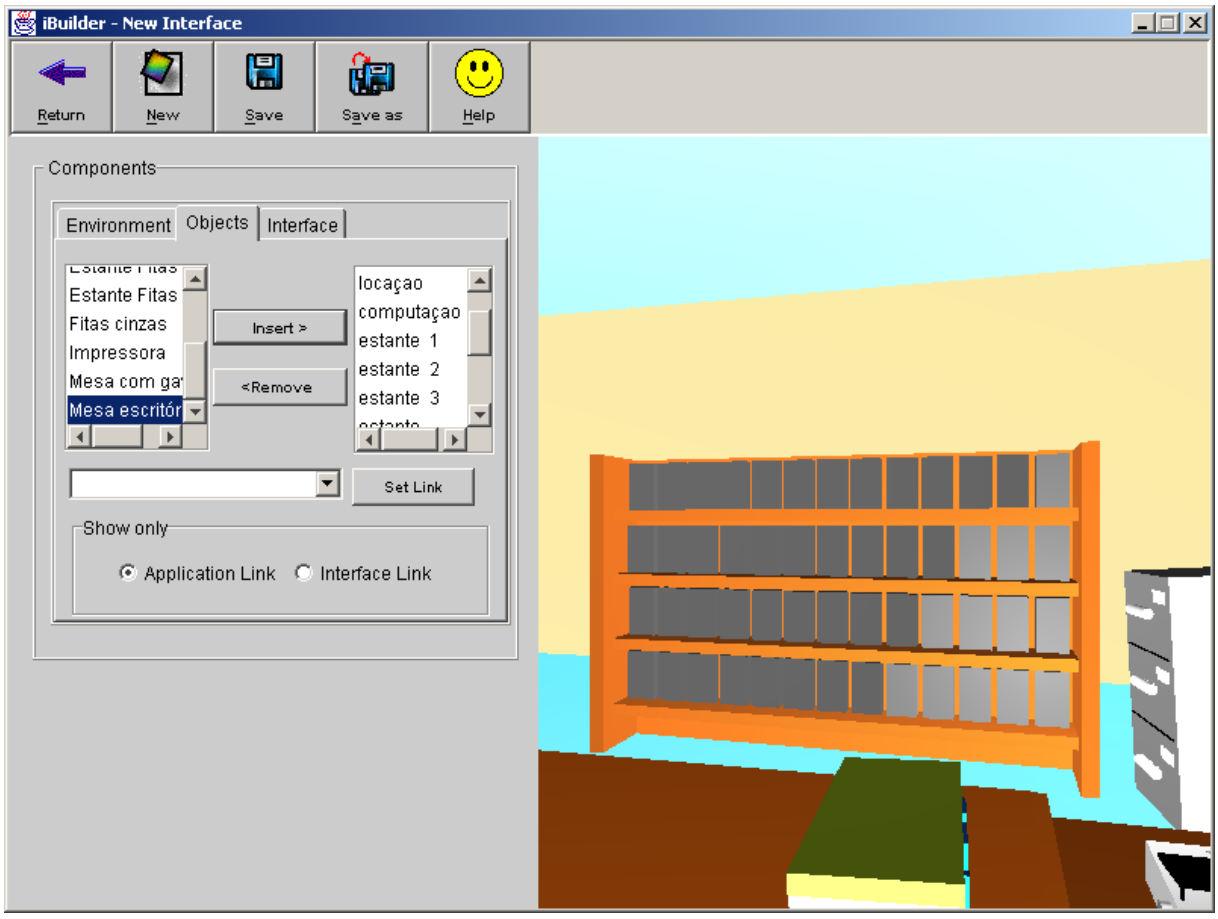


Figura 4-3. Uma outra visão da interface construída pelo usuário.

Terminada a construção da interface, o usuário atribuiu a ela o mesmo nome de sua locadora de vídeo. Em seguida, o pacote de aplicação foi gerado e ele pôde navegar pela aplicação através da execução do InterViewer. A Figura 4-4 mostra o InterViewer em execução.

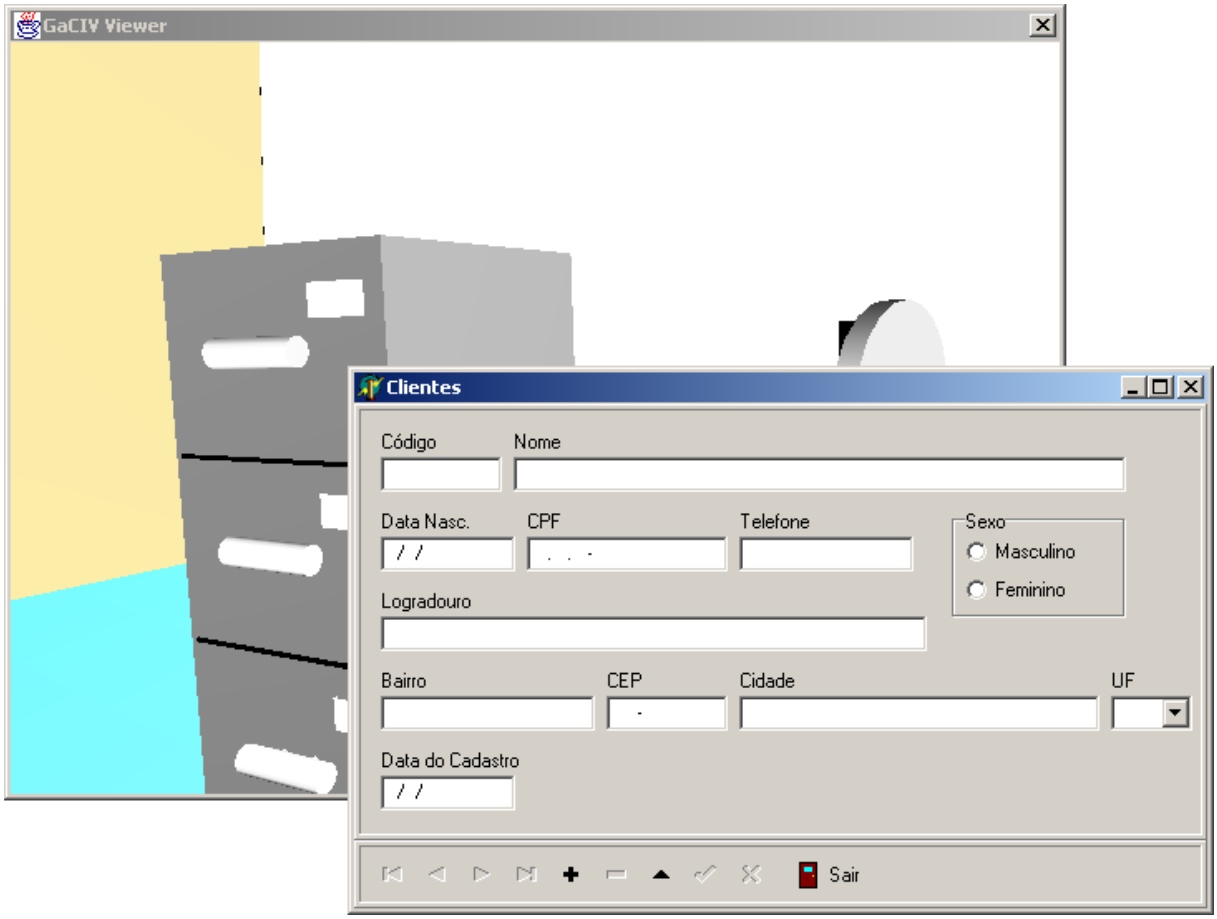


Figura 4-4. A visualização da interface no InterViewer e a execução do programa para cadastro de clientes, acionado através de um clique sobre o arquivo de aço.

O usuário não considerou necessário o uso de *links* de interface. Isso porque, de acordo com ele, a interface construída e os programas associados aos respectivos objetos são suficientes para que uma locadora de vídeo funcione adequadamente, principalmente uma de pequeno porte como a dele. De qualquer forma, é interessante citar que a partir de algum objeto que compõe a interface para a locadora de vídeo, poderia ser associado um *link* para uma outra interface. Conforme foi comentado, toda interface salva é automaticamente registrada como um *link* de interface. Assim, por exemplo, o objeto que representa o computador na interface do sistema para locadora de vídeo e que o usuário denominou "computaçao" (sem acento) poderia ser associado ao *link* de interfaces referente ao escritório criado anteriormente, na seção 4.2, "Um exemplo de aplicação", deste capítulo, para oferecer outras aplicações de apoio a funções administrativas, como editores de texto, calculadora, entre outras. O resultado, durante a execução no InterViewer, é mostrado na Figura 4-5. Nesse exemplo, a interface para escritório é exibida caso seja dado um clique sobre o computador da interface para locadora de vídeo.

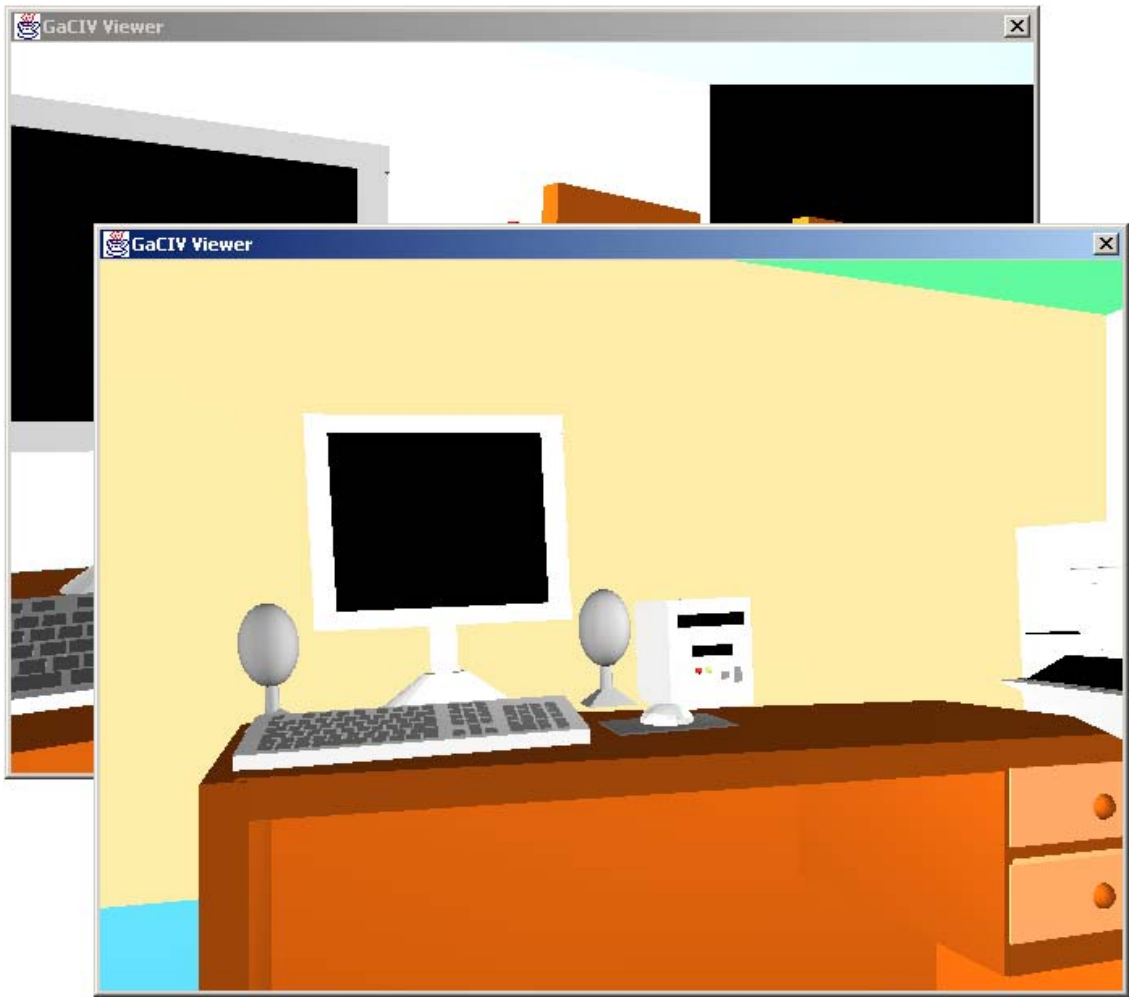


Figura 4-5. A execução de uma árvore de menu no InterViewer.

4.3.2. A segunda etapa do estudo de caso

O usuário foi convidado para a execução de uma segunda etapa do estudo de caso, na qual foi proposto que ele sugerisse a inclusão de novas funcionalidades ao sistema, não contempladas na primeira etapa, e efetuasse as alterações necessárias.

Em uma entrevista que também antecedeu a fase de utilização da ferramenta GaCIV, com o intuito de se levantar novos requisitos e antecipar a disponibilização das novas funcionalidades do sistema, o usuário solicitou a inclusão de cinco programas: um para abertura e fechamento diários do caixa, um para a impressão do relatório periódico de movimentação do caixa (semelhante a um livro-caixa), um para consulta da relação de filmes assistidos por um cliente, um para a manutenção de contas a pagar e outro para a manutenção das contas a receber. Dessa vez, nenhum novo objeto foi modelado. O usuário preferiu

visualizar a interface construída anteriormente para realizar as alterações e sugerir a inclusão de novos objetos, caso fosse necessário.

Antes de o usuário utilizar a ferramenta GaCIV nesta segunda etapa, um breve treinamento, similar ao da primeira etapa, também foi efetuado a fim de reforçar a sua experiência. Em seguida, bem mais à vontade que na primeira etapa do estudo de caso, ele abriu a interface construída naquela ocasião. A partir daí, começou a analisar quais alterações poderia efetuar para inserir as novas funções do sistema. Primeiramente, afirmou que o programa para abertura e fechamento do caixa deveria ser associado à caixa de sapatos sobre uma das mesas, fazendo uma analogia entre a função e o objeto (uma caixa cuja tampa pode ser aberta e fechada). Como essa caixa tem a inscrição "LOC", ele preferiu inserir uma outra no lugar, com a inscrição "CAIXA", já que a primeira tinha apenas uma função decorativa no ambiente. Como essa modificação implicaria na alteração do gabarito, ele a anotou para depois efetuá-la e prosseguiu na tentativa de inserir os demais programas. O programa para impressão do relatório de movimentação do caixa foi associado à impressora, que também já fazia parte da interface. No caso dos programas para manutenção das contas a pagar e das contas a receber, a associação foi feita de acordo com as características do ambiente real. O usuário comentou que, na locadora da qual é proprietário, as contas a pagar e o dinheiro proveniente do pagamento das locações (contas a receber) são depositados nas gavetas de uma das escrivaninhas. Por essa razão, ele considerou que a associação dos respectivos programas ao objeto referente à escrivaninha com duas gavetas seria o ideal, por ser intuitivo. Como essa escrivaninha é um único objeto, não seria possível associar a ela dois programas. Foi sugerido, então, que uma caixa de diálogo fosse exibida com duas opções de acesso – contas a pagar e contas a receber – a fim de que a escolha pudesse ser realizada para a execução do respectivo programa. A maior dificuldade do usuário foi associar o programa para consulta da relação de filmes assistidos por um cliente a algum objeto na interface. Nenhum dos objetos inseridos no ambiente, de acordo com ele, lembrava intuitivamente essa função. Até que ele finalmente sugeriu a inclusão de um objeto que lembrasse uma pessoa (o cliente), porque assim seria fácil associar o objeto à função. Infelizmente nenhum objeto que representasse uma pessoa estava disponível e, por isso, essa alteração foi adiada até que o objeto estivesse disponível.

Depois de definir as alterações, o usuário abriu o gabarito definido para o domínio de locadoras de vídeo para incluir a caixa de sapatos com a inscrição "CAIXA". Esse objeto pôde ser disponibilizado facilmente através da alteração da caixa para os recibos de locação. O usuário, então, inseriu esse novo objeto no gabarito, conforme mostra a Figura 4-1.

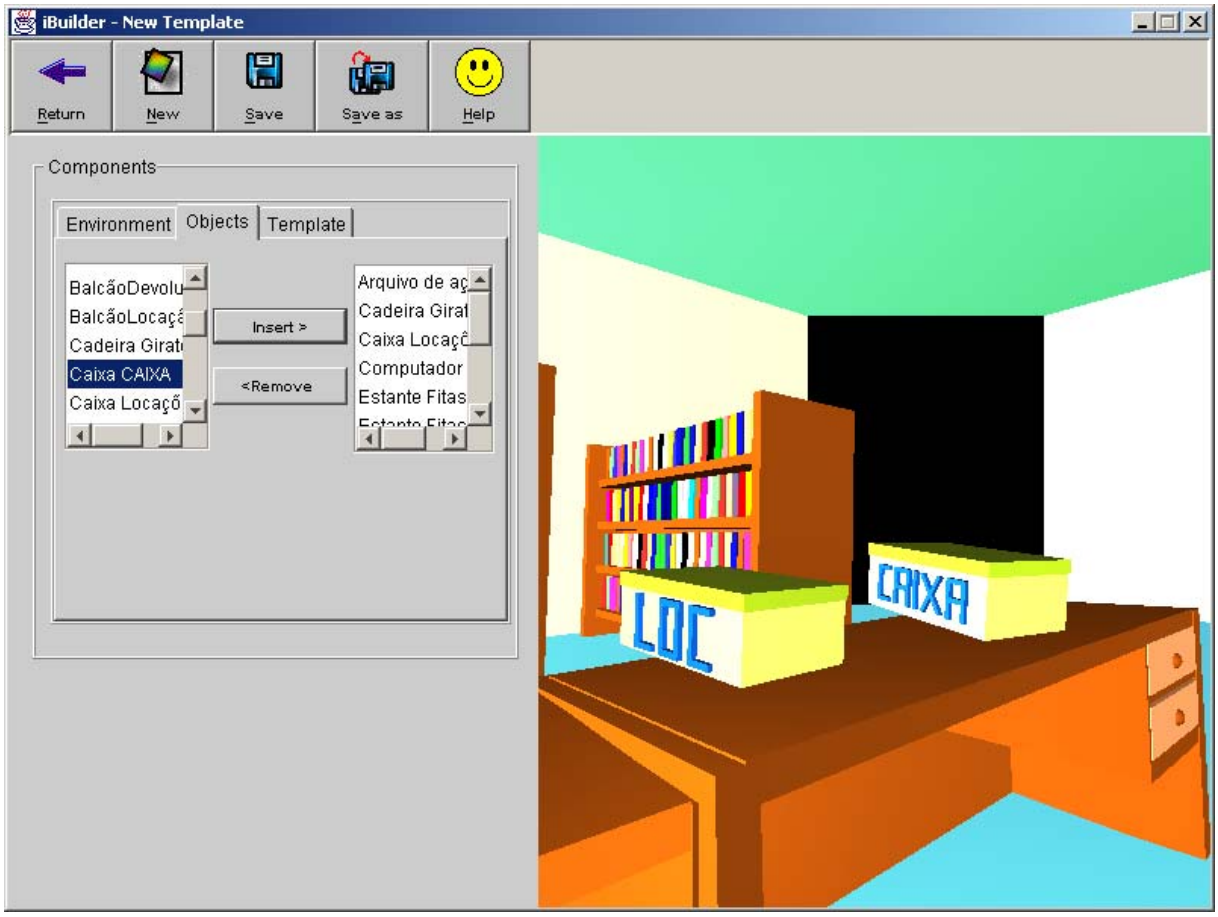


Figura 4-1. A inclusão da caixa com a inscrição "CAIXA" no gabarito para construção de interfaces para locadoras de vídeo.

Essa foi a única alteração necessária no gabarito. Uma outra alteração seria a inclusão do objeto representando uma pessoa, que, conforme explicado anteriormente, não pôde ser modelado durante a realização dessa segunda etapa do estudo de caso, devido à sua maior complexidade. Por isso, o usuário apenas indicou onde o objeto poderia ser posicionado.

Antes que o usuário abrisse a interface da locadora de vídeo, um novo *link* de aplicação foi cadastrado como referência aos programas para manutenção das contas a pagar e das contas a receber. A inclusão da caixa de diálogo com as opções de acesso a um desses programas também pôde ser disponibilizada a tempo, devido à simplicidade de implementação dessa alteração.

Em seguida, o usuário abriu a interface da locadora de vídeo e incluiu a nova caixa com a inscrição "CAIXA". Na seqüência associou esse objeto ao *link* de aplicação referente ao programa de abertura e fechamento de caixa, conforme mostra a Figura 4-2. Depois, associou o objeto correspondente à escrivaninha com duas gavetas ao *link* de aplicação referente aos programas de controle de contas a pagar e contas a receber,

finalizando as alterações, já que o programa para impressão do relatório de movimento de caixa já havia sido associado ao objeto que representa a impressora.

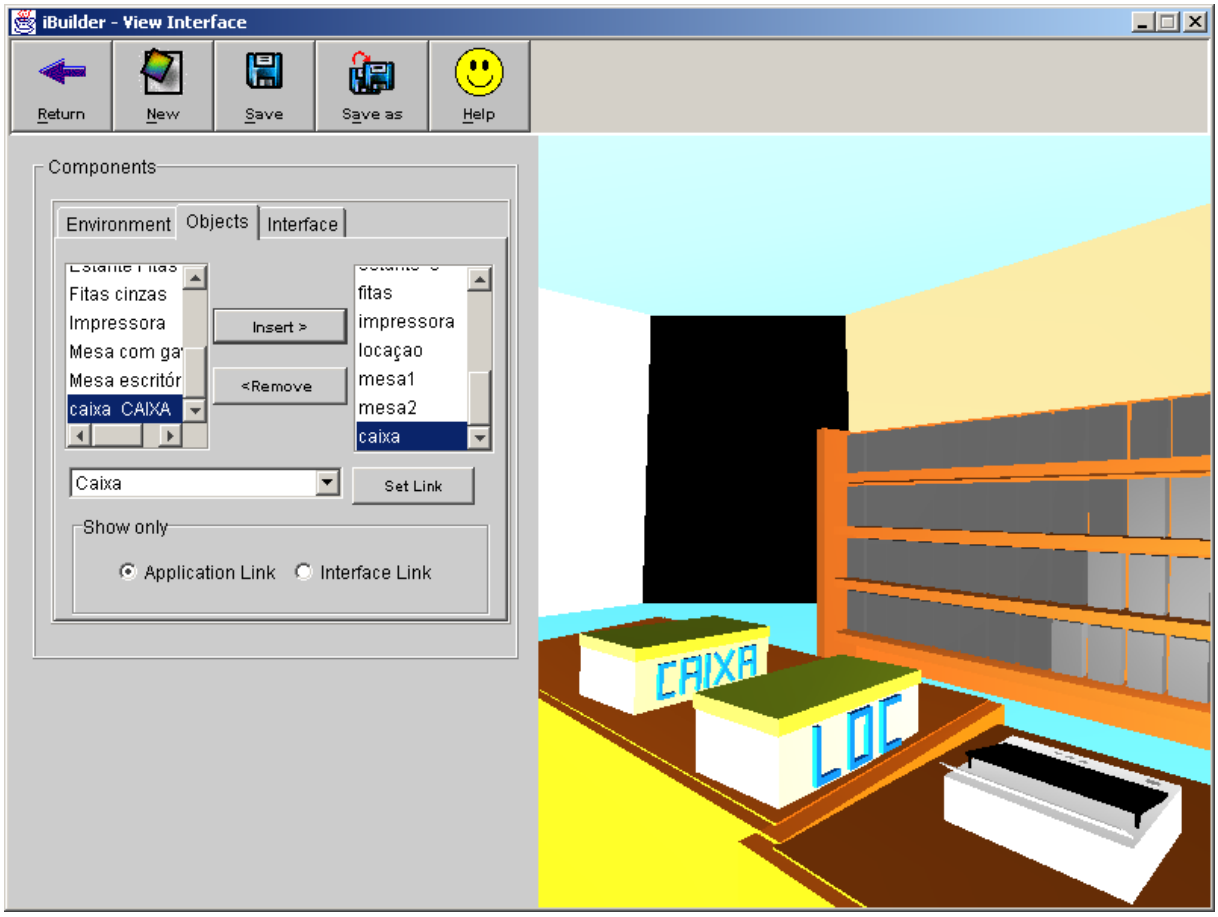


Figura 4-2. A inclusão de um novo objeto na interface e a sua associação a um *link* de aplicação.

4.3.3. Considerações gerais sobre o estudo de caso

Como introdução às considerações apresentadas nesta subseção, é interessante analisar como, de fato, a EUP esteve presente no apoio ao desenvolvimento do sistema com RV pelo usuário. Para essa finalidade, é apresentada a Tabela 4-1, destaque de parte da Tabela 2-1, que compreende justamente a técnica de configuração de parâmetros, abordada neste trabalho.

Tabela 4-1. O Paradigma Paramétrico de EUP

Paradigma	Técnica	Características	Vantagens	Desvantagens
Paramétrico	Configuração de parâmetros	c1. Ativação/desativação de parâmetros predefinidos c2. mudança dos valores para os parâmetros disponíveis c3. seleção, ativação e agrupamento de um conjunto de parâmetros e sua associação a um nome (criação de estilos)	v1. personalização da interface da aplicação v2. criação de estilos (personalizações) que podem ser compartilhados entre usuários	d1. as configurações são restritas às situações previstas pelo projetista da aplicação d2. poucos parâmetros podem resultar em um baixo potencial de extensão d3. muitos parâmetros podem dificultar a previsão dos resultados

Em relação às características (c1) e (c2) da técnica em questão, a ferramenta GaCIV oferece parâmetros cujos valores devem ser definidos tanto para a construção quanto para a configuração de gabaritos e interfaces. Portanto, no projeto de interfaces com RV discutido ao longo deste trabalho, essa técnica oferece não só os meios para a configuração da aparência de gabaritos e interfaces, mas também para a construção desses componentes (no sentido mais amplo da palavra).

Para a construção de gabaritos, estão disponíveis os seguintes parâmetros, conforme ilustra a Figura 4-6, apresentada na seção 4.2, "Um exemplo de aplicação", deste capítulo:

a) escolha do valor correspondente ao nome do ambiente para compor o gabarito;

b) composição da lista de objetos que compõem o gabarito, através da seleção de um objeto disponível no repositório do GaCIV e a sua inserção (ativação) na lista (e, conseqüentemente, no gabarito exibido pelo *browser* do InterBuilder);

c) seleção e remoção (desativação) de elementos da lista de objetos que compõem o gabarito (e, conseqüentemente, a remoção desse objeto do gabarito exibido pelo *browser* do InterBuilder);

d) digitação do valor correspondente ao nome que identifica o gabarito.

Para a construção de interfaces, os parâmetros são praticamente os mesmos, sendo que as diferenças são: a definição do valor correspondente ao nome do ambiente para compor a interface é feita automaticamente, de acordo com o valor do respectivo parâmetro do gabarito; a composição da lista de objetos que compõem a interface é realizada a partir da lista de objetos configurada durante a construção do gabarito e; cada um dos objetos pode ser associado ou a um *link* de aplicação ou a um *link* de interface.

Ainda em relação à construção de gabaritos e interfaces, podem ser configurados os parâmetros referentes à posição, rotação e orientação dos objetos dentro de um ambiente. Essa configuração, no entanto, é realizada exclusivamente através de manipulação direta, uma característica destacada por Shneiderman (2001) como um importante facilitador de interação entre o usuário e o sistema.

O agrupamento desse conjunto de parâmetros resulta na criação de gabaritos e interfaces completas. Por isso, o projeto de interfaces com RV segundo as diretivas embutidas na ferramenta GaCIV (essas diretivas são discutidas no capítulo 5) não são afetadas negativamente pelas desvantagens da técnica de configuração de parâmetros, apresentadas na Tabela 4-1. A restrição de configurações (d1) às situações pertinentes à construção de gabaritos e interfaces e a pequena quantidade de parâmetros (d2) não são aspectos negativos nesse contexto, mas características importantes para facilitar o trabalho do usuário, sem deixar de oferecer a ele as condições necessárias à tarefa proposta. Conseqüentemente, não há a necessidade de um número excessivo de parâmetros, fator que evita a desvantagem (d3).

As vantagens da técnica podem ser percebidas na facilidade de personalização das interfaces (v1), através da manipulação direta dos parâmetros dos objetos dentro de um ambiente. Além disso, a criação de estilos (v2) é caracterizada pela criação de gabaritos configuráveis, que podem ser reusados e refinados por outros usuários.

Uma vez que esses comentários reforçam a discussão apresentada na subseção 3.3.2, “Classificação da ferramenta GaCIV no contexto da EUP”, e demonstram que o GaCIV atende aos requisitos para sua classificação no Paradigma Paramétrico, é possível citar algumas considerações importantes, como resultados do estudo de caso. A análise apresentada a seguir é baseada na observação de todos os passos executados pelo usuário para a construção da interface para a aplicação de locadora de vídeo, de seus comentários durante todo o projeto e do questionário de avaliação, apresentado no Apêndice A deste trabalho.

Em primeiro lugar, é interessante ressaltar que o usuário avaliou satisfatoriamente a ferramenta GaCIV, conforme as notas atribuídas às questões da parte 3 do questionário de avaliação. Essa satisfação está relacionada principalmente à possibilidade de interação com um ambiente virtual que “imita” o mundo real.

Inicialmente, o usuário não estava muito à vontade. A partir do momento em que ele realmente assimilou os objetivos da ferramenta GaCIV, sentiu-se mais confortável e avaliou positivamente a organização das telas e funções para a execução das operações, conforme as respostas atribuídas às questões da parte 4 do questionário de avaliação. Conseqüentemente, segundo ele, a facilidade de aprendizagem também passa a ser um fator

positivo da ferramenta (parte 6 do questionário de avaliação). Desse momento em diante, foi muito interessante observar o entusiasmo do usuário enquanto acompanhava a construção do gabarito, mas principalmente durante a construção da interface. Ao final do estudo de caso, o usuário até sugeriu a inserção de objetos que lembrassem cartazes colados nas paredes da locadora de vídeo. A RV parece ter colaborado eficientemente para o entusiasmo do usuário, porque enquanto construía a interface, ele parecia estar visualizando o espaço do novo prédio da locadora de vídeo, até mesmo porque tinha bastante cuidado para organizar os objetos no ambiente virtual de acordo com aquilo que ele esperava para o ambiente real. Ele comentava constantemente quais eram as melhorias que seriam obtidas, em comparação à organização do prédio atual.

Um outro fator muito interessante é que o usuário considerou a interação com o ambiente virtual mais prática do que a interação com um sistema com menus convencionais, chegando mesmo a afirmar que, para a nova versão do sistema de sua locadora, poderia ser considerada a possibilidade de se construir uma interface com RV. Isso torna possível a afirmação de que o projeto de interfaces é um domínio propício para o uso da RV, porque o uso de analogias pode tornar a interação humano-computador mais intuitiva.

Quanto ao envolvimento ativo do usuário no projeto de interfaces, foi possível notar que o levantamento de requisitos e a avaliação da usabilidade tendem a ser atividades muito beneficiadas. No estudo de caso realizado, o usuário orientava e acompanhava atentamente todo o projeto. Muitos detalhes possivelmente não teriam sido considerados na ausência do usuário. Um exemplo disso é a solicitação que ele fez para que fosse disponibilizada uma estante com caixas pretas, representando as fitas. Até então, só haviam dois objetos que representavam as estantes com as capas coloridas dos filmes. Na questão da usabilidade, conforme já foi comentado, o usuário teve um grande cuidado para: a) organizar o ambiente virtual de acordo com o ambiente real, a fim de facilitar a navegação pelo ambiente; b) associar adequadamente cada programa com um objeto que realmente pudesse proporcionar uma fácil percepção da localização das funcionalidades do sistema.

Embora tenha demonstrado considerável satisfação com a ferramenta GaCIV, o usuário também teve dificuldades durante o uso. Uma delas foi em relação ao movimento de rotação dos objetos. O movimento de rotação padrão oferecido pela API Java 3D é muito flexível, o que acaba sendo uma desvantagem, porque um movimento brusco do mouse pode causar uma rotação indesejável no objeto. Além disso, quando um objeto é inserido no ambiente, dependendo da posição da visão do usuário, esse objeto não é imediatamente visualizado, porque aparece em uma posição diferente da visão do usuário. É necessário,

então, que o usuário gire a sua visão do ambiente, a fim de localizar o objeto. Esses são problemas de ordem técnica que não foram solucionados nessa primeira implementação da nova versão da ferramenta GaCIV, mas que merecem atenção numa próxima fase de melhorias dessa versão, a fim de tornar a usabilidade da ferramenta mais adequada. Mas a usabilidade, de forma geral, foi bem aceita pelo usuário, principalmente porque as telas não são sobrecarregadas de opções e funções e porque, segundo ele próprio, a seqüência lógica para a realização das tarefas é simples, facilitando o aprendizado (parte 4 do questionário de avaliação).

Embora o usuário tenha solicitado que o gabarito fosse criado sem a sua participação direta, isso permitiu analisar o quanto o reuso dos modelos proporcionados pelos gabaritos configuráveis agiliza a construção de interfaces, um aspecto positivo no contexto da EUP, conforme discutido na seção 3.3.1, "A estrutura da ferramenta GaCIV", deste trabalho. Isso também indica que a construção do gabarito é uma atividade que cabe ao projetista de interfaces, embora deva ser realizada com o acompanhamento do usuário, a fim de que ele ofereça as orientações necessárias para que os objetos virtuais sejam adequadamente disponibilizados segundo o domínio de aplicações considerado no projeto. Esse indício é reforçado pela observação de que o usuário se sentiu mais à vontade para construir a interface justamente porque já contava com o gabarito, ou seja, um modelo semipronto que ofereceu mais facilidade e agilidade à sua tarefa.

Na segunda etapa do estudo de caso, a principal verificação foi em relação à prototipação, ou seja, a possibilidade de construção de gabaritos e interfaces de uma forma iterativa, contínua, à medida que novos requisitos precisam ser incorporados ao sistema e novos objetos precisam ser modelados. Além disso, ainda foi possível verificar a independência de diálogo, aspecto também discutido na seção 3.3.1, "A estrutura da ferramenta GaCIV", deste trabalho. Em razão dessa característica, os programas executáveis podem ser desenvolvidos paralelamente ao projeto de interfaces. Foi o que aconteceu nessa etapa: a criação de uma caixa de diálogo com as opções de execução ou do programa para controle das contas a pagar ou das contas a receber foi realizada enquanto o usuário efetuava alterações no gabarito para interfaces de locadoras de vídeo.

Apesar da inclusão de novas funcionalidades ao sistema, o usuário não considerou necessária a utilização de *links* de interface. Ficou claro que esse mecanismo não foi utilizado porque no ambiente real da locadora de vídeo não existe, por exemplo, uma sala separada da área de atendimento ao cliente para as funções exclusivamente administrativas. Além disso,

mesmo com os novos programas, o número de funcionalidades não se tornou expressivo a ponto de sobrecarregar o ambiente virtual.

De um modo geral, pode-se afirmar que a satisfação do usuário justificou os esforços para a concretização dos objetivos deste trabalho. Conforme era esperado, a RV destacou-se como um importante diferencial para o envolvimento do usuário no projeto de interfaces, principalmente pelo estímulo visual, permitindo a exibição de um ambiente virtual com características semelhantes ao ambiente real. Foi interessante observar o quanto o usuário se sentiu motivado ao navegar pelo sistema através do InterViewer. Por diversas vezes, ele destacou a interação mais intuitiva proporcionada pela RV e conseqüentemente a facilidade de localização das funcionalidades do sistema. A Figura 4-1 mostra a execução do sistema gerado pelo usuário. Essa figura, no entanto, exibe inclusive uma alteração feita posteriormente ao estudo de caso, sem a participação do usuário, porém de acordo com sua orientação: a inserção de um objeto representando o cliente, ao qual foi associado o programa para exibição da relação dos filmes assistidos pelo cliente. É interessante ressaltar que, caso o usuário não concordasse com a posição do cliente dentro do ambiente, ele poderia alterá-la facilmente.

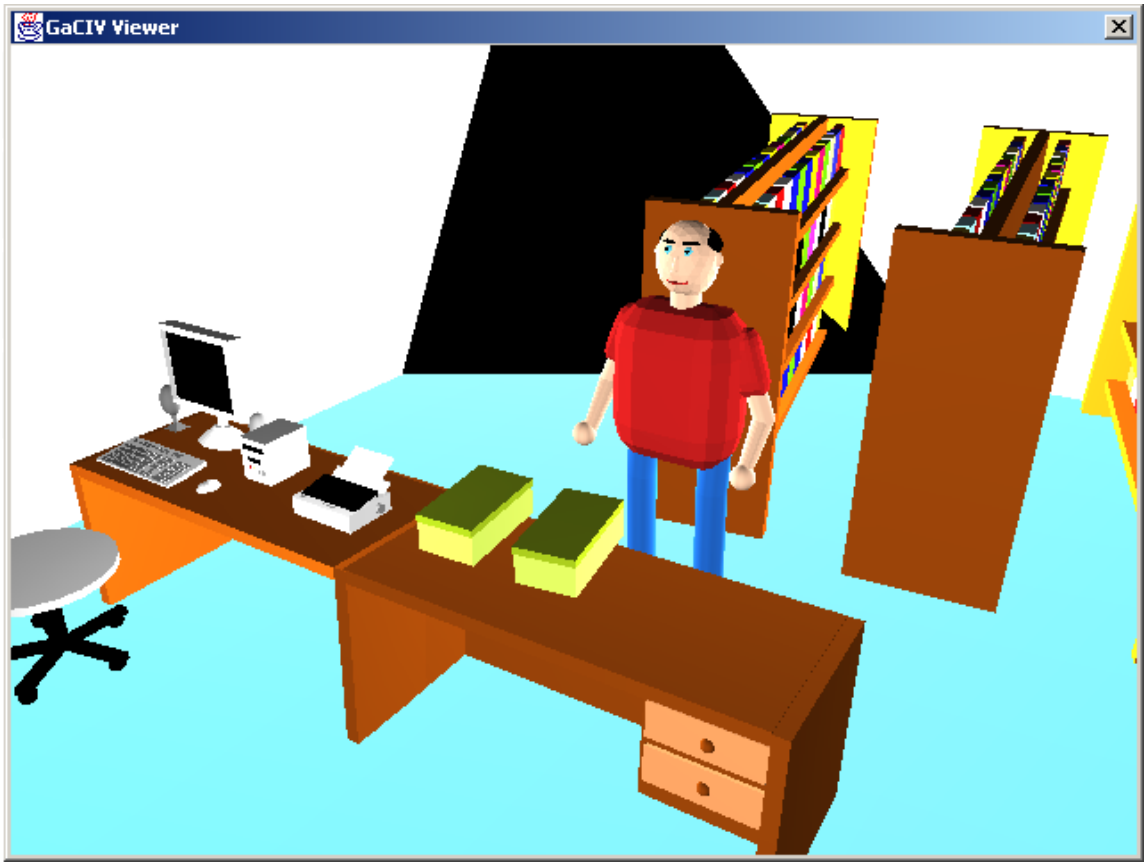


Figura 4-1. O sistema completo para locadora de vídeos em execução.

4.4. Considerações finais

Neste capítulo foram apresentados um exemplo de projeto de interfaces na ferramenta GaCIV e um estudo de caso que teve a participação de um usuário, proprietário de uma locadora de vídeo. O estudo de caso auxiliou na identificação de algumas considerações que validam a proposta deste trabalho: envolver o usuário ativamente no projeto de interfaces, por meio de ferramentas CASE apoiadas pela EUP. Além disso, a RV também mostrou ser um fator estimulante ao envolvimento do usuário e na sua satisfação quanto aos resultados obtidos, principalmente ao se considerar que a interação com o sistema pode ficar mais agradável e intuitiva.

No próximo capítulo, são apresentadas as conclusões finais deste trabalho e algumas propostas para trabalhos futuros.

5. Conclusões

5.1. Considerações iniciais

Em poucas palavras, pode-se dizer que a proposta apresentada na introdução deste trabalho (capítulo 1) contempla o envolvimento efetivo do usuário no desenvolvimento de sistemas com RV, por meio da ferramenta CASE GaCIV. Sustentado principalmente por duas importantes áreas de pesquisa – A EUP e a RV, esse objetivo foi alcançado satisfatoriamente, sendo que os resultados obtidos se mostram suficientes para oferecer diretivas de orientação a outros trabalhos de pesquisa com objetivos afins.

Neste capítulo são apresentadas as principais conclusões deste trabalho, que podem ser comentadas de acordo com os estudos realizados e a partir dos esforços para a adaptação da ferramenta GaCIV no contexto da EUP.

Na seção 5.2 e subseções, são apresentados os resultados obtidos com o trabalho, as principais contribuições e publicações. Na seção 5.3 e subseções são apresentados alguns problemas a serem superados, relacionados à implementação e usabilidade. Finalmente, na seção 5.4 são apresentadas sugestões para trabalhos futuros.

5.2. Resultados obtidos

Normalmente, os usuários não são vistos com a devida importância pelos profissionais de computação e, por isso, estes não dão àqueles a chance de terem um envolvimento completo nas atividades de desenvolvimento de software, contribuindo para a orientação de todo o processo a partir de suas necessidades básicas [Leonard'2002]. Apesar das pesquisas de IHC estarem sendo cada vez mais difundidas e aceitas, ainda existe bastante resistência quanto à real importância de se dar ao usuário a atenção ideal, a fim de garantir maior qualidade dos sistemas, conforme afirmam vários trabalhos compilados por Gulliksen *et al* (1999). O uso de técnicas de EUP em uma ferramenta CASE para apoiar o envolvimento efetivo do usuário no projeto de interfaces é uma das principais contribuições deste trabalho. A participação ativa do usuário oferece a ele uma interessante oportunidade de poder construir a interface para suas aplicações de acordo com a sua visão do domínio em questão. Dessa

forma, as funcionalidades da aplicação podem ser distribuídas da maneira mais adequada para ele, e os caminhos de navegação podem ser determinados com mais naturalidade. A interação com o sistema tende, portanto, a ganhar maior conformidade com as expectativas do usuário.

A RV, aliás, pode ser vista como um fator muito interessante no projeto de interfaces, o que constitui mais uma contribuição deste trabalho. O motivo é que, conforme comentado na subseção 2.3.1, "A RV e a EUP: em direção aos mesmos objetivos", deste trabalho, nem sempre a RV pode garantir um nível de usabilidade adequado aos sistemas, dependendo do domínio de aplicação. Através do estudo de caso apresentado no capítulo anterior, foi possível observar a motivação do usuário durante a construção de um ambiente virtual que permitiu a visualização não só de características do ambiente real em que ele trabalha, mas também do ambiente que ele projeta para o novo prédio da locadora de vídeo. Uma outra observação, apoiada por comentários do usuário, também deve ser feita em relação à interação com o ambiente tridimensional: a analogia com o mundo real facilita a navegação, e quanto mais semelhante for a organização do ambiente virtual em relação ao ambiente real, maior poderá ser essa facilidade.

Dos esforços para ocultar as dificuldades do uso da RV através da ferramenta GaCIV, destaca-se mais uma contribuição deste trabalho, ao prover uma ferramenta CASE que manipula os recursos de RV de maneira acessível a engenheiros de software, projetistas de interface e usuários.

Finalmente, é interessante ressaltar que este trabalho colabora com um desafio ainda pouco explorado em pesquisas: a integração da EUP com recursos de RV, procurando contribuir com a aproximação dessas duas promissoras áreas de pesquisa. Essa integração pode, conforme analisado ao longo deste trabalho, aumentar de forma eficiente as potencialidades do usuário na interação com o computador.

5.2.1. Diretivas de projeto e implementação

Como visto, não só o envolvimento do usuário é importante, como também os mecanismos que são oferecidos para que sua atuação seja a mais proveitosa possível. Assim, algumas idéias validadas através da ferramenta GaCIV destacam-se como diretivas interessantes para a disponibilização de ferramentas CASE para o usuário. São elas:

- O uso dos recursos da RV como um fator de contribuição à usabilidade do sistema e também para a motivação do usuário;

- O uso da RV não-imersiva, possibilitando que os recursos da computação gráfica sejam acessíveis a um número bem maior de usuários, pelo fato de não impor restrições quanto ao uso dos computadores pessoais;
- O uso de técnicas da EUP, como meio de oferecer mecanismos mais simples para a execução das tarefas propostas pela ferramenta através de uma interação intuitiva do usuário com o sistema;
- O uso de gabaritos configuráveis como um repositório de modelos semiprontos para reutilização. Isso agiliza a construção de interfaces e o usuário pode visualizar os resultados mais rapidamente. Conseqüentemente, o conceito de gabaritos configuráveis é um diferencial para convencer o usuário a participar ativamente do projeto de interfaces, justamente porque o tempo exigido para essa atividade passa a ser menor;
- A abordagem de prototipação, que permite que o projeto de interfaces seja realizado gradativamente, em ciclos iterativos. A cada ciclo, o usuário pode analisar os resultados e propor melhorias, e assim sucessivamente, até que a interação com o sistema atenda, de fato, às suas necessidades. Além disso, alterações nos requisitos do sistema podem ser acomodadas mais facilmente;
- A necessidade de se configurar poucas opções de parâmetros para a construção de gabaritos e interfaces. Essa é uma característica possível de se implementar no projeto de interfaces com RV apoiada pela ferramenta GaCIV, porque uma pequena quantidade de parâmetros é suficiente para a construção de interfaces completas. Portanto, quanto menor for a quantidade de parâmetros e combinações que deve ser configurada, melhor será a usabilidade da aplicação. No entanto, quando se tem a necessidade de uso de um número expressivo de parâmetros, é importante que a aplicação ofereça a visualização adequada dos efeitos das escolhas e modificação das combinações e valores informados. Assim, o usuário estará ciente do efeito da configuração de cada parâmetro ou conjunto de parâmetros e, por isso, possíveis problemas de usabilidade também podem ser evitados;
- A organização das funções em tarefas mínimas e o mais logicamente distribuídas possível. Essa característica se refere à divisão da construção dos gabaritos e interfaces em partes menores: a escolha do ambiente, a escolha dos objetos para compor um gabarito ou uma interface, a associação dos objetos de uma interface a

links de aplicação ou de interface e, por fim, a atribuição de um nome que identifique o gabarito ou a interface construída. Isso permite que o foco de atenção do usuário esteja em atividades específicas para atingir o objetivo maior.

5.2.2. Publicações

Durante a realização deste trabalho, foram obtidas duas publicações nacionais e duas internacionais, além da participação, na sessão de painéis, em um congresso de pós-graduação. A lista a seguir apresenta a relação de publicações:

- Balbino, F.C., Silva, J.C.A., Penteado, R.A.D. O Desafio de Prover Ambientes para EUP e Realidade Virtual com Usabilidade. In *Proceedings of IHC 2002*, p. 351-354. Fortaleza, Ceará, Brasil, Outubro 2002. /artigo resumido/
- Balbino, F.C., Silva, J.C.A., Penteado, R.A.D. *End-User Programming como Apoio ao Desenvolvimento de Sistemas com Realidade Virtual*. II Congresso de Pós-Graduação da UFSCar, Sessão de Painéis, São Carlos, Brasil, Setembro 2003.
- Balbino, F.C., Silva, J.C.A., Penteado, R.A.D. GaCIV: apoiando a construção de interfaces com Realidade Virtual por usuários finais. In *XXIX Latin-American Conference on Informatics (CLEI)*. Proceedings. CD-ROM. La Paz, Bolivia, 29 de Setembro a 2 de Outubro de 2003.
- Albertin, J.C.L., Balbino, F.C., Silva, J.C.A., Penteado, R.A.D. Apoio ao Desenvolvimento de Sistemas com Realidade Virtual baseado em reuso de Componentes. In *XVII Simpósio Brasileiro de Engenharia de Software (SBES), Sessão de Ferramentas, Anais – Caderno de Ferramentas*, p. 43-48. Manaus, Amazônia, Brasil, Outubro 2003.
- Balbino, F.C., Albertin, J.C.L., Silva, J.C.A., Penteado, R.A.D. End-User Programming Aided by a CASE Tool for Designing Interfaces with Virtual Reality. In *III Workshop on Software Engineering, Chilean Computing Week*. Chillán, Chile, Novembro 2003.

5.3. Dificuldades a serem superadas

O estudo de caso apresentado no capítulo 4, além de fornecer subsídios para a análise dos resultados deste trabalho, também auxiliou na observação de dificuldades que ainda devem ser superadas para que o protótipo da ferramenta GaCIV ofereça melhores condições para a construção de interfaces com RV. Isso é importante principalmente pela proposta de envolvimento efetivo do usuário no projeto de interfaces. A seguir, são comentadas as dificuldades encontradas durante o estudo de caso e outros testes realizados ao longo da implementação.

5.3.1. Dificuldades relacionadas à implementação

O problema maior relacionado à implementação está no tratamento de erros. Apesar de ao longo do código de programação existir o tratamento de exceções, as mensagens de erros não são exibidas em caixas de diálogo, mas na tela de *prompt* do MS-DOS, a partir da qual as aplicações Java são executadas. Além disso, não houve o tratamento específico de exceções, para que as mensagens de erro pudessem esclarecer, de fato, o erro específico ocorrido.

Um outro problema está relacionado às classes responsáveis pela carga dos objetos 3D (denominadas *loaders*). Os *loaders* são conjuntos de classes implementadas por terceiros, e apresentam alguns erros durante o carregamento dos objetos, muitas vezes inesperados. Objetos do mesmo tipo (VRML, por exemplo) algumas vezes não são carregados, ou se carregados uma vez, um erro é gerado na tentativa de carregá-los uma segunda vez. Esse problema foi percebido com alguns objetos, modelados em um mesmo software para modelagem 3D.

5.3.2. Dificuldades relacionadas à usabilidade

O principal problema relacionado à usabilidade, inclusive de acordo com o usuário que participou do estudo de caso, refere-se ao movimento de rotação dos objetos. Conforme foi comentado na subseção 4.3.3, “Considerações gerais sobre o estudo de caso”, no capítulo anterior, um movimento brusco do mouse pode provocar uma rotação indesejada

do objeto. Uma possível solução para esse problema será limitar a rotação dos objetos aos eixos X (movimento circular para direita e para esquerda) e Y (movimento circular para cima e para baixo).

Uma outra melhoria que certamente beneficiará a usabilidade refere-se à ação de exclusão de objetos de um gabarito ou uma interface ou a sua associação com um *link* de aplicação ou um *link* de interface. Para essas duas ações, é necessário selecionar o objeto correspondente. Por enquanto, a seleção só pode ser realizada na lista com o nome dos objetos. No entanto, é viável que essa seleção possa ser feita no próprio *browser* de exibição do ambiente virtual. Dessa forma, o usuário poderá selecionar o objeto desejado sem precisar se lembrar do nome atribuído a ele, principalmente nos casos em que um mesmo objeto é utilizado mais de uma vez em uma mesma interface.

Os termos em inglês também foram apontados pelo usuário como um problema no uso da ferramenta (conforme as respostas atribuídas às questões da parte 5 do questionário de avaliação no Apêndice A), principalmente quando as pessoas não têm um conhecimento suficiente desse idioma. Nesse caso, uma melhoria no protótipo da ferramenta seria embutir uma opção de configuração do idioma. Um arquivo texto, por exemplo, pode armazenar uma lista de variáveis, correspondentes aos diversos elementos da interface da ferramenta, como botões e títulos das janelas, e as respectivas mensagens em um idioma específico. Durante o carregamento do programa, o arquivo de texto correspondente pode ser lido a fim de configurar as mensagens de acordo com a opção de idioma do usuário.

É interessante ressaltar que uma avaliação de usabilidade formal deve ser realizada para que as questões dessa ordem possam ser melhor analisadas.

5.4. Trabalhos futuros

Com base nos resultados obtidos e nas pesquisas realizadas ao longo deste trabalho, surgem algumas propostas para trabalhos futuros, comentados a seguir:

- **Adequação da ferramenta GaCIV ao Paradigma Imitativo:** conforme comentado na subseção 3.3.3, “Em direção ao Paradigma Imitativo”, a possibilidade de se embutir na ferramenta GaCIV os mecanismos da PBD surgem como resultados de outro trabalho [Albertin’2002] desenvolvido do DC-UFScar, em que os atuais objetos são substituídos por componentes 3D, ou seja, um conjunto composto por um objeto 3D e suas funcionalidades. Com isso, podem ser implementados mecanismos

que permitam ao usuário criar novos comportamentos para os componentes registrados no repositório da ferramenta GaCIV;

- **Realização de um conjunto de experimentos para avaliação formal de usabilidade do projeto de interfaces para aplicações de diferentes domínios com envolvimento efetivo do usuário:** neste trabalho, foi realizado um estudo de caso com um único usuário, proprietário de uma locadora de vídeo. Um conjunto de experimentos pode ser realizado considerando-se domínios diferentes, para verificação formal dos benefícios, problemas e desafios relacionados ao projeto de sistema com RV, através da aplicação de métodos de avaliação de usabilidade. Desse estudo, podem emergir diretivas para a aplicação de um método que direcione projetos de interface com RV e que assegure uma participação realmente colaborativa do usuário;
- **Adaptação da ferramenta GaCIV para utilização na Web:** a popularização da Internet fez com que o desenvolvimento de software passasse a ter uma grande concentração para aproveitamento dos recursos da Web. Além disso, pessoas com uma enorme diversidade de perfis, gostos, culturas, níveis de instrução, também passaram a ter o acesso à informação disponibilizada na Web. Por essa razão, a usabilidade é um fator de extrema importância nesse contexto, já que tem como objetivo atender às pessoas com as mais diferentes expectativas. Assim, um estudo dos benefícios do uso da RV no projeto de interfaces discutidos neste trabalho podem ser estendidos à Web. Com isso, outras perspectivas de extensão das potencialidades da ferramenta GaCIV podem surgir como, por exemplo, o projeto de sistemas com RV em um ambiente distribuído;
- **Construção de um editor de objetos 3D na ferramenta GaCIV:** todos os objetos registrados no repositório da ferramenta GaCIV devem ser construídos com o auxílio de outros softwares específicos para modelagem de modelos 3D. A construção de um editor de objetos 3D próprio pode viabilizar a possibilidade de que o projeto de interface com RV seja realizada completamente com a ferramenta GaCIV. É um trabalho complexo, que deve contemplar principalmente mecanismos que tornem a modelagem 3D uma tarefa mais simples, não exigindo um esforço tão grande para essa tarefa como muitos softwares comerciais disponíveis no mercado.

APÊNDICE A

A seguir, é apresentado o questionário de avaliação da satisfação do usuário em relação à ferramenta GaCIV, aplicado no final da segunda etapa do estudo de caso, apresentado no capítulo 4 deste trabalho. Para cada questão, o usuário tinha que atribuir uma nota no intervalo de 1 a 9 (o significado dos valores é dado, entre parênteses, à frente do espaço para a resposta). Quando a questão não se aplica à ferramenta, a resposta é assinalada pelas iniciais NA (não se aplica).

Questionário para avaliação da satisfação do usuário

Nome: José da Silva

Sistema: GaCIV

Idade: 42

Curso: -

Ano de ingresso: -

Sexo: ☒ Masculino ☐ Feminino

Parte 1: Experiência com o sistema

1.1. Quanto tempo você já trabalhou com o sistema? 5 horas ~ 2 dias

1.2. Na média, quanto tempo você gasta por semana trabalhando com o sistema? NA

Parte 2: Experiência passada

2.1. Com quantos sistemas operacionais você já trabalhou? 1

Parte 3: Reações gerais do usuário

Selecionar o nível que reflete mais apropriadamente suas impressões sobre a utilização deste sistema computacional. Não Aplicável = NA

3.1. Reações gerais ao sistema: 9 - maravilhoso (1-terrível .. 9-maravilhoso)

3.2. Sua sensação foi: 9 - satisfatória (1-frustrante .. 9-satisfatória)

3.3. O software é: 9 - estimulante (1-desestimulante .. 9-estimulante)

3.4. A utilização do software é: 8 (1-difícil .. 9-fácil)

Parte 4: Telas

4.1. Caracteres na tela do computador: 9 - fácil de ler (1-difícil de ler .. 9-fácil de ler)

4.1.1. Imagem dos caracteres: 9 - precisa (1-distorcida .. 9-precisa)

4.1.2. Fontes dos caracteres: 9 - bastante legíveis (1-pouco legíveis .. 9-bastante legíveis)

4.2. Destaques utilizados na tela como negrito, sublinhado, etc, foram NA para a aprendizagem e a utilização do software. (1-inútil .. 9-útil)

4.3. Os layouts da tela foram úteis para a aprendizagem do software? 8 (1-nunca .. 9-sempre)

4.3.1. A quantidade de informação mostrada na tela é: 8 (1-inadequado .. 9-adequado)

4.3.2. A organização de informações mostrada na tela é: 9 - lógica (1-ilógica .. 9-lógica)

4.4. A seqüência das telas que aparecem durante a execução de uma tarefa é: 8 (1-confuso .. 9-clara)

4.4.1. A próxima tela que aparece na execução da maioria das tarefas é: 2 (1-presumível .. 9-impresumível)

4.4.2. Volta para a tela anterior: 9 - fácil (1-impossível .. 9-fácil)

4.4.3. É possível verificar o que o software está fazendo, ou seja, como é a progressão de tarefas relacionadas com o trabalho: 9 - claramente marcada (1-confuso .. 9-claramente marcada)

Por favor, escreva seus comentários sobre as telas aqui:

Parte 5: Terminologia e informação do sistema

5.1. Uso da terminologia ao longo do sistema: 3 (1-confusa .. 9-clara)

5.2. A terminologia se relaciona bem com o trabalho que você está fazendo? 3 (1-confusa .. 9-clara)

5.2.1. A terminologia técnica/computacional é utilizada: 5 (1-muito freqüentemente .. 9-apropriadamente)

5.2.2. A terminologia na tela é: 3 (1-ambígua .. 9-precisa)

5.3. As mensagens que aparecem na tela: 3 (1-inconsistente .. 9-consistente)

5.3.1. A posição das instruções na tela é: 6 (1-confusa .. 9-clara)

5.4. As mensagens que aparecem na tela são: 3 (1-confusa .. 9-clara)

5.4.1. As instruções dadas pelas mensagens são: 2 (1-confusa .. 9-clara)

5.4.2. As instruções para correção de erros dadas pelas mensagens do software são: 3 (1-confusa .. 9-clara)

5.5. O computador deixa você informado sobre o que está acontecendo: 6 (1-nunca .. 9-sempre)

5.5.1. Qual a freqüência que os cursores animados deixam você informado do que está acontecendo, ou seja, indica que algo está ocorrendo, como, por exemplo, cursores em forma de ampulheta: 1 - nunca (1-nunca .. 9-sempre)

5.5.2. A execução de uma operação leva a um resultado presumível: 9 - sempre (1-nunca .. 9-sempre)

5.5.3. O intervalo de tempo entre as operações é: 9 - aceitável (1-inaceitável .. 9-aceitável)

5.6. As mensagens de erro do software são: 1 - inútil (1-inútil .. 9-útil)

5.6.1. Mensagens de erros esclarecem o problema: 1 - nunca (1-nunca .. 9-sempre)

5.6.2. As frases das mensagens de erro são: 2 (1-ofensiva .. 9-“prestativa”)

Por favor, escreva seus comentários sobre terminologia e informação do sistema aqui:

Os termos e mensagens em inglês dificultam o entendimento das pessoas que não conhecem o idioma.

Parte 6: Aprendizagem

6.1. Aprendizagem da operação do sistema é: 9 - fácil (1-difícil .. 9-fácil)

6.1.1. Introdução ao sistema: 7 (1-difícil .. 9-fácil)

6.1.2. Aprendizado das funções avançadas é: 7 (1-difícil .. 9-fácil)

6.2. A exploração das características por tentativa e erro é: 8 (1-desencorajador .. 9-encorajador)

6.2.1. Exploração das características: 5 (1-com risco .. 9-segura)

6.2.2. Descoberta de novas características: 7 (1-difícil .. 9-fácil)

6.3. Lembrança de nomes e uso de comandos: 8 (1-difícil .. 9-fácil)

6.3.1. Lembrança de funcionalidades específicas é: 8 (1-difícil .. 9-fácil)

6.4. Tarefas podem ser realizadas de maneira direta: 7 (1-nunca .. 9-sempre)

6.4.1. Número de passos por tarefa: 9 - suficiente (1-excessivo .. 9-suficiente)

6.4.2. Passos para completar uma tarefa seguem uma seqüência lógica: 8 (1-nunca .. 9-sempre)

6.4.3. Retorno ao completar a seqüência de passos: 8 (1-confuso .. 9-claro)

Por favor, escreva seus comentários sobre aprendizagem aqui:

(comentário do autor) O usuário destacou a facilidade de aprendizado da aplicação em si, ou seja, terminado o projeto, a interface com RV facilita consideravelmente a interação com o sistema.

Parte 7: Capacidades do sistema

7.1. Velocidade do sistema: 8 (1-baixa .. 9-alta)

7.1.1. Tempo de resposta para a maioria das operações: 8 (1-baixa .. 9-alta)

7.1.2. Taxa que a informação é mostrada: 7 (1-baixa .. 9-alta)

7.2. Corrigindo os erros: 7 (1-difícil .. 9-fácil)

7.2.1. Tipos de correção: 5 (1-complexo .. 9-simples)

7.2.2. Capacidade de desfazer operações: 7 (1-inadequado .. 9-adequado)

7.3. Fácil operação depende do seu nível de experiência: 9 - freqüentemente (1-raramente .. 9-freqüentemente)

7.3.1. É possível realizar as tarefas conhecendo poucos comandos: 8 (1-com dificuldade .. 9-facilmente)

Por favor, escreva seus comentários sobre as capacidades do sistema aqui:

Parte 8: Manuais técnicos e ajuda on-line

8.1. Manuais técnicos são: NA (1-confuso .. 9-claro)

8.1.1. A terminologia utilizada no manual: NA (1-confuso .. 9-claro)

8.2. A informação do manual é facilmente entendida: NA (1-nunca .. 9-sempre)

8.2.1. Descobrir uma solução usando o manual é: NA (1-impossível .. 9-fácil)

8.3. Quantidade de ajuda dada pelos manuais: NA (1-inadequada .. 9-adequada)

8.3.1. Posição das mensagens de ajuda na tela: NA (1-confuso .. 9-claro)

8.3.2. Acesso às mensagens de ajuda é: NA (1-difícil .. 9-fácil)

8.3.3. O conteúdo das mensagens de ajuda on-line é: NA (1-confuso .. 9-claro)

8.3.4. A quantidade de ajuda dada é: NA (1-inadequada .. 9-adequada)

8.3.5. Ajuda define aspectos específicos do sistema: : NA (1-inadequada .. 9-adequada)

8.3.6. Descobrindo informações específicas do sistema: NA (1-difícil .. 9-fácil)

8.3.7. Ajuda on-line: NA (1-inútil .. 9-útil)

Por favor, escreva seus comentários sobre manuais técnicos e ajuda on-line aqui:

Referências Bibliográficas

[AgentSheets'2001] AgentSheets, Inc. *Getting Started with AgentSheets*. 2001.

[AgentSheets'2003] AgentSheets Web site. URL: http://agentsheets.com, acesso em maio/2003.

[Albertin'2002] ALBERTIN, J.C.L. *Reuso de Componentes para Interfaces com Realidade Virtual*. Exame de Qualificação. Orientadora: Júnia Coutinho Anacleto Silva. Departamento de Computação. UFSCar. São Carlos, SP, 2002.

[Alice'2003] Alice Web site. URL: http://www.alice.org, acesso em maio/2003.

[Assis *et al*'2000] ASSIS, A.F.S.R., SILVA, J.C.A. Meeting the Challenge of Systems Development with Virtual Interfaces. In *2000 International Conference on Information Society in the 21st Century: Emerging Technologies and New Challenges (IS2000)*. The University of Aizu, Aizu-Wakamatsu City, Fukushima, Japan. Proceedings, pp. 562-566, November 2000.

[Assis'2001] ASSIS, A.F.S.R. *Um Ambiente Computacional para Desenvolvimento de Interfaces Utilizando Realidade Virtual*. Dissertação de Mestrado. Orientadora: Júnia Coutinho Anacleto Silva. Departamento de Computação. UFSCar. São Carlos, SP, 2001.

[Astheimer *et al*'1999] ASTHEIMER, P, ROSENBLUM, L. A Business View of Virtual Reality. In *IEEE Computer Graphics and Applications*, vol. 19, pp. 28-29, 1999.

[Balbino *et al*'2002] BALBINO, F.C., SILVA, J.C.A., PENTEADO, R.A.D. O Desafio de Prover Ambientes para EUP e Realidade Virtual com Usabilidade. In *Proceedings of IHC 2002*, pp. 351-354. Fortaleza, Ceará, Brazil, Outubro 2002.

[Barbosa'1999] BARBOSA, S.D.J. *Programação Via Interface*. Tese de Doutorado. Orientadora: Clarisse Sieckenius de Souza. Departamento de Informática. PUC-Rio. Rio de Janeiro, RJ, 1999.

[Baron *et al*'2001] BARON, M., GIRARD, P. Bringing Robustness to End-User Programming. In *Proceedings of 2001 IEEE Symposia on Human-Centric Computing Languages and Environments*. Stresa, Italy. Entergraphica, 2001, pp. 142-149.

[Bicho *et al*'2002] BICHO, A.L., SILVEIRA JR, L.G., CRUZ, A.J.A., RAPOSO, A.B. Programação Gráfica 3D com OpenGL, Open Inventor e Java 3D. In *Revista Eletrônica de Iniciação Científica (REIC)*, vol. II, n. I, Março 2002. Sociedade Brasileira de Computação, Brasil.

[Bowman *et al*'2001] BOWMAN, D.A., KRUIJFF, E., LAVIOLA, J.J., POUPYREV, I. An Introduction to 3-D User Interface Design. In *Presence*, vol. 10, n.º I, February 2001, 96-108.

[Brooks'1987] BROOKS, F.P. No Silver Bullet, Essence and Accidents of Software Engineering. In *IEEE Computer*, vol. 20, n.º 4, April 1987, pp 10-19.

[Brooks'1999] BROOKS, F.P. What's Real About Virtual Reality?. In *IEEE Computer Graphics and Applications*, vol. 19, pp. 16-27, 1999.

[Cheung'2003] CHEUNG, W.M.A. Compare Tools in 3D. In *Website for the 3rd Annual CM316 Conference on Multimedia Systems*. Southampton University, UK, January 2003. URL: http://mms.ecs.soton.ac.uk/papers/34.pdf, acesso em maio/2003.

[Conway'1997] CONWAY, M.J. *Alice: Easy-to-Learn 3D Scripting for Novices*. Tese de Doutorado. University of Virginia, 1997.

[Conway *et al*'2000] CONWAY, M.J. *et al*. Alice: Lessons Learned from Building a 3D System for Novices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, The Hague, The Netherlands, pp. 486-493, 2000.

[Cunha'2001] CUNHA, C.K.V. *Um Modelo Semiótico dos Processos de Comunicação Relacionados à Atividade de Extensão à Aplicação por Usuários Finais*. Tese de Doutorado. Orientadora: Clarisse Sieckenius de Souza. Departamento de Informática. PUC-Rio. Rio de Janeiro, RJ, 2001.

[Cypher'1993] CYPHER, A. *Watch What I Do: Programming by Demonstration*. The MIT Press. Cambridge, MA, 1993.

[Cypher *et al*'1995] CYPHER, A., SMITH, D.C. KidSim: End User Programming of Simulations. In *Proceedings of CHI'1995*. Denver, Colo. New York: ACM Press, 1995.

[da Silva'2001] da SILVA, S.R.P. *Um Modelo Semiótico para Programação por Usuários Finais*. Tese de Doutorado. Orientadora: Clarisse Sieckenius de Souza. Departamento de Informática. PUC-Rio. Rio de Janeiro, RJ, 2001.

[Deitel *et al*'2002] DEITEL, H.M., DEITEL, P.J. *Java, Como Programar*. 4 ed. Porto Alegre: Bookman, 2003.

[Drake *et al*'1997] DRAKE, J.M., XIE, W.W., TSAI, W.T. Approach and case study of requirement analysis where end users take an active role. In *Proceedings of the 15th International Conference on Software Engineering*, Baltimore, Maryland, United States, pp. 177-186, 1997.

[Goodell'2002] GOODELL, H. *What is End-User Programming?*. URL: http://www.cs.uml.edu/~hgoodell/EndUser/whatsEUP.htm, acesso em fevereiro/2002.

[Gulliksen *et al*'1999] GULLIKSEN, J., LANTZ, A., BOIVIE, I. *User Centered Design in Practice – Problems and Possibilities*. Technical Report TRITA-NA-D9813, CID – Centre for User Oriented IT Design, Stockholm, Sweden, January 1999.

[Hix *et al*'1993] HIX, D., HARTSON, H.R. *Developing User Interfaces: Ensuring Usability Through Product & Process*. John Wiley & Sons, Inc., 1993.

[Jacobson'1994] JACOBSON, L. *Realidade Virtual em Casa*. Rio de Janeiro: Berkeley, 1994.

[Java'2003] Java Web Site. URL: http://java.sun.com/, acesso em junho 2003.

[Java 3D'2002] Java 3D API Tutorial. URL: http://java.sun.com/products/java-media/3D/collateral, acesso em julho/2002.

[Kahn'2001] KAHN, K. Generalizing by Removing Detail: How Any Program Can Be Created by Working with Examples. In LIEBERMAN, H. (2001). *Your Wish is my Command: Programming by Example*. Morgan Kaufmann. San Francisco, CA, 2001, pp. 22-43.

[Kaur *et al*'1998] KAUR, K., SUTCLIFFE, A., MAIDEN, N. Improving Interaction with Virtual Environments. In *The 3D Interface for the Information Worker* (Digest n.º 1998/437), IEE Colloquium, May 1998.

[Lakoff *et al*'1980] LAKOFF, G., JOHNSON, M. *Metaphors We Live By*. The University of Chicago Press: Chicago, 1980.

[Lakoff'1987] LAKOFF, G. *Women, Fire, and Dangerous Things*. The University of Chicago Press: Chicago, 1987.

[Leonard'2002] LEONARD, A. Enabling end users to be more efficient during systems development. In *Proceedings of the 2002 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists on Enablement Through Technology*, Port Elizabeth, South Africa, pp. 156-162, 2002.

[Lieberman'2001] LIEBERMAN, H. *Your Wish is my Command: Programming by Example*. Morgan Kaufmann. San Francisco, CA, 2001.

[Machado'1997] MACHADO, L.S. *A Realidade Virtual em Aplicações Científicas*. Dissertação de Mestrado. Orientador: Luiz Alberto Vieira Dias. INPE – Instituto Nacional de Pesquisas Espaciais. São José dos Campos, SP, 1997.

[Madsen'1999] MADSEN, K. H. The Diversity of Usability Practices. In *Communications of the ACM*, vol. 42, Issue 5, p. 60-62, May 1999.

[Mian *et al*'2001] MIAN, P.G., NATALI, A.C.C., FALBO, R.A. Ambientes de Desenvolvimento de Software e o Projeto ADS. In *Revista Engenharia, Ciência e Tecnologia*, vol 04, n.º 04, Julho/Agosto 2001, pp 3 – 10, Vitória, Espírito Santo, Brasil.

[Nielsen'1993] NIELSEN, J. *Usability Engineering*. Academic Press, 1993.

[Noyes *et al*'1995a] NOYES, J.M., STARR, A.F. Working with users in system development: some methodological considerations. In *Integrating HCI in the Lifecycle*, IEE Colloquium, April 1995.

[Noyes *et al*'1995b] NOYES, J.M., HARRIMAN, J.C. User involvement in the design process: a case for end-user evaluation of software packages. In *Human Centred Automation*, IEE Colloquium, June 1995.

[Pimentel *et al*'1995] PIMENTEL, K. TEIXEIRA, K. *Virtual Reality – Through the New Looking Glass*. 2 ed. New York: McGraw-Hill, 1995.

[Pinho'2000] PINHO, M.S. *Técnicas de Interação em Ambientes Tridimensionais*. Minicurso. Workshop de Realidade Virtual. Gramado, RS, 2000.

[Pressman'2002] PRESSMAN, R.S. *Engenharia de Software*. 5 ed. Rio de Janeiro: McGraw-Hill, 2002.

[Repenning *et al*'2001] REPENNING, A., PERRONE, C. Programming by Analogous Examples. In LIEBERMAN, H. (2001). *Your Wish is my Command: Programming by Example*. Morgan Kaufmann. San Francisco, CA, 2001, pp. 352-369.

[Robertson *et al*'1993] ROBERTSON, G.G, CARD, S.K., MACKINLAY, J.D. Non-immersive Virtual Reality. In *IEEE Computer*, vol. 26, pp. 81-83, 1993.

[Royce'70] ROYCE, W.W. Managing the Development of Large Software Systems: Concepts and Techniques. In *Proceedings of WESCON*, August 1970.

[Shneiderman'2001] SHNEIDERMAN, B. Foreword. In Lieberman, H. *Your Wish is my Command: Programming by Example*. Morgan Kaufman. San Francisco, CA, 2001.

[Silva'1999] SILVA, J.C.A. Development of Virtual Interfaces Using Configurable Templates. In *ICCIMA'99 – 3rd International Conference on Computational Intelligence and Multimedia Applications*, New Delhi, India, Proceedings, pp. 354-358, September 1999.

[Smith *et al*'2001] SMITH, D.C., CYPHER, A., TESLER, L. Novice Programming Comes of Age. In LIEBERMAN, H. *Your Wish is my Command: Programming by Example*. Morgan Kaufmann. San Francisco, CA, 2001, pp. 8-19.

[Soares'2002] SOARES, C.L. *Integração da Engenharia Reversa e Interfaces com Realidade Virtual*. Dissertação de Mestrado. Orientadora: Júnia Coutinho Anacleto Silva. Departamento de Computação. UFSCar. São Carlos, SP, 2002.

[Stagecast Creator'2003] Stagecast Creator Web site. URL: http://www.stagecast.com, acesso em maio/2003.

[Tanriverdi *et al*'2001] TANRIVERDI, V., JACOB, J.K. VRID: a Design Model and Methodology for Developing Virtual Reality Interfaces. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, Baniff, Alberta, Canada, November 2001.

[ToonTalk'2003] ToonTalk Web site. URL: http://www.toontalk.com, acesso em maio/2003.

[Vince'1995] VINCE, J. *Virtual Reality Systems*. Cambridge: Addison-Wesley, 1995.

[Wavefront'2003] Wavefront® Maya Web site. URL: http://www.aliaswavefront.com/en/products/maya/index.shtml, acesso em maio/2003.

[Wiklund'1994] WIKLUND, M.E. *Usability in Practice: How Companies Develop User-Friendly Products*. Academic Press, 1994.