Cesar Tadeu Pozzer

Um Sistema para Geração, Interação e Visualização 3D de Histórias para TV Interativa

TESE DE DOUTORADO

DEPARTAMENTO DE INFORMÁTICA
Programa de Pós–graduação em Informática

Rio de Janeiro
Março de 2005

Pontifícia Universidade Católica
do Rio de Janeiro

Cesar Tadeu Pozzer

Um Sistema para Geração, Interação e Visualização 3D de Histórias para TV Interativa

Tese de Doutorado

Tese apresentada ao Programa de Pós–graduação em Informática do Departamento de Informática da PUC–Rio como parte dos requisitos parciais para obtenção do título de Doutor em Informática.

Orientador: Prof. Bruno Feijó
Co–Orientador: Prof. Antonio L. Furtado
Co–Orientador: Prof. Angelo E. M. Ciarlini

Rio de Janeiro
Março de 2005

Pontifícia Universidade Católica
do Rio de Janeiro

Cesar Tadeu Pozzer

Um Sistema para Geração, Interação e Visualização 3D de Histórias para TV Interativa

Tese apresentada ao Programa de Pós–graduação em Informática do Departamento de Informática do Centro Técnico Científico da PUC–Rio como parte dos requisitos parciais para obtenção do título de Doutor em Informática. Aprovada pela Comissão Examinadora abaixo assinada.

Prof. Bruno Feijó
Orientador
Departamento de Informática — PUC–Rio

Prof. Antonio L. Furtado
Co–Orientador
Departamento de Informática — PUC–Rio

Prof. Angelo E. M. Ciarlini
Co–Orientador
Departamento de Informática Aplicada — UniRio

Prof. Marcelo de Andrade Dreux
Departamento de Engenharia Mecânica — PUC–Rio

Prof. Waldemar Celes Filho
Departamento de Informática — PUC–Rio

Prof. Soraia Raupp Musse
UNISINOS

Prof. Roberto de Beauclair Seixas
IMPA

Prof. José Eugenio Leal
Coordenador Setorial do Centro Técnico Científico — PUC–Rio

Rio de Janeiro, 30 de Março de 2005

Cesar Tadeu Pozzer

Graduou–se em Informática pela UFSM - Universidade Federal de Santa Maria. Durante sua graduação, foi pesquisador do CNPq em trabalhos de iniciação científica no LACESM. Durante o Mestrado no ITA foi bolsista da CAPES, realizando estudos em Síntese de Imagens. Atualmente faz pesquisas em Storytelling e jogos para computador.

Ficha Catalográfica

Pozzer, Cesar T.

Um Sistema para Geração, Interação e Visualização 3D de Histórias para TV
Interativa/ Cesar Tadeu Pozzer; orientador: Bruno Feijó; co–orientador: Antonio L. Furtado, Angelo E. M. Ciarlini. — Rio de Janeiro : PUC–Rio, Departamento de Informática, 2005.

156 f. : il. ; 30 cm

1. Tese (doutorado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática.

Inclui referências bibliográficas.

1. Informática – Teses. 2. Computação Gráfica. 3. Inteligência Artificial. 4. Agentes. 5. TV Interativa. I. Feijó, Bruno. II. Furtado, Antonio. III. Ciarlini, Angelo. IV. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. V. Título.

CDD: 004

Dedico esta tese ao Prof. Marcelo Dreux, pelo incentivo decisivo ao longo de todo o Doutorado.

Agradecimentos

A meus pais.

Ao professor Marcelo Gattass, meu primeiro contato na PUC, por ter me oferecido uma vaga no TECGraf como pesquisador, trabalho este que teve grande valia financeira e, principalmente, profissional.

Ao professor Marcelo Dreux, meu maior incentivador para entrar no programa de Doutorado da PUC-Rio e também pela colaboração fornecida ao longo de todo o Doutorado.

Aos meus orientadores Bruno Feijó, Antônio Furtado e Angelo Ciarlini, que me acompanharam durante todo o desenvolvimento da tese.

Ao Ricardo Smarzaro, pela elaboração dos modelos 3D usados na primeira versão do motor gráfico, ao João, pelos modelos usados na segunda versão e ao Pablo, pelos modelos usados na terceira e atual versão.

Ao pessoal do ICAD, pela amizade, colaboração e companheirismo nas horas mais difíceis e saborosas. Em especial ao Chico, Börje, Rodrigo e ao Binder, pela grande ajuda na solução de problemas durante a implementação do motor gráfico.

À minha esposa Ranice, pela paciência, compreensão e incentivos durante todo este trajeto. E ao Fernando, o meu guri, que ainda é muito pequeno para compreender o motivo da ausência permanente de seu pai em casa.

Ao CNPq e à FINEP pelo apoio financeiro, e aos laboratórios ICAD/IGames e VisionLab pelo ambiente de trabalho oferecido.

Resumo

Pozzer, Cesar T.; Feijó, Bruno; Furtado, Antonio; Ciarlini, Angelo. **Um Sistema para Geração, Interação e Visualização 3D de Histórias para TV Interativa**. Rio de Janeiro, 2005. 156p. Tese de Doutorado — Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Esta tese visa o desenvolvimento de um ambiente integrado para o controle da geração e representação de histórias interativas dinâmicas. A geração é realizada por um processo de simulação, resultando em um conjunto de operações parcialmente ordenado que define o enredo da história. Esta história deve então ser representada graficamente por meio de um motor gráfico. Estão sendo utilizadas técnicas cinematográficas para capturar a essência das cenas, compostas por um ambiente virtual 3D, que possuem personagens e objetos. Para o desenrolar da história, os personagens, implementados como agentes reativos, interagem entre si em um ambiente multiagente e com a cena. Cada agente encapsula recursos que os permitem fazer a representação gráfica dos eventos típicos das histórias. A arquitetura como um todo é projetada para servir como meio de geração de conteúdo para a TV interativa.

Palavras–chave

TV Interativa, Agentes, Inteligência Artificial, Computação Gráfica, Histórias Interativas.

Abstract

Pozzer, Cesar T.; Feijó, Bruno; Furtado, Antonio; Ciarlini, Angelo. **A System for Generation, Interaction and 3D Visualization of Stories for Interactive TV** . Rio de Janeiro, 2005. 156p. PhD. Thesis — Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

This thesis aims at the development of an integrated tool for managing both the generation and representation of dynamic interactive stories (storytelling). The story generation is accomplished by a simulation process resulting in a set of partially ordered operations that define the plot of the story. This story should then be graphically represented by means of a 3D engine. It has been used cinematographic techniques to capture the essence of the scenes, which are composed by a virtual 3D environment, including characters and objects. Characters, implemented as reactive agents, interact among each other in a multi-agent system and with the scene to accomplish the plot of the narrative. Each agent encapsulates resources that allow them to graphically represent typical events of stories. The overall architecture is designed as a source for Interactive TV content.

Keywords

Interactive TV, Agents, Artificial Intelligence, Computer Graphics, Storytelling

Conteúdo

Lista de Figuras

Lista de Tabelas

1
Introdução

Desde o surgimento dos primeiros jogos eletrônicos em meados dos anos 50, uma infinidade de aparatos eletrônicos foram desenvolvidos, principalmente referentes a jogos e entretenimento digital. Telefone celular, consoles de jogos, PDAs (*Personal Digital Assistants*), computadores pessoais e televisão estão entre os dispositivos que, mesmo sendo fisicamente diferentes, têm pelo menos uma característica comum: levar de alguma forma o entretenimento aos seus usuários. Os jogos digitais são uma das formas mais difundidas de entretenimento eletrônico, mas que atualmente não se restringem apenas a computadores e vídeo games. Com o surgimento da TV Interativa, que já existe em vários países desenvolvidos, é possível que o telespectador tenha mais autonomia sobre a programação. A nova mídia, que gradativamente está chamando a atenção do público, abre a porta para o desenvolvimento de novos aplicativos interativos, dentre os quais figuram-se os jogos.

O desenvolvimento de um aplicativo de entretenimento digital pode englobar Computação Gráfica, Inteligência Artificial, Banco de Dados, processamento de áudio, interfaces, Engenharia de Software, tecnologias de redes e Sistemas Distribuídos. Devem ser considerados também os aspectos do(s) hardware(s) para onde este aplicativo será portado.

Tecnologias de hardware e software são fundamentais para a evolução da mídia digital. Porém, não basta apenas tecnologia digital para a construção de sistemas de entretenimento. Novos paradigmas de conteúdos interativos precisam ser desenvolvidos para adequar os sistemas às necessidades dos usuários.

Assim como existem pesquisas no intuito de convergir tecnologias de modo a realizar a integração de vários dispositivos em um único, o mesmo vale para a mídia digital. Segundo Forman [53], TV, filmes, vídeo games e Internet tendem a convergir em direção a um grande fluxo de entretenimento digital onde autores, público e agentes virtuais interajam em uma experiência colaborativa.

A TV Interativa (*Interactive TV* - iTV) pode ser uma solução para essa convergência de mídias. Porém, mais uma vez, o conteúdo é peça-chave da questão. Quando nos referimos a conteúdo, devemos pensar tanto na forma como é gerado e apresentado, quanto nos meios pelos quais o usuário/telespectador poderá interagir com ele.

Partindo dessa visão de futuro, busca-se, neste trabalho, abordar temas relacionados com a especificação de conteúdo interativo que possa ser utilizado especialmente como programação para TV interativa. No contexto de iTV, a exibição de histórias interativas que possam ser criadas e guiadas pelo próprio usuário se mostra como uma boa alternativa de conteúdo interativo. Nas histórias interativas, o usuário pode agir como autor e diretor. O desenvolvimento da história pode assumir múltiplas combinações. É, sem dúvida, um passo adiante em relação à interatividade atualmente presente em alguns programas de TV, onde o telespectador, por meio de uma linha telefônica, pode selecionar entre um elenco fixo de possibilidades, o final desejado. A área que trata de histórias interativas é conhecida como Narrativa Interativa de Histórias (*Interactive Storytelling*) [62][102].

A literatura em Narrativa Interativa de Histórias ainda é um pouco escassa e os resultados das pesquisas ainda deixam muitos aspectos em aberto. Muitos trabalhos apresentam conceitos interessantes, mas não fornecem uma estrutura formal para a geração de enredos [123][64], nem exibem uma visualização robusta das histórias [129]. Estes vazios na literatura motivam os desenvolvimentos propostos neste trabalho.

1.1 Proposta

O objetivo desta pesquisa de tese é estudar conceitos de áreas gerais como TV interativa, Agentes, Computação Gráfica e Storytelling, e propor novas técnicas e paradigmas de conteúdos interativos, baseados em histórias interativas, que possam ser aplicados a iTV, entretenimento, além de jogos e programas educativos. Todos estes conceitos estão fortemente interligados e visam a construção de um sistema que permita alta interatividade e que, ao mesmo tempo, exija pouco esforço do usuário.

As histórias simuladas nesta tese adotam o gênero contos de fadas, onde figuram dois heróis, um vilão e uma vítima. Estes personagens podem desempenhar um conjunto fixo de ações, o que permite criar histórias onde ocorrem eventos como raptos, libertações, duelos e casamentos.

Para a construção deste sistema, é necessário desenvolver pesquisas em três linhas distintas, mas fortemente relacionadas, conforme esquema proposto na Figura 1.1. A linha de TV Interativa corresponde ao estudo de conteúdos interativos e técnicas relacionadas com a exibição e interação do conteúdo a ser gerado. A linha de Agentes e IA trata da geração de personagens autônomos, bem como geração dinâmica de histórias, que ao mesmo tempo sejam interessantes e coerentes. Por fim, a linha de Computação Gráfica é necessária para transformar as representações simbólicas das histórias em informação gráfica que possa ser apresentada como um conteúdo interativo, fazendo uso de câmeras virtuais e modelos 3D animados.

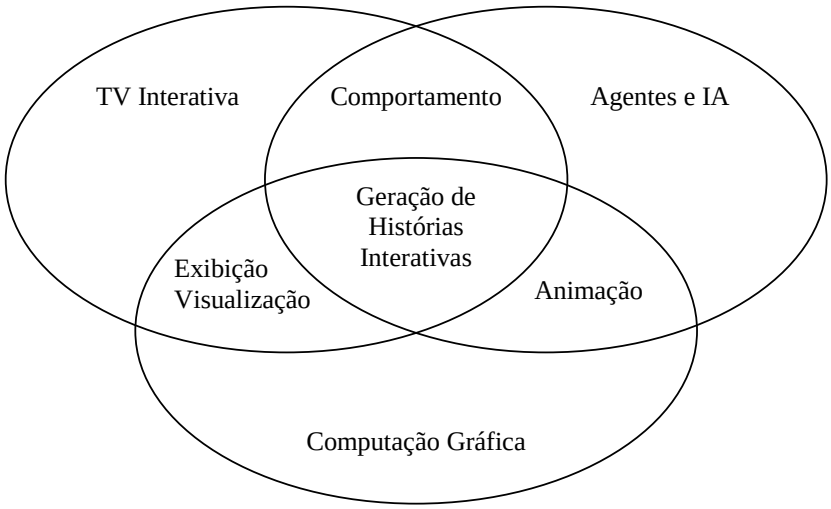

Figura 1.1: Linhas de pesquisa que compõem o sistema de geração e visualização de histórias interativas.

A história a ser representada é o componente central do processo. Ela é criada pelo IPG (*Interactive Plot Generator*) [33], um gerador interativo de histórias, implementado em Prolog, estendido com programação em lógica com restrições (*Constraint Logic Programming*) que, por meio de planejamento e regras de inferência de objetivos, gera conjuntos parcialmente ordenados de eventos. A criação de um enredo é realizada por um processo de simulação, onde tanto a descrição da situação inicial dos personagens, seus objetivos e as operações que caracterizam o gênero, são levadas em consideração. Uma operação descreve os fatos válidos antes e depois do evento em termos de pré- e pós-condições. Estas operações, no IPG, são funções que mudam o estado do mundo. Estas funções são baseadas numa classificação de eventos típicos proposto por Propp [115]. Para que uma operação seja executada, é necessário que suas pré-condições

sejam satisfeitas, ou no próprio estado inicial, ou então pelas pós-condições de operações que a precedem [33].

O usuário pode participar ativamente da geração das histórias, tanto na definição da ordem total dos eventos, pela inserção de novos eventos ou situações, bem como pela requisição de alternativas, dentro do elenco de possibilidades que a história pode assumir. Neste ponto, o IPG também tem o papel de garantir a integridade e coerência da história. A Figura 1.2(a) ilustra um usuário interligando eventos, por meio do Gerenciador de Enredos (Ver Capítulo 6 - Arquitetura do Sistema), para construir uma determinada versão de uma história, onde a princesa Marian é raptada pelo dragão Draco.

Figura 1.2: a) Gerenciador de Enredos; b) Visualização da história.

Definida uma ordem total dos eventos, estes devem então ser transformados em ações gráficas animadas, suficientemente realistas para o propósito em questão. Para isso, faz-se uso de personagens autônomos (atores), que habitam um mundo virtual 3D. A Figura 1.2(b) ilustra a cena do rapto dramatizada pelo Visualizador de Enredos (Ver Capítulo 6).

Para a etapa de visualização da história (dramatização), são definidos outros dois módulos: o de Inteligência Artificial (IA) e o de Computação Gráfica. A IA se faz necessária para assegurar que cada personagem, aqui representado como um agente de software, seja capaz de expressar um comportamento inteligente e realista, por meio de animações gráficas condizentes com as ações a serem representadas. O resultado final é a exibição gráfica da representação da história simulada. O módulo de Computação Gráfica é o responsável por esta tarefa. Para isso, este módulo faz uso de algoritmos de renderização, iluminação e técnicas de modelagem de personagens, bem como do cenário. Neste módulo é utilizada a linguagem C++, juntamente com a biblioteca OpenGL [147]. A Figura 1.3 apresenta

um diagrama mais detalhado da integração dos diversos módulos que definem o sistema que está sendo proposto.

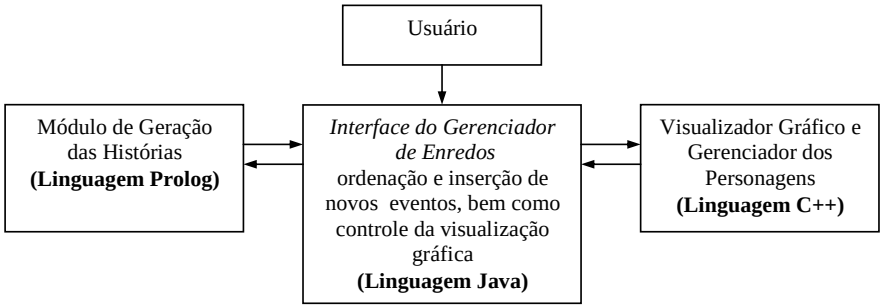

Figura 1.3: Integração dos Módulos que definem o sistema de visualização de histórias interativas.

Para a implementação do sistema proposto, faz-se uso de diversas técnicas desenvolvidas para jogos, visto que ambos paradigmas de entretenimento possuem características comuns. A maior diferença entre eles está na forma como o usuário interage. Ao contrário de um jogo, onde o jogador controla um personagem ou um grupo de personagens, no sistema proposto o papel do usuário é controlar a geração de uma história ou somente acompanhar a sua visualização, que por sua vez faz uso de personagens autônomos (agentes controlados), que interagem entre si e com o ambiente para assegurar uma fiel representação da história até então simulada.

Esta tese visa atingir, de forma resumida, as seguintes contribuições originais. Descrições mais detalhadas das contribuições são apresentadas no Capítulo 8:

- Desenvolver um sistema que integra geração, direcionamento lógico e exibição de histórias interativas considerando possíveis arquiteturas de TV Interativa e que possa atender a diferentes níveis de interatividade;
- Desenvolver um modelo comportamental de atores que respondam às necessidades de tempo real e dinamismo requeridos por aplicações que tratam com histórias interativas, bem como propor extensões para incorporação de drives e emoções;
- Especificar um modelo de câmera inteligente, necessário à dramatização das histórias.

1.2 Estrutura

Esta tese está organizada em 8 capítulos. No Capítulo 2, faz-se uma exposição geral de tecnologias existentes referentes a hardware, software, bem como a conteúdo, que estão sendo atualmente utilizadas em TV Interativa. Mediante este estudo, pode-se então melhor avaliar a viabilidade técnica e cognitiva (humana) deste meio de entretenimento. Espera-se assim conseguir especificar um modelo de conteúdo baseado em histórias Interativas passível de implementação e implantação sob a tecnologia existente nesta área.

O Capítulo 3 apresenta um levantamento e uma discussão das técnicas atuais de Narrativa Interativa de Histórias (*Interactive Storytelling*), sob os aspectos de geração, interação e visualização das histórias. Faz-se também uma análise das abordagens comumente utilizadas.

Sobre o estudo apresentado no Capítulo 3, o Capítulo 4 propõe um novo modelo de geração de histórias, que faz uso do IPG, desenvolvido por Ciarlini [33]. Um dos objetivos deste capítulo é também a definição de um conjunto de eventos adequados para gerar histórias interessantes, interativas e que possam ser facilmente representadas e compreendidas unicamente por meio de recursos gráficos.

No Capítulo 5 faz-se um apresentação de tecnologias de agentes usadas na implementação dos personagens (atores) que habitam as histórias, tanto a nível de geração como de visualização. No que se refere à visualização, apresenta-se um modelo comportamental de implementação simples para gerenciamento de ações presentes nas histórias a serem visualizadas. A câmera virtual também faz uso de conceitos de agentes e é brevemente descrita neste capítulo. Por fim, apresenta-se uma discussão da incorporação de *drives* e emoções nas etapas de geração e visualização das histórias.

O Capítulo 6 descreve a arquitetura do sistema, que é composto por três grandes módulos, implementados em diferentes linguagens de programação: Prolog, Java e C++. São apresentadas as atribuições que cada módulo deve desempenhar, bem como detalhes da integração e da troca de informações entre eles. Apresenta-se também uma arquitetura para incorporação deste sistema em um ambiente de TV Interativa.

O Capítulo 7 apresenta a implementação do sistema e os resultados obtidos. São descritas as técnicas e algoritmos empregados para a implementação dos módulos que compõem o sistema. Faz-se também uma descrição do cenário, personagens, câmeras virtuais e estratégias usadas para

controle temporal da visualização gráfica das histórias.

No Capítulo 8 são apresentadas as conclusões gerais, contribuições, bem como perspectivas de trabalhos futuros.

2
TV Interativa

Com a substituição das linhas analógicas convencionais por linhas digitais, a TV tradicional está passando por fortes modificações que vão muito além de apenas melhorias na qualidade de imagem [42]. A mídia digital permite que, além de nitidez e resolução, uma grande gama de conteúdos possa ser inserida e manipulada neste novo paradigma de TV.

A interatividade é, sem dúvida, a maior beneficiada com o avanço desta tecnologia. Com a utilização de novos hardwares de transmissão e decodificação, além de TVs digitais, pode-se fornecer ao usuário a possibilidade de, em termos gerais, interagir com aplicativos que executam localmente, permitindo, desta forma, que conteúdos desenvolvidos para computadores e video games possam estar disponíveis ao acesso de qualquer telespectador frente à TV. O grau de interação é determinado pela tecnologia empregada e pelos recursos de hardware e software disponibilizados, bem como pela natureza do conteúdo recebido.

Esse novo paradigma de TV, também conhecido como "interactive TV – iTV" ou "enhanced TV", é resultado de enormes avanços em tecnologia digital, TV digital, redes de alta velocidade, processamento de áudio e vídeo, softwares de compressão e serviços de *broadcasting* [55][137]. A TV convencional, que é passiva e linear, está se tornando sob demanda, participativa, não-linear, bidirecional, ou em outras palavras, está permitindo que cada telespectador possa definir o conteúdo e a forma como a informação lhe é apresentada. iTV é a reinvenção da existente e passiva TV. Segundo [138], iTV é, essencialmente, a incorporação de programação ao vídeo que permite algum nível de interatividade. Segundo [12], iTV é a junção das tecnologias associadas à difusão de vídeo com tecnologias interativas viabilizadas por um "canal de retorno" ligando terminais de acesso avançados com as emissoras de televisão.

Esta nova tecnologia vai permitir que o telespectador possa receber informações adicionais sobre assuntos de interesse e até mesmo agendar o horário em que deseja ver seu programa favorito [42]. Com a iTV, o

telespectador poderá, num futuro próximo, ter boa parte dos recursos disponíveis em um PC, tais como jogos, Internet, áudio e vídeo, com a mesma eficiência. Entretanto para que tudo isso ocorra, é necessária a criação de padrões, tanto na forma de transmissão digital, quanto no formato do conteúdo a ser exibido.

Para que a iTV possa se consolidar e vir a substituir a TV tradicional, uma série de requisitos deve ser atendida. O elemento central de toda a infra-estrutura é o conteúdo final que é apresentado ao telespectador (usuário). Em função deste conteúdo, necessidades de banda passante, processamento computacional e nível de interação devem ser determinados e garantidos. Esses diversos itens são discutidos neste capítulo e são usados para validar a viabilidade técnica do uso de histórias interativas como conteúdo para iTV.

2.1 Comunicação de Dados

Um dos requisitos necessários para a construção e consolidação da iTV é a substituição da transmissão analógica por linhas digitais de alta velocidade. Este novo meio, além de ser necessário para a iTV, é um passo para a evolução da velha TV analógica na direção da TV digital (DTV - *Digital Television*). DTV é um dispositivo que pode processar (decodificar) mídias digitais, que em sua grande maioria são comprimidas, para um aproveitamento melhor da banda passante. O padrão de compressão adotado é o MPEG-2 [30][1] para imagem e o MP3 para áudio.

Não existe uma obrigatoriedade do uso de DTV para acesso a conteúdos interativos. Estes podem ser acessados por meio de TVs analógicas, desde que acopladas a hardwares conversores de sinal digital em analógico. Esta conversão pode ser realizada pelos set-top boxes (URD - Unidade Receptora-decodificadora [126]), que são dispositivos que conectam a TV aos provedores (*content providers*) e que também processam os dados e conteúdos interativos. Estes dispositivos são detalhados na Seção 2.4.

DTV não é sinônimo de HTDV (*High definition TV*), que é um padrão de alta definição e, conseqüentemente, representa um maior consumo de banda. Esta banda, em DTV, pode ser variada, o que não ocorre em transmissões analógicas. A resolução em HDTV é em torno de 1K de linhas

[1]MPEG-4 é um padrão de compressão bem mais avançado, porém, por questões de padronização já estabelecidas no mercado, o MPEG-2 é a referência em TV digital.

de escaneamento ativas (*scan lines*)[2], o que é um salto significativo em relação ao 0.5K da SDTV (*Standard Definition TV*)[3]. As tecnologias de 4K, como a *Ultra High Definition Vídeo* (UHDV) da NHK (*Japan Broadcasting Corporation*) e câmeras tipo HDC-950, ainda são experimentos em TV[4].

Diversos serviços interativos, como VoD (*Video on Demand*), transmissão de aplicativos, imagens, gráficos e outros elementos de grande uso em programações interativas, somente são possíveis com a tecnologia digital.

Independente do padrão de transmissão (americano, europeu ou japonês), os tipos de transmissões de TV Digital são os seguintes: cabo, satélite e terrestre. No Brasil, por suas características geográficas e de base instalada, a transmissão para a TV aberta deve ser via terrestre[5]. A TV por assinatura não está presa a uma política do governo e cada emissora pode adotar o padrão que achar conveniente (Aliás, a Sky já oferece TV digital de baixa interatividade, por satélite, desde 1996 [133]). A banda para transmissão terrestre é de 6 MHz, o que permite uma taxa de transmissão de 20 Mbps. Com a compressão MPEG-2, esta banda permite que uma emissora de TV transmita uma qualidade HDTV (com som *Dolby Digital*[6]) em um único programa consumindo toda a banda. Entretanto, ao contrário da transmissão analógica[7], a emissora pode optar por usar a banda de 6 MHz para transmitir 4 canais SDTV simultâneos. Na realidade, a banda de 6 MHz pode comportar várias opções combinando canais de vídeo e dados (Figura 2.1).

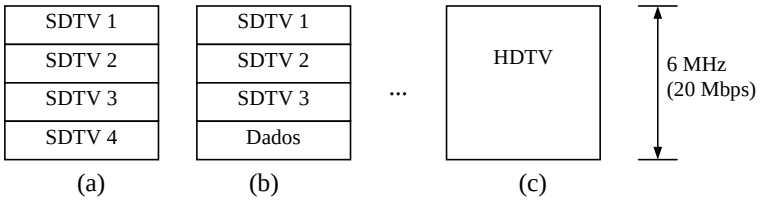

Figura 2.1: Opções de uso de um canal de TV digital.

Cada uma das opções apresentadas na Figura 2.1 representa um

[2]1080x1920 (USA), 1000x1778 (Europa), 1080x11920 (Japão).

[3]484x720 (Padrão NTSC, USA), 575x767 (Padrão PAL, Europa).

[4]A indústria de cinema americana vê o cinema digital 4K em LSDI (*Large Screen Digital Imaging*) como a solução da sobrevivência ao cinema doméstico (*Home Theaters*) e à TV digital [75].

[5]Chama-se terrestre à transmissão feita pela rede de torres de transmissão.

[6]5 direções e 1 subwoofer.

[7]A transmissão analógica não suporta canais próximos em freqüência por causa da interferência.

modelo de negócios diferente[8]. Há basicamente dois tipos de interação em TV digital: com o sinal recebido e com o canal de retorno. A interação através do sinal corresponde a uma navegação restrita às opções e aos serviços transmitidos[9]. A interação com o canal de retorno permite que o telespectador envie informação de volta a emissora, geralmente utilizando uma linha telefônica (este tipo de comunicação ainda pode evoluir para uma comunicação de banda mais larga através de redes e Internet). A linha telefônica é normalmente suficiente, devido ao pouco volume de dados de retorno transmitidos por telespectadores.

O uso de linhas digitais para uso em TV é ainda um conceito novo. Por isso, ainda não existe um consenso sobre a forma de utilização desta larga banda passante. Atualmente, e como ilustra a Figura 1.1, existem 3 correntes que propõem diferentes usos desta nova tecnologia:

- Maior resolução: Implicaria na substituição do padrão tradicional de dados (*SDTV - Standard TV*) pelo padrão HDTV (Figura 2.1(c));
- Aumento de canais: Nesta estratégia, continua-se usando o padrão SDTV, porém aumenta-se o número de canais disponíveis (Figura 2.1(a));
- Adição de recursos interativos: Parte da banda passante seria usada para enviar conteúdos interativos ou, até mesmo, conteúdos personalizados para cada telespectador (se houver uma integração com redes de serviço e Internet). Em outras palavras, a consolidação da iTV.

Uma opinião intermediária sugere variar a resolução em função do conteúdo a ser apresentado. Apesar das opiniões discordantes, certamente, parte da banda passante vai ser usada para iTV, visto o grande sucesso da Internet e desejo de uma grande massa de pessoas em interagir com a informação e com outras pessoas, por meio de jogos, programas interativos, dentre outros.

Linhas de alta velocidade, capazes de suportar transmissão de vídeo e outros conteúdos digitais em tempo real, se fazem necessárias frente à possibilidade de transmissão de conteúdos personalizados a cada

[8]Na ocasião da confecção desta tese (2004), o governo brasileiro apontava para o modelo de 3 canais SDTV com foco em interação (Figura 2.1(b)). A Rede Globo, por ser a única empresa brasileira capaz de suprir programação HDTV de qualidade, forçava o modelo HDTV completo (Figura 2.1(c)). O autor desta tese aposta na adoção do modelo que contempla a interatividade (Figura 2.1(b)).

[9]A indústria costuma designar este tipo de interação como "walled garden" (jardin cercado).

usuário, principalmente no caso de VoD. Segundo Furht [55], o futuro das redes de comunicação será baseado em sub-redes ATM hierárquicas e interconectadas, distribuídas em áreas geográficas distintas[10]. No topo da hierarquia, existirá uma rede global, que irá conectar várias redes nacionais. Por sua vez, as redes nacionais vão englobar redes metropolitanas, que poderão ser conectadas a redes locais. Toda esta estrutura será implementada sobre as redes já existentes, que passarão a ser substituídas gradativamente por redes de fibra óptica.

Um exemplo do uso de redes ATM para iTV pode ser visto em [104]. Neste sistema, cada usuário é conectado a um servidor, que atende a um determinado número de clientes, localizados em uma determinada vizinhança. Para evitar falhas, os serviços são disponibilizados em servidores que possuem réplicas, que entram em operação quando o servidor corrente falha. Os serviços disponibilizados nesta plataforma, desenvolvidos para a Time Warner, incluem VoD, comércio eletrônico e jogos multi-usuários. Atualmente, transmissões digitais em broadcasting operam sobre satélites, cabos e linhas telefônicas de alta velocidade [76].

O projeto I2TV (Infra-estrutura Internet2 para Desenvolvimento e Teste de Programas e Ferramentas para TV Interativa), que está sendo desenvolvido no Brasil em âmbito nacional por um conjunto de universidades e emissoras de TV [12], também baseia-se no uso da Internet como meio de difusão do conteúdo digital. Este projeto visa a formação de um consórcio para desenvolvimento e teste de aplicações e ferramentas para TV Interativa em ambiente Internet2. Ele aborda a geração de conteúdo interativo, *middleware*, exibição (desenvolvimento de formatadores e distribuidores de vídeo [13]), e sistemas de rede necessários à distribuição do conteúdo interativo. A abordagem deste projeto sugere uma integração de tecnologias de broadcast de vídeo digital com a WWW, tendo em vista a quantidade de aplicações interativas que podem surgir como conseqüência desta integração.

Outro projeto, coincidentemente com a mesma sigla, é o i2TV (*Interactive Internet TV*) [105]. Este é um projeto experimental que visa criar um ambiente de "Realidade Mesclada" (*Mixed Reality*) que permita a interação de usuários da Internet (participantes *on-line*) com eventos ocorrendo em um local real (participantes on-site), como por exemplo uma conferência. Os dois tipos de participantes devem se sentir envolvidos (imersos neste novo ambiente mesclado), e vivenciar detalhes das duas

[10] ATM deixou de ser uma tecnologia atrativa face às novas possibilidades das redes gigabit. O conceito a ser mantido é o de Qualidade de Serviço (QoS - *Quality of Service*).

situações que estão sendo integradas (on-line e on-site). Nesse projeto faz-se uso de uma combinação de ambientes 3D, TV digital, interfaces de comunicação móveis, *broadcast Internet streaming*, e mediadores (*medial staging on-site*) de modo a integrar Internet e o modelo passivo de trasmissão de TV por broadcast em uma novo meio de TV interativa baseada em *Mixed Reality*. Os mediadores se fazem necessários para gerenciar múltiplas interações provenientes dos usuários on-line sob o evento, sendo realizado em tempo real, em um espaço real, de modo que possam se tornar uma parte integrante do evento.

2.2 Usuário versus Telespectador

Com as propostas da iTV, uma grande mudança deve ocorrer na forma como a TV é usada e manipulada. A possibilidade do usuário poder interagir com o conteúdo que está sendo apresentado, bem como poder escolher o que deseja assistir, é sem dúvida um grande avanço para uma tecnologia que permaneceu inalterada durante várias décadas.

Entretanto, antes de desenvolver todo um aparato tecnológico de redes, processadores e aplicativos, deve-se perguntar ao usuário sobre sua intenção de transformar o mais comum instrumento de diversão, entretenimento e cultura em um dispositivo que, possivelmente, não será operado em sua total funcionalidade sem a intervenção humana. Além disto, põe-se em questão se as pessoas estarão dispostas a pagar mais pelos novos serviços interativos disponibilizados pelos provedores [138]. Este novo paradigma vai exigir das pessoas uma mudança da velha e tradicional atitude passiva por outra onde a interatividade é a palavra chave.

Independente da resposta do usuário ser positiva ou negativa, várias frentes de pesquisa estão avançando na tentativa de definir as linhas mestras desta revolução tecnológica. Entretanto, nem todos têm a mesma opinião sobre os rumos da iTV, o que está levando ao surgimento de tecnologias com dois enfoques divergentes, cujo ponto chave é a determinação do grau de interatividade que será oferecido ao telespectador. Será vencedora a solução que melhor atender às necessidades dos usuários e do mercado.

A primeira, encabeçada pela Microsoft [86], resume a interatividade com o termo "interatividade preguiçosa" (*lazy interactivity*), ou seja, a iTV terá sucesso se exigir pouca atenção e esforço do usuário. Já a outra corrente, liderada pela Sun Mircrosystems, sustenta a definição de uma arquitetura

que possa oferecer alto poder de processamento e, conseqüentemente, maior liberdade de interação ao usuário.

A Microsoft está se concentrando no desenvolvimento de interfaces intuitivas e divertidas para os usuários. O principal desafio é que estas interfaces precisam ser mais eficientes do que as usadas em computadores atuais e ao mesmo tempo devem ser mais fáceis de aprender e usar. Para isso, está sendo feito uso de computação gráfica de alta qualidade [19].

À medida que a iTV evolui, ela se torna mais semelhante a um computador no que diz respeito tanto a dispositivos de hardware, softwares, poder computacional, potencialidades e, conseqüentemente, nível de interação, quanto na susceptibilidade a erros, o que pode resultar em eventual insatisfação do usuário.

Até o momento, não foi feita uma distinção entre usuário e telespectador. Os dois termos foram usados de forma indistinta. Entretanto, existe uma diferença muito clara entre os dois termos. Usuário, em termos computacionais, é a pessoa que usa um dispositivo para realizar uma determinada tarefa. Por outro lado, o telespectador é somente um membro passivo, ou seja, que não interage - apenas recebe informação.

Em se tratando de iTV, o termo telespectador já não é o mais apropriado, uma vez que ele necessita interagir, transformando-se em usuário de um serviço ou aplicativo com que interage. O nível máximo de interação coincide com a equiparação do uso da iTV com um computador pessoal.

Atualmente, o usuário pode escolher o grau de participação, ou seja, pode se enquadrar como um telespectador, como um usuário, ou em uma posição intermediária. À medida que o poder computacional dos set-top boxes aumenta, há uma tendência ao aumento do grau de interatividade oferecido e de potencialidades de novas aplicações e conteúdos.

Neste trabalho, o telespectador da TV tradicional será considerado como um usuário, visto que este poderá, em algum momento, interagir com a informação recebida ou, em termos mais genéricos, com o conteúdo recebido. Esse aspecto é um dos mais polêmicos referentes à iTV e está intimamente relacionado com o tipo de conteúdo oferecido, discutido em mais detalhes na próxima seção.

Cabe ressaltar que é muito mais fácil desenvolver processadores poderosos e redes de alta velocidade do que mudar a mentalidade de uma massa de usuários que, durante décadas, viam a TV apenas com um meio passivo de transmissão de informações, diversão e lazer.

2.3 Geração de Conteúdo Interativo

A possibilidade de interação do telespectador/usuário com o conteúdo recebido depende não apenas dos recursos de software e hardware disponíveis no cliente para permitir esta interação. É necessária a criação de conteúdo interativo específico para tal fim.

A geração de conteúdo interativo está trazendo à tona muitos desafios e, como conseqüência, novas áreas de pesquisa estão surgindo. Muitas delas se concentram em adaptar interfaces e programas desenvolvidos para computadores aos hábitos do novo público-alvo emergente.

Os provedores devem se familiarizar com as mudanças necessárias para transformar a velha transmissão broadcasting em algo que possa ser personalizado e direcionado a cada usuário. Além da adaptação, também devem ser levados em conta os novos custos de produção da programação [138], o que em muitos casos pode resultar em prejuízos. Para contornar este problema, uma solução é associar conteúdo interativo com comércio eletrônico [22]. A questão de tecnologias para a geração de conteúdo interativo é analisada na Seção 2.3.2.

2.3.1 Aplicações e Serviços para iTV

Atualmente, um dos principais usos da iTV é para guia de programação (EPG - *Electronic Program Guide*), que consiste em exibir uma interface gráfica com informações que auxiliam o usuário na escolha de programas, canais, *pay-per-view*, VoD (*video-on-demand*) e diversos outros conteúdos disponibilizados nesse ambiente digital [77]. Serviços mais avançados de EPG oferecem recursos de Internet, como navegação e e-mail.

Os EPGs são aplicativos que podem ser configurados pelo provedor ou pelo próprio usuário e, por geralmente serem serviços gratuitos, são uma ótima opção para a exibição de comerciais e anúncios. Um exemplo é a WebTV for Windows [144]. Pelo fato dos EPGs estarem se tornando um portal para outros conteúdos interativos, é de vital importância que os provedores dominem esta tecnologia e possam disponibilizá-la da melhor forma [138].

O conteúdo interativo pode estar mesclado com a programação corrente ou pode ser acessado separadamente. Podem-se citar, como exemplos de conteúdos não mesclados, as EPGs que informam a previsão do tempo, placares de jogos e notícias.

Em situações onde o conteúdo é mesclado, geralmente usam-se elementos gráficos semitransparentes, preferencialmente localizados em um canto da tela da TV. Esta técnica pode ser usada para visualização de estatísticas de jogos, competições automobilísticas, ou qualquer outra aplicação em que o telespectador tenha interesse em receber informações adicionais, além dos fluxos convencionais de áudio e vídeo presentes na TV. Outra estratégia de exibição de conteúdo é o uso de caixas de diálogo, que podem ser usadas para exibir ou capturar informação do usuário, muito utilizadas para anúncios e comércio on-line. Além das aplicações já mencionadas, podem-se ainda citar recursos de acesso a contas bancárias, escolha do ângulo de visão em partidas de futebol, acesso a cenas de capítulos anteriores, dentre outras.

Segundo Furht [55], os principais serviços disponibilizados atualmente, além dos já mencionados, incluem entretenimento interativo, serviços de navegação, jogos interativos mono e multi-usuários, serviços educacionais e instrucionais como ensino à distância, bem como versões eletrônicas de jornais, revistas e páginas amarelas.

2.3.2 Tecnologias para Geração de Conteúdo Interativo

Os principais problemas para a geração de conteúdo para iTV estão no desenvolvimento de formatadores e editores que trabalhem com a sincronização de mídias. O grupo Telemídia (www.telemidia.puc-rio.br) apresenta excelentes propostas para uma linguagem hipermídia (*Nested Context Language* - NCL), um formatador (*HyperProp*) [128] e uma ferramenta de autoria (*JEdictor*). O comprometimento deste grupo de pesquisa com a linguagem Java possibilita o uso das ferramentas propostas em qualquer plataforma.

Existem várias outras tecnologias para dar suporte ao desenvolvimento de conteúdo interativo. Um exemplo do uso de VRML (*Virtual Reality Modeling Language*) [143] pode ser encontrado no "*The Steerable Media Project*" [95]. Nessa pesquisa, são usados os conceitos de camadas para mesclar vídeos e gráficos 2-3D, por meio de uma linguagem de marcação chamada Blendo, que estende a linguagem VRML. Quando estes elementos chegam ao cliente, é realizada a composição, em tempo dinâmico, de acordo com as preferências do usuário, por meio de um motor de síntese (*Blendo engine*)[11], que também faz uso da biblioteca gráfica OpenGL [147]. Nesse

[11] Blendo é um motor de apresentação multimídia desenvolvido pela Sony que deriva

projeto, se aposta na contínua queda da relação preço/desempenho de processadores gráficos 3D, bem como de hardwares em geral, para a criação de dispositivos capazes de sintetizar vídeo com alta qualidade, com uso de técnicas de Computação Gráfica, a um custo que os usuários estarão dispostos a pagar.

Além de VRML, é também usada a linguagem Flash [51], com extensões que permitem trabalhar com fluxos múltiplos de vídeo, habilidade para renderizar elementos gráficos 2-3D, além de permitir desenvolvimento de aplicações em linguagem C++ e Java. A idéia do projeto é a criação de um cenário para a apresentação de notícias, cujo conteúdo possa ser guiado pelo usuário. Este conteúdo a ser apresentado não deve se parecer nem com a tradicional TV, nem com páginas WEB, o que ocorre no caso da WebTV [144], desenvolvida pela Microsoft.

A WebTV, por sua vez, faz uso de HTML para estruturação da informação exibida. Com esta tecnologia podem-se apresentar game shows interativos, notícias e eventos esportivos. Todos os elementos da informação, tais como vídeo e textos, são apresentados em janelas distribuídas na tela. Textos e gráficos são renderizados em janelas separadas, o que faz com que o conteúdo apresentado seja muito semelhante ao de uma página Web. O grau de interatividade e liberdade de geração de conteúdos interativos é também muito limitado.

Gibbs et al. [60] estendem o motor Blendo para suportar transmissões esportivas em TV interativa. O protótipo desenvolvido permite desde a exibição de estatísticas simples sob demanda, como, por exemplo, visualizar a telemetria de um determinado carro, até a integração com jogos de computador. Tanto em [95] como em [60], os experimentos foram realizados sobre estações de trabalho SGI rodando Windows NT.

2.4 Processamento e Exibição de Conteúdos Interativos

A infra-estrutura para iTV consiste na geração, transmissão e exibição de conteúdos interativos. Esse conteúdo, como já foi mencionado, pode consistir de fluxos simples de áudio e vídeo, mas pode também agregar conteúdos interativos, tais como aplicativos, gráficos, imagens e caixas de diálogo. A manipulação desse conteúdo digital exige processamento computacional adicional no cliente. Essa tarefa é realizada pelos set-top boxes, que são dispositivos que agem como meio de comunicação entre a

do VRML.

TV, digital ou analógica, e o meio de transmissão dos dados (via satélite, cabo, terrestre) [76]. Segundo Driscoll [42], "O Set-top Box está situado entre a supervia de informação digital e o aparelho de TV". Diversas outras nomenclaturas para estes dispositivos podem ser encontradas em [78]. Neste trabalho será usado o termo set-top box.

O set-top box recebe o sinal digital comprimido e codificado e o decodifica, transformando-o em sinais analógicos que possam ser exibidos nas TVs. Eles podem ser vistos como computadores dedicados com interfaces via display e outros dispositivos de interação como mouse, teclado e controle remoto, por onde o usuário pode interagir com o conteúdo em apresentação. Dependendo da ação do usuário, faz-se necessário o uso de uma linha de retorno de dados com o provedor para que as novas requisições ou comandos possam ser atendidos. Esta linha de retorno é geralmente muito mais lenta do que a de transmissão de streams, e pode ser implementada por uma linha telefônica, *cable modem*, DSL, comunicação via satélite, etc.

A TV interativa é muito mais que vídeo sob demanda (VoD). Ela é um novo meio pelo qual o telespectador tem tanto educação quanto entretenimento. O set-top provê acesso a esse meio, e serve como uma ponte de comunicação entre um vasto repositório de informação multimídia e uma ferramenta que pode navegar nesta informação, criando para cada usuário uma apresentação diferenciada [19]. Todos estes recursos necessitam de processamento adicional, que deve estar presente no set-top box e que pode ser assim considerado como um motor de síntese.

2.4.1 Arquitetura de Hardware do Set-top Box

Os set-top boxes possuem interfaces de rede e decodificadores para capturar e processar os fluxos (vídeo, áudio ou dados) que, para economia de banda passante, são comprimidos. Também devem possuir buffers para garantir continuidade de exibição dos fluxos, em casos de atrasos da rede, e mecanismos de sincronização de exibição de áudio e vídeo. Além disso, a existência de barramentos, memórias, CPUs, unidades de armazenamento, processadores gráficos e dispositivos de entrada e saída, tornam a arquitetura de um set-top muito parecida com a arquitetura de computador pessoal. Segundo Furht [55], vários dispositivos poderão estar conectados aos set-top boxes tais como VCR, controle remoto, CD-ROM, Impressoras, HD e teclados.

Os set-top boxes podem oferecer recursos de atualização de software como browsers e EPGs. Alguns podem também armazenar localmente programação a ser exibida no momento que o usuário desejar [77].

O conceito de set-top não é totalmente novo. Ele já existe há algum tempo, porém se limita a apenas receber informação analógica e mostrar na TV. Esta primeira versão de set-top box não possui linha de retorno e disponibiliza baixo poder computacional. Em termos cronológicos, este é considerado o set-top de primeira geração, também chamado de *Broadcast TV Set-top Box*.

A geração seguinte de set-top, também conhecida como *Enhanced TV Set-top Boxes*, já possui canal de retorno lento, implementado por meio de uma linha telefônica, e permite serviços de VoD, comércio eletrônico e recursos de Internet, como navegação, e-mail e Chat.

Da terceira geração em diante, os set-tops já disponibilizam recursos de armazenamento de dados e possibilidade de executar jogos, tudo isso aliado a redes de maior velocidade. O que diferencia também as gerações de set-top boxes é o poder computacional, que se torna maior a cada nova geração.

2.4.2 Arquitetura de Software do Set-top

A arquitetura de software de um set-top é muito semelhante à de um computador. Geralmente, ela é divida em 3 camadas: sistema operacional, *middleware* e aplicativos. Estas camadas de software se fazem necessárias para dar ao set-top a possibilidade de executar as ações do usuário, bem como processar e exibir o conteúdo interativo.

2.4.2.1 Sistemas Operacionais

Set-top boxes fazem uso de sistemas operacionais (SO) de tempo real (também denominados RTOS - *Real-Time OS*), especialmente porque devem processar mídias contínuas, como áudio e vídeo, sem interrupção e em tempo real. Como os set-tops ainda têm recursos de hardware limitados quando comparado com microcomputadores, o SO precisa ser robusto, compacto e principalmente confiável, visto que telespectadores não estão acostumados com bugs, resets, nem com pausas para CPU, disco ou rede, resultado de tarefas intensas. Os usuários de iTV esperam que o tempo de

resposta nunca seja maior que meio segundo. Além disso, esperam que a TV e seus recursos estejam sempre disponíveis [104]. Para rápida inicialização, o SO deve ser preferencialmente armazenado em ROM. Outra exigência do SO é a capacidade de processar concorrentemente diversas tarefas, que vão desde processar o vídeo de entrada até a validação de mensagens.

A abstração do hardware é realizada pela definição do SO em camadas. A camada mais baixa consiste em um conjunto de drivers e abstrações de software que fazem interface diretamente com o dispositivo físico. Com esta estratégia, desenvolvedores podem portar mais facilmente o SO para múltiplas plataformas de hardware. Também é comum a presença de drivers para acesso de teclados, portas, modems e discos [42]. Outra camada comum é o kernel, que tem a função de gerenciar os recursos do set-top, como memória e prioridade de processos.

A seguir são apresentados alguns sistemas operacionais que compartilham das mesmas características desejáveis em um SO para set-top, que incluem: arquitetura de tempo real, organização em camadas para abstrações de hardware, tamanho reduzido, confiabilidade e, principalmente, disponibilização de uma API ou gerenciadores de SO para desenvolvimento de aplicativos.

PowerTV [112]: Desenvolvido pela empresa Scientific-Atlanta especialmente para rodar em set-top, é armazenado em ROM e pode ser atualizado facilmente pela rede. Foram incorporados no PowerTV gerenciadores de sistema operacional para facilitar o desenvolvimento de aplicações, bem como acesso a periféricos.

VxWorks [141]: Desenvolvido pela empresa Wind River Systems, este sistema operacional pode ser usado em set-top boxes, telefones inteligentes, sistemas de navegação para carros e agendas portáteis (*handhelds*). Oferece diversos recursos de acesso à rede, incluindo TCP/IP, FTP e Telnet. É exportado para várias plataformas de hardware, incluindo PowerPC, Intel, Sparc e MIPS. Para manter a compatibilidade de hardware, são fornecidos códigos fonte dos drivers. Oferece suporte a vários browsers também.

Microsoft Windows CE: O Windows CE é uma versão compacta do Windows, feita para rodar em dispositivos com recursos de hardware limitados, como set-tops e *handhelds*. A versão para set-top segue a filosofia cliente-servidor. Uma característica importante deste SO é a possibilidade de executar aplicativos Java ou applets usando uma

JVM - *Java Virtual Machine* - específica para Windows CE. Outro ponto de destaque é o suporte à tecnologia DirectX [93], que permite acesso direto da aplicação aos recursos de hardware para processar multimídia [42].

JavaOS [82]: A tecnologia Java para sistema operacional também é fundamentada na definição de camadas. A base do sistema é um kernel, baseado na tecnologia ChorusOs [31], que é amplamente usada em dispositivos como celulares e sistemas de comunicação. Como camada superior, existe a J2ME - *Java Micro Edition*, que é destinada a fazer a comunicação com aplicativos usados no set-top [42].

Linux: O Linux está chamando a atenção dos fabricantes de set-tops e está se tornando um dos principais concorrentes do Windows CE, principalmente por ser gratuito e ter baixa exigência de hardware. Um exemplo de projeto nacional de pesquisa em iTV que faz uso deste sistema é o I2TV [121], que visa desenvolver tecnologias e ferramentas de suporte à TV Interativa no ambiente Internet2. Além do Linux, neste projeto faz-se uso da API JAVATV como middleware entre as aplicações e a sub-camada de hardware.

2.4.2.2 Middleware

Middleware é um termo relativamente novo em se tratando de set-top. Ele é o centro da nova arquitetura de software, consistindo de uma camada de conexão que age como uma ponte de comunicação entre o SO do set-top e a aplicação do usuário. Esta abstração facilita a migração das aplicações entre diferentes sistemas operacionais.

Em um ambiente set-top, o middleware consiste de alguns componentes, comumente conhecidos como máquinas virtuais. Pode-se dividir os padrões de middleware em 3 categorias [131]:

- Baseada em HTML, Javascript e CSS (*Cascade Style Sheet*), que é definida pelo padrão ATVEF (*Advanced Television Enhancement Forum*). Este padrão de middleware é a tecnologia adotada pela Microsoft na WebTV;
- Tecnologias proprietárias como OpenTV [106] e WebTV, e;
- Java TV [22], que é vista como a tecnologia da nova geração no que se refere ao desenvolvimento de conteúdo para TV interativa.

Sobre a tecnologia de middleware é que são implementadas a maioria das aplicações para iTV. O middleware permite às aplicações operarem transparentemente sobre uma rede sem ter que se preocupar com os protocolos subjacentes. Esta consideração reduz a complexidade de desenvolvimento porque as aplicações podem ser escritas tirando proveito de uma API comum.

Padrão HTML: HTML é sem dúvida a linguagem dominante na Internet, devido à sua simplicidade [86]. Foi apostando nesta característica que a Microsoft focou seus esforços em produzir sua versão de TV interativa. Para estender os recursos da linguagem HTML, usa-se a linguagem JavaScript, que associadas à norma ATVEF, permitem que conteúdos interativos baseados nesta tecnologia possam ser distribuídos para diversas arquiteturas de set-tops. Como resultado de sua concepção, o HTML oferece recursos para diversos propósitos simples. Ele porém não foi projetado para ser usado em TV interativa, pois apresenta limitações no controle de como o layout da tela é renderizado. Outro ponto fraco se refere a elementos dinâmicos, como animações. Pelo fato de não haver processamento local, a maioria dos recursos de interação só pode ser realizada com uma comunicação bidirecional com o servidor (paradigma cliente/servidor). Uma das maiores limitações desta tecnologia está no desenvolvimento de jogos. Segundo [131], o HTML não é um verdadeiro padrão de middleware, pois não existem certificações formais quanto ao seu interpretador, principalmente no que se refere a extensões da linguagem, o que produz, de forma semelhante à Internet, diversos bugs e incompatibilidades. Desta forma, os set-tops que não foram desenvolvidos no padrão WebTV, mas que têm interpretadores HTML, poderão ter problemas de incompatibilidade. Apesar da grande quantidade de pontos fracos, esta tecnologia é adotada por grandes empresas devido à sua simplicidade e pela influência de marketing exercida pela Microsoft. Os principais usos desta tecnologia são: comerciais, propaganda, notícias, clima, programas de auditório, comércio eletrônico, enfim, nada que exija grandes recursos de processamento no lado do cliente.

Padrão Java: Combinando-se a robustez da linguagem Java com as iniciativas da Sun em favor da TV Interativa, nota-se que a plataforma Java oferece um poder de expressão muito grande em relação ao desenvolvimento de aplicações para set-tops. Ele é composto por uma versão mais leve da máquina virtual Java (J2ME - *Java 2 Micro*

Edition) e por um conjunto de bibliotecas. O Java é sem dúvida uma das tecnologias mais modernas e mais poderosas em se tratando de TV interativa, pois disponibiliza uma API completa de programação que independe de plataforma. Outra característica importante diz respeito à segurança, pois sendo uma linguagem interpretada, ocorre uma verificação dos bytecodes antes de sua execução. Isso assegura ao Java alta confiabilidade, o que é uma característica essencial em TV [22]. Cabe, porém, observar que a linguagem Java, pelo fato de ser interpretada, é fortemente penalizada pela baixa eficiência quando comparada com linguagens compiladas como C e C++.

Padrão MHEG-5: Além das máquinas virtuais HTML, Javascript e Java, existe também um outro padrão de middleware chamado MHEG-5 [20], adotado na Europa, o qual foi desenvolvido para tratar com informações multimídia e hipermídia. O termo multimídia refere-se às diversas mídias suportadas, incluindo áudio, vídeo, gráficos e texto. O termo hipermídia é descrito como um aperfeiçoamento que permite navegar na tela por meio de links. Além da definição de objetos multimídia e hipermídia, o padrão também aborda a transferência destes objetos da base de dados até às redes de comunicação. Uma vez que os dados multimídia/hipermídia sejam agrupados no servidor, em um fluxo de bits, são enviados por uma rede até o set-top, onde reside o motor MHEG-5, que extrai a informação, a interpreta e faz a sua exibição na tela do aparelho de TV. O motor requer apenas 300Kb, logo pode ser implantado em sistemas com baixos recursos. Um novo recurso adicionado a este padrão é a possibilidade de interpretar código Java, o que dá a ele a possibilidade de executar operações complexas e acessar serviços em dispositivos remotos, como servidores, localizados nos provedores. Além deste recurso, também estão sendo adicionados ao padrão características como a presença de canal de retorno e dispositivos de entrada, como teclados sem fio. Como principais aplicações deste padrão podem-se citar: EPG, VoD, comércio eletrônico e jogos on-line.

Tecnologias Proprietárias: A maioria dos especialistas em iTV concorda que nenhum padrão de middleware em particular vai controlar o mercado de set-top num futuro próximo, o que está despertando algumas companhias de software a desenvolverem produtos de middleware para set-top. Pode-se citar como exemplos a OpenTV, PowerTV, PlanetWeb e Liberate Navigator. Uma lista completa pode

ser encontrada em [79]. Estas novas tecnologias proprietárias resolvem muitos dos problemas apresentados no HTML, porém impõem outros, principalmente no que se refere à natureza não padronizada destes ambientes. As tecnologias proprietárias na sua maioria dão suporte aos padrões vistos anteriormente. Cada uma oferece suas próprias APIs, ferramentas e serviços. Isso torna as aplicações não portáveis, exigindo que versões específicas sejam desenvolvidas para diferentes plataformas [131]. Entretanto, segundo [106], a versão EN2 do middleware da OpenTV é independente de plataforma, modular, extensível e dá suporte a diversos SOs, como pSOS, VxWorks, Nucleus Plus, microTOS, OS2O, etc. Esta versão também oferece uma biblioteca que inclui funções gráficas, de rede, manipulação de fluxos de vídeo, criptografia, dentre outros. Mesmo sendo multiplataforma, o desenvolvedor do OpenTV não disponibiliza os códigos fontes.

2.4.2.3 Aplicativos

Na camada mais alta do ambiente de software estão os aplicativos. A idéia é que a aplicação possa fazer uso das camadas inferiores, como mostrado na Figura 2.2. Esta característica pode trazer ganho de desempenho, porém ao mesmo tempo impossibilita sua execução em diversas plataformas.

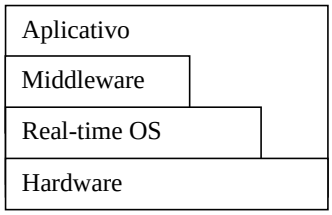

Figura 2.2: Camadas de software em um set-top box.

Como já foi mencionado, o poder computacional e o desempenho destes aplicativos dependem fortemente das linguagens de programação disponibilizadas, bem como das APIs e middlewares presentes na arquitetura.

2.5 Conclusões e Discussões

Atualmente as potencialidades e possibilidades de uso que a iTV pode oferecer não estão completamente definidas. Ainda assim parece haver um consenso de que a iTV constitui uma tecnologia muito promissora, visto que este novo paradigma de TV está chamando a atenção de emissoras de TV, instituições de pesquisa acadêmica, bem como diversos ramos da indústria que possuem alguma ligação tecnológica com toda a infra-estrutura necessária para disponibilizar conteúdos interativos. No Brasil, a iTV ainda se encontra em um estágio de estudo por parte do governo e instituições de pesquisa, que têm a incumbência de adotar algum padrão já existente, ou desenvolver um próprio. Três padrões estão sendo estudados: o padrão europeu (DVB - *Digital Video Broadcasting*) [47], o americano (ATSC - *Advanced Television System Committee*) [6] e o Japonês (ISDB - *Integrated Services of Digital Broadcasting*). Maiores detalhes destes padrões podem ser vistos em [126].

O grau de evolução da iTV não está associado somente com a evolução do hardware (no caso, o set-top box), mas também com as linhas digitais de transmissão de dados. A quebra de paradigmas é também outro fator muito importante, senão crucial, para a evolução e definição de novas gerações de conteúdos e programas interativos.

Segundo Becker [13], a especificação do que deve ser o conteúdo interativo é o principal fator que deve ser estudado e pesquisado, ao contrário da especificação de padrões tecnológicos a serem utilizados (Europeu, Americano ou Japonês), como ressaltado por [71], uma vez que todos estes padrões estão cada vez mais parecidos.

Neste trabalho propõe-se o uso de um ambiente de Histórias Interativas como conteúdo para esta nova mídia, o que pode ser visto também como uma contribuição original desta tese. Pelos dados apresentados neste capítulo, pode-se observar que a infra-estrutura de hardware e software dão suporte à exibição deste tipo de conteúdo, pois disponibilizam linhas de transmissão digitais (em muitos casos o uso da própria Internet) e dispõem de hardware dedicado ao processamento da mídia digital. Em termos de aplicativos, observa-se a existência de diversos middlewares que abstraem detalhes específicos do hardware. Para garantir maior compatibilidade de execução em diversas plataformas, neste trabalho optou-se pelo uso de linguagens e bibliotecas multiplataformas, como o Java, C++ e OpenGL. Não é o foco deste trabalho criar software para uma plataforma específica de set-

top box. Todos os processamentos necessários são realizados em um PC. Apesar destes terem um desempenho superior aos modelos atuais de set-top boxes, assim como [95], também aposta-se na existência futura e breve de set-tops com recursos computacionais abundantes, que possam permitir simulações que atualmente são possíveis apenas em computadores de última geração.

Detalhes mais específicos da integração entre TV interativa e a ferramenta proposta são discutidos no Capítulo 6, que apresenta a especificação da arquitetura do sistema.

3
Narração de Histórias (Storytelling)

Tanto o conteúdo como os novos paradigmas de interface estão revolucionando a maneira como as pessoas interagem com computadores. No que tange à narração de histórias, a evolução para um meio que proveja maior acessibilidade, em substituição ou complementação às antigas interfaces, é alvo de pesquisas recentes. Pode-se observar que conteúdos lineares (como livros) não se mostram mais suficientes para atender à nova demanda de aplicações. Devem-se prover recursos que permitam ir além da narrativa tradicional composta por simples ramificações ou *hiperlinks* [99]. Dentro destas novas expectativas de conteúdo e interface, Glassner [62] discute o significado real de expressões como "hipertexto interativo", "histórias não lineares", "ficção interativa" e "narrativas participativas", no sentido dos pré-requisitos que o conteúdo deve disponibilizar para poder realmente ser chamado interativo.

Storytelling é um novo paradigma de entretenimento digital que está avançando a passos largos com a criação de técnicas e ferramentas que permitem que histórias interativas possam ser criadas, visualizadas e guiadas com o auxílio do computador. Segundo Mateas [97], um sistema de *storytelling* engloba modelos de conhecimento e processamento necessários para contar uma história.

Segundo Spierling [129], agentes conversacionais, personagens virtuais, bem como interfaces de jogos são exemplos do resultado das pesquisas em *storytelling*. O campo de aplicação é muito vasto, variando deste entretenimento, como jogos [151], a aplicações comerciais [129]. Glassner [62] vê *storytelling* como uma forma de unir histórias e jogos, no sentido de aumentar o conteúdo narrativo destes.

As pesquisas em *storytelling* geralmente concentram-se em três linhas distintas: geração, interação e visualização das histórias.

Geração: Refere-se à forma como a história é gerada, ou seja, como se dá a criação da estrutura que vai guiar aspectos mais gerais, como personagens, ações, objetos e relacionamentos;

Interação (direcionamento): Estabelece como se dá a interação entre usuário, história (enredo) e personagens. Também é responsável pelo gerenciamento das ações dos personagens autônomos (agentes), de modo a manter a história coerente. Está intimamente relacionado com o elemento que controla a IA e o nível de autonomia e decisão que cada agente pode desempenhar;

Exibição: Trata a forma de representação gráfica da história, ou seja, a transformação das abstrações das estruturas internas dos personagens em ações realistas dentro de um espaço gráfico, comumente 3D.

Geralmente, todos estes processos são encapsulados por motores de enredo (*story engines*), que são ferramentas que gerenciam e sincronizam os eventos nos mais diversos níveis de abstração, assegurando o direcionamento lógico da história. Drama interativo (*Interactive drama*) é a classe de narrativas geradas por estes motores [64]. Segundo Mateas [96], drama é a junção de Personagem, História e Apresentação. O projeto "OZ" [108], que atualmente não está mais em andamento, foi um dos trabalhos precursores na quebra de paradigmas relativos à criação de mundos de histórias interativas. Procurava integrar drama interativo e agentes realistas, que pudessem exibir personalidades, emoções, comportamento social, motivações e objetivos.

Existem diversas abordagens na literatura para encarar este desafiante e novo meio de interação com a mídia digital. As próximas sessões fazem uma análise destes processos, procurando mostrar diversas estratégias adotadas na implementação dos motores de enredo.

3.1 Geração de Histórias

Qualquer sistema de *storytelling* opera sobre algum tipo de dado que possa representar a história com a qual o usuário interage. Estes dados podem consistir, por exemplo, de regras, personagens, planos, tipos de cenas e eventos. A forma de geração desta configuração inicial é algo questionado por Grasbon [64]. Segundo ele, máquinas não têm capacidade de criar dinamicamente histórias com detalhes convincentes, visto que é uma tarefa que envolve um ciclo iterativo de especificações e testes, de modo a assegurar que o conteúdo gerado seja ao mesmo tempo interativo e coerente. Como apresentado na Figura 3.1, o motor proposto por Grasbon deixa a cargo do

autor, em uma fase de pré-processamento, a criação de dados que definem o modelo global da história.

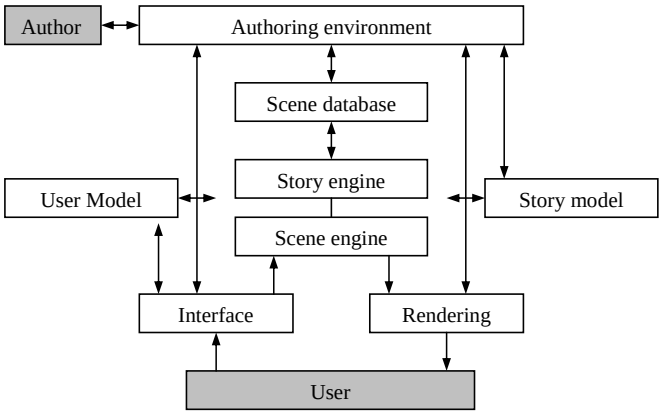

Figura 3.1: Diagrama esquemático da estrutura de um motor. Reproduzido de Grasbon [64], pg. 338.

No diagrama da Figura 3.1 deve-se observar que o usuário não é o autor da história e, conseqüentemente, a interação ocorre somente sobre os dados que já foram definidos pelo autor. Este aspecto é discutido em mais detalhes por Spierling [129], que procura localizar onde ocorre a "interação", dentre os processos de geração e apresentação da história. Segundo ele, somente o "contar" ("*telling*"), é realizado de modo interativo. Como os motores operam sobre histórias pré-definidas, com um enredo que descreve fatos e ocorrências, apenas o autor da história tem o poder de alterar e definir elementos que regem aspectos mais globais da narrativa. O papel do usuário é fornecer parâmetros usados pelo motor para direcionar a história, que consiste em "contar" fatos que foram definidos em uma etapa anterior.

Mesmo deixando a cargo do autor o processo de definição da estrutura básica da história, este ainda é um processo complexo. Segundo Spierling [129], para criar histórias interessantes, o artista deve iterativamente, ir construindo modelos e ao mesmo tempo ir interagindo com os mesmos, em um trabalho integrado, geralmente utilizando uma abordagem ascendente (*bottom-up*). Devido a esta constatação, ele define um modelo em quatro níveis de abstração, oriundo do estudo de vários projetos já criados em *storytelling* (Figura 3.2).

Na proposta de Spierling [129], parte-se de modelos e dados pré-definidos, sobre os quais o autor deve interagir iterativamente. Este modelo visa tratar problemas inerentes da estrutura da narrativa, estrutura de agentes e, finalmente, aspectos da representação gráfica dos elementos animados. Tanto a influência de usuários como a de autores podem ser

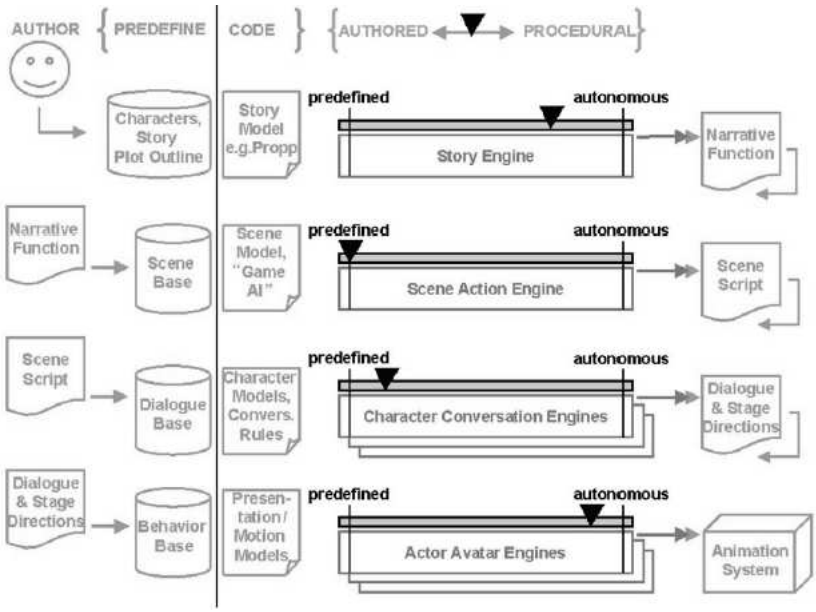

Figura 3.2: Modelo abstrato de quatro níveis com autonomia configurável para geração de enredos interativos. Reproduzido de Spierling [129], pg 35.

testadas em cada estágio, o que conduz à geração de conteúdo interativo com maior atratividade.

Na Figura 3.2, cada camada inferior faz uso dos dados provenientes da camada superior. O *Story Engine* controla o fluxo da história, ou a ordem dos eventos - que são funções proppianas [115] - no nível mais abstrato. Estas funções são específicas para determinados tipos de histórias. O *Scene Action* faz a escolha de cenas, definidas em scripts, que incluem detalhes de câmera e ações concretas de personagens. No próximo nível, são tratados relacionamentos, personalidades, diálogos e outras operações entre agentes. No nível mais baixo (*Presentation*), disponibilizam-se recursos para a representação de animações, diálogos e sons.

O autor pode definir em cada estágio o nível de autonomia que o respectivo motor pode exercer, o qual pode ser totalmente autônomo, totalmente manual ou uma combinação de ambos. Dependendo do grau de autonomia em cada motor, gêneros diferentes podem ser criados.

3.2 Interação, Direcionamento e Modelo da História

A estrutura interna de representação da história é um fator que tem grande influência sobre a forma como ocorre a interação do usuário. O nível

de interação, ou controle na direção do fluxo da narrativa que está sendo exibida, pode ser tratado sob diferentes abordagens.

Em uma abordagem orientada a enredos (*plot-based*), tanto a interação como a geração do enredo são realizados sobre operações em alto nível, como em Grasbon [64], ou seja, o usuário não tem permissão de a qualquer momento alterar qualquer atributo. Neste tipo de abordagem, tem-se um maior controle sobre a história que está sendo apresentada, de forma a evitar que fuja do contexto inicial definido pelo autor. A estrutura geral da história é previamente gerada pelo autor e pode já definir um início, um meio e um fim. O usuário tem a liberdade de realizar interferências "sutis" que apenas têm como objetivo guiar o andamento para chegar aos pontos predefinidos. Geralmente faz-se uso de uma estrutura de nós, dispostos em forma de um grafo ou rede, onde cada nó tem, como conteúdo, um evento da história, informação sobre um personagem ou uma posição discreta no ambiente. Existem conexões entre os nós que, ao serem percorridos, definem o formato da apresentação da narrativa [100].

Em uma abordagem orientada a personagens (*character-based*) [25][98][150][26][24][28][27], a interação é em mais baixo nível, permitindo maior possibilidade de alteração de atributos que influenciam no desenvolvimento do enredo. A geração do enredo é baseada no comportamento e na interação dos personagens (agentes) autônomos. Esta abordagem, também chamada de "narrativa emergente" [129], apesar de ser muito importante em *storytelling*, pode ser perigosa, visto que tecnologias existentes podem não ser capazes de criar roteiros coerentes e histórias com muito significado [64]. Abordagens que permitem que o usuário faça um controle muito detalhado podem conduzir ao mesmo tipo de problema.

Segundo Cavazza [25], a estratégia orientada a personagens se mostra unificada tanto para tratar geração como interação de histórias. Por isso, não restringe a interação somente a pontos chave durante a representação, que é o que ocorre com abordagens orientadas a enredos. Entretanto, formalismos de planejamento ainda fazem-se necessários para especificar aspectos de autoria relacionados com a estrutura geral da narrativa.

Em se tratando de modelos orientados a enredos, a autonomia intrínseca dos agentes pode ser vista como um problema e não como uma solução, visto que o comportamento reativo e local de agentes reativos pode confrontar com aspectos globais e deliberativos da natureza da história [98]. O personagem deve tomar ações necessárias para mover a história na direção esperada. A escolha das ações depende não somente do estado interno do

agente e do estado do mundo, mas também do estado corrente da história, que engloba informações dos outros personagens e ações passadas.

As abordagens orientadas a enredo e personagens tratam, respectivamente, *storytelling* como narração e simulação. No primeiro caso, tem-se uma história parcialmente definida que deve ser contada para o usuário, com algumas possibilidades de interação, desde que limitadas ao escopo do enredo. Tendo-se à disposição um conjunto de parâmetros reais, geralmente em alto nível, a abordagem orientada a enredos significa a possibilidade de processar dados já conhecidos que foram projetados com o intuito de serem ajustados [129]. Na segunda abordagem, a história é criada à medida que os personagens interagem entre si e com o mundo, cujas ações são fortemente influenciadas pela interação do usuário. Apesar de permitir um nível de interatividade muito mais fino, pode tornar a interação um tanto trabalhosa e complexa.

Existem também abordagens híbridas, como no caso de Crawford [38], que faz uso de verbos como componentes básicos das ações, que devem ser definidos pelo autor, visto que também acredita na impossibilidade da máquina gerar histórias coerentes automaticamente. Esta abordagem parece muito adequada, visto que permite combinar eventos dinâmicos com ações previamente definidas.

Em termos mais gerais, existem diversos problemas que devem ser tratados na implementação de sistemas interativos de *storytelling*, dentre os quais podem-se destacar: o papel do usuário, o nível de representação e controle da narrativa, modelos de interação, relações entre personagens e enredos [28]. Alguns destes problemas resultam do conflito entre interação e narrativa, o que classifica a forma como o usuário interage com a história: personagem ou espectador. De qualquer forma, o usuário deve estar de alguma forma envolvido com o controle e direcionamento da história.

Uma solução intermediária para o problema da interação é apresentada por Charles [28], que propõe a criação de um sistema onde existe limitação no envolvimento do usuário com a história, porém permite que a interação ocorra a todo o momento. A história é guiada pelo comportamento de personagens (agentes) autônomos (*character-based*). Como em Cavazza [26], a estrutura central do planejamento e representação do conhecimento faz uso de HTN (*Hierarchical Task Network*), que também é usada como meio de intervenção pelo usuário, tanto em termos de objetivos da narrativa, como ações físicas. Uma HTN pode ser vista como uma representação implícita de um conjunto de possíveis soluções para uma tarefa. Desta forma, cada personagem está associado a uma HTN que contém todas as regras possíveis

para um personagem, para uma dada história. Cada tarefa está associada com pré- e pós-condições.

Na Figura 3.3 é apresentada uma HTN típica (representada por um grafo E/OU) para um personagem, que deve ser pesquisada da raiz (ações genéricas) em direção às folhas (ações terminais) para a geração de planos. Camadas inferiores de um plano correspondem aos vários modos de atingir os objetivos. Para o exemplo da figura, o diário (*diary*) somente pode ser consultado se não estiver sendo usado por outro personagem e estiver em seu local de origem. Como estas condições podem mudar dinamicamente, representam uma boa fonte de interatividade.

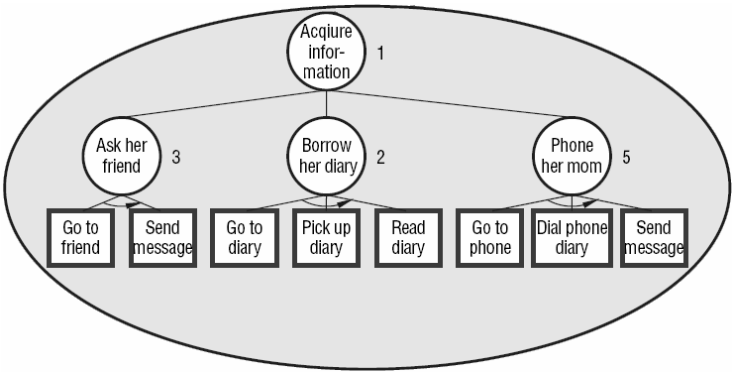

Figura 3.3: Ramo de um HTN de um personagem para a ação de aquisição de informação sobre outro personagem. Reproduzida de Cavazza [25], pg. 19

Cavazza [26] apresenta dois mecanismos diferentes para interação. Por meio de interação física direta com objetos virtuais da cena, o usuário pode alterar a configuração de objetos (adição, remoção ou deslocamento) que têm influência na tomada de decisões dos personagens para a realização de determinadas ações. Outra estratégia utilizada consiste no uso de mensagens lingüísticas, que permitem influenciar o personagem por meio de informações, que podem ser usadas para resolver alguns objetivos. Estas mensagens podem ser instruções para a realização de ações diretas ou dicas do comportamento mais adequado no dado momento. Em vez de estar imerso na história, o usuário é tratado como um espectador.

Estas abordagens de Cavazza [26] e Charles [28] são usadas em cenários com representações de personagens em situações cotidianas (como ocorre em seriados televisivos norte-americanos) baseadas em um romance entre dois personagens principais, como tentativas para marcar encontros, diálogos para descobrir preferências de outros personagens, etc. O protótipo permite quatro personagens, sendo todos baseados em planos HTN. Como ponto

de partida, tem-se a trama que define as regras dos personagens, que são dinamicamente alteradas, dentro de limites do gênero, pela interação entre personagens e pela intervenção do usuário.

Para garantir que as ações sejam relevantes, dado o enredo, em Cavazza [26] o controle do comportamento dos personagens é descrito em termos de regras, com associação entre objetivos e ações correspondentes. Um conjunto ordenado de regras forma um plano. Para garantir diversidade de enredos, vários fatores contribuem para tornar as ações imprevisíveis a partir da perspectiva do usuário:

- A distribuição inicial dos personagens, que tem forte influência na determinação da duração de cada ação, o que conseqüentemente conduz situações diferentes;
- As regras de vários personagens podem ser dinamicamente combinadas, ou seja, pode haver interações entre os planos dos personagens, o que neste caso pode levar a situações onde vários personagens competem por um mesmo recurso;
- Resultados randômicos gerados por ações terminais;
- Estado emotivo do personagem (*mood*);
- Interação do usuário.

Nas abordagens de Grasbon [64] e Spierling [129], faz-se uso de funções morfológicas, definidas por Propp [114]. Estas funções, definidas em alto nível, são usadas para a criação de regras e algoritmos para serem usados pelo motor apresentado na Figura 3.1. A abordagem de Propp se mostra vantajosa para *storytelling* pois integra em um modelo unificado todas as variantes (ramos) que a história pode assumir. A ordem de execução dos eventos é baseada em regras. Maiores detalhes sobre o trabalho de Propp são apresentados no capítulo seguinte.

Em uma proposta de modelo de um sistema de *storytelling*, Mateas [98] expõe alguns requisitos importantes na construção deste tipo de sistema. Em sua abordagem, o usuário interage, em primeira pessoa, com os personagens da história que está sendo processada em tempo real, principalmente por meio de diálogos. Esperam-se transições suaves, resultantes de freqüentes intervenções do usuário, que continuamente direcionem o rumo da história. Ações isoladas devem ser combinadas para formar um estado global do sistema. Em caso de exibição de uma mesma história várias vezes, um fator randômico deve ser adicionado aos eventos, para que a representação seja diferenciada, porém transmitindo a mesma mensagem. Como em outros

sistemas, existe um gerenciador de ações (*drama manager*), que controla o comportamento global dos personagens. Caso, em um determinado momento, um personagem não esteja participando do enredo, este pode assumir seu comportamento autônomo.

Dando continuidade a esta pesquisa, Mateas [99][100] apresenta Façade, um sistema interativo que procura combinar vantagens das abordagens baseadas em enredos e personagens, de forma a minimizar as fraquezas das duas estratégias. No geral, a simulação é o principal elemento de controle dos personagens sobre a interação com o mundo e com os demais personagens. Por debaixo deste controle autônomo, o *drama manager* continuamente monitora a simulação, de forma a assegurar ao usuário uma experiência envolvente. Os agentes têm a habilidade de realizar várias atividades inteligentes em paralelo, como falar, gesticular, expressar emoções. Para isso, faz-se uso da linguagem ABL (*A Behavior Language* [91]).

3.3 Exibição (Representação Gráfica)

A exibição, representação gráfica ou dramatização, é o processo final realizado, em tempo real, pelos motores de enredo. Nesta fase, os fatores de tempo e espaço são adicionados à cena para dar ao usuário imersão temporal, espacial e emocional.

Na abordagem de Spierling [129], é desenvolvido um protótipo de um personagem virtual, presente em um quiosque digital, que interage com o usuário por meio de diálogos. Adota-se uma adaptação da estrutura de contos de fadas para um contexto de diálogo de negócios. O conteúdo do diálogo é relativo a negócios e informações sobre produtos. O principal aspecto abordado se refere à modelagem conversacional e social do comportamento do agente. Alguns exemplos da interface da aplicação são apresentadas na Figura 3.4. Neste sistema, o autor especifica regras que direcionam o diálogo e que são específicas da personalidade do personagem. É utilizada a linguagem Java3D na modelagem dos personagens, em conjunto com técnicas de *morphing* para animação facial e de mãos. Entre duas cenas, pode haver saltos de tempo, o que pode ser desejável para evitar mostrar cenas com poucas ações e eventos desinteressantes.

Nos trabalhos de Cavazza [26][24] e Charles [28], faz-se uso do motor usado no jogo Unreal. As histórias podem ter até 3 minutos de duração. Ações que envolvem recursos como ler uma revista, por exemplo, estão

Figura 3.4: Exemplos da interface do protótipo de conversação. Reproduzidas de Spierling [129], pg. 43.

associadas a um timer. Na Figura 3.5, são apresentadas algumas telas de diferentes configurações de cenários e interações entre personagens. Eventos diferentes, em locais diferentes, podem ocorrer simultaneamente no decorrer da história [29] e podem ser visualizadas pelo espectador sob a perspectiva de qualquer personagem. É permitida ainda a navegação livre pela cena. Além destes recursos, devido a restrições espaciais e temporais, o sistema incorpora um módulo de câmera, que desempenha um papel de diretor virtual, que apresenta ao espectador as situações mais relevantes a cada momento. Para a escolha das cenas, leva-se em consideração os tipos de eventos, número de participantes, além dos contextos emocional e afetivo da história no dado momento.

Figura 3.5: Exemplos de cenas de diferentes histórias. Reproduzidas de Cavazza [25], pg. 20 e 23.

Mateas [99][100] apresenta um sistema interativo onde o usuário é um personagem em primeira pessoa que interage com um casal de

amigos, em seu apartamento. Este casal está frente a uma discussão sobre a dissolução do casamento. O papel do usuário é intervir nesta discussão para interferir no rumo da história. Esta interferência motiva o usuário a sucessivas visualizações da história para sentir como sua interferência influencia na finalização da narrativa. O usuário pode conversar em linguagem natural, além de poder mover-se e gesticular livremente no mundo 3D representado pelo apartamento do casal. O diálogo é o mecanismo principal de comunicação e intervenção na história. O usuário deve digitar a mensagem, que é exibida na tela. O sistema automaticamente analisa o texto e realiza as ações necessárias, inclusive respondendo às perguntas. Na Figura 3.6 são apresentados alguns exemplos destas interações.

Figura 3.6: Cenas de interação com o casal. Reproduzidas de Mateas [99], pg. 9

3.4 Conclusão e Observações

Pelos modelos estudados e apresentados nesta seção, pode-se ter um panorama das tecnologias, técnicas e abordagens atualmente sendo utilizadas nas pesquisas em *interactive storytelling*.

É um consenso o fato de que, com a tecnologia existente, a máquina não tem condições de criar histórias dinâmicas interessantes, sem se basear em alguma estrutura formal com regras e eventos que devem ser realizados. Independente da abordagem utilizada, tem-se em comum a existência de uma estrutura de narrativa, criada previamente pelo autor da história, onde as ações principais são baseadas. Esta estrutura é o que diferencia sistemas de *storytelling* de sistemas baseados unicamente em agentes autônomos puros, os quais podem apresentar comportamentos reativos ou deliberativos.

Em geral, existe uma restrição quanto ao número de personagens, visto que cada um deve estar associado a um conjunto grande de regras próprias,

que determinam, com certo grau de detalhes, elencos de ações que podem ser executadas para alcançar os objetivos. A determinação destas regras pode ser um tanto complexa e trabalhosa, tendo em vista a complexidade de modelar comportamentos e estruturas de decisão que atendam a um objetivo específico. Essa dificuldade também pode ser constatada pela análise de propostas de mecanismos de criação de histórias em diversos níveis de abstração, onde, por meio de processos iterativos, pode-se continuamente ajustar os parâmetros que regem determinados aspectos dos personagens e que moldam seu modo de agir e raciocinar sobre o mundo [129].

Diferentes formas de interação com a história são adotadas nos trabalhos apresentados, que podem ser tanto em primeira como terceira pessoa, dependendo da abordagem utilizada na concepção do sistema e propósito a que se destina. Na abordagem utilizada por Cavazza [26], a intervenção pode ocorrer a todo momento, inclusive com os elementos do cenário, o que, por sua vez, pode conduzir a história a situações sem solução.

No presente trabalho, relativo a aspectos gráficos e de cenário, o sistema proposto se assemelha ao trabalho de Cavazza [26], que dentre os sistemas estudados, é o que apresenta maior riqueza de detalhes. Entretanto, em termos de geração e interação com a história, faz-se uso de abordagens semelhantes a Grasbon e Spierling [64][129], que são fundamentadas em lógica para garantir a coerência do conteúdo gerado.

No próximo capítulo, são apresentadas a estrutura, as características e as potencialidades do IPG, que é o módulo do sistema resultante desta tese, utilizado para prover suporte aos processos de autoria, planejamento e interação das histórias interativas. O sistema proposto, como um todo, visa conciliar vantagens das abordagens baseadas em personagens e enredos, ao mesmo tempo garantindo diferentes níveis de interatividade com o usuário nos processos de geração e direcionamento das histórias. Além disso, o próximo capítulo apresenta o contexto dos enredos escolhido como exemplo, incluindo o elenco das operações usadas para representar as possíveis ações.

4
Modelo de Geração de Histórias

Como mencionado no capítulo anterior, a base de todo o processo de se trabalhar com histórias interativas tem início com a especificação dos eventos e fatos que podem ser combinados para gerar o conjunto de cenas que compõe a história. Neste capítulo, inicialmente é apresentado o IPG, o módulo do sistema que dá suporte e apoio lógico à autoria e geração das histórias. Na seqüência, são apresentados aspectos relativos à geração de conteúdo interativo e diversificado, bem como detalhes da estrutura interna da base de dados usada nos exemplos de histórias abordados nesta tese.

4.1
O IPG (Interactive Plot Generator)

Um passo inicial para a geração automática de enredos teve início com os estudos do pesquisador russo Vladimir Propp [115], que observou que, em textos literários de gêneros específicos (no caso de contos de fadas russos), é muito comum a ocorrência de eventos típicos e de padrões de encadeamento entre os eventos.

Propp sugeriu a caracterização dos textos de um determinado gênero pela associação de funções a pequenos trechos das narrativas. A ocorrência das funções ao longo de uma narrativa obedeceria sempre a determinadas seqüências, que poderiam envolver todas ou apenas parte das funções típicas do gênero. As funções proppianas seriam, então, os eventos típicos básicos da narrativa, descrevendo-se os enredos como seqüências de ocorrências das funções. O encadeamento entre os eventos é um processo que tem de respeitar uma lógica, segundo a qual um evento só ocorre na presença de determinadas condições. Cada evento, por sua vez, pode também criar as condições necessárias para a execução de outros eventos [33].

Em geral, os eventos da história não ocorrem aleatoriamente. Estes eventos ocorrem essencialmente em vista dos objetivos que levam os personagens a agirem. Por sua vez, esses objetivos surgem quando

determinadas situações se configuram, exigindo a ação dos personagens. Os enredos resultam, portanto, da interação cooperativa ou competitiva entre os personagens na busca de seus objetivos. Deve-se destacar que as situações mais interessantes, que conferem dramaticidade aos textos literários e demandam mais atenção em situações reais, decorrem justamente dos conflitos entre personagens ou dos conflitos entre diferentes objetivos de um mesmo personagem [33]. No total, Propp caracterizou 31 funções, apresentadas na Tabela 4.1.

Baseado no trabalho de Propp, Ciarlini [33] propôs um método formal para a especificação dessas funções (eventos típicos), representando-as por operações lógicas, com pré e pós-condições. Para permitir a geração semi-automática de narrativas de um certo gênero, Ciarlini [33] desenvolveu o IPG (*Interactive Plot Generator*). O IPG é um módulo de geração de enredos, implementado com o uso de técnicas de Inteligência Artificial e Bancos de Dados, por meio da linguagem Prolog. Ele é composto por dois sub-módulos: um de inferência de objetivos e um outro de planejamento. No planejamento, o IPG faz a ligação entre os eventos de acordo com a lógica, explicitada por suas pré e pós-condições. O planejador do IPG é uma extensão do planejador ABTweak [149], que tem as seguintes características:

- É hierárquico - permite estabelecimento de pré-condições mais importantes, de modo a chegar mais rápido a uma solução;
- É não-linear, ou seja, estabelece relações de ordem entre eventos apenas quando necessário, o que em geral leva a uma maior eficiência. Além disso, a busca e conciliação de vários objetivos simultâneos fica facilitada.

O IPG usa um processo de simulação para a criação de um enredo de um certo gênero, que tem como ponto de partida a descrição da situação inicial dos personagens, o modelo de comportamento atribuído a eles (especificado em termos de objetivos a serem perseguidos em situações previstas) e as alternativas que cada personagem tem para atingir seus objetivos, que são especificadas em termos das operações (ou padrões típicos) que caracterizam o gênero. Estes dois últimos caracterizam o gênero da narrativa. Uma operação descreve os fatos válidos antes e depois do evento. Para que uma operação seja executada, é necessário que suas pré-condições sejam satisfeitas, ou no próprio estado inicial, ou então pelas pós-condições de operações que a precedem.

O processo de geração dos enredos faz parte de um ciclo com múltiplos estágios, onde alternam-se fases de inferência de objetivos e planejamento.

Tabela 4.1: Lista das 31 funções típicas de contos de fadas Russos [115]

absence (1): Um dos membros da família da vítima se ausenta de casa.
interdiction (2): Uma proibição é feita à vítima.
violation (3): A proibição é violada.
reconnaissance (4): O vilão busca informações sobre a vítima.
delivery of information (5): A informação sobre a vítima é obtida.
fraud (6): O vilão tenta enganar a vítima para tomar posse dela ou de seus bens.
complicity (7): A vítima, sem querer, colabora com o vilão.
villainy or lack (8): O vilão causa dano à vítima (ou a membro de sua família). Pode ser, por exemplo, um rapto ou um assassinato.
mediation (9): O herói toma conhecimento do infortúnio.
counteraction (10): O herói concorda em ir em busca da vítima.
departure (11): O herói sai de casa.
proof (12): O herói é testado por um doador para saber se merece receber o objeto mágico ou um auxiliar.
reaction (13): O herói reage às ações do futuro doador.
receipt of magical object (14): O herói adquire um objeto mágico ou um auxiliar.
translocation (15): O herói é transportado para as vizinhanças de onde está a vítima.
struggle (16): O herói combate o vilão.
marking (17): O herói recebe uma marca.
victory (18): O herói vence o vilão.
liquidation (19): A vítima é libertada ou a causa de infelicidade é eliminada.
return (20): O herói retorna.
pursuit (21): O herói é perseguido.
rescue (22): O herói é resgatado da perseguição.
arrival (23): O herói chega em casa sem ser reconhecido.
pretentions (24): Um falso herói se apresenta reclamando a recompensa.
task (25): Uma tarefa difícil é proposta ao herói.
solution (26): O herói resolve a tarefa.
recognition (27): O herói é reconhecido.
exposure (28): O vilão ou falso herói é exposto.
transfiguration (29): O herói ganha uma nova aparência.
punishment (30): O vilão é punido.
wedding (31): O herói é recompensado, usualmente casando-se.

Como diversos desencadeamentos diferentes podem ocorrer, o usuário, após cada fase de planejamento, pode interagir com o sistema para selecionar as alternativas mais interessantes, na medida em que enredos parciais são criados. É importante também que o autor possa interagir, impondo

condições correspondentes a estados dos personagens ao longo da narrativa. Se o modelo de comportamento definido inicialmente não produz bons resultados, o autor deve finalmente interagir com o sistema, em um nível mais profundo, corrigindo o modelo de comportamento de seus personagens e reiniciando o processo. O usuário pode também forçar a ocorrência de eventos e especificar que algumas situações devem ser verdadeiras em certos momentos durante a narrativa. Este tipo de interação é permitido tanto no início como durante o processo de simulação.

A Figura 4.1 mostra o esquema de geração de enredos proposto por Ciarlini et al. [35]. Nesse esquema, um novo enredo pode ser obtido a partir de uma simulação completa, da instanciação de um padrão típico de acordo com informações fornecidas pelo usuário ou por um processo misto no qual usa-se simulação para adaptar um enredo típico a condições específicas.

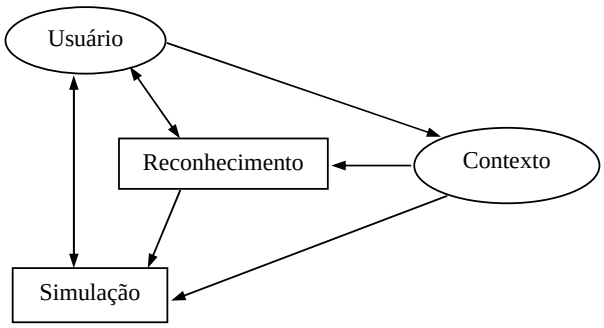

Figura 4.1: Esquema de geração de enredos.

Como o IPG trabalha com planejamento não-linear, são gerados planos compostos por um conjunto de operações parcialmente ordenadas. O exemplo apresentado na Tabela 4.2, baseado em casos das funções de Propp, ilustra esta ordenação. Para este exemplo, a operação 2 somente pode ser executada após a 1, as operações 7 e 8 somente após a 5, a operação 3 a qualquer momento, e assim por diante. A ordenação total é definida pelo usuário, que pode assumir diferentes combinações desde que respeitadas as precedências, representadas por setas na Figura 4.2.

4.2 Contexto dos enredos

O IPG já foi utilizado em dois tipos de contextos: contextos de banco de dados empresariais [36], no apoio à tomada de decisões[1] e em contextos

[1]Neste caso, o IPG é usado para tratar objetivos entre uma empresa, que deseja obter mais clientes e melhorar o serviço oferecido a eles, e seus funcionarios, que desejam

Tabela 4.2: Ordem parcial das operações

0: `Init()`
1: `Absence_of_younger_people(princess)`
2: `Kidnapping_of_a_person(princess, dragon)`
3: `Call_for_help(hero, tsar)`
4: `Departure_of_seeker_hero(hero)`
5: `Fight_in_an_open_field(hero, dragon)`
6: `Out_of_earth_receipt(hero)`
7: `Victory_in_open_battle(hero, dragon)`
8: `Return(hero)`
9: `Reward(hero)`
$1 < [2]$
$5 < [7, 8]$
$7 < [9]$
$4 < [5]$
$6 < [7]$
$8 < [9]$

literários, no caso contos de fadas [35]. Para a modelagem de contos de fadas, foi utilizado, com algumas adaptações, um subconjunto das funções catalogadas por Propp, cada uma associada a uma operação com pré e pós-condições. Estas condições são especificadas por um conjunto de predicados, que devem permitir que qualquer conto do gênero possa ser gerado pelo IPG.

O processo de simulação, realizado pelo IPG, por operar em um nível lógico, trabalha de forma transparente sobre qualquer tipo de operação válida e consegue gerar planos dentro dos limites estabelecidos. Com isso, novos enredos podem ser criados sem alteração do kernel do IPG. Apenas o arquivo com as especificações dos predicados, operações e objetivos deve ser reescrito.

Como ilustrado no exemplo anterior, o resultado do processo de simulação gera um conjunto de eventos, em formato texto, que possuem uma ordem temporal parcial. Neste trabalho, usa-se o IPG como um gerador de planos para alimentar um motor gráfico, onde personagens não mais lógicos, representados por atores, devem interagir "fisicamente" em um ambiente de representação gráfica. Isso traz fortes implicações tanto no contexto dos enredos, bem como nas ações realizadas pelos personagens. Simulações de Banco de Dados empresariais, usadas no apoio à decisão [33], por exemplo, são definidas por operações que não podem ser facilmente

obtenção de estabilidade e ganho de promoções.

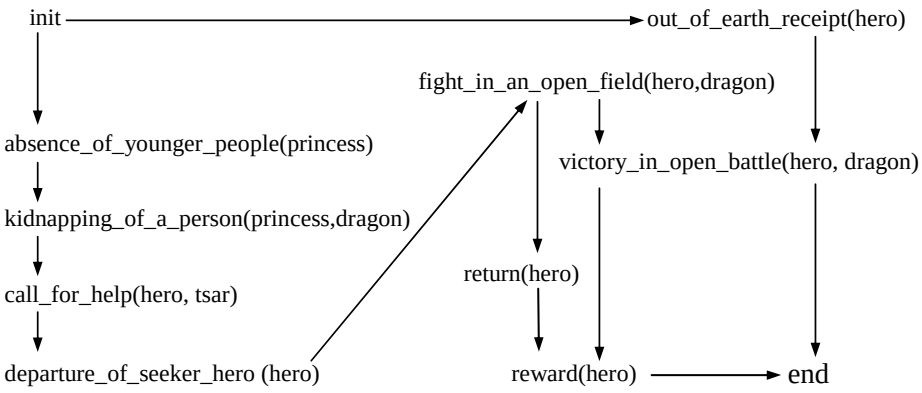

Figura 4.2: Representação gráfica de um enredo parcialmente ordenado.

expressas somente por recursos gráficos, por meio de um motor de jogos. Como exemplo, pode-se citar um evento característico deste gênero onde um personagem ganha uma promoção. A expressão de informação numérica ou textual, por meio de áudio ou janelas de diálogos, também se faz necessário para estes casos. O IPG ainda não possui recursos para geração de diálogos associados às operações de uma narrativa. Porém, já existe um protótipo, ainda rudimentar, que gera textos a partir de enredos [56]. Este recurso, entretanto, é de grande valia, mesmo em narrativas que podem ser facilmente expressas somente por meio de recursos gráficos.

Exemplos de contextos cuja representação gráfica expressa informação suficiente para o entendimento são histórias baseadas em cenas de ação, onde ocorrem interações físicas entre personagens e com o mundo que os rodeia, como no caso de histórias de contos de fadas. Como observado no decorrer desta pesquisa, muitas das 31 funções catalogadas para contextos de contos de fadas podem ser facilmente e suficientemente representadas por animações gráficas. Deste modo, este foi o gênero adotado nesta tese como base para o desenvolvimento do sistema.

4.3 Estrutura dos Enredos

Uma vez definido o gênero das histórias, partiu-se para a investigação e definição de um conjunto de operações com fácil representação gráfica e que, ao mesmo tempo, fossem suficientes para gerar enredos interessantes e com certo grau de diversidade. O objetivo foi delinear o que seria um enredo interativo, ou seja, um enredo que permita que tanto o IPG, quanto o usuário, possam disponibilizar de um grande repertório de possibilidades

na geração de histórias diversificadas e interessantes, cujo tempo de duração da representação gráfica dure um período considerável de tempo.

Como ponto de partida a esta investigação, considerou-se um exemplo tradicional de contos de fadas catalogado por Propp, como se segue:

"*Um vilão, no caso um dragão, rapta uma princesa desprotegida. O rei, em desespero, requisita ajuda a um herói que parte em busca da princesa, mata o dragão e, como recompensa, casa-se com a princesa.*"(Figura 4.2)

Para a dramatização deste simples exemplo, uma série de eventos podem ser tratados. Para que o dragão possa raptar a vítima, de forma resumida, ele deve saber onde ela se encontra, deslocar-se até ela, desviando de obstáculos, raptá-la e levá-la ao cativeiro. O rei, por sua vez, deve, de alguma forma, comunicar-se com o herói, que deve deslocar-se até o local do cativeiro, lutar com o dragão, vencê-lo, libertar a princesa e levá-la de volta para sua casa. Para que o casamento se realize, primeiramente os personagens devem ir até uma igreja. Se o casamento for realizado dentro da igreja, deve-se dispor de recursos para tratar com cenas de interiores.

Para identificar as dificuldades e melhor compreender os processos necessários para criar uma representação gráfica para esta simples história, foi desenvolvido um motor experimental. Diversas questões e problemas foram levantados. Eis os mais impactantes:

- Mecanismo de controle da duração de cada evento: a duração de cada evento pode ser fixa ou decorrente da interação entre os personagens. Em função disso, parâmetros para a determinação da finalização de uma ação devem ser descritos;
- Formas de interação entre os personagens: para este exemplo em especial, há personagens que se locomovem no chão (humanos) e um que voa (dragão). Isso traz várias conseqüências na forma como eles interagem entre si e com o ambiente;
- Mecanismo de sincronização entre a história simulada e a história que está sendo exibida graficamente: deve-se observar que a representação gráfica não consiste somente em exibir cenas, mas também na reconstrução do encadeamento de eventos e na sincronização de atributos presentes tanto no visualizador gráfico como no IPG;
- Nível de interação do usuário: o usuário deve ter o recurso de selecionar partes da história a serem visualizadas, bem como a reexibição de cenas de maior interesse;

– Nível de variação das histórias: como observado nos primeiros testes realizados, o nível de variação das histórias era relativamente pequeno. Uma questão a ser respondida era se deveriam ser incorporadas mais operações à base de dados ou se novas operações, mais específicas ao contexto, deveriam ser criadas;
– Comportamentos necessários que os personagens devem dispor: dependendo da quantidade e do nível de detalhamento das operações, comportamentos específicos para cada tipo de ação ou interação entre personagens deveriam ser implementados (como ocorre no jogo The Sims [48]).

Muitas destas questões são respondidas somente no Capítulo 7, que trata da implementação da ferramenta. No momento, se está preocupado em definir recursos que permitam ao IPG gerar histórias coerentes e que apresentem alto grau de diversidade e que permitam ao usuário a seleção de um grande conjunto de alternativas que conduzam ao final desejado.

Inicialmente, pode-se cogitar em especificar um grande conjunto de operações. Esta abordagem apresenta dois tipos de problemas: a nível de simulador, à medida que o número de operações cresce, cresce também o tempo de planejamento necessário para gerar a história. A nível de representação gráfica, cada nova operação implica na implementação de novas ações, bem como animações gráficas correspondentes. Em termos mais gerais, o nível de detalhamento de operações pode chegar rapidamente a níveis impraticáveis de serem representados, tanto a nível lógico como gráfico.

Visto que somente o uso de uma grande gama de operações não é uma boa solução para definir uma base que desse suporte a histórias diversificadas e com vários caminhos possíveis, procurou-se investir na definição de operações mais gerais e na agregação massiva de atributos numéricos, tanto a personagens, quanto a locais e operações. A definição de operações gerais, tais como "`go(CH,PLACE)`" (para representar a ida do personagem `CH` até o local `PLACE`) são claramente mais úteis que operações do tipo "`absence_of_younger_people(VICTIM)`" (que indica a ida da vítima a um local desprotegido). A operação "`go`" pode ser usada para representar os deslocamentos de todos os personagens, incluindo a ida da vítima a um local desprotegido. Consegue-se assim economizar tanto na especificação lógica do contexto quanto na implementação de recursos para a dramatização.

O conceito de locais também foi inserido na base de dados. Até então, nenhuma ação ou personagem estava associado a um local específico. Com a

inclusão deste elemento, personagens sempre estão relacionados com o local onde se encontram, o local onde residem e o local onde devem realizar as ações. Os locais possuem dois atributos numéricos: um que indica o nível de proteção e o outro para representar se o local é do bem ou do mal. Este mesmo atributo, visto sob a forma de caráter, é também associado a cada personagem. Todos os personagem possuem este atributo igual ao local onde residem, que é especificado no arquivo de inicialização dos personagens (Ver Sessão 7.3.1).

As operações mais gerais e os atributos numéricos associados aos personagens e locais permitem uma grande variedade de alternativas para a realização e combinação de ações.

Para permitir o uso de pré-condições com relações numéricas entre os atributos, o IPG suporta *constraint logic programming* [94]. Essa técnica combina os mecanismos de busca típicos de um planejador com a resolução de conjuntos de equações e inequações numéricas. Quando o conjunto de equações e inequações não tem solução para uma determinada alternativa do processo de busca, essa alternativa é descartada.

Os atributos numéricos são usados para modelar caráter, força, relações de afinidade e nível de proteção. Como apresentado no próximo capítulo, estes atributos também podem ser usados para a modelagem de *drives* e emoções.

Uma grande vantagem do uso de atributos numéricos é a possibilidade de tratar a variação contínua de suas intensidades, como resultado das pós-condições da execução de uma operação. Por exemplo, toda vez que um local é atacado, sua proteção é reduzida por um fator estipulado; quando ocorre luta entre dois personagens, suas forças são reduzidas, toda vez que a vítima é libertada, sua afeição aumenta em relação ao herói.

Muitas operações têm como pré-condições *thresholds* mínimos ou máximos para que possam ser realizadas. Para que o rapto ocorra, por exemplo, o nível de energia do vilão deve ser maior que a soma da energia da vítima com a proteção do local onde ela se encontra. Para este exemplo, o vilão poderá ter que realizar vários ataques contra o local da vítima ou então tornar-se mais forte.

Pretende-se mostrar que desta forma, pode-se construir situações complexas pela combinação de restrições numéricas, locais, personagens, níveis de proteção, relacionamento entre personagens, dentre outros. Com o uso de atributos numéricos, bem como pela estipulação de locais, pôde-se reduzir o número de operações e ao mesmo tempo, garantir a diversidade de histórias.

As operações trabalham sobre fatos previamente estipulados na base de dados do sistema. Fatos não podem ser criados durante o processo de simulação da história, porém podem ser alterados livremente pela execução das operações, como resultado de suas pós-condições. Os fatos são especificados por meio de predicados, usando a notação Prolog, como mostrado na Tabela 4.3.

Tabela 4.3: Predicados definidos na base Prolog

Predicados	Descrição
`character(CH,KIND)`	`CH` é um personagem com caráter do tipo `KIND` (natureza ou índole), que pode assumir dois valores: 1 (bem) ou -1 (mal)
`hero(CH)`	O personagem `CH` é um herói
`victim(CH)`	O personagem `CH` é uma vítima
`villain(CH)`	O personagem `CH` é um vilão
`place(PL)`	`PL` é um local
`home(CH,PL)`	O personagem `CH` reside no local `PL`
`current_place(CH,PL)`	O personagem `CH` está localizado em `PL` no dado momento
`protection(PL,KIND,L)`	O local `PL` tem um nível de proteção `L` do tipo `KIND` (natureza: bem ou mal)
`strength(CH,L)`	O personagem `CH` tem um nível de força `L`
`affection(CH1,CH2,L)`	O nível de afeição de `CH1` por `CH2` vale `L`
`alive(CH)`	O personagem `CH` está vivo

No nosso exemplo de utilização (Seção 7.9), os predicados anteriormente definidos foram usados para especificar a configuração inicial do simulador, que consiste na definição dos personangens (Marian - a vítima, Draco - o vilão, Brian e Hoel - os heróis) e seus atributos.

Na Tabela 4.4 são apresentadas as operações especificadas. Elas procuram ser genéricas, de simples representação gráfica e suficientes para gerar histórias interessantes. No Apêndice 1 são apresentadas as especificações destas operações, incluindo suas pré- e pós-condições, definidas através dos predicados listados anteriormente e de relações numéricas entre atributos.

A base de dados do IPG também contém a especificação das Regras de Inferência de Objetivos (*goals*). Elas representam os objetivos a serem atingidos pelos personagens e são usadas como meio de geração automática das histórias pelo IPG, mesmo sem a interferência direta do usuário. Cada regra representa um conjunto de fatos que um personagem passa a desejar tornar verdadeiros quando certas situações ocorrem no enredo [33]. Por meio

Tabela 4.4: Conjunto de operações (adaptadas de Propp)

Operações	Descrição
`Go(CH,PL)`	Locomoção do personagem `CH` até o local `PL`. Geralmente ocorre em combinação com outras operações. Faz-se necessária visto que personagens e ações estão associados a locais
`Reduce_protection(PL)`	Redução da proteção do local `PL` pela redução voluntária do número de guardas. Cada local possui um nível de proteção associado. Algumas operações têm como pré-condições que o nível de proteção do local onde a ação deve ocorrer assuma valores dentro de certos limites
`Attack(CH,PL)`	O personagem `CH` deve atacar o local `PL` para reduzir a proteção, lutando com os guardas do local. `CH` e `PL` devem estar associados. Para que o ataque ocorra, existe a pré-condição de que a natureza (boa ou má) de `CH` deve ser diferente da natureza da proteção corrente do local
`Get_stronger(CH)`	O personagem `CH` recebe poderes que aumentam a sua força
`Fight(CH1,CH2)`	Luta entre os personagens `CH1` e `CH2`. Foi criada para representar as lutas entre os mocinhos (no caso os cavaleiros que fazem o papel de heróis) e o vilão (dragão). Para que a luta ocorra, o local da luta deve estar desprotegido (baixa proteção) e os personagens devem ter naturezas opostas
`Kidnap(VIL,VIC)`	Um personagem vilão rapta o personagem vítima. Para isso, o nível de força do vilão deve ser maior que a soma da força da vítima com a proteção do local onde ela se encontra
`Kill(CH1,CH2)`	O personagem `CH2` é morto pelo `CH1`. `CH1` deve ser mais forte que `CH2`. Devem ser também de naturezas opostas
`Free(HERO,VICTIM)`	O herói liberta a vítima que estava raptada. Tem como pré-condição a morte do raptor e como pós-condição o aumento da afeição da vítima pelo herói
`Marry(CH1,CH2)`	Casamento entre dois personagens. É resultado da premiação por uma ação realizada, neste caso a libertação. Logo, para que haja um casamento, deve haver um rapto e o vilão deve ser derrotado

delas, o autor pode modelar a estrutura geral da narrativa. Para o contexto de contos de fadas de nosso exemplo foram definidas, usando uma lógica temporal modal [34], 4 regras:

1. Se no início da história existe uma vítima, será tomada alguma medida para torná-la frágil. Segundo o elenco de operações apresentadas, isso pode ser realizado de várias formas, como por exemplo: pela redução da proteção onde a vítima se encontra, pelo ataque do vilão a este local ou fazendo com que a vítima vá a um local desprotegido;

2. Se a vítima ficar desprotegida, o vilão desejará raptá-la. Para isso, o vilão deve ir até a vítima, reduzir a proteção do local onde ela se encontra, caso seja necessário, raptá-la e levá-la para o cativeiro;

3. Se a vítima for raptada, o(s) herói(s) tentará(ão) salvá-la. Para que isso ocorra, o herói deve reduzir a proteção do local onde o vilão reside, lutar com ele e vencê-lo;

4. Se a afeição entre herói e vítima é alta, eles desejarão se casar. A principal pós-condição da libertação é o aumento da afeição da vítima pelo herói que a liberta. Neste caso, pelas operações apresentadas, a única forma do casamento se realizar é que haja um rapto seguido de uma libertação.

4.4 Conclusões e Discussões

O IPG é um versátil e poderoso módulo para geração de histórias, que usa a lógica como forma de garantir a integridade e coerência das seqüências de eventos gerados por mecanismos de planejamento e inferência de objetivos, bem como pela especificação explícita de fatos pelo usuário, o que torna a geração da história um processo assistido.

Para o propósito que esta ferramenta se destina - servir como fonte de dados a um visualizador gráfico - investiu-se na especificação de uma base de dados compacta e que ao mesmo tempo fosse suficiente para gerar histórias cuja interatividade, atratividade e tempo de duração da representação gráfica fossem valores expressivos, principalmente quando comparado com outros modelos de *story engines*.

Fazendo-se uso de uma abordagem *plot-based*, o IPG apresenta características que permitem um controle rígido no direcionamento da história, mas que nem por isso limita a forma como o usuário pode interagir

com a mesma. Detalhes sobre a interação com o usuário são apresentados no Capítulo 7, que trata da implementação da ferramenta, onde são expostas as interfaces gráficas por onde o usuário interage com a história que está sendo processada.

Um recurso que ainda não foi explorado na ferramenta, mas que já está incorporado ao IPG, é a possibilidade do usuário poder, além de inserir e ordenar operações e goals, alterar atributos dos personagens. Diversas questões interessantes podem surgir deste tipo de interação. Pode-se por exemplo, transformar um herói em um vilão.

Outro recurso que pode ser mais explorado é o uso de mais de um personagem do mesmo papel ou de um personagem com mais de um papel. No exemplo, há dois heróis: Brian e Hoel. Pode haver competições entre eles para decidir quem vai salvar a princesa, ou situações mais interessantes onde um se encarrega de enfraquecer as defesas do castelo do vilão, enquanto o outro derrota o vilão e leva o prêmio. Mais situações com certeza podem surgir se for permitido que haja mais de uma vítima, mais de um vilão ou que um herói também possa ser uma vítima.

No próximo capítulo são fundamentados aspectos de agentes, comportamentos e estratégias para a implementação dos atores, cujo papel é fazer a representação dos eventos da história, gerados pelo IPG e definidas em formato texto, por meio de conteúdo gráfico animado.

5
Agentes em Histórias Interativas

Segundo Mateas [97], com o interesse recente por pesquisa em agentes autônomos, têm surgido muitos estudos na interseção entre agentes e narrativas, dentre os quais podem-se destacar o uso de agentes como personagens em histórias e agentes que podem contar histórias. Essa visão se aplica diretamente na abordagem deste trabalho, tanto a nível de geração como de visualização das histórias.

A tecnologia de agentes tem sua origem em IA distribuída [15], em trabalho cooperativo suportado por computador [7], em animação comportamental [119][145][136], em personagens emocionais [9][10][11][91][118] e em robótica [18]. Apesar de programação orientada a agentes ter sido proposta em 1993 como uma paradigma pós-objeto [124], somente em 2003-2004 é que agentes passaram a ser tratados com rigor e abrangência da Engenharia de Software [59]. Uma referência clássica sobre teorias, arquiteturas e linguagens de agentes, sob o enfoque da comunidade de IA, pode ser encontrada em [146]. Outras referências mais recentes são [2] e [85].

Uma definição precisa de agentes só é possível de ser dada em função do propósito pelo qual o agente é construído, uma vez que agentes são entidades que encapsulam conhecimento sobre algum domínio [85]. Entretanto, agentes podem ser genericamente definidos como entidades de software que têm as seguintes características: autonomia, habilidade social, reatividade e pro-atividade. Racionalidade, habilidade de aprender e mobilidade são características adicionais que não são nem necessárias nem suficientes para caracterizar agentes.

Para o contexto deste trabalho, em especial tratando a geração de histórias, a idéia principal do uso de agentes é a especificação de uma modelagem que os permita alcançarem seus objetivos em circunstâncias que podem envolver competição. Deseja-se que o comportamento dos personagens baseie-se na busca de seus objetivos, os quais surgem quando condições particulares aparecem nas histórias [33]. Neste trabalho, o

objetivo do agente é um conjunto de fatos que este agente deseja tornar verdadeiros em um certo instante.

Em se tratando da visualização gráfica das histórias, o agente é visto como uma entidade (personagem autônomo) que possui objetivos (*goals*), que é capaz de perceber certas propriedades do ambiente onde se encontra (*sensing*), podendo executar ações específicas neste ambiente (*acting*), sendo que algumas destas ações e/ou percepções podem/devem ser feitas através da cooperação com outros agentes [103]. A Figura 5.1 ilustra esta definição.

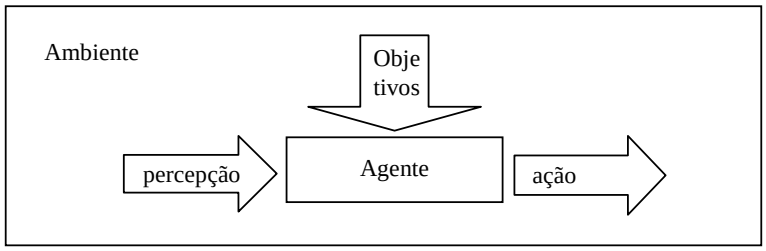

Figura 5.1: Agentes na representação gráfica de histórias.

Este capítulo destina-se a fazer um estudo das tecnologias de agentes, voltadas à implementação do sistema de *storytelling* que está sendo proposto. Na Seção 5.1 são apresentados conceitos gerais da teoria de agentes e sistemas multi-agentes, bem como discussões das técnicas mais adequadas à implementação do trabalho proposto. Na Seção 5.2 faz-se uma apresentação sobre comportamentos de agentes e a proposição de um modelo comportamental que é usado para o gerenciamento das ações dos agentes na etapa de visualização gráfica. Na Seção 5.3 apresentam-se aspectos de implementação de câmeras autônomas, bem como especificações das estratégias usadas neste trabalho. Por fim, na Seção 5.4, faz-se um estudo de drives e emoções, bem como discussões de como estas funcionalidades podem ser incorporadas no sistema que está sendo proposto. Drives são necessidades básicas como fome, sede, interação social, entre outros.

5.1 Modelos de Agente para Storytelling

Autonomia, habilidade social, capacidade pró-ativa e reatividade são propriedades comumente aceitas que tornam um agente diferente de um simples objeto de software. A autonomia permite ao agente executar a maior parte de suas ações sem interferência direta de agentes humanos ou de outros agentes computacionais. A habilidade social é a propriedade

que permite que agentes interajam com outros agentes (humanos ou computacionais) para completarem a resolução de seus problemas ou para auxiliarem outros agentes usando alguma linguagem de comunicação de agente[1]. Esta interação deve-se à impossibilidade de resolução de certos problemas ou a algum tipo de conveniência. A capacidade pró-ativa refere-se ao comportamento orientado a objetivos, que leva o agente a tomar iniciativas quando julgar apropriado. Por último, reatividade é a capacidade do agente reagir à percepção do ambiente, através de seus sentidos, tais como visão e audição.

Quando se cria um agente para uma aplicação específica, deve-se ter em mente a seleção das habilidades que permitam aos agentes agirem de maneira autônoma. Estas habilidades podem incluir percepção e interpretação de mensagens, raciocínio baseado em crenças, tomada de decisão, planejamento e habilidade para executar planos incluindo a passagem de mensagens.

Os agentes podem ser categorizados quanto à arquitetura que suporta a resolução de problemas: cognitivos, reativos e híbridos. Agentes cognitivos, também conhecidos como Intencionais, Racionais ou Deliberativos, têm a habilidade de raciocínio sobre suas intenções e crenças, e podem, por um processo explícito, criar e escolher ações e planos a serem realizados. Neste grupo, existe uma categoria baseada em estados mentais, chamada BDI, que considera três estados mentais para a descrição do processamento interno de um agente: crença, desejo e intenção (*belief, desire, intention*). As crenças representam o conhecimento do agente acerca do ambiente onde está situado. Os desejos, segundo Rao [117], representam o estado motivacional do sistema, ou seja, estados ou ações que o agente quer que se verifiquem. As intenções são os desejos a serem executados e são usadas no processo de decisão do curso de ações que devem ser tomadas.

Muitos destes conceitos se aplicam na modelagem do IPG, visto que tem-se claramente especificado o estado motivacional que vai levar o agente a agir para atingir seus objetivos. As crenças podem representar os atributos especificados na base de dados que são previamente modelados de forma a direcionar a geração dos planos que conduzirão a história.

A arquitetura reativa é uma abordagem que rompe com o paradigma de IA simbólica tradicional. Esta arquitetura tem raízes em alguns trabalhos pioneiros em animação [119], mas surge mais formalmente nos trabalhos de R. Brooks [17]. Brooks propõe que entidades reativas devem responder

[1]KQML (*Knowledge Query and Manipulation Language*) [152] ou FIPA ACL (*Agent Communication Language*) [1], por exemplo.

dinamicamente a mudanças no ambiente. Ademais, Brooks lança os primeiros princípios de emergência e *situatedness* para as entidades reativas.

O princípio de emergência declara que a inteligência de um sistema de agentes emerge da interação de agentes com eles mesmos e com seus ambientes. Cada agente isoladamente é tão simples quanto os agentes sem mente propostos por Minsky [101]. Neste princípio, a inteligência surge da interação. O princípio de *situatedness* afirma que a inteligência de um agente está situada no mundo e não em algum modelo formal do mundo construído no agente. Portanto, um agente usa a sua percepção do mundo, ao invés de usar deduções baseadas em alguma representação simbólica deste mundo.

Nesta arquitetura, o agente não tem capacidade de raciocínio sobre suas intenções, reagindo tão-somente sobre regras e planos pré-definidos. Esta é uma abordagem que se adequa perfeitamente às histórias baseadas no enredo (*plot-based*), visto que os planos já foram previamente definidos na fase de geração das histórias. Não é desejado que os personagens raciocinem e nem tomem decisões inteligentes, para evitar que executem ações espontâneas que mudem o rumo da história já simulada (Os agentes devem agir como atores guiados [132][23]). Os agentes devem ter habilidades suficientes para cumprir ordens, que neste caso são eventos da história, bem como apresentar características reativas mediante a interação com outros agentes. Aspectos de aprendizagem também não se fazem necessários.

Tratando-se o gerador e o visualizador das histórias como um sistema único e integrado, pode-se dizer também que este faz uso de uma arquitetura híbrida, onde a escolha das ações é realizada utilizando-se uma combinação de técnicas de arquiteturas reativas e cognitivas. A arquitetura híbrida foi proposta como uma alternativa para solucionar as deficiências cognitivas (incapacidade de reação rápida e adequada a situações não previstas) e reativas (incapacidade de descobrir alternativas para o seu comportamento quando a situação do mundo diverge dos seus objetivos). Segundo Nareyek [103], os agentes híbridos utilizam planejamento de alto nível de abstração em uma fase de pré-processamento, enquanto decisões sobre alternativas de refinamento menos significativas são tratadas por sistemas reativos.

No sistema proposto, também existem diversos aspectos de Sistemas Multi-agentes (SMA) [148]. Estes sistemas estudam o comportamento de um conjunto de agentes autônomos cujo objetivo comum é a solução de um dado problema cuja resolução está além das capacidades de um único indivíduo [67]. Para o trabalho em questão, deve-se estudar como se dá a interação entre personagens dentro de enredos e no ambiente de representação gráfica. Os personagens interagem com outros personagens, positiva ou

negativamente, na busca de objetivos e, para que sejam consistentes, devem usar uma lógica que possibilite inferir objetivos e adotar planos para atingi-los [33]. Como o objetivo geral é a criação de enredos e não a satisfação dos objetivos específicos de cada personagem, Ciarlini [33] faz uso de um único planejador que é usado para inserir operações de todos os agentes na busca de seus objetivos.

Quando se trata da representação gráfica das histórias, as características de sistemas multi-agentes se mostram mais visíveis ainda. Cada personagem é tratado como uma unidade autônoma e deve dispor de mecanismos de coordenação, interação e comunicação de modo que seus conhecimentos, objetivos, habilidades e planos individuais sejam usados de modo organizado, em favor da execução de uma seqüência ordenada de eventos gerados na etapa de criação da história.

Segundo Jennings [83], a coordenação entre os agentes se faz necessária por uma série de fatores. Geralmente existem dependências entre as ações dos agentes, ou seja, a ação de um agente pode ser pré-requisito da ação de outro agente. Além disso, nenhum indivíduo tem competência, recursos ou informação suficientes para resolver um problema completo de forma independente, onde deve ser garantido o respeito às restrições globais, à solução do problema e à viabilização dos procedimentos que garantem a harmonia quando da execução de uma tarefa de forma conjunta por mais de um agente.

No presente trabalho, a coordenação é implementada de forma centralizada, ou seja, existe um módulo responsável pelo controle da ordem de execução das operações especificadas no enredo. Como diversas operações requerem mais de um agente, mecanismos de interação também fazem-se presentes.

Para que uma interação ocorra, deve ser conhecida a necessidade da interação para que o conteúdo da interação (mensagem) possa ser definido, dentro de uma gama de recursos disponíveis a serem utilizados para tal fim. A comunicação é um meio de realizar a interação entre agentes. É composta por duas partes: o envio da mensagem (ação) e a recepção da mensagem (percepção). Faz-se uso de uma arquitetura nivelada, ou seja, a comunicação entre eles ocorre de forma direta. Para finalizar, tem-se o ambiente, que constitui o contexto onde todas as interações entre os agentes ocorrem. Através do ambiente, ocorre a disseminação do controle, dos dados e do conhecimento pela comunidade de agentes [3]. O nível de acessibilidade do ambiente determina a quantidade de detalhes que o aparato sensorial do agente pode detectar. Neste trabalho, o ambiente é considerado efetivamente

acessível, ou seja, os sensores detectam todos os aspectos relevantes para a escolha da ação. Neste caso, não é necessário que o agente mantenha qualquer representação interna do mundo, visto que qualquer informação necessária pode ser imediatamente obtida.

5.2 Modelagem comportamental

O comportamento de um agente engloba as possíveis ações que este agente pode desempenhar. Essas ações são resultado da combinação de um elenco de regras que foram associadas ao agente. A especificação destas regras é resultado da avaliação das principais características do sistema onde o agente está inserido e, conseqüentemente, do que é esperado que o agente seja capaz de realizar. Para um melhor entendimento do comportamento de um agente, Reynolds [120] sugere quebrá-lo em várias camadas, como apresentado na Figura 5.2, que retrata o comportamento de movimentação em uma hierarquia de 3 camadas.

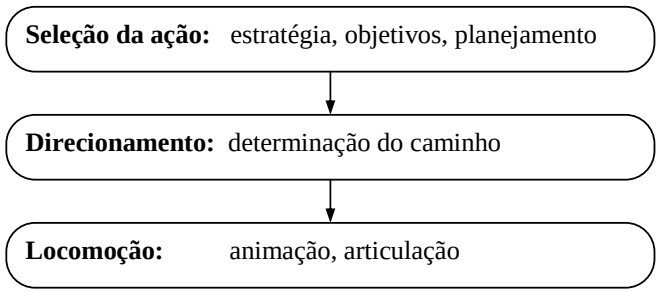

Figura 5.2: Uma hierarquia do comportamento movimentação [120].

Para esta pesquisa de tese, este é o principal comportamento desempenhado pelos agentes, uma vez que a locomoção é pré-requisito para diversas outras operações dentro do elenco de eventos das histórias. A idéia de separação em hierarquia se aplica perfeitamente no escopo desta tese. Pela análise da Figura 5.2, no topo da hierarquia está a seleção da ação. Como já apresentado no Capítulo 4, este processo é definido ou pelo IPG ou como resultado do comportamento reativo resultante da interação entre os agentes. Além disso, novas ações podem surgir da interação do agente com o cenário. A camada de direcionamento (*steering*), refere-se à forma como o agente adquire informação do ambiente para que possa chegar ao local determinado, evitando possíveis obstáculos. Por último, tem-se a animação gráfica da ação que está sendo realizada.

Operações relacionadas à movimentação de personagens referem-se sempre a um único indivíduo. Desta forma, comportamentos de movimentação em grupo não são tratados. Também não é objetivo desta tese fazer um estudo aprofundado de comportamentos de movimentação/manobras. Reynolds [120] apresenta uma descrição mais detalhada destes tópicos.

Na Seção 5.2.1 é apresentado um modelo comportamental simples e ao mesmo tempo eficiente para gerenciar seqüências de ações estáticas ou dinâmicas necessárias à dramatização dos eventos estipulados dentro dos enredos.

5.2.1 Modelo Comportamental

Máquinas de Estados Finitos (*Finite State Machines* - FSM) são provavelmente o padrão de software mais utilizado em jogos para controlar o comportamento de agentes reativos. As FSMs são geralmente representadas por diagramas de transição de estados. Elas são fáceis de implementar e depurar [116], além de apresentarem alta eficiência, uma característica essencial da implementação de agentes reativos, cuja característica marcante é a resposta rápida a estímulos externos.

Entretanto, à medida que a complexidade do comportamento dos agentes aumenta, as FSMs tendem a crescer de forma descontrolada, o que torna a implementação impraticável. Uma solução possível para este problema é o uso de máquinas hierárquicas (HFSM), também chamadas *Behavioral Transition Networks* [72], como proposto por vários pesquisadores [122][54][107]. Nesta abordagem, níveis mais altos lidam com ações mais genéricas, enquanto níveis mais baixos lidam com ações mais específicas. Entretanto, a hierarquia nem adiciona mais poder ao modelo, nem reduz o número de estados. Ela pode somente reduzir significativamente o número de transições e tornar a FSM mais intuitiva e simples de compreender [61]. É importante observar que qualquer HFSM pode ser reescrita como uma FSM sem hierarquia [66].

Uma característica observada durante o desenvolvimento do modelo de comportamento do agente foi que geralmente as ações a serem realizadas, em muitos casos, possuem pré-condições. Essas pré-condições são expressas por meio de ações auxiliares que devem ser realizadas para garantir a integridade e veracidade da ação previamente estabelecida (ação terminal). Quando se está trabalhando com a representação de ações complexas e abstratas,

onde a execução de uma ação depende da execução de um conjunto de ações precedentes, HFSMs apresentam as mesmas limitações que as FSMs. Como exemplo, suponha que um agente (personagem) tem como objetivo a libertação de outro personagem que está raptado. Para que isso ocorra pode ser necessário, por exemplo, que o personagem deva executar um conjunto de ações auxiliares, como andar, lutar, conversar, dentre outras. Muitas destas ações podem envolver outros agentes que habitam o mesmo ambiente. Quando o personagem cumprir todos os pré-requisitos necessários, o agente pode então realizar a ação à qual está designado.

Para lidar com estes tipos de situações, a arquitetura proposta incorpora uma estrutura de pilha a cada agente como memória auxiliar à FSM, de modo a evitar a criação de nós adicionais que seriam necessários para tratar situações onde ações possuem como pré-condições a execução de outras ações. A pilha possui apenas duas posições. Na base (*slot 0*) é guardada a ação que representa o objetivo final (*goal*) do agente, ou seja, o evento que representa o fim da missão, que, para o exemplo anterior, é a libertação do prisioneiro. No topo da pilha (*slot 1*) é armazenada a ação intermediária corrente. A meta do agente é sempre executar a ação que está no topo da pilha, que é sempre a última inserida. Uma ação permanece na pilha enquanto sua execução estiver ocorrendo ou até que outra ação mais importante (geralmente pré-condição) faça-se necessária.

Na Figura 5.3 é apresentado a arquitetura, composta basicamente por 5 módulos, usada no controle dos comportamentos dos agentes:

***Perceptor*:** Continuamente adquire informação do mundo. O sistema visual do agente realiza testes de colisão com objetos na cena. O sistema de mensagens processa mensagens enviadas por outros agentes. Estas mensagens podem mudar o estado interno de modo que o agente possa participar de uma ação coletiva;

Pilha de duas posições: A pilha é usada como um repositório de ações que devem ser executadas;

Fila de história: Mantém um histórico das últimas N ações executadas. Com este repositório, no caso da inexistência de ações específicas, o Gerenciador de Ações pode selecionar ações aleatórias não repetitivas para o agente. A pilha, juntamente com a fila, representam a memória de trabalho do agente;

Gerenciador de Comportamentos (FSM): É o elemento central da arquitetura. Ele processa entradas, determina novas ações auxiliares

e designa ações ao effector que melhor refletem o estado corrente do agente;

***Effector*:** É o meio pelo qual o agente muda o ambiente. Mensagens podem invocar outros agentes e, conseqüentemente, mudar seus atributos. Além disso, este módulo é o responsável pela movimentação do agente.

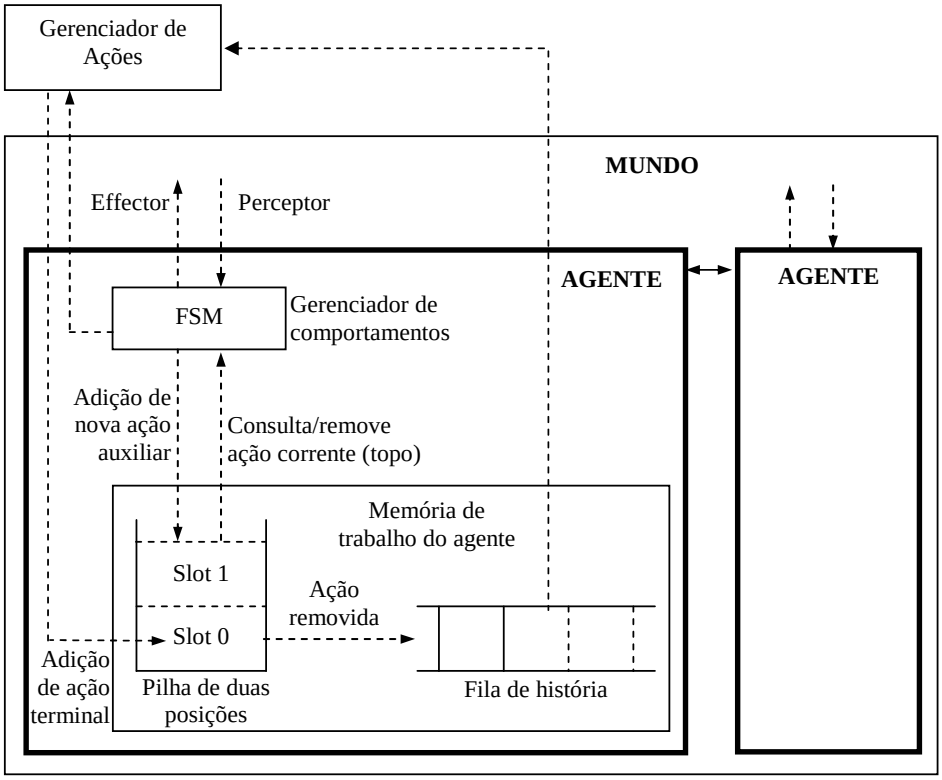

Figura 5.3: Modelo de gerenciamento de tarefas do agente.

A memória de trabalho do agente é representada pela pilha e fila de história. O estado corrente do agente é representado unicamente pela pilha. Tanto o Gerenciador de Ações, como o próprio agente, podem inserir operações na pilha. O Gerenciador sempre insere uma nova ação terminal na base toda vez que a pilha estiver vazia. O próprio agente deve comunicar ao gerenciador que não tem mais ações a executar. Garantindo que as ações terminais são sempre inseridas na base, a pilha ainda tem uma posição livre para tratar ações auxiliares que se fizerem necessárias. Essas ações auxiliares são sempre inseridas pelo próprio agente, mediante a impossibilidade do cumprimento da ação terminal, segundo regras implementadas em cada nó da FSM. Como exemplo, se o agente estiver se locomovendo, ações candidatas correspondem a outras ações de deslocamento, no caso, para evitar colisão com possíveis objetos. Se o agente deve atacar um determinado

local, ações de deslocamento e luta podem ser continuamente inseridas até que a ação de ataque seja finalizada.

Na Figura 5.4 é apresentado um comportamento simplificado da pilha para o exemplo da libertação da vítima. No estado inicial, o herói recebe como ação **terminal** a libertação da vítima. Esta ação é então armazenada na base da pilha do herói, e somente será removida com o final de sua execução. No estado `Free` da FSM, duas pré-condições estão associadas: o herói deve estar próximo da vítima e deve derrotar o vilão (raptor da vítima). Quando a FSM do herói é executada, as pré-condições são avaliadas. Supondo que ele já esteja no mesmo local da vítima, este cria uma ação auxiliar para lutar com o vilão, que é colocada no topo da pilha. A ação de lutar, por sua vez, tem como pré-condições estar próximo e ser mais forte que o oponente. Caso os personagens estejam próximos e o vilão seja mais forte que o herói, uma ação para tornar o herói mais forte é adicionada em sua pilha pelo estado `Fight` da FSM. Neste momento, a pilha já está com as duas posições ocupadas (pilha cheia). Neste caso, a ação do topo (`Fight`) é removida para a inserção da nova (`Get_stronger`), mesmo que esta ainda não tenha sido finalizada. Quando a operação `Get_stronger` finalizar, a execução volta à base da pilha (`Free`), o que faz com que a operação `Fight`, por ainda nao ter sido executada e por ser pré-condição da libertação, seja novamente reinserida no topo da pilha, desta vez com suas pré-condições já satisfeitas. A ação `Fight` é então executada até que o herói derrote o vilão. Quando o `Fight` é finalizado, a ação `Free` pode então ser executada. Quando a operação `Free` é finalizada, o personagem não tem nenhuma ação a realizar. Neste momento, ele solicita ao Gerenciador o envio de uma nova ação.

Em termos gerais, o agente deve ter a habilidade de parar a execução da ação corrente em favor da execução de uma ação de mais alta prioridade. Isso não trás nenhum problema, porque, se for necessário, a ação removida pode ser reinserida na pilha pela FSM tantas vezes quanto forem necessárias.

Fazendo-se uma analogia à máquinas hierárquicas, cada posição da pilha age como uma FSM, que armazena a configuração local do contexto sendo executado. A pilha não necessita ter mais do que duas posições, pois com um slot variável é possível armazenar uma ação auxiliar. Ações auxiliares, resultantes de outras ações auxiliares não necessitam de espaço dedicado, visto que a partir da ação da base é possível reconstruir a hierarquia de eventos necessários a sua execução. Devido a esta estratégia, pode haver um pequeno overhead na reinserção de ações auxiliares que foram removidas em função de outras ações auxiliares mais atuais.

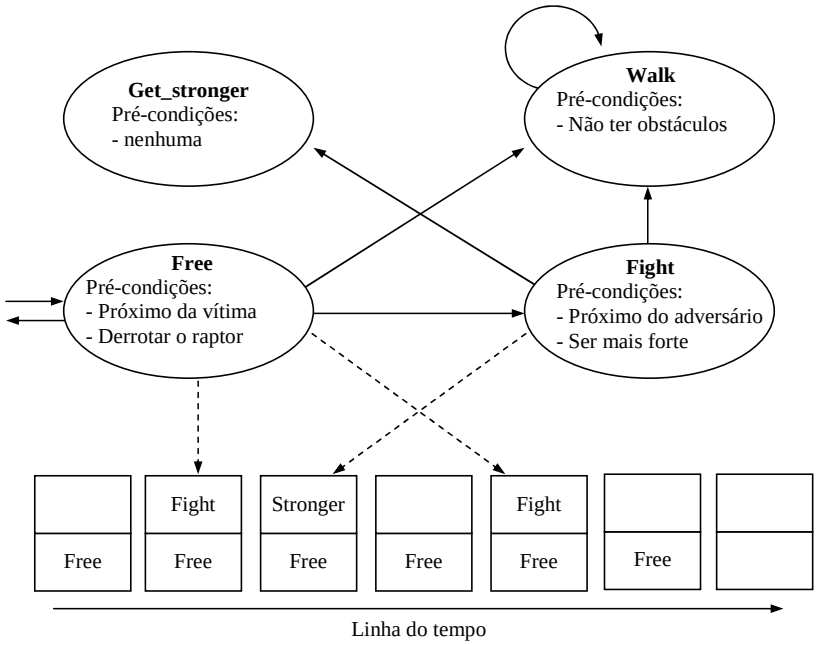

Figura 5.4: Gerenciamento da pilha pela FSM. Pré-condições de cada ação estão associados aos nós do grafo.

O uso de mais de duas posições pode levar a situações onde ações auxiliares, mesmo não mais válidas e necessárias, sejam executadas. Como exemplo, suponha que em um dado momento, o agente está executando a ação A_1 que consiste em ir a um determinado *waypoint* para desviar de um obstáculo. Neste mesmo tempo, ele detecta a presença de um inimigo que o faz fugir para uma direção perpendicular. Após a fuga, uma nova rota ao destino final deve ser criada. Se A_1 ainda estivesse na pilha, estaria no topo da pilha e deveria ser executada. Entretanto, após a fuga, a ação A_1 pode estar totalmente obsoleta, o que poderia resultar em uma ação sem lógica do ponto de vista do usuário.

Este mecanismo de pilha permite também que após o término da ação terminal, novas ações terminais sejam inseridas pelo próprio agente. Isso pode ocorrer em várias circunstâncias. Pode-se especificar que, após uma libertação, por exemplo, ambos os personagens devem ir para suas respectivas moradias.

Uma abordagem que também faz uso de uma pilha como forma de expandir as potencialidades das FSM foi proposta por [135], com a diferença que a pilha tem tamanho ilimitado. Neste presente trabalho, todos os exemplos testados indicaram que uma pilha com apenas duas posições é a melhor solução. De fato, na arquitetura proposta, ações complexas como "perseguir um inimigo, desviando de obstáculos, para iniciar um combate",

podem ser adequadamente tratadas.

Mais exemplos que mostram o gerenciamento de situações complexas, que envolvem comportamentos de manobra, são apresentados na próxima seção.

5.2.2 Comportamentos de manobra

O modelo proposto incorpora explicitamente um único tipo de manobra: atingir uma posição específica (*Seek*). Apesar de parecer um tanto simplificado e limitado, este comportamento é utilizado implicitamente como base para execução de outros comportamentos como: perseguir, seguir caminho e evitar colisão.

Para melhor compreender como estes comportamentos são implementados, deve-se observar como está estruturado o ambiente onde os agentes estão inseridos. A estratégia proposta de movimentação do mundo é baseada em *waypoints*[2] e análise de terreno (*Terrain Reasoning*). *Waypoints*, de modo geral, são sinalizações de locais que podem ajudar o agente a navegar no mundo, uma vez que eles agem como nós de um grafo que representam os caminhos pelos quais o agente pode navegar [87]. Análise de terreno é uma técnica de IA para jogos que auxilia o raciocínio sobre informações do terreno, de modo que planejamento, tomada de decisões, controle de ações, dentre outros, podem ser processados [130].

Todas as ações são associadas com *waypoints*, que podem ser estáticos, no caso de objetos dispostos no cenário (como moradias e construções), ou dinâmicos, no caso de ações que envolvem outros personagens. No presente trabalho, a cada objeto do cenário estão associados 5 *waypoints*, como mostrado na Figura 5.5. Os quatro presentes nos cantos representam um caixa envolvente (*bounding box*) estendida do objeto e o *waypoint* central representa a porta, que é usado para guiar o personagem para dentro e para fora do objeto. O volume envolvente é usado para testes de colisão com o caminho corrente do agente. Em frente ao local de entrada, é definida uma região proporcional ao tamanho do objeto que é considerada área externa do objeto. Tanto a área externa como a interna podem ser usadas para designar alvos em ações do tipo `Walk`.

O motivo principal para o uso da rede de *waypoints* como estratégia de locomoção, em oposição a algoritmos de planejamento de caminho do tipo A* [21], é a possibilidade de se utilizar o modelo comportamental,

[2]Esta tese sugere a tradução de *waypoint* para pontos de caminho.

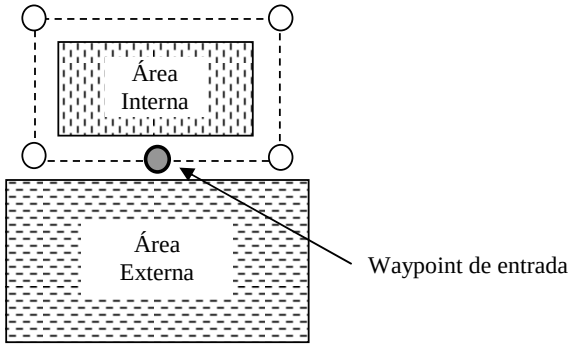

Figura 5.5: *Waypoints* associados aos objetos.

apresentado na seção anterior, como suporte à movimentação e desvio de obstáculos, bem como para o tratamento de situações que involvam na locomoção do agente para o interior de um objeto.

A idéia é quebrar o caminho completo em pequenos segmentos, como mostrado na Figura 5.6. A criação dos segmentos ocorre sempre que o caminho selecionado (corrente) fizer colisão com algum objeto da cena (que inclusive pode ser dinâmico). O novo segmento adicionado deve momentaneamente mudar a rota de modo a evitar a colisão. Para a escolha do novo segmento, faz-se uso da visão do agente. Se o caminho tiver interseção com a caixa envolvente do objeto, procura-se pelo *waypoint* mais próximo do destino que seja visível pelo agente. A nova rota geralmente representa a menor distância ao destino. A Figura 5.7 mostra um exemplo deste algoritmo.

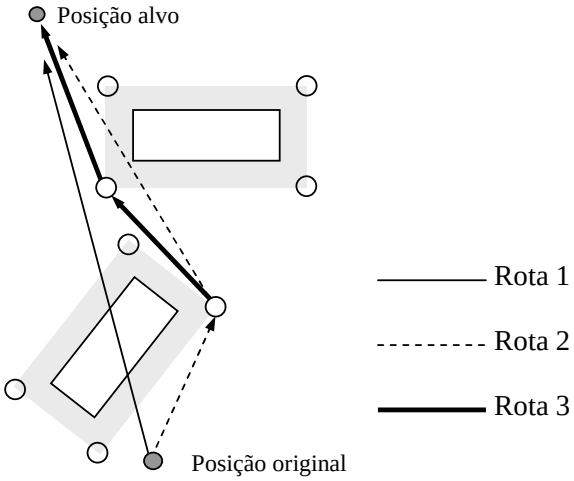

Figura 5.6: Composição incremental da rota.

Cada segmento, à medida que o agente se move no ambiente, é convertido em uma ação do tipo `Walk`, cujo destino imediato é a extremidade

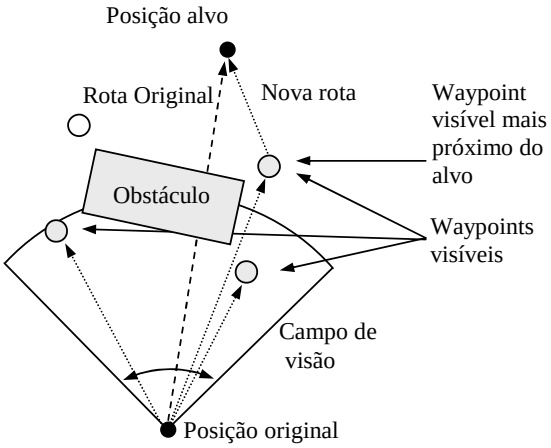

Figura 5.7: Algoritmo para seleção do melhor *waypoint*.

final do segmento. Uma vez que o caminho é organizado como um conjunto de pequenos caminhos individuais, essas ações de `Walk` podem ser facilmente inseridas e removidas na pilha do agente, sempre que se fizerem necessárias para assegurar a finalização da ação terminal. Desta forma, a FSM trabalha transparentemente com qualquer tipo de ação, incluindo planejamento de caminho e tratamento de colisão com objetos fixos e móveis.

Para situações mais complexas, como fazer o agente entrar em uma casa, por exemplo, faz-se uso da mesma estratégia. O agente inicialmente encontra o *waypoint* visível mais próximo do destino final. Quando ele chega a esta posição, tenta novamente ir ao destino final. Se uma nova colisão for detectada, um novo *waypoint* (diferente do corrente) é selecionado. Este processo continua até que o agente atinja o *waypoint* de entrada. Neste momento, os testes de colisão são desabilitados e o agente pode então entrar em casa. Na figura 5.8 é apresentado um exemplo do funcionamento do algoritmo proposto.

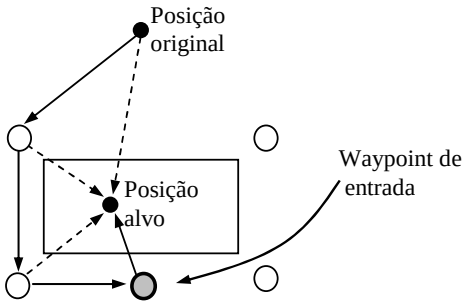

Figura 5.8: Exemplo de uma rota para entrar em um objeto do cenário.

Esta estratégia de tratamento de colisão falha quando as caixas envolventes de objetos têm interseção. Na figura 5.9 é apresentado um

exemplo desta configuração. Pode-se observar em (b) que o trajeto selecionado conduz o agente a um *waypoint* não alcançável, impedindo-o de atingir o alvo.

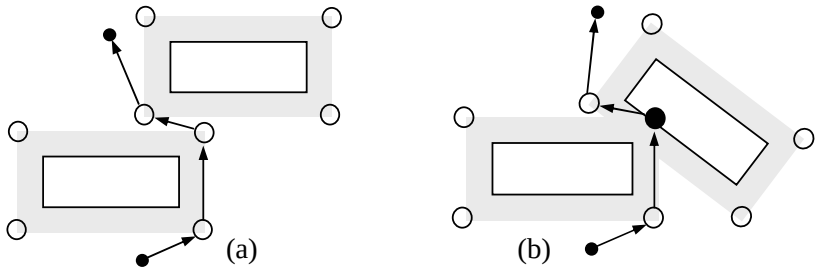

Figura 5.9: Exemplo de caminhos atingível (a) e não atingível (b).

5.3 Câmera Virtual

Conceitos de agentes também fazem-se presentes na estipulação de comportamentos que a câmera virtual deve apresentar. A câmera é um dos principais módulos de visualização, pois é responsável por determinar qual cena, a cada momento, deve ser exibida. A câmera deve desempenhar o papel de um diretor que, como em um filme, deve definir diversos parâmetros para ressaltar, da melhor forma possível, os principais fatos que estão ocorrendo. Ela deve ser autônoma o suficiente para selecionar o conteúdo a exibir, se posicionar, se locomover e fazer transições entre tomadas sucessivas.

Existem diversos trabalhos com o intuito de criar sistemas que procuram determinar o melhor posicionamento da câmera em cenas interativas 3D [63][111]. Para situações de tomadas individuais, diversas estratégias e restrições de tomadas podem ser encontradas em Drucker et al [43][45]. Uma tomada é o intervalo de tempo que a câmera grava continuamente. Aspectos matemáticos para definição de parâmetros de câmera, dada a geometria da cena e posição do personagem desejado, podem ser vistos em [14][68][44]. Diversas heurísticas e restrições referentes a tomadas sucessivas na criação das cenas são propostas [69][32]. Para definição da ordem destas tomadas, existem fórmulas específicas que melhor capturam a essência da cena. Uma longa discussão destas fórmulas, com situações práticas de uso, pode ser encontrada em [5].

Dois trabalhos merecem atenção especial, pois têm como foco a definição de sistemas automáticos de cinematografia para serem usados em ambientes virtuais interativos. Tomlinson [134] apresenta um sistema que é

responsável pelo controle de uma câmera e luzes em um mundo virtual 3D habitado por personagens autônomos e controlados pelo usuário, de forma que o conteúdo emocional possa ser evidenciado. A câmera é implementada por um agente chamado *CameraCreature*. Ele encapsula sensores para extrair informações sobre os estados emocionais e motivacionais, além das ações que estão sendo realizadas pelos agentes do ambiente. Além dos sensores, esta câmera autônoma também possui emoções, motivações e ações. O modelo de emoção da câmera é usado na escolha da tomada a ser realizada e dos parâmetros relacionados. Cada estado emocional causa um efeito visual característico. O estado emocional da câmera é calculado por uma função que leva em consideração o seu temperamento, a taxa de variação da emoção no tempo e fatores internos e externos que afetam o estado emocional. Uma vez que o estado emocional dos agentes da cena tem influência no estado emocional da câmera, os estilos de tomadas refletem o estado emocional dos personagens. Para ressaltar este estado emocional do grupo, são usados tanto estratégias de posicionamento de câmeras como recursos de iluminação

Em He [69], é descrito o *Virtual Cinematographer* (VC), uma arquitetura que automaticamente gera especificações de câmera para captura de eventos em ambientes virtuais 3D, em tempo real, a partir de informações e eventos passados pela aplicação. A estrutura geral é apresentada na Figura 5.10.

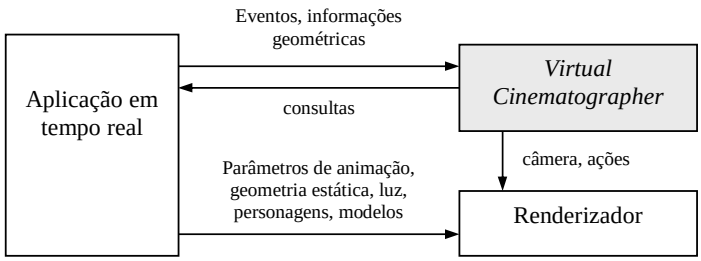

Figura 5.10: Arquitetura de uso do Virtual Cinematographer [69].

O conhecimento necessário para determinação das seqüências de tomadas para cada cena, bem como as condições necessárias para fazer a transição de tomadas, são encapsulados em *idioms*. Os *idioms* são implementados por máquinas de estados finitas hierárquicas (HFSM), onde cada estado está associado com um módulo de câmera. Cada *idiom* tem a capacidade de capturar um tipo de situação particular, como o diálogo entre dois personagens ou o deslocamento de um personagem de um lugar para outro.

Nesta tese, faz-se uso de uma arquitetura semelhante a He [69]. Entretanto existem vários aspectos que diferenciam os dois modelos. Um deles está no tipo de cena a ser visualizada. Em He [69], a câmera deve monitorar ações de um personagem específico, que geralmente é o que está sob o controle do usuário. No sistema proposto, a câmera deve decidir, entre um elenco de personagens que compõem a história que está sendo visualizada, qual deve ser exibido e de que forma. Estes personagens podem estar realizando ações individuais ou em dupla. Além de selecionar o(s) personagem(s), a câmera deve também se posicionar e se locomover para acompanhar personagens em movimento.

Para a seleção do(s) personagem(s), a câmera consulta o Gerenciador de Ações para saber qual a operação corrente. Com esta informação, descobrem-se quais são os personagens e, desta forma, a ação que está sendo processada. Maiores detalhes sobre a câmera são apresentados no Capítulo 7, que trata da implementação.

5.4 Drives e emoções

A representação de agentes com comportamentos realistas é fundamental em aplicações de entretenimento, como cinema, iTV [42], jogos de computador e realidade virtual [39]. Segundo Brezeal [16], o comportamento, juntamente com expressões faciais, são o meio pelo qual o agente pode expressar sua motivação, que é determinada por drives e emoções.

Os drives influenciam na seleção de comportamento e no estado emotivo [57]. Uma vez que a expressão do agente reflete seu estado emotivo, o grau de satisfação dos drives influencia indiretamente nas expressões faciais [16]. As expressões faciais, em aplicações de entretenimento, podem ser usadas para ativar o aspecto emotivo do observador.

Além de expressões faciais, o agente também pode fazer uso de gestos físicos e expressões vocais como rir, chorar, bocejar (*non-language sounds*) para exteriorizar seu estado emocional [142]. As expressões faciais podem ser representadas, por exemplo, pelas sobrancelhas, olhos, pálpebras, orelhas e boca.

Drives não podem se satisfazer sozinhos. Eles se tornam satisfeitos quando o agente é habilitado a ativar um comportamento adequado para supri-los. Por exemplo, para suprir o drive de fome, o agente deve prover-se de recursos de alimentação, assim como deve dormir para suprir o drive

de fadiga. Quando tal comportamento é ativado, a intensidade do drive é reduzida [16]. Um drive ou emoção é considerado ativo quando sua intensidade for superior a um dado limiar (*threshold*).

Geralmente os drives têm um comportamento cíclico temporal, ou seja, sem estímulos um drive tende a aumentar em intensidade, a não ser que seja satisfeito. A cada drive corresponde um valor desejável, com limite inferior e superior aceitáveis, dentro dos quais é considerado permanecer em regime homeostático (drive satisfeito).

Os drives têm influência direta sobre as emoções. Emoções positivas, como a felicidade, ocorrem quando os drives são satisfeitos. Emoções negativas, como a tristeza, ocorrem quando um drive não está satisfeito, ou quando é super-satisfeito (excedendo o limite superior aceitável).

As emoções podem ser provenientes de estímulos externos (eventos do ambiente ou de outros agentes) e internos (intensidade dos drives) [142]. Segundo Izard [81], a ativação de emoções, assim como o aumento ou redução de sua intensidade, pode ocorrer por estímulos de sensores cognitivos e não cognitivos.

Segundo Velazquez [142], as emoções também têm influência no comportamento de agentes autônomos. Esta constatação resulta do fato das emoções não serem provenientes somente da ação de drives, mas também de estados internos do agente, bem como da interação com outros agentes.

A intensidade de uma emoção precisa ultrapassar um limiar para ser expressa externamente ao agente. Além do nível de ativação, diversas questões devem ser consideradas ao se trabalhar com emoções [37]:

- Como as emoções são mantidas e como é a sua alteração no tempo - Cada emoção deve possuir um limiar (*threshold*) de ativação e uma função de decaimento, que controla a duração da emoção;
- Como elas interferem em outras - Algumas emoções podem ativar ou inibir outras emoções (ex: o medo pode inibir a felicidade);
- Como elas surgem da interação com outros agentes;
- Como elas são influenciadas pelas características internas do agente.

Segundo modelos teóricos sobre emoções [49][80], existe um conjunto de emoções básicas ou primárias, que possuem como características determinados aspectos que as diferenciam das demais e que estão relacionados com eventos fundamentais da vida cotidiana. Estas emoções também apresentam características que podem ser facilmente mapeadas em traços distintos por meio de expressões faciais [41]. São elas: raiva,

tristeza, alegria, medo, repulsa e surpresa (*distress/sadness, anger, enjoyment/happiness, fear, disgust, surprise*). Elas possuem traços característicos, principalmente nos olhos e boca, e em locais onde há formação de rugas, como mostrado na Figura 5.11.

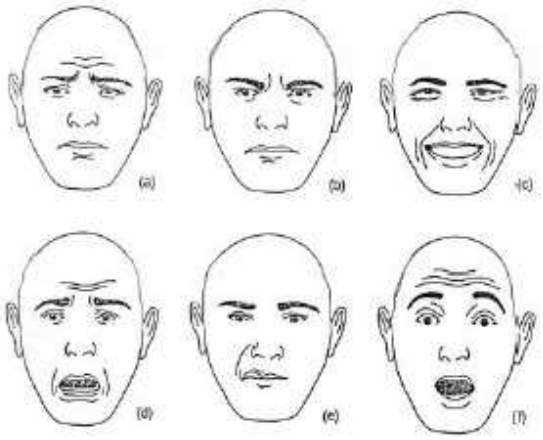

Figura 5.11: Exemplos das expressões faciais universais [109].

Durante o desenvolvimento deste trabalho, investigou-se a possibilidade da incorporação de drives e emoções, bem como expressões faciais aos agentes. Drives e emoções podem ser considerados tanto na geração como na representação de histórias interativas, sob três aspectos: forma como são gerados, sua influência no comportamento dos agentes e o efeito sobre o direcionamento da história. As expressões faciais estão relacionadas somente com a representação gráfica.

Tanto drives como emoções podem ser facilmente representados internamente ao agente por meio de valores numéricos, dentro de um intervalo predefinido, que indicam sua intensidade no momento[3]. Em se tratando da geração da história, podem ser facilmente associados a pré- e pós-condições das operações, bem como podem ser usados nas regras de inferência de objetivos. Suas intensidades podem ser explicitamente definidas, da mesma forma como ocorre com os demais atributos, em função da operação que está sendo processada e de características inerentes de cada personagem.

A nível de representação gráfica, o agente está imerso em um ambiente real e, por isso, propenso a interações com o meio e com outros agentes, o que pode influenciar fortemente na alteração de drives e emoções e, conseqüentemente, no seu comportamento. Como nesta etapa

[3]Uma representação mais robusta pode ser implementada usando lógica nebulosa (*fuzzy logic*) e variáveis lingüísticas.

são considerados aspectos do cenário, como distâncias e obstáculos, bem como personagens figurantes, pode-se determinar a intensidade de drives e emoções para cada ação que os personagens realizam. Desta forma, eles podem ter um papel fundamental na geração de novos objetivos, podendo assim influenciar no direcionamento da história sendo criada pelo IPG.

Alterar o rumo da história previamente simulada e certificada, pela inclusão de novas ações que visem suprir algum drive, por exemplo, é uma questão que ainda deve ser estudada e é sugerida como trabalho futuro. Em um momento inicial, drives e emoções podem ter apenas um efeito passageiro, servindo assim como enriquecedores da apresentação gráfica da história. Pode-se considerar a geração de ações que levem o drive ao regime normal, mas após a finalização destes eventos, as ações que foram interrompidas devem voltar a ser executadas, sem comprometer de forma alguma o direcionamento da história.

Referente às emoções, a afeição é a única que está atualmente implementada no IPG. Na etapa de visualização, as emoções podem ser usadas apenas para efeito da própria representação, ou seja, fazer uso de expressões faciais para descrever o estado emocional do agente associado ao comportamento que está sendo apresentado a cada momento. Nesta abordagem, as emoções são resultado da execução de uma tarefa concreta (*task-oriented emotions*) [64].

Sob o aspecto de implementação, a incorporação dos drives é um processo relativamente simples. Uma vez que ações para suprir drives são geradas pelo próprio agente, este pode usar a estrutura da pilha para gerenciar as ações auxiliares. Desta forma, a ação terminal continua na base da pilha e será executada assim que drives forem supridos e pré-condições satisfeitas, observando-se sempre qual tem maior prioridade. No caso das emoções, uma alternativa simples seria o uso de diferentes texturas, que seriam mapeadas sobre a face do personagem. Representações mais complexas geralmente utilizam-se estruturas que tratam tanto a configuração dos ossos [109] como dos músculos presentes na face [110]. Os músculos faciais podem tanto ser usados para representação das expressões bem como da fala do personagem [92]. Neste caso, deve-se dispor de modelos complexos de personagens, o que está fora do escopo desta tese. Além disso, para ressaltar a expressão facial, o módulo de câmera, em comunicação com os agentes, deve fazer um enquadramento que possa evidenciar esta característica.

Na Figura 5.12 é apresentado um esquema conceitual para a incorporação de drives e emoções no sistema proposto. Maiores detalhes

sobre o **Gerenciador de Ações** podem ser vistos no Capítulo 6.

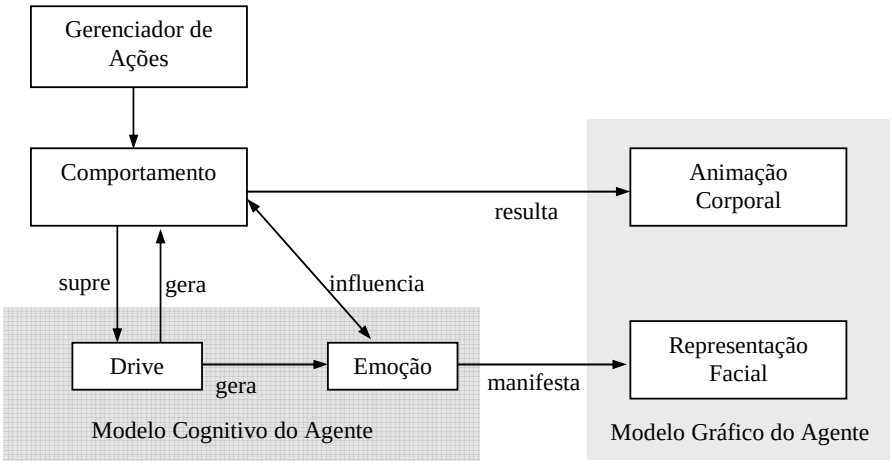

Figura 5.12: Esquema conceitual para tratamento de drives e emoções.

5.5 Conclusões

Neste capítulo apresenta-se uma abordagem geral sobre teorias de agentes e sistemas multi-agentes necessárias para dar suporte à exibição interativa de histórias criadas dinamicamente por um processo de simulação. Estas teorias podem ser tanto aplicadas ao processo de geração da história, como, principalmente, à exibição da história gerada. O uso de agentes tem mostrado ser uma boa alternativa para atender à complexidade do sistema.

Para o tratamento de comportamentos, o uso da pilha, associada à máquina de estados finitos (FSM), tem se apresentado como uma ótima solução ao tratamento de ações complexas. A limitação em duas posições mostra-se como melhor alternativa para o tratamento de ações intermediárias que surgem como forma de suprir pré-requisitos das ações terminais. O mesmo vale para ações resultantes do efeito que drives e emoções têm sobre os agentes.

Apesar de não terem sido incorporados na implementação, drives e emoções mostram um forte potencial para aumentar o grau de realismo das histórias, tanto a nível de geração como de visualização. Atualmente, o grupo de pesquisa do ICAD está estudando a possibilidade do tratamento de drives e emoções para a definição de comportamentos mais realistas. Este grupo pretende brevemente fazer testes reais e definir qual o nível de influência adequado que drives e emoções devem ter sobre as ações dos personagens. O mesmo vale para as expressões faciais, que, além de

estarem associadas a drives e emoções, também possuem uma relação muito forte com o modelo geométrico usado na representação do personagem. O conjunto mínimo necessário de expressões somente vai ser definido mediante experimentos reais, em função das possíveis ações que cada agente poderá realizar, levando também em conta as potencialidades da câmera virtual, que desempenha um papel fundamental para a visualização das cenas.

6
Arquitetura do Sistema

Nos capítulos anteriores são apresentados diversos aspectos relacionados com a geração das histórias (conteúdo, geração, níveis de interatividade, diversidade), que têm como apoio lógico o IPG (*Interactive Plot Generator* [33]). Para permitir que o usuário possa interativamente guiar e visualizar as histórias sendo geradas, neste capítulo faz-se a proposição de um sistema que integra os processos de geração, interação e visualização de histórias. Este sistema tem como intuito permitir que o usuário possa interagir com o conteúdo tanto na geração como na visualização das histórias, por meio de interfaces gráficas.

Como um dos objetivos desta tese é a proposição de conteúdo baseado em histórias interativas para ser usado em um ambiente de TV interativa, o sistema proposto contempla características que permitem fazer uso da tecnologia corrente de hardware e software para iTV como forma de disponibilizar histórias interativas personalizadas a cada usuário.

6.1
Arquitetura Geral

A Figura 6.1 apresenta um diagrama geral do sistema de histórias interativas em iTV. Ele é baseado em uma arquitetura distribuída cliente-servidor, onde o servidor age somente como o provedor de conteúdos (base de dados), enquanto o cliente age como receptor, gerador e visualizador do conteúdo interativo.

No servidor são definidas duas bases de dados, onde são armazenados conteúdos necessários tanto para a geração como para a vizualização dos enredos. Na base Prolog são especificados os contextos lógicos dos enredos a serem disponibilizados aos usuários como forma de conteúdo interativo. Cada enredo é armazenado em um único arquivo texto, que contém os predicados (personagens, lugares, atributos), operações e regras

de inferência dos objetivos a serem perseguidos pelos personagens, como pode ser visto no Apêndice A.

Cada enredo está associado a elementos gráficos que são usados para transformar as representações simbólicas das operações em ações interpretadas por personagens habitando um mundo virtual 3D. Esses elementos gráficos são armazenados em uma base gráfica, que contém os modelos 3D de personagens e objetos do cenário, bem como texturas e definições do próprio ambiente, por meio da geometria do terreno.

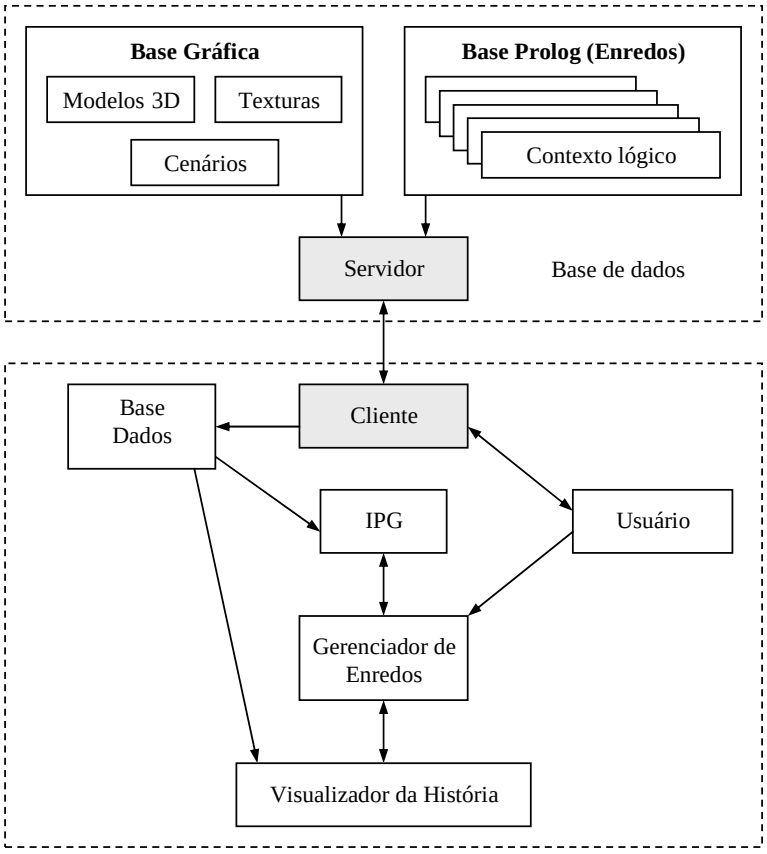

Figura 6.1: Estrutura para disponibilização de histórias interativas em iTV.

A Figura 6.2 apresenta um modelo de implantação do sistema dentro de um ambiente de iTV. No nível mais baixo encontra-se o provedor de conteúdos e no nível mais alto, o usuário. O Set-top Box está localizado no centro e é o intermediador de todo o processo. Ele realiza todo o processamento computacional referente à solicitação, ao recebimento e armazenamento de conteúdo interativo, bem como a geração e exibição gráfica das histórias, mediante interação com o usuário.

A arquitetura apresentada na Figura 6.1 permite que o usuário possa selecionar o contexto de histórias com o qual deseja interagir. Por meio de uma interface com o cliente, ele pode saber quais contextos estão disponíveis

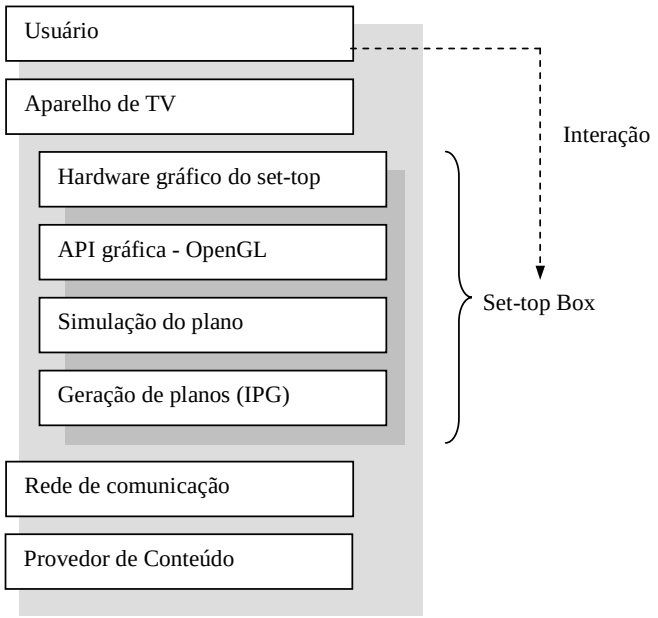

Figura 6.2: Modelo de implantação do sistema em um contexto real de iTV.

na base do provedor e selecionar o que mais lhe interessar. Uma vez selecionado o contexto, os dados referentes a ele são enviados pelo provedor ao cliente e então armazenados em uma base local. O **Gerenciador de Enredos** é o componente centralizador, que coordena tanto a geração como a visualização.

A colocação do IPG no lado do cliente tem dois objetivos: permitir que cada usuário tenha sua história personalizada e garantir maior eficiência, visto que a geração de planos é um processo computacionalmente caro. Deste modo, cada usuário faz uso de processamento local do set-top box, o que assegura menor tempo de resposta, bem como evita comunicações freqüentes com o provedor de conteúdos.

O uso de banda passante, geralmente um problema em diversos tipos de conteúdos interativos, não apresenta maiores impactos, visto que a comunicação com o servidor ocorre apenas num momento inicial, quando o usuário seleciona uma história a interagir. Neste momento, dados referentes à história, bem como modelos gráficos, devem ser levados ao cliente e então armazenados na base local para futura utilização. Durante todo o resto do processo de interação com a história, todo o processamento é realizado localmente. Antes de realizar qualquer requisição de dados ao provedor, pode-se fazer uma consulta ao cache local do set-top box, para verificar se o dado requisitado, ou parte dele, já se encontra disponível localmente.

Como mencionado no Capítulo 2, tanto o provedor de conteúdos como a camada de rede são abstraídos com o uso de um único computador, que

desempenha o papel de provedor e de set-top Box. A Figura 6.3 apresenta os módulos da arquitetura que atualmente estão implementados. A base de dados local já é inicializada com todos os dados necessários. Para o usuário, a interação ocorre de forma transparente. Nesta figura, o componente de visualização da história está apresentado de modo mais detalhado.

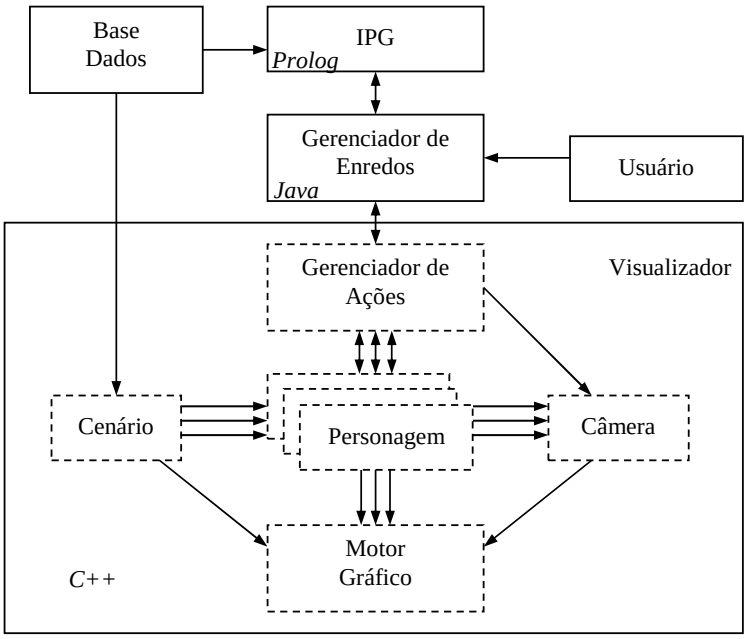

Figura 6.3: Módulos implementados do sistema. O módulo de visualização é apresentado em maiores detalhes.

O **Gerenciador de Ações** tem a função de processar os eventos fornecidos pelo Gerenciador de Enredos, transformando-as em ações que são então delegadas aos respectivos personagens. Ele também realiza todo o sincronismo de visualização das ações de personagens para a representação dos eventos. Toda vez que um evento é processado, um novo é requisitado ao Gerenciador de Enredos.

6.2 Os Módulos do Sistema

Os três grandes módulos do sistema (IPG, Gerenciador de Enredos e Visualizador) foram implementados usando as linguagens mais adequadas para cada tipo de tarefa, como apresentado na Figura 6.4. A linguagem JAVA utilizada no Gerenciador de Enredos é utilizada como base para a comunicação entre os módulos. Cada módulo faz uso de linguagens/bibliotecas multiplataformas, um requisito fundamental para

viabilizar a implantação desta tecnologia nas diversas arquiteturas de set-top boxes e sistemas operacionais.

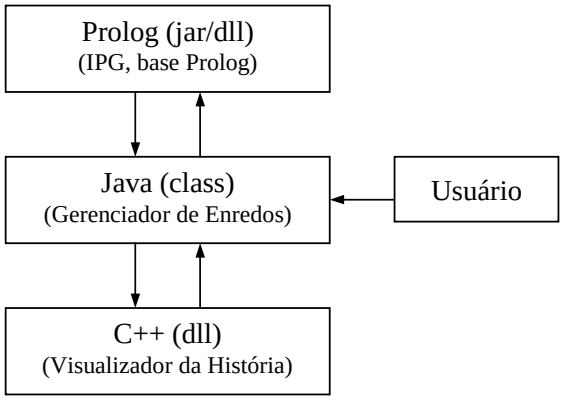

Figura 6.4: Linguagens usadas para implementar os módulos.

6.2.1 O Gerenciador de Enredos

Gerenciador de Enredos age como centralizador e integrador da arquitetura, servindo como ponte de comunicação entre o IPG (implementado em Prolog) e o **Visualizador de Histórias** (implementado em C++). Totalmente implementado em Java, ele concentra todos os recursos de interação com o usuário. A funcionalidade do Gerenciador de Enredos engloba os seguintes itens:

- Interação com o IPG para guiar a geração da história, segundo comandos do usuário. Pode requisitar ao IPG novas alternativas de enredos parciais, a continuação de um enredo, ou a adição de novos eventos e ordens temporais a um enredo que está sendo processado;
- Interface com o usuário para o gerenciamento das operações geradas pelo IPG, ou inseridas pelo próprio usuário. Por meio de uma interface gráfica, o usuário pode requisitar a renderização gráfica, fazer a inserção/remoção de eventos e forçar a ocorrência de situações específicas no enredo, bem como definir a ordem total dos eventos;
- Interação com o módulo gráfico para gerenciar a visualização da história. Gerencia também o envio de eventos a serem representados, o que pode ser resultado de uma ação explícita do usuário, ou por uma requisição do módulo gráfico.

6.2.2 O IPG

Para executar o IPG, faz-se uso do SICStus Prolog [125], um interpretador Prolog multiplataforma[1] desenvolvido pelo Instituto de Computação da Suécia (*Swedish Institute of Computer Science*). O SICStus é executado em código nativo, em formato de biblioteca de ligação dinâmica, podendo assim ser incorporado a outros aplicativos.

O SICStus disponibiliza uma interface em Java chamada Jasper que permite que uma aplicação Java possa acessar o SICStus de forma transparente (Isso é realizado via chamadas JNI - *Java Native Interface* [84]).

Sobre o Jasper, implementou-se em Java o **Componente de Consulta**. Ele tem a função de interagir com o Jasper no intuito de converter os dados gerados pelo IPG em um formato que possa ser facilmente interpretado pelo Gerenciador de Enredos (em Java). O **Componente de Consulta** também tem como função fazer a inicialização do IPG, com a história a ser processada, bem como fornecer acesso simplificado a eventuais consultas feitas sobre a base Prolog.

Na Figura 6.5 é apresentada a hierarquia de componentes que permite a comunicação entre o Gerenciador de Enredos e o IPG.

Figura 6.5: Camadas entre o Gerenciador de Enredos e o IPG.

[1] Atualmente, as seguintes plataformas e sistemas operacionais são disponíveis: Intel/x86 (Windows 2000/XP, Linux, Solaris 7), Compaq/Alpha (Tru64 Unix 5.1), HP9000/7XX (HP-UX 11.x), IBM RS/6000(AIX 4.3.3, AIX 5L 5.1 (32/64 bits)), Silicon Graphics (IRIX 6.5), Sun Sparc (Solaris), Macintosh (MacOS).

6.2.3 O Visualizador da História

O módulo Visualizador da História, implementado em C++, cuida de todas as funcionalidades necessárias à visualização gráfica das histórias. Como mostrado na Figura 6.3, ele incorpora o **Gerenciador de Ações**, o cenário, os personagens, a câmera e o motor gráfico. O módulo faz uso tanto de processos de Inteligência Artificial como de Computação Gráfica.

Quando o usuário requisita a visualização da história, o **Visualizador da História** recebe do Gerenciador de Enredos um conjunto de eventos a serem representados graficamente. Cada operação está associada a um evento, que por sua vez está associado a personagens. O Gerenciador de Ações faz o gerenciamento de todos os personagens. Ele tem como função avaliar cada evento e invocar os respectivos personagens a realizar o evento. Os personagens, implementados como agentes reativos, devem realizar as ações associadas e enviar seu estado geométrico (representado pela malha de polígonos corrente) ao motor gráfico para ser então desenhado na tela. A câmera virtual tem como papel exibir a ação que está sendo processada pelo personagem encarregado de representar a operação corrente.

A Figura 6.6 ilustra as camadas para a implementação do Visualizador de Histórias e a comunicação deste com o Gerenciador de Enredos. O produto final do Visualizador são malhas poligonais que representam a geometria da cena no dado momento. Estas malhas são renderizadas pela biblioteca OpenGL [147]. Todos os componentes deste módulo são definidos em C++ e encapsulados em uma dll, que é carregada pelo Gerenciador de Enredos. Para que este módulo fale com o Gerenciador de Enredos, disponibilizam-se, via interface JNI, um conjunto de funções.

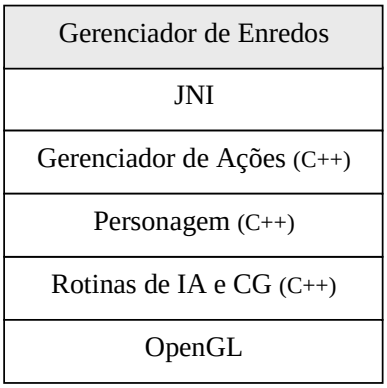

Figura 6.6: Hierarquia em camadas do módulo C++.

6.3 Conclusões e Observações

Por fazer uso de diversas tecnologias de software, o sistema utiliza três linguagens diferentes de programação, bem como uso de várias outras bibliotecas. Procurou-se fazer escolhas que permitissem o seu uso em múltiplas plataformas.

A integração de toda esta estrutura em um único sistema é uma tarefa complexa que exigiu longo período de estudos, planejamento e implementação. A linguagem Java, por disponibilizar uma API para comunicação com aplicações escritas em código nativo (JNI), é o elemento centralizador, que também inclui interface com o usuário, que é o meio por onde o usuário controla tanto a geração, organização e visualização das histórias.

Neste processo de integração, o módulo que apresentou os maiores desafios foi a integração com o IPG, não somente sob o aspecto de linguagem de programação, mas principalmente sob o aspecto de formas de interação com o usuário. No que se refere à implementação, as versões mais novas do SICStus Prolog disponibilizam APIs tanto em C como em Java, permitindo assim que as aplicações conversem diretamente entre si. Mesmo com a existência desta API, foi necessário desenvolver mais uma camada em Java, que além de traduzir mensagens trocadas entre os dois módulos, também incorpora recursos que permitem a sincronização da história que está sendo processada pelo IPG com a história sendo manipulada pelo usuário na interface gráfica. A integração dos módulos Java e C++ é mais simples pois ambos os módulos compartilham dados muito similares, uma vez que Java e C são linguagem procedimentais e de sintaxe muito semelhantes.

No próximo capítulo apresentam-se detalhes da implementação dos módulos, como ocorre a comunicação entre eles, bem como detalhes das bases de dados usadas para a geração e a visualização das histórias.

7
Implementação e Resultados

Níveis de abstração são uma realidade na construção de motores (*engines*) para tratar histórias interativas [129]. Um mesmo personagem pode ser visto, em diferentes níveis de abstração, de forma totalmente diferente. No nível de geração da história (IPG), os personagens são apenas entidades lógicas, que possuem objetivos a serem cumpridos. Quando tratados no nível de interação com o usuário, são entidades fictícias que podem ser manipuladas e organizadas com certa ordem para gerar encadeamentos de eventos. No nível de representação gráfica, a abstração é substituída por uma forma física, que permite expressar graficamente seus comportamentos. O usuário pode presenciar cada ação executada e todo o nível de detalhes que o ambiente onde o personagem está inserido disponibiliza.

O nível de detalhes da implementação dos personagens (que neste caso são vistos como atores guiados [132][23]) deve ser suficiente para permitir que eles tenham uma representação "realista", saibam cumprir ordens e, ao mesmo tempo, possuam autonomia para tomar certas decisões (nível de planejamento em baixo nível de detalhes), dentro de um espectro limitado pela concepção do sistema. Estas decisões levam o personagem ao cumprimento de sua meta da maneira mais adequada do ponto de vista do usuário que está guiando a história interativa.

Como apresentado no capítulo anterior, existem vários módulos, tanto de IA como de CG, que trabalham conjuntamente para permitir que as histórias simuladas no IPG possam ser visualizadas interativamente. Este capítulo aborda aspectos de implementação do sistema proposto, ou seja, detalhes mais específicos de como cada módulo funciona e como se dá o sincronismo entre eles para garantir a coerência entre os dados processados a nível lógico com os dados apresentados por meio de recursos gráficos.

Inicialmente, apresentam-se discussões relativas à especificação do motor gráfico usado na representação das histórias. São abordados aspectos da escolha de um modelo gráfico adequado, bem como da especificação do

cenário. Do mesmo modo, apresentam-se detalhes mais específicos dos tipos de personagens utilizados, tratamento de ações individuais e em grupo, bem como mecanismos de comunicação entre personagens.

A Seção 7.4 apresenta os atributos que cenário e personagens incorporam, quando tratados a níveis de simulação e visualização. Juntamente, discutem-se estratégias de sincronismo destes atributos entre a geração e a visualização.

Na seqüência, apresentam-se detalhes da especificação da câmera virtual, que é o módulo responsável pela seleção automática do conteúdo a ser exibido ao usuário.

Na Seção 7.6, apresenta-se o ciclo de renderização das histórias ou, mais especificamente, as etapas que o motor gráfico realiza para transformar os eventos em animação gráfica. Apresenta-se também como é realizado o controle temporal durante a visualização de um evento.

Na Seção 7.7 apresentam-se as diferentes formas pelas quais o usuário pode interagir com o sistema, usando interfaces gráficas e, ao final, os resultados gráficos obtidos.

Os detalhes relativos a códigos e documentação do sistema podem ser encontrados no repositório ICAD/IGames/VISIONLAB [74].

7.1 Escolha de um Modelo Gráfico

Durante o desenvolvimento desta tese, o primeiro item levantado na especificação do motor foi o modelo gráfico a ser adotado: 2D ou 3D. Esta escolha define todos os demais aspectos referentes ao tipo de cena utilizada, tipos de personagens, ações que podem ser realizadas e, conseqüentemente, os recursos que a câmera virtual pode evidenciar.

Como a exibição da história pretende tratar de forma bastante realista a interação entre os personagens, as limitações do espaço 2D impedem a visão macroscópica do mundo de simulação, a visualização do cenário e a interação entre grupos de personagens. O uso de recursos 3D oferece maiores possibilidades, tanto referentes à visualização bem como a algoritmos, dispositivos de hardware, técnicas e modelos 3D amplamente utilizados e estudados para aplicações em jogos. Nesta escolha, não foi considerado o modelo isométrico (2.5D).

Uma vez escolhido o modelo 3D como forma de representar as histórias, buscaram-se alternativas para a sua implementação. Pode-se apontar de imediato duas abordagens possíveis: utilizar algum motor gráfico

já desenvolvido ou desenvolver um usando uma API gráfica, como OpenGL [147] ou DirectX [93]. Existe atualmente uma grande quantidade de motores criados principalmente para o desenvolvimento de jogos (como o Fly3D [52] e o 3D GameStudio [153]. Uma lista completa de motores pode ser encontrada em [50]), que aparentemente podem ser utilizados no processo de exibição das histórias. A maioria destes motores disponibiliza diversos recursos para animação de personagens, tratamento de colisão, simulação física, otimizações de renderização e scripts de IA.

Para melhor entender os critérios usados na avaliação entre a construção de um motor e a utilização de um já existente, são observados não somente os recursos disponíveis mas, principalmente, o nível de customização oferecido. Ao se trabalhar com histórias interativas, deseja-se criar cenários dinâmicos para que o usuário possa usufruir de histórias que definitivamente sejam personalizadas. A solução adotada deve fornecer recursos que permitam a total manipulação de atributos de personagens e do cenário, permitir fácil expansão e adaptação a qualquer tipo de história que um sistema de representação gráfico possa exibir, além de garantir exibição, em tempo real, de cenas que podem ser complexas e dinâmicas. Além disso, o sistema deve ser facilmente adaptado a diferentes arquiteturas de hardware (*set-top boxes*) usadas para prover poder computacional que permita a exibição de conteúdo iterativo em iTV. Deseja-se ter o máximo de customização sobre todos os processos envolvidos. Por estes motivos, apesar do desenvolvimento de um motor próprio ser uma tarefa demorada e complexa, a presente tese apostou nesta direção.

O motor está escrito em Linguagem C++, juntamente com a API gráfica OpenGL [147]. Sempre que possível, o motor incorpora bibliotecas de domínio público para tratar a manipulação de personagens e objetos do cenário (como no caso dos formatos MD2 e 3DS).

7.2 Estrutura do Cenário

O cenário é o local onde a história é representada graficamente. Ele serve como base para a simulação da interação entre personagens. Esta simulação, no presente trabalho, é realizada com um enfoque um tanto diferente, se comparado com o processo de geração da história pelo IPG. São considerados aspectos físicos, como distâncias, obstáculos e, conseqüentemente, aspectos do personagem, como velocidade e autonomia para determinar meios de realização das ações a eles designadas.

O cenário é definido como um ambiente aberto. O elemento de suporte aos demais componentes da cena é o terreno, onde são dispostos os objetos estáticos, como as construções, além dos objetos móveis, no caso, os personagens. Os objetos do cenário têm como principal papel servir como auxílio à navegação dos personagens em comportamentos de manobra. Para isso, todos os objetos possuem a mesma estrutura lógica baseada em *waypoints*, como apresentado na Seção 5.2.2. Os *waypoints* também são usados como auxílio ao posicionamento da câmera virtual (Seção 7.5). Os objetos são representados por modelos 3D e, para o contexto dos enredos adotado nesta tese, são usados para caracterizar casas, igrejas e castelos. Além do terreno e objetos, para tornar o ambiente mais realista, faz-se uso de uma biblioteca para a criação de um céu realista [4].

Na base Prolog, cada história a ser representada tem um elenco de personagens. Cada personagem, por sua vez, está sempre associado a locais, que podem representar residências ou objetos onde o personagem deve realizar uma ação. Como a quantidade de locais a serem representados no cenário depende da especificação da base Prolog, o terreno, associado à história, deve ter características que possam permitir uma distribuição adequada destes objetos sobre ele. Ele pode ser não-planar, desde que não possua regiões com altas irregularidades que impeçam a passagem dos personagens. Para assegurar estas restrições, a definição do terreno fica a cargo do autor da história, que também tem a função de definir a localização que cada objeto deve assumir.

A geração de terrenos não-planares é uma tarefa relativamente simples, pois consiste em gerar uma malha de triângulos a partir de um mapa de alturas (*height field*), que pode ser oriundo de uma função randômica ou de uma imagem, como mostrado na Figura 7.1. Para tornar a renderização mais eficiente, geralmente faz-se uso de algoritmos de LOD (*Level of Detail*) [88][58][46][89][139][140][90]. Esta é uma técnica que consiste em reduzir a complexidade da malha (número de triângulos), procurando manter, ao máximo, a geometria visual da mesma. Neste trabalho, foi utilizado um algoritmo proposto por Pozzer et al. [113].

7.2.1 Inicialização do cenário

O carregamento do cenário é realizado no início do processo de visualização, em função do contexto da história que está sendo processada. Cada contexto de história está associado a um arquivo texto que contém

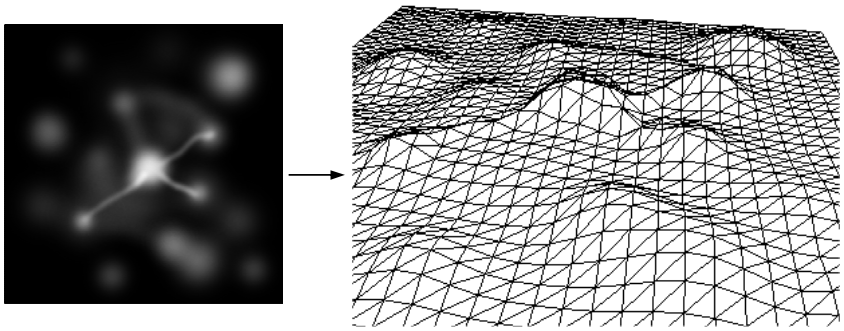

Figura 7.1: Geração de uma malha regular a partir de uma imagem.

a definição do terreno (mapa de altura e textura), dos objetos (arquivo, formato e escala) e de seus atributos (tipo, nome, posição, modelo 3D e orientação). A figura 7.2 apresenta um modelo deste arquivo.

```
//type   heightmap  texture
terrain  map.raw    textmap.jpg

//type    name            scale
modelPOZ  castle1.poz     7000
modelPOZ  castle2.poz     5000
modelPOZ  castle3.poz     5000
modelPOZ  church.poz      6000

//type       name             pos_x  pos_z  model        rotation
buildingPOZ  castle           18300  21000  castle1.poz  -30
buildingPOZ  knight_castle    24300  32700  castle1.poz   180
buildingPOZ  dragon_castle    34000  32000  castle2.poz   160
buildingPOZ  princess_castle  16300  20700  castle3.poz  -30
buildingPOZ  church           35500  19500  church.poz    160
```

Figura 7.2: Arquivo de especificação do cenário.

Esta estrutura permite que um mesmo modelo 3D possa representar mais de um elemento no cenário. O sistema faz uso de três bibliotecas para carregamento de modelos 3D: um para modelos MD2[1] [70], um para 3DS[2] [73] e um para modelos POZ[3].

Dispondo-se dos objetos 3D e do terreno, pode-se facilmente, por meio deste arquivo, criar vilarejos realistas com ruas (que podem ser representadas pela textura do terreno), igrejas, casas, comércio e fábricas. O cenário pode ter objetos (locais) que não sejam de conhecimento do IPG, porém o inverso não pode ocorrer.

[1]O formato MD2 foi criado para ser usado na modelagem das animações dos personagens do Jogo Quake II.

[2]Formato exportado pelo programa 3D Studio Max.

[3]O formato POZ foi desenvolvido pelo autor desta tese como forma de contornar limitações apresentadas pelo carregador 3DS quando está tratando modelos complexos. O arquivo 3DS é carregado pelo programa Deep Exploration [40], que exporta a malha do modelo em formato CPP, que é então convertida para o formato POZ.

7.3 Personagens

Existem dois tipos de personagens implementados no visualizador gráfico: os principais e os figurantes. Os principais estão definidos na base Prolog e são usados pelo IPG no processo de simulação da história. Os personagens figurantes existem apenas no motor gráfico e são usados para tornar a representação das histórias mais realista, uma vez que comumente existem poucos personagens principais. Até o momento, não se tem conhecimento de nenhum trabalho em *storytelling* que faça uso de personagens figurantes com este intuito.

Pela concepção do sistema integrado de simulação/visualização, como o próprio nome diz, toda a parte de geração da história deve se concentrar na simulação, sobrando para a visualização apenas a representação gráfica das ações fornecidas pelo simulador. Entretanto, é importante poder adicionar um certo nível de simulação no módulo de visualização, visto que diversos fatos interessantes podem surgir da interação física entre personagens. Isso resulta principalmente da existência dos personagens figurantes, que podem estar em contínua interação com os personagens principais da história.

A solução ideal nesta configuração é que os personagens principais possam interagir de forma um tanto realista com os figurantes, desde que o direcionamento da história não mude mediante esta forma de interação. Para que isso seja possível, trata-se a interação física de maneira normal, porém fica o personagem figurante impedido de alterar qualquer atributo de um personagem principal. Deste modo, por exemplo, um figurante jamais pode, inesperadamente, matar um personagem principal.

Os personagens figurantes podem ser usados para dois fins distintos: servir como auxílio à realização de determinadas ações que os personagens principais devem desempenhar, bem como para tornar o cenário de visualização mais povoado. Eles podem desempenhar diversas profissões. Até o momento da escrita desta tese, os guardiões são os únicos personagens figurantes implementados. Na especificação lógica da história, cada local está associado a um nível de proteção, que é um valor numérico que varia de 0 a 100. Como já mencionado, diversas ações fazem uso destes atributos como pré-condições para a sua execução. No motor gráfico, faz-se uso de figurantes, que representam guardiões, para implementar este atributo. Se a proteção de um local for 50, devem existir 5 guardiões (1 guardião equivale a uma proteção igual a 10). Assim, por exemplo, a operação `Reduce_protection` consiste em ordenar que um guardião, de um local

especificado, vá para sua casa. A operação `Attack`, por sua vez, faz com que o atacante mate dois guardiões. Com esta estratégia, fica visível para o usuário o quão protegido cada local é.

Além dos guardiões, outros tipos de figurantes poderiam ser pensados tais como:

- Vendedores - Podem ser usados para diversos fins. Como exemplos, considerando-se que os personagens tenham drives, os vendedores de alimento podem ser a forma de suprir a fome. Caso um personagem necessite de poderes para derrotar o inimigo, esse poderia ir até um mago para comprar objetos mágicos que o tornassem mais forte; e
- Trabalhadores - Podem ser usados para povoar o cenário. Cada personagem figurante deve ter um local específico no cenário para trabalhar. Desta forma, uns podem ir para as fazendas, enquanto outros, para as fábricas, por exemplo.

Além de poderem desempenhar diversas funções, os personagens figurantes, exceto os guardiões, que estão indiretamente associados à história, também podem exibir comportamentos rotineiros, como ir para o trabalho de manhã e voltar para casa ao final do expediente. Ações entre figurantes, realizando histórias paralelas, também aparenta ser um recurso adicional para tornar a representação gráfica mais realista.

7.3.1 Inicialização dos Personagens

Todos os personagens, principais e figurantes, são especificados em um arquivo, da mesma forma como ocorre com o cenário. O arquivo de personagens deve ser processado após o do cenário, visto que referencia locais que já devem estar definidos dentro do visualizador. Na Figura 7.3 é apresentado um exemplo deste arquivo e, na Tabela 7.1, o significado de cada parâmetro. Para permitir que um mesmo modelo 3D (no formato MD2) possa representar mais de um personagem no cenário, estes são definidos no início do arquivo e então referenciados nas definições dos personagens que se seguem.

```
//      model              texture             scale
model   knight.md2         knight.pcx          15  //actors
model   princess.md2       princess.pcx        15
model   dragon.md2         dragon.pcx          5
model   soldier.md2        soldier.pcx         15
model   skeleton.md2       skeleton.pcx        15

model   knightW.md2        knightW.pcx         15  //weapons
model   soldierW.md2       soldierW.bmp        5
model   skeletonW.md2      skeletonW.bmp       15

//  type name     num  home            work            model         weapon
actor P  brian     1   knight_castle   nil             knight.md2    knightW.md2
actor P  hoel      1   princess_castle nil             knight.md2    knightW.md2
actor P  marian    1   princess_castle nil             princess.md2  nil
actor P  draco     1   dragon_castle   nil             dragon.md2    nil
actor F  soldier  10   princess_castle princess_castle soldier.md2   soldierW.md2
actor F  skeleton 10   dragon_castle   dragon_castle   skeleton.md2  skeletonW.md2
```

Figura 7.3: Arquivo de especificação dos personagens.

7.3.2 Sistema de Troca de Mensagens entre Personagens

Para que um personagem execute uma ação coletiva, este deve enviar uma mensagem ao personagem com o qual deseja interagir. A mensagem é transmitida via uma chamada de método ao personagem alvo. Quando um personagem recebe uma mensagem de início de ação coletiva e a aceita, é criada uma nova ação terminal com os dados passados no corpo da mensagem, que em sua grande maioria, são iguais aos dados presentes na ação do personagem que originou a ação coletiva.

Atualmente, as mensagens têm como único propósito a convocação para uma luta a dois. A ação de luta ocorre em duas circunstâncias: quando o IPG gera uma ação `Fight` explícita, ou de forma indireta, no caso de uma ação do tipo `Attack`. No primeiro caso, o evento já contém os personagens que devem interagir. No segundo, dentro do estado `Attack` da FSM, o personagem atacante deve perguntar ao **Gerenciador de Ações** com qual personagem deve lutar. Este personagem deve ter três características: ser um figurante guardião do local onde o personagem principal deve atacar, estar próximo do personagem atacante e deve estar vivo. Na implementação corrente, a ação `Fight` nunca ocorre entre dois figurantes. Entretanto, este tipo de ação, onde os guardiões de um local atacam guardiões de outro local, já foi simulado, gerando resultados muito interessantes. O grande problema é que não se tem controle sobre o resultado deste tipo de ação, pois ambos os personagens são figurantes e também porque isso afeta a proteção dos dois locais, uma vez que vai haver baixas nos dois lados. Para representações mais realistas deste tipo de confronto, podem-se usar estratégias de simulação de confrontos [127].

Além de mensagens trocadas, também existe um mecanismo

Tabela 7.1: Descrição dos parâmetros do arquivo de personagens

Label	Descrição
Type	Tipo do personagem: Principal (P) ou Figurante (F)
Num	Número de personagens com os mesmos atributos. Para os personagens que representam guardiões, o valor deve ser 10, para permitir que o nível de proteção possa ser ajustado posteriormente conforme a especificação da base Prolog.
Home	Localização (objeto do cenário) onde o personagem reside. Na inicialização do motor, todos os personagens estão dentro de suas casas.
Work	Local do cenário onde o personagem trabalha. Somente é usado para Figurantes. Quando a visualização tem início, todos os figurantes recebem a ordem para ir trabalhar, o que faz com que os respectivos locais tenham o nível de proteção correto.
Model	Modelo MD2 usado para representar o corpo do personagem.
Weapon	Modelo MD2 usado para representar a arma do personagem.

implementado para alteração do estado dos personagens, que podem assumir três valores: `DEAD`, `FREE` e `LOCKED`. O estado `DEAD` representa que o personagem não pode mais realizar nenhuma ação. O estado `LOCKED` significa que o personagem foi raptado e que somente poderá executar novas ações quando algum outro personagem, no caso o herói da história, fizer sua libertação. Por default, todos os personagens estão em estado `FREE`, o que os permite realizar qualquer ação.

7.4 Integração dos Módulos de Geração e Visualização

Na Tabelas 7.2 e 7.3 são apresentados os atributos que são associados aos dois tipos de personagens e objetos do cenário, respectivamente. A definição destes atributos, bem como principalmente do módulo onde devem estar localizados, é resultado de um longo estudo e de várias discussões. Como apresentado nestas tabelas, são poucos os atributos presentes em ambos os módulos, o que facilita o sincronismo dos mesmos.

Para garantir que a história gerada pelo IPG seja corretamente representada no motor gráfico, os atributos que são comuns aos dois módulos devem estar sempre sincronizados. Para os atributos de personagens, isso ocorre normalmente à medida que os personagens realizam as ações. O efeito de uma ação realizada no simulador deve ter o mesmo efeito no visualizador.

Tabela 7.2: Atributos de Personagens na Visualização e no IPG ("x" indica atributo incluído)

Atributos	**Visualização**	**IPG**	**Comentários**
Localização	x	x	Na visualização, a localização representa uma posição absoluta (x,y) do personagem, enquanto no Prolog, um local (*place*).
Estado	x	x	Representa se o personagem está vivo.
Modelo 3D	x		Modelos MD2 que representam o corpo e utensílios do personagem.
Velocidade	x		Velocidade que o personagem se desloca no cenário.
Afeição		x	Parâmetro usado apenas na simulação da história, como pré-condição para o casamento.
Natureza (índole)		x	Parâmetro usado apenas na simulação da história, para impedir luta entre personagens com mesma índole (bem x mal).
Objetivos		x	No visualizador, os personagens são apenas agentes reativos que cumprem ordens.
Drives	x		*Atributo não implementado*
Emoções	x	x	*Atributo não implementado*

O atributo proteção é o único que é sincronizado automaticamente quando a aplicação é inicializada. Assim que os arquivos de definição do cenário e personagens são carregados, o visualizador realiza consultas diretamente ao IPG para saber o nível de proteção de todos os objetos definidos no arquivo. Em função deste valor, guardiões podem ser removidos da representação.

7.5 Câmera Virtual

Todos os personagens ficam continuamente executando ações fornecidas pelo **Gerenciador de Ações** em função do evento corrente.

Tabela 7.3: Atributos de Objetos do cenário na Visualização e no IPG ("x" indica atributo incluído)

Atributo	**Visualização**	**IPG**	**Comentários**
Localização	x		A localização representa a posição absoluta (x, y) do objeto no cenário.
Proteção	x	x	Único atributo comum. No Prolog é representado por um valor numérico, que é mapeado no visualizador em número de guardiões.
Modelo 3D	x		Modelos 3DS e POZ que representam a estrutura física do objeto.
Tipo		x	Possui o mesmo valor da natureza (ou índole) dos personagens que nele residem. É usado para garantir que um personagem não ataque locais dominados por personagens com índole igual à sua.

Quando não existem eventos a serem processados, ações defaults são delegadas, em função do tipo de personagem.

O papel da câmera é ficar continuamente verificando o evento corrente. Por meio deste evento, descobre-se o personagem corrente, que por sua vez, pode informar a ação específica que está desempenhando (topo da pilha - Seção 5.2).

Uma vez detectado o personagem e sua ação corrente, a câmera deve automaticamente se posicionar de modo a capturar a representação em andamento. A posição do personagem é usada como referencial para auxiliar no posicionamento da câmera, cuja localização específica é dada em função da ação. Apenas três ações têm câmera implementada, como mostrado na Tabela 7.4. Todas as demais ações, por fazerem uso de ações primitivas ou serem ações que não têm representação gráfica específica, fazem uso das demais.

A câmera deve procurar ressaltar duas características da cena: o local onde a ação está ocorrendo e os personagens que fazem parte da ação. Tomadas muito próximas (*close ups*) dos personagens não se fazem necessárias visto que estes não implementam emoções. Para melhor situar o usuário do que está ocorrendo na representação, deve-se procurar sempre contextualizar o local onde a ação vai ocorrer, pois como a câmera se baseia

Tabela 7.4: Tipos de câmeras para diferentes ações

Ação	**Implementação**	**Ações que a implementam**
`Walk`	x	
`Reduce_protection`		`Stand`
`Attack`		`Walk, Fight`
`Fight`	x	
`Kidnap`		`Walk`
`Free`		`Walk`
`Work`		`Walk, Stand`
`Kill`		`Fight`
`Marry`		`Fight`
`Stand`	x	
`Get_stronger`		`Stand`

em tomadas, com certa freqüência ocorrem cortes entre tomadas sucessivas, principalmente quando há mudança do personagem-foco.

Durante o desenvolvimento desta tese, o gerenciamento de cortes bruscos foi um dos maiores problemas enfrentados na especificação do módulo de câmera. Para reduzir o efeito negativo de cortes bruscos entre duas tomadas consecutivas, faz-se uso de uma estratégia que usa um repositório (*pool*) de três câmeras. Nesta abordagem, tem-se a câmera corrente (a que está sendo usada) e duas auxiliares que são candidatas ao cargo de câmera corrente. Cada câmera contém especificações próprias de local e orientação. A idéia de se ter várias câmeras é a possibilidade de se poder selecionar a mais adequada a cada momento. Esta escolha é dada por uma função heurística, que leva em consideração a ação antiga, a ação corrente, a posição do personagem, o peso das câmeras e a mudança de personagem-foco.

7.5.1 Criação das Câmeras

Sempre que ocorre troca de personagem ou ação, as duas câmeras que não estão sendo utilizadas são reconfiguradas para a nova situação. A mudança de ação nem sempre significa troca de personagem, porém o contrário é sempre verdadeiro. A determinação da nova posição, além da posição do personagem e ação corrente, envolve também possíveis objetos

localizados na vizinhança, uma vez que a câmera não pode ficar localizada dentro de um objeto. Exceto quando os personagens estão se locomovendo entre dois locais, eles sempre estão em um local específico. Desta forma, podem-se usar informações de *waypoints* dos objetos como forma de auxílio ao posicionamento da câmera. As três ações básicas de câmera são definidas a seguir:

Ação Walk: Esta é a única ação que envolve movimento de personagens no cenário. Quando o personagem recebe esta ação, ele deve planejar o caminho ao destino, como apresentado na Seção 5.2.2. Tendo-se o personagem e sua ação corrente, a câmera consulta a pilha do personagem e faz uma avaliação do caminho planejado. Existem dois casos possíveis, como mostrado na Figura 7.4. Se o caminho não tiver desvios, duas novas posições perpendiculares às posições inicial e final são criadas. Caso haja desvio, a posição final é dada pelo vetor normal aos dois caminhos intermediários. Para este e todos os demais tipos de câmeras, sempre que o personagem-foco se mover, a câmera corrente faz um novo enquadramento por meio de uma rotação em seu eixo vertical. Tanto para esta ação, como para as demais, caso a posição selecionada esteja dentro de um objeto do cenário, esta posição é substituída pelo *waypoint* do objeto mais próximo da posição originalmente estabelecida. Neste caso, há uma redução de seu peso, visto que a posição selecionada é apenas uma aproximação da posição originalmente estabelecida.

Ação Fight: Para a ação `Fight`, as câmeras são posicionadas de modo a capturar ambos os personagens vistos de dois pontos distintos, como mostrado na Figura 7.5. Considera-se um deslocamento tanto na direção da reta que define a linha de ação dos personagens, como na direção perpendicular a esta linha.

Ação Stand: Sempre que um personagem está parado, ele está nas proximidades de um objeto do cenário (área interna ou externa). Deste modo, faz-se uso da estrutura de *waypoints* do objeto onde ele se localiza para posicionar a câmera de modo a contextualizar o personagem e o local onde ele se encontra.

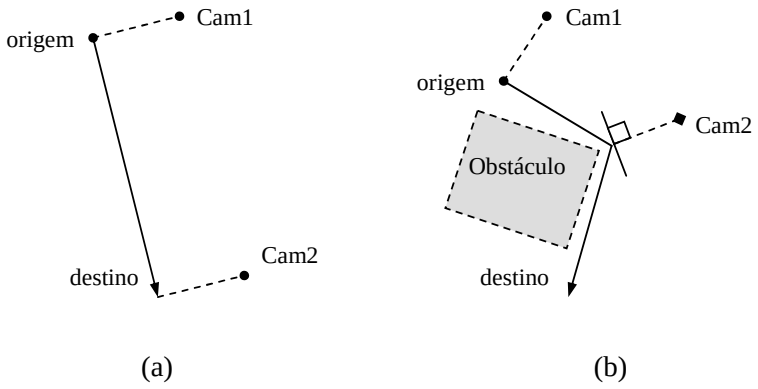

Figura 7.4: Especificação das câmeras para a ação Walk a) sem devio e b) com desvios.

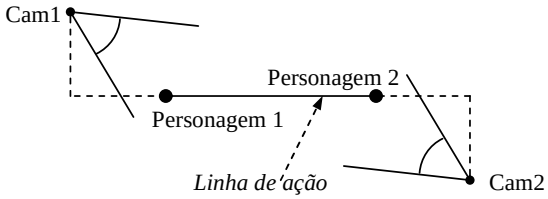

Figura 7.5: Especificação das câmeras para a ação Fight.

7.5.2 Escolha da Câmera Ativa

A cada frame, o módulo da câmera deve selecionar, entre o elenco de câmeras disponíveis, a mais adequada. Faz-se uso de funções heurísticas simplificadas para fazer a escolha das câmeras. Existem três situações claras onde pode ocorrer a troca de câmeras: quando há mudança do personagem-foco, quando há mudança da ação corrente e quando o personagem se desloca no cenário. O caso mais simples de ser tratado ocorre quando há mudança de personagem-foco, visto que a câmera deve rapidamente capturar a nova cena. Nos outros dois casos, a câmera tem maior liberdade na seleção da tomada. A câmera procura, sempre que possível, amenizar cortes bruscos quando novas situações se fazem presentes, segundo regras de cinematografia [69]. A principal regra implementada diz respeito à contextualização da nova cena, ou seja, como situar o observador do novo evento a ser apresentado.

A função heurística é selecionada em função da ação atual que está sendo desempenhada. São usadas várias regras para a seleção. Estas regras levam em conta também o peso associado às câmeras quando são criadas:

– Se o personagem mudar, a câmera deve mudar imediatamente, mesmo que haja um corte brusco e que a nova cena não seja contextualizada adequadamente;

- Se não existe um evento a ser processado, a câmera deve permanecer exibindo a última cena, caso exista;
- Em uma situação de deslocamento, caso o personagem esteja muito afastado da câmera corrente, ou se não esteja mais visível devido a oclusões de objetos do cenário, a câmera associada ao destino final/intermediário do caminho do personagem pode tornar-se ativa;
- Caso duas câmeras tenham pesos semelhantes, a preferência é sempre da câmera corrente, de modo a evitar cortes desnecessários;
- Em caso de mudança somente de ação, ocorre uma redução gradual do peso da câmera corrente com o passar do tempo, de modo que após a contextualização da nova ação pela câmera corrente, câmeras mais adequadas possam ser usadas.

7.6 Ciclo de Renderização das Cenas

O processo de visualização das cenas segue o mesmo paradigma de outras aplicações gráficas em tempo real, como no caso dos jogos. A cada laço de renderização, toda cena é processada pelo motor gráfico e, na seqüência, enviada à placa gráfica para ser desenhada na tela. O processamento do motor é dividido em várias etapas, como mostrado no seguinte algoritmo:

1. Para cada personagem:
   - Verificar término da ação corrente. Em caso afirmativo, verificar se o término da ação implica no término do evento corrente. Se isso ocorrer, solicitar ao **Gerenciador de Ações** um novo evento a ser processado. Na seqüência, delegar uma nova ação ao personagem em função do evento corrente. Se o evento corrente for nulo, associar ação default (`Stand`);
   - Invocar a FSM (*Finite State Machine*) do personagem para processar um passo de sua ação corrente. Neste processo, ações intermediárias podem ser geradas, bem como finalizadas.
2. Configurar a câmera em função da ação em execução pelo personagem vinculado ao evento corrente:
   - Se houve alteração de personagem ou ação corrente, criar duas novas câmeras;

- Chamar a função heurística para selecionar a câmera mais adequada para a situação corrente.

3. Renderizar cada elemento da cena, ou seja, objetos, terreno e céu.

4. Para cada personagem:

   - Renderizar seu modelo segundo estado atual de animação do modelo 3D.

O mapeamento dos eventos para as ações é sempre de um para um, ou seja, cada evento é convertido em uma ação terminal, que deve ser colocada na base da pilha de ações do personagem (Seção 5.2).

O corpo da ação deve incluir obrigatoriamente o seu tipo e o identificador do evento que lhe deu origem (`eventID`). Toda vez que um personagem requisita uma nova ação ao Gerenciador de Ações, esse verifica qual foi o `eventID` associado à última ação do personagem. Caso o `eventID` seja igual ao do evento corrente, significa que o evento corrente foi finalizado e que o Gerenciador de Ações deve requisitar ao Gerenciador de Enredos um novo evento. O corpo da ação também pode conter, dependendo da ação em execução, localizações estáticas (objetos), dinâmicas (referências a outros personagens - para ações coletivas) e também tempo de duração. Este mesmo elenco de parâmetros é usado em ações temporárias criadas pelos próprios personagens.

O tempo de duração de algumas ações é pré-definido (aleatório ou fixo). Por exemplo, a ação `Stand` pode assumir valores quaisquer. Já, para a ação `Fight`, o tempo é sempre fixo. Entretanto, para ações baseadas em espaço, como no caso de `Walk`, esta regra não pode ser aplicada. O tempo de duração vai depender da distância a ser percorrida e da velocidade com que o personagem se locomove. Ações de `Walk` são finalizadas quando o personagem atinge o destino. Ações `Attack` finalizam quando o atacante derrota um número estipulado de guardiões.

7.6.1 Controle temporal

A câmera virtual deve sempre buscar eventos que sejam mais importantes a cada momento, caso contrário, o usuário pode perder o interesse pela cena que está sendo apresentada. Para que esta regra se aplique, devem existir obviamente diversas ações ocorrendo em paralelo. No presente trabalho, existe no máximo uma única ação principal ocorrendo,

visto que cada evento da história é processado sequencialmente. Os personagens figurantes, quando não estão interagindo com um principal, executam somente ações de patrulha, que consistem em se locomover de um lugar para outro no local a ser guarnecido.

Cenas onde o personagem principal deve se deslocar até um determinado local para realizar uma ação específica podem se tornar monótonas caso durem por um longo período de tempo. Neste aspecto, uma questão que foi levantada ao longo desta tese, era se todas as cenas devem ser continuamente processadas ou se partes, menos interessantes, podem ser suprimidas da representação gráfica. Neste caso, a ação tem início em um local, ocorre um corte e na seqüência, a ação volta a ser exibida no local de sua finalização. Para tratar cortes de ações, mudanças significativas devem ser incorporadas principalmente à estrutura dos personagens. O maior problema refere-se à estimativa de onde o personagem deve ser reposicionado após o corte, para dar continuidade à ação. Para a câmera, basta seguir o personagem em sua nova posição. Entretanto, cortes bruscos podem tornar confuso o entendimento pelo usuário da ação que voltou a ser exibida, agora em outro local e sob outros parâmetros de visualização.

Devido a estes problemas, o presente trabalho considera a execução de cada ação de maneira contínua. Para contornar possíveis problemas desta abordagem, define-se um cenário com tamanho adequado para que ações, mesmo envolvendo o percorrimento de longos trajetos, sejam completadas em um tempo relativamente pequeno.

Já pensando na incorporação futura de drives e emoções ao modelo, bem como em interações sociais com outros personagens, a simulação contínua de cada cena permite avaliar, com mais rigor, a variação destes atributos. Entretanto, estas questões de simulações contínuas vs comprimidas devem ser alvo de trabalhos futuros, onde a representação e a manipulação da variável tempo devem ser rigorosamente pesquisadas.

7.7 Interface com o Usuário

A idéia geral do sistema é prover meios eficientes ao usuário, não apenas para guiar a ordem dos eventos a serem visualizados, mas também, e principalmente, para explorar alternativas coerentes que um dado contexto de histórias permite.

Toda a interação é realizada com o Gerenciador de Enredos, que por sua vez, interage com o IPG, para controlar a geração da história, bem como

com o motor gráfico, para controlar a visualização.

A intervenção do usuário pode ser forte ou fraca. Em uma intervenção fraca, o direcionamento da história fica mais a cargo do IPG, pois o usuário se limita a escolher alternativas de enredos parcialmente gerados pelo IPG, que sob sua visão parecem ser interessantes. Em uma intervenção forte, o usuário pode especificar a ocorrência de eventos e forçar a ocorrência de situações em determinados instantes, o que torna a história mais personalizada.

Nos dois tipos de intervenção, o usuário não tem controle direto sobre a cena, nem sobre os personagens. Além disso, a intervenção do usuário é sempre indireta, no sentido que qualquer intervenção deve ser validada pelo IPG antes de ser incorporada ao plano corrente.

7.7.1 A interface do Gerenciador de Enredos

O Gerenciador de Enredos engloba toda a interface gráfica de interação, por meio da qual o usuário pode participar na escolha dos eventos que fazem parte da história, bem como na ordem que devem ser visualizados (Figura 7.6). Em termos gerais, o papel do Gerenciador de Enredos é também servir como um canal de comunicação e conversão de dados entre o IPG e o Visualizador de Histórias.

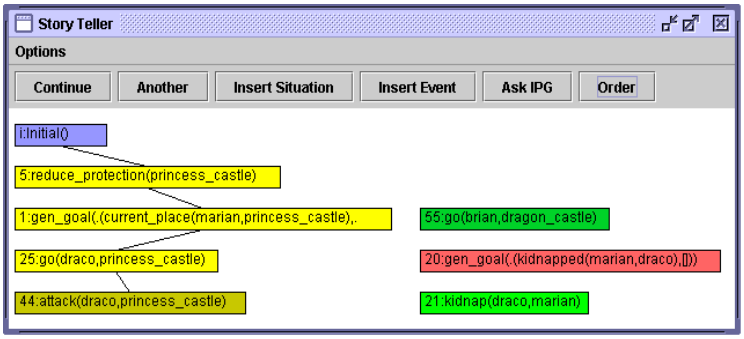

Figura 7.6: Interface de interação com o usuário onde o número à esquerda de cada evento representa o tempo do evento.

Tanto eventos como objetivos são representados por caixas retangulares (ícones) que assumem uma cor específica em função do estado corrente. Esta cor é usada para guiar o usuário no encadeamento de eventos da história. Os ícones podem assumir 7 cores distintas, conforme a Tabela 7.5.

Tabela 7.5: Esquema de cores utilizado para guiar o usuário na manipulação dos eventos

Cor	Significado
Azul	Representa apenas o ponto de partida para encadeamento dos demais eventos.
Amarelo Claro	Eventos que já estão com a ordenação total concluída. São os únicos que podem ser visualizados graficamente.
Amarelo Escuro	Evento corrente de onde o usuário pode conectar outros eventos ativos.
Verde	Representa eventos ativos que podem ser conectadas ao evento corrente, ou seja, eventos que não violam restrições temporais.
Vermelho	Eventos inativos, que não podem ser conectadas ao evento corrente pois têm restrições temporais que devem ser supridas anteriormente (oposto da cor verde).
Ciano	Evento que está sendo representado pelo motor gráfico. Quando a renderização do evento finaliza, a cor do ícone retorna ao estado anterior à renderização.
Verde escuro	Eventos inseridos pelo usuário e que ainda não foram certificados pelo IPG.

A interface de ordenação de eventos define a estrutura mais fundamental por onde o usuário interage com o sistema e é uma etapa que deve ser realizada obrigatoriamente antes de iniciar a visualização da história. Os ícones são manipuláveis no sentido de que podem ser ligados uns aos outros (definição da ordem total) para formarem encadeamentos lineares de eventos. Este encadeamento linear define o direcionamento da história. Somente pode haver a criação de um novo elo entre o evento corrente e um ativo.

Caso o usuário tente estabelecer um elo entre o evento corrente e um inativo (cor vermelha), uma mensagem de erro indicando todas as restrições temporais violadas é exibida.

7.7.2 Interação com o IPG

Quando o usuário deseja a geração de histórias sem que seja exigida interação constante e efetiva, ele pode deixar este trabalho a cargo do IPG. O

IPG disponibiliza dois meios para geração de histórias de forma automática, por meio dos comandos *Continue* e *Another* (Figura 7.6).

O comando *Continue* faz com que o IPG inicialmente infira um novo objetivo e na seqüência, por meio de algoritmos de planejamento, encontre um conjunto de eventos que levem à satisfação deste objetivo. Estes eventos são enviados ao Gerenciador de Enredos para aprovação do usuário. Caso o usuário esteja satisfeito com a solução encontrada para os objetivos inferidos, ele pode solicitar ao IPG a continuação da história (inferência de novos objetivos e subseqüente planejamento), ou definir a ordem total e requisitar a visualização da história até então gerada. Caso o usuário não esteja satisfeito com a história, pode requisitar ao IPG a geração de outra alternativa, pelo comando *Another*. Neste caso, o IPG tenta descobrir outra seqüência de eventos que levem ao cumprimento dos objetivos inferidos. O usuário pode alternar entre as operações *Continue* e *Another*, porém deve-se observar que o *Another* tem efeito somente sobre a última operação *Continue* realizada. Eventos gerados pelo IPG não podem ser removidos pelo usuário.

A todo momento, tanto o IPG, como o Gerenciador de Enredos, guardam a configuração corrente dos eventos e das ordens parciais. Quando a ação *Continue* é executada, caso o usuário não tenha feito nenhuma alteração de ordem total ou inclusão de novos eventos, o IPG é chamado sem nenhum argumento. Neste caso, um novo objetivo é inferido e a história tem prosseguimento. Caso o usuário altere somente a ordem parcial dos eventos, as novas restrições são passadas ao IPG por meio de uma lista. Estas novas restrições são levadas em conta pelo IPG, um novo objetivo é inferido e a história tem sua continuação.

Quando todos os objetivos estiverem satisfeitos, o IPG não pode mais ser usado de forma automática para gerar novos eventos, visto que, sob sua visão, a história já está finalizada. Entretanto, formas mais manuais de interação ainda podem ser usadas. A partir deste momento, o IPG age apenas como uma ferramenta de apoio lógico a autoria.

O IPG suporta dois meios adicionais para intervenção mais efetiva do usuário, de modo a permitir a geração de histórias mais personalizadas. Primeiramente, o comando *Insert Situation* permite que o usuário especifique situações que devem ocorrer em momentos específicos da história, pela inclusão de algum objetivo adicional a ser alcançado. Os detalhes específicos de como o objetivo é alcançado são deixados a cargo do IPG, que é o responsável por encontrar uma solução, se alguma existir, usando algoritmos de planejamento. Deve-se observar que, em termos de

desempenho, um plano válido computável pode não ser alcançado se os limites de busca atualmente configurados no IPG forem excedidos. Os objetivos que o usuário define têm o mesmo efeito que objetivos presentes na estrutura da história criada pelo autor.

No nível mais baixo de interação, por meio do comando *Insert Event*, o usuário pode explicitamente inserir eventos no enredo e adicionar restrições de ordem relativas a eles. Para tornar os novos eventos válidos, o usuário deve chamar o IPG pelo comando *Continue*. Neste momento, todas os eventos definidos pelo usuário, bem como possíveis ordens temporais, são submetidos ao IPG, que executa algoritmos de planejamento para verificar se eles são consistentes com o enredo corrente. Caso não sejam, o IPG pode inserir ainda outros eventos e restrições temporais de modo a garantir a consistência. Neste tipo de interação, o IPG também não faz nenhuma inferência de objetivos. Apenas certifica a validade dos eventos inseridos e as possíveis restrições de ordem também inseridas. Desta forma, o usuário pode criar histórias totalmente desvinculadas dos objetivos presentes na base de dados da história sendo manipulada.

Tanto na inserção de eventos quanto de situações, se o usuário não estiver satisfeito com uma solução apresentada pelo IPG, ele pode solicitar a geração de outras alternativas com o comando *Another*.

Quando um evento ou um objetivo é inserido pelo usuário, um novo ícone é criado dentro da interface. A principal diferença entre ícones definidos pelo usuário e pelo IPG está no fato que os inseridos pelo usuário, em seu estado inicial, não possuem restrição alguma de ordem associada, o que o permite que sejam conectados a qualquer outro ícone. Desta forma, o novo ícone pode ser inserido tanto no início da história, ao final, ou entre dois eventos quaisquer, que inclusive já podem ter uma ordenação total definida. Somente após a certificação do IPG (pela ação *Continue*), é que são definidas as restrições temporais que este novo evento deve assumir. Outra diferença reside no fato que o usuário somente pode remover ícones por ele definido, desde que ainda não incorporados (ou que foram rejeitados) pelo IPG.

Para tornar o processo de definição de situações e eventos mais transparente e fácil, o módulo de Interface disponibiliza dois diálogos por onde o usuário pode visualmente e iterativamente definí-los (Figuras 7.7 e 7.8). Eles são compostos por comboboxes que definem grande parte dos parâmetros que o usuário pode manipular. Assim que o usuário seleciona o primeiro combo, os demais são ajustados para suprir os parâmetros que o evento possui. Muitos objetivos possuem parâmetros numéricos que

devem ser preenchidos pelo usuário. Entretanto, a interface representa tais parâmetros assinalados com o símbolo "XX", como mostrado na Figura 7.8. As situações inseridas pelo usuário podem ser compostas de vários objetivos. Para isso, a interface possui o comando *Add*.

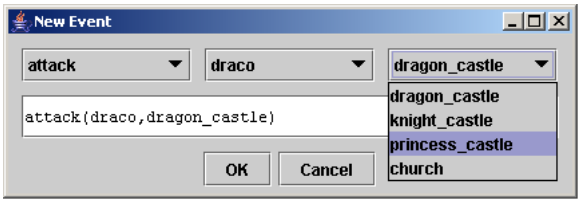

Figura 7.7: Interface para inserção de eventos.

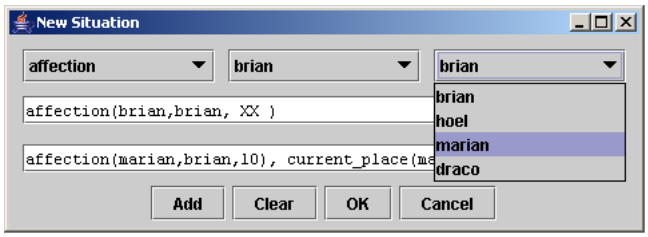

Figura 7.8: Interface para inserção de situações.

A lista de parâmetros nos comboboxes é estática. Não foram implementados ainda mecanismos para consultar o IPG, dado o contexto de histórias corrente, para saber quais são os personagens principais e os possíveis eventos e parâmetros. Estas interfaces permitem que o usuário possa também escrever seus próprios eventos e objetivos, uma vez sabendo a sintaxe utilizada pelo IPG.

7.7.3 Interface de Ícones

Clicando-se com o botão direito do mouse sobre cada ícone, é exibido um menu que disponibiliza as ações que podem ser realizadas. Estas ações incluem remoção (*Remove*), consulta (*Ask*) e renderização (*Render*) do evento. Dependendo do estado do ícone, algumas ações podem estar desabilitadas (Figura 7.9).

Ao executar a operação de consulta (*Ask*), que está sempre habilitada, pode-se consultar o IPG sobre o estado de qualquer elemento da narrativa relativo a um tempo específico T_i, por meio da lógica modal temporal do IPG [34] (Figura 7.10). Este recurso é útil para usuários avançados

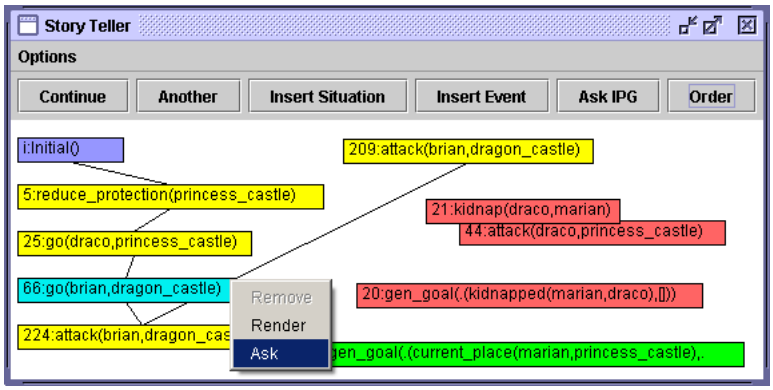

Figura 7.9: Exemplo de menu de ações sobre ícones.

descobrirem, por exemplo, porque um evento ou objetivo não é permitido, bem como para autores que desejam sintonizar os requisitos da história. Esta mesma interface pode ser ativada pelo comando *Ask IPG*, da barra geral de ferramentas. Como todo ícone está associado a um identificador numérico, que é usado como forma de definição das restrições temporais, a consulta realizada sobre os ícones já possui este argumento preenchido. Pode-se consultar por exemplo, o nível de proteção de um local no tempo 66, que para o exemplo da Figura 7.9, corresponde ao instante quando Brian vai para o castelo do Draco. Para isso, a seguinte consulta deve ser feita: `h(t(66), protection(PL,KIND,PROT))`. Neste exemplo, quase todas as variáveis são livres, o que permite, por meio do comando *Another*, desta interface, saber o nível de proteção de todos os locais no tempo 66 (Figura 7.10). A consulta `h(t(T), protection (church,KIND,PROT))` pode ser usada para saber o nível de proteção da igreja em todos os tempos já definidos pelo IPG. Esta interface deve, em trabalhos futuros, apresentar textos em linguagem natural mais amigáveis do que sentenças em sintaxe Prolog.

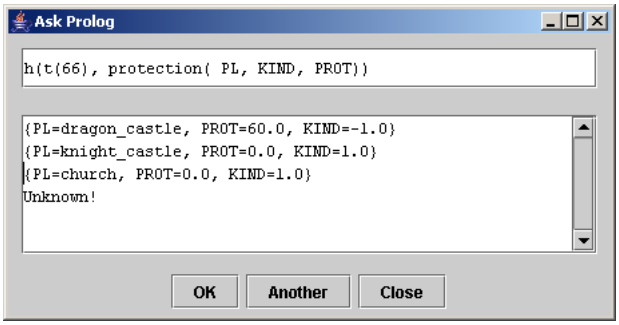

Figura 7.10: Interface para consultar o IPG.

Outra ação disponível no ícone é a renderização. Para que esta opção

esteja disponível, uma série de fatores deve ser verificada. Inicialmente, somente podem ser renderizados ícones que representem eventos. Objetivos não têm este recurso e somente estão presentes na interface como forma de auxílio ao usuário na definição da história. Para que um evento esteja ativo, ele deve ter sido gerado pelo IPG, ou já certificado pelo mesmo, no caso de ter sido definido pelo usuário. A visualização ocorre em tempo real. Atributos variáveis são alterados à medida que os eventos são dramatizados. De modo a manter as representações lógicas e gráficas compatíveis (esquema de sincronização apresentado na Seção 7.4), os valores das variáveis, antes do início da visualização de um evento, devem ser compatíveis com as pré-condições da operação, bem como os valores finais com as pós-condições.

A dramatização tem início pela seleção de um evento específico. Todos os eventos subseqüentes encadeados a partir do ponto inicial são visualizados de modo contínuo, exceto se o usuário interromper o processo. Quando um evento é selecionado para ser visualizado, o motor gráfico usa os valores correntes dos atributos pertinentes como ponto de partida para a representação. Para garantir que a cena se mantenha logicamente compatível com o enredo, não é permitido ao usuário visualizar eventos a partir de pontos aleatórios da história, que não tenham sido previamente renderizados. Um evento somente pode ser renderizado se o seu precedente já foi finalizado, exceto para o evento conectado ao nó raiz. Entretanto, se o usuário deseja visualizar eventos já renderizados anteriormente, pode-se usar qualquer evento como ponto de partida. Para que isso seja possível, quando cada evento tem sua visualização iniciada, o motor gráfico armazena, em uma base local, todos os atributos de todos os personagens e objetos da cena, tendo como chave de acesso o tempo associado a cada evento. Caso o usuário deseje visualizar novamente um evento em ordem aleatória, os dados associados a este evento são usados para atualizar o estado corrente do motor, garantindo assim a mesma representação originalmente realizada.

Como o usuário pode alternar entre simulação e visualização, novos eventos e restrições temporais podem ser inseridos tanto pelo usuário como pelo IPG. Se a visualização for reativada, ela pode iniciar apenas em eventos anteriores às modificações.

7.8 Operações de Videotape (VCR)

Uma vez definida uma ordem total e selecionado o evento inicial a ser renderizado, o motor requisita um novo evento ao Gerenciador de

Enredos assim que o corrente é finalizado. Quando não existem mais eventos conectados ao encadeamento, é enviada uma mensagem ao motor para parar a renderização.

Durante a reprodução dos eventos, o usuário dispõe de algumas operações de videotape (VCR) como *pause*, *play* e *fast forward* (FF). Estas ações têm efeito na velocidade do andamento da visualização gráfica.

A velocidade com que a história é exibida depende tanto da duração de cada evento, como do fator de animação (`FA`), que regula a velocidade com que os personagens realizam as animações, pelo controle da passagem do tempo (com efeito em ações baseadas em tempo, como `Wait` e `Fight`) e velocidade de realização das tarefas (ações baseadas em deslocamentos). A Equação 1 mostra o cálculo para determinação da velocidade com que cada personagem realiza as animações. O fator 1.0/fps assegura que, independente da taxa de quadros por segundo (FPS = *frames per second*) atingida pela aplicação, a velocidade de animação permanece constante.

```
animationSpeed = (1.0/fps)*FA   Eq (1)
```

Para uma operação de pause, o valor de `FA` assume valor zero. Isso faz congelar o estado e posição de cada personagem, que podem ser reativados com a operação *Play*, que restaura o valor de `FA` ao seu valor default. Para fazer com que a animação acelere, basta aumentar o valor de `FA`. A interface por onde o usuário controla a animação é motrada na barra de comandos inferior da Figura 7.11.

Figura 7.11: Interface para controle das operações de videotape.

7.9 Resultados e Utilização do Sistema

Para mostrar os resultados obtidos e exemplificar a utilização do sistema, apresentam-se neste capítulo alguns exemplos de histórias geradas

e cenas dramatizadas (visualizadas). Em especial, deseja-se mostrar as diferentes opções de interatividade que o usuário dispõe e os recursos que a ferramenta disponibiliza para tornar os processos de geração das histórias e visualização intuitivos e com alto poder de expressão.

O nível de diversidade e personalização das histórias geradas tem como principal elemento o nível de interação imposto pelo usuário. Como já apresentado, o sistema permite que o usuário intervenha na geração da história de maneira forte ou fraca, adequando assim as expectativas e interesses de diferentes tipos de usuários.

Para a geração das histórias, foi definida a seguinte situação inicial:

- Marian é uma princesa (no caso uma vítima em potencial), que vive em seu castelo (princess_castle);
- Draco é o único malfeitor (vilão) da história. Ele reside em seu castelo (dragon_castle);
- Brian e Hoel são os cavaleiros (heróis). Ambos residem no mesmo local (knight_castle);
- Cada local da história tem um nível de proteção (número de guardiões);
- Cada personagem tem um nível específico de força para lutar;
- Os heróis têm uma alta afeição pela princesa;
- A princesa não tem nenhuma afeição especial pelos heróis.

Partindo-se dessa situação e usando apenas o comando *Continue*, i.e. com a intervenção mais fraca possível, o processo de geração fica apenas a cargo do IPG, o que leva à geração da seguinte história:

"*A proteção do castelo da Marian é reduzida. Draco vê isso como uma boa oportunidade para raptá-la. Draco então vai para o castelo da Marian, o ataca e após rapta Marian. Como nobre cavaleiro, Brian sente-se obrigado a salvá-la. Ele vai até o castelo do Draco, ataca-o e então luta com Draco. Finalmente, Brian mata Draco e liberta Marian, que neste momento se apaixona por Brian. Motivados pela mútua afeição, Brian e Marian vão para a igreja para se casarem.*"

Esta história é mostrada na Figura 7.12, por meio do **Gerenciador de Enredos**.

Esta mesma história também pode assumir várias alternativas, com baixa interação do usuário, pelo uso dos comandos *Another* e *Continue*

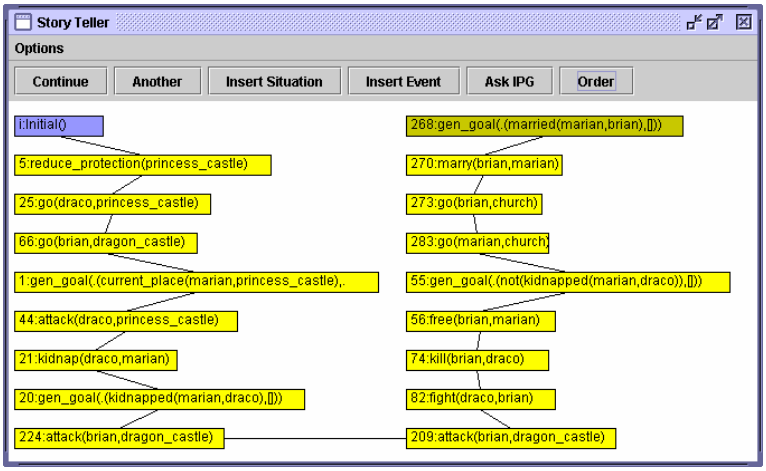

Figura 7.12: História gerada sem interferência do usuário.

combinados. O sistema, por exemplo, gera automaticamente as várias alternativas de fragilização de Marian (tais como ataques do dragão, ida a um local desprotegido) e de salvamento de Marian, onde os dois heróis cooperam de alguma forma. Na figura 7.13 é mostrada uma história onde os dois heróis deslocam-se para o castelo de Draco. Brian ataca o castelo, derrota o dragão e o mata, mas Hoel é quem liberta a princesa e casa-se com ela. A história assume este desfecho pois um dos efeitos do evento `Free` é fazer com que a afeição da vítima pelo seu libertador aumente. Esta história apresenta o exemplo clássico do falso herói.

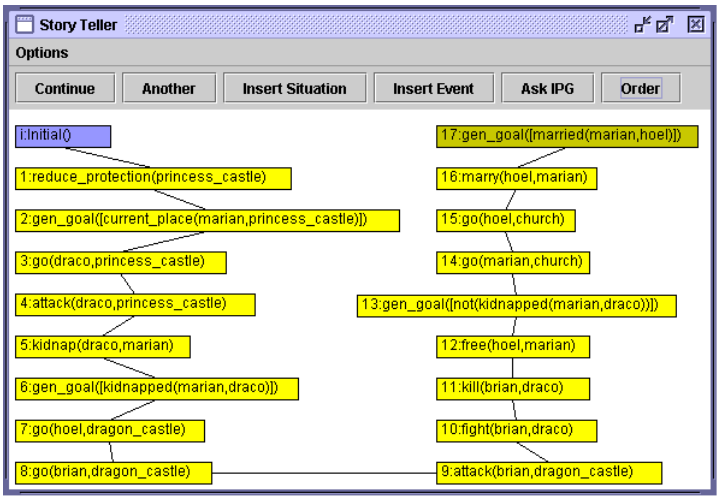

Figura 7.13: Interação pelo uso do comando *Another*.

Caso o usuário deseje histórias mais personalizadas, ele pode intervir no processo com interações fortes, pela inserção de eventos e/ou situações. Podem ser obtidas, dessa forma, histórias completamente personalizadas.

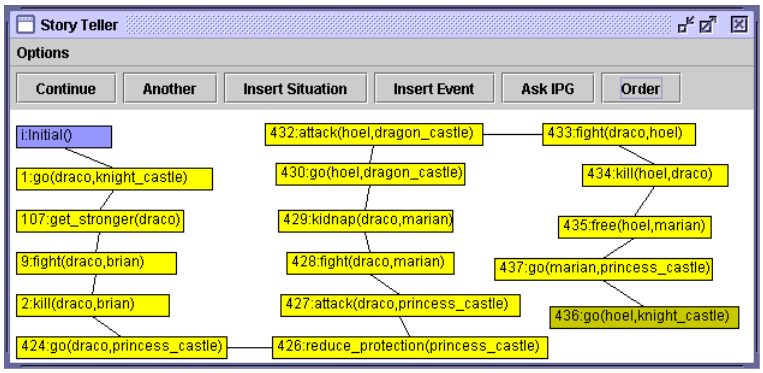

Figura 7.14: Exemplo de história com forte intervenção do usuário.

Na Figura 7.14 é apresentado um exemplo que retrata a seguinte história obtida com interações fortes:

"*Draco deseja raptar a princesa. Sabendo Draco que existe um herói chamado Brian, ele vai até o castelo de Brian, recebe poderes para tornar-se mais forte que Brian e em uma luta, mata Brian. Achando que não existem mais heróis, vai até o castelo da Marian para raptá-la. Chegando lá, tem que enfrentar as defesas do castelo e ao final, ainda sofre um forte ataque da princesa, que luta para não ser raptada. Sendo mais forte que a princesa, Draco consegue raptá-la. Entretanto, um outro herói oculto, Hoel, vai até o castelo do Draco, reduz sua proteção e o vence em uma luta. Após libertar a princesa, Hoel decide não casar-se com ela pois sabe que o verdadeiro amor da Marian era por Brian*".

Se modificada a situação inicial, podem ser geradas inúmeras variações de histórias, todas elas obedecendo à lógica definida no contexto.

Nos exemplos apresentados até este momento, a interação com a história ocorreu somente a nível de geração, pela utilização integrada do IPG com o **Gerenciador de Enredos**. Uma vez que os eventos da história estejam devidamente ordenados, o usuário pode ativar o motor gráfico para que seja realizada a dramatização destes eventos.

Para melhor contextualizar os eventos apresentados ao usuário, é escolhido um cenário que retrata histórias medievais. A estrutura geral do cenário onde os eventos são processados é apresentada na Figura 7.15.

Referente à Figura 7.15(a), o castelo da princesa é o maior e mais próximo, o dos cavaleiros é o mais à direita e o do Draco é o mais ao fundo (cor avermelhada). O usuário pode personalizar este cenário alterando o arquivo de definição do cenário, como mostrado na Seção 7.2. Para a captura destas duas figuras, fez-se uso da câmera em modo manual (controlada

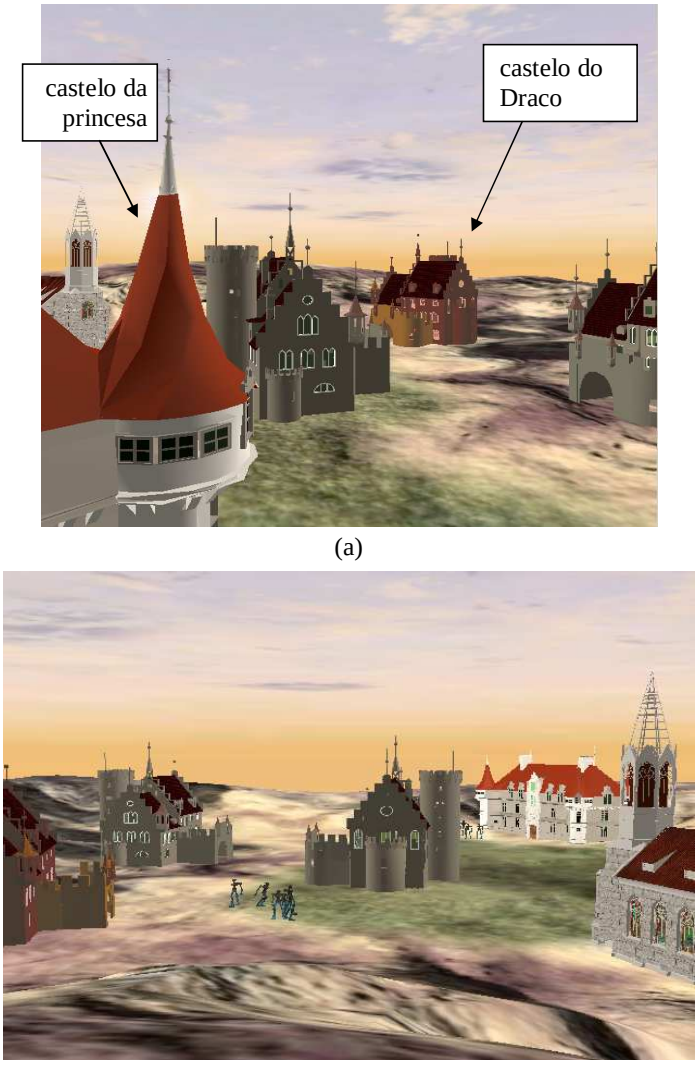

Figura 7.15: Cenário sob os pontos de vista (a) do castelo da princesa e (b) da igreja.

pelo usuário), modo este que, aliás, confere um pouco mais de caráter de videogame à dramatização.

Na Figura 7.16 são apresentados, respectivamente, os ataques de Draco sobre o castelo da princesa e de Hoel sobre o castelo de Draco. Na Figura 7.17 apresenta-se a cena onde Brian vai até a igreja para casar-se com a Marian.

No protótipo, por questões de tempo de modelagem, estão definidas animações de personagens menos elaborados do que os das Figuras 7.16 e 7.17 (com exceção do dragão). Isto, entretanto, não compromete os resultados. No repositório de documentos e aplicações do ICAD/IGames/VisionLab [74] são constantemente introduzidos melhores modelos e animações, bem como atualizações e correções do sistema.

Figura 7.16: Luta do Draco com os soldados do castelo da Marian e de Hoel com os soldados do castelo do Draco.

Figura 7.17: Casamento de Brian e Marian.

7.10 Conclusões e observações

Este capítulo provê detalhes de implementação referentes a diversos aspectos do sistema. No que se refere ao modelo gráfico, sem sombra de dúvidas, a escolha do modelo 3D trouxe benefícios em relação à qualidade do conteúdo exibido ao usuário, como observado nos resultados obtidos. Apesar de exigir maior esforço de programação, o paradigma 3D permite criar configurações de câmera mais variadas. O mesmo vale para a definição do conjunto de ações que os personagens podem desempenhar.

Os mesmos comentários podem ser aplicados à escolha do motor de visualização. Várias dificuldades foram enfrentadas, principalmente

referente a modelos 3D de personagens e objetos. Possivelmente, motores já desenvolvidos poderiam amenizar alguns dos problemas enfrentados relativo a este aspecto. Entretanto, vários outros, inerentes à estrutura de implementação do motor e às limitações no nível de personalização de rotinas e funcionalidades, poderiam ser de solução muito mais difícil. Pela decisão adotada, tem-se agora um sistema configurável e adaptável a outros tipos de histórias, bem como maior disponibilidade de ser exportado para outras plataformas de hardware, como requer a TV interativa.

A estrutura de especificação do cenário e personagens permite que o usuário do sistema possa configurá-los de maneira bastante dinâmica. Pode-se fazer alteração dos modelos 3D, sua localização e quantidade. O usuário pode também mudar o terreno e também inserir novos objetos e personagens.

Quanto à interface gráfica, pode-se observar que oferece ao usuário uma gama grande de alternativas de interação. A interação pode ser fraca ou forte, dependendo do nível de esforço que o usuário deseja empregar, bem como do tipo de história que espera obter. Por meio da interface, pode-se também controlar a visualização gráfica que, para esta implementação, é um processo separado da simulação da história.

O único módulo que não apresenta os resultados esperados é o da câmera. Apesar de incorporar recursos para tratar enquadramento e cortes de tomadas, observa-se que em situações onde ocorre a mudança de personagem, a estratégia implementada não permite que possa ser garantida uma boa contextualização da nova tomada antes que a ação tenha início, visto que a câmera não sabe de antemão quais são as próximas ações a serem realizadas. Isso resulta do fato da câmera não ser pro-ativa, a ponto de seguir as ações e não determinar quando elas devem iniciar. O problema da câmera requer uma revisão de paradigma e uma abordagem possível é discutida no Capítulo 8, na seção de trabalhos futuros.

8
Conclusões

Este capítulo apresenta as considerações gerais sobre a presente pesquisa, as contribuições mais relevantes e os trabalhos futuros mais recomendados.

8.1
Considerações Gerais

O presente trabalho apresenta a especificação e a implementação de um sistema integrado que permite a geração, interação e visualização de histórias interativas. O sistema é composto por três módulos: o IPG [33] que cuida da geração da história, o módulo de interação com o usuário (Gerenciador de Enredos) e o módulo de visualização 3D (motor gráfico).

A definição destes módulos, ou mais especificamente da arquitetura como um todo, é fruto de uma extensa pesquisa de aspectos relativos ao contexto dos enredos, operações a serem implementadas, interfaces de manipulação das histórias, estrutura interna da geometria dos modelos 3D, definição do ambiente de simulação gráfica a ser adotado e níveis de interatividade de que o usuário dispõe. Durante o desenvolvimento da presente tese, foram implementados protótipos iniciais, que mostraram detalhes mais específicos dos requisitos necessários para levar ao usuário conteúdo interativo por meio de informação gráfica animada.

Um dos maiores problemas enfrentados diz respeito às formas de integração entre módulos tão distintos que formam a estrutura do sistema. Em especial, a integração (em termos práticos de implementação) entre o Java e o Prolog, foi a que apresentou os maiores desafios. Graças à API Java do Sicstus Prolog e ao dinamismo do IPG, puderam-se criar mecanismos robustos para assegurar interatividade e, principalmente, coerência das histórias geradas. Os recursos implementados permitem que o usuário possa direcionar de várias formas o rumo da história, em vários níveis de interatividade. Neste aspecto, o Gerenciador de Enredos tem um papel

muito importante, pois torna mais clara e amigável a interação do usuário na especificação dos eventos que fazem parte da história, bem como da ordem com que devem ser representados.

O elenco de operações disponibilizadas, apesar de ser um tanto pequeno, mostrou-se muito satisfatório. A incorporação de níveis de proteção aos locais durante a geração dos enredos é um fator que contribui positivamente nesse aspecto. Os locais, assim especificados, trazem mais dinamismo e aumento de possibilidades de realizar-se um mesmo evento. O uso de restrições numéricas também é um outro fator que contribuiu para a dinâmica que histórias interativas requerem.

Quanto à implementação do motor gráfico, verifica-se que a abordagem adotada, apesar de ter sido trabalhosa, apresenta várias vantagens. Uma delas é a customização de todos os processos envolvidos, como gerenciamento de ações (estrutura de pilha), comunicação entre os módulos e especificação da câmera virtual. Em especial, a estrutura de pilha sugerida tem-se mostrado como melhor alternativa para o tratamento de ações complexas, bem como para o futuro tratamento de drives e emoções. Esta estrutura apresenta fácil implementação e se apresenta como uma poderosa solução para expandir as potencialidades das máquinas de estados finitos. Outra vantagem do desenvolvimento do motor 3D, ao invés de utilizar motores prontos, é ele se tornar totalmente desvinculado do sistema operacional. Essa é uma meta a ser alcançada, visto que, desta forma, a ferramenta pode ser facilmente portada para outras plataformas, especialmente as de set-top boxes.

Em termos visuais, este trabalho adota modelos 3D que podem retratar, com certo nível de fidelidade, as ações realizadas pelos personagens. O nível de detalhes de ações não se aproxima do nível de requinte apresentado pelo jogo de simulação The Sims [48], mas é suficiente para o entendimento das ações que estão sendo representadas. Da mesma forma, procura-se também contextualizar os objetos do cenário de modo a ressaltar aspectos específicos do enredo, que, no caso, relatam histórias medievais.

O modelo de gerenciamento e seleção de câmeras mostra-se muito prático para capturar eventos aleatórios onde figuram diferentes personagens executando diferentes ações a cada momento. As funções heurísticas implementadas fazem a seleção da melhor câmera a cada momento, procurando evitar cortes desnecessários. Entretanto, em situações onde ocorre a troca de personagem, cortes bruscos ainda estão presentes, o que não permite contextualizar a nova cena antes de dar início à exibição da ação corrente.

Finalmente, em relação à TV interativa, uma das principais características do sistema é a possibilidade de que tanto o nível de interação, como o conteúdo apresentado, sejam personalizados para cada usuário. Pela arquitetura proposta, exceto para o gerenciamento da base de dados, todos os demais processos utilizam o processamento do set-top box. Isso também evita sobrecarga de rede, um problema de difícil solução para diversos tipos de conteúdos interativos. Durante o desenvolvimento desta tese, todos os testes foram realizados com o uso de um computador pessoal. Não se tem como objetivo fazer avaliação de desempenho do sistema, entretanto, por questões de exemplificação, obtém-se uma taxa média de 60 fps (*frames per second*), em um Pentium 4 de 2.6 GHz, com uma placa de vídeo GeForce FX 5800, com 128 Mb de memória. Não buscou-se fazer testes reais em set-top boxes, visto que isso causaria uma sobrecarga desnecessária de tempo e dinheiro para o propósito desta tese.

Um paradigma que norteia o presente trabalho refere-se ao conceito de que TV interativa pode ser descrita como:

$$TV\ Interativa = TV + Jogo$$

onde jogo é a face interativa do processo. Entretanto, não se trata de incorporar jogos à TV digital. Os conceitos e modelos apresentados nesta tese criam um novo olhar para histórias interativas na TV digital, assim como apontam para um novo estilo de jogo que vai muito além do RPG convencional.

8.2 Contribuições Alcançadas

Direcionamento lógico com representação coerente. Tanto o IPG, a interface com o usuário, bem como o motor gráfico, trabalham conjuntamente e de forma sincronizada para garantir a geração, direcionamento e exibição das histórias de modo coerente. Todos estes recursos estão integrados em um único sistema multiplataforma, que faz uso de diversas tecnologias de software e linguagens de programação e que permite ao usuário brincar tanto de autor como espectador de histórias interativas. O fato dos módulos estarem implementados em diferentes linguagens de programação não implica na existência de mecanismos de comunicação fracos entre eles. Pelo contrário, cada módulo possui uma interface de comunicação comum,

que permite acoplá-los a outras ferramentas ou aplicativos, para prover o tipo de suporte necessário. Na literatura consultada, não foi encontrado nenhum sistema similar que use tanto esquemas fundamentados em lógica, para garantir coerência, bem como recursos gráficos, que permitem a fácil e parametrizada interação do usuário com a história que está sendo manipulada.

Mecanismos com vários níveis de interferência sobre a história.

Pela concepção do sistema, o usuário pode interagir com a geração da história de maneira fraca ou forte. No primeiro caso, faz-se uso dos recursos que o IPG disponibiliza para definir os eventos da história. No segundo caso, o usuário tem maior controle sobre a história, o que permite a geração de conteúdo mais personalizado, pois permite que eventos e objetivos sejam inseridos na história. Neste caso, o IPG age como ferramenta de apoio lógico à autoria, cuja função é validar e, se necessário, criar novos eventos para garantir a coerência. Dos trabalhos correlatos discutidos nesta tese, o que mais se assemelha ao sistema proposto, relativo ao nível parametrizável de interação, é o proposto por Spierling [129], que define uma arquitetura que trata a especificação das histórias em quatro níveis de abstração (Figura 3.2). Entretanto, estes recursos de interação parametrizáveis são oferecidos a nível de autoria da história e têm como principal propósito dar ao autor, e não ao usuário do sistema, recursos que permitam a prototipação da história em camadas, desde a estrutura geral, até detalhes da animação dos personagens.

Eventos e atributos adequados para a representação gráfica.

Como primeiros passos desta tese, foram investigados o contexto e possíveis eventos que as histórias abordariam. Para a especificação dos possíveis eventos, foram usadas, inicialmente, operações diretamente baseadas nas funções do trabalho de Propp para a validação do IPG, quando este foi implementado. Em seguida, procurou-se especificar um elenco de operações mais adequado tanto no que tange à geração de histórias quanto à sua representação através de animações gráficas, onde os personagens são interpretados por atores virtuais 3D. Propõe-se um pequeno elenco de operações, e regras de inferência de objetivos, mas que juntamente com atributos selecionados, formam a base para a criação de histórias diversificadas e que apresentam alto nível de interatividade e uma representação gráfica consistente. Em termos gráficos, deve-se ressaltar, em especial, a estrutura da cena utilizada,

que faz uso de elementos do cenário como forma de contextualizar os eventos que estão sendo apresentados. Todos os eventos ocorrem em um local específico do cenário, geralmente em frente aos objetos, o que facilita a compreensão pelo usuário do local onde o evento está sendo dramatizado. Neste aspecto, o sistema proposto oferece vantagens de personalização dos cenários, personagens e modelos, até então não observados na literatura.

Criação de um modelo comportamental baseado em pilha. O modelo proposto de pilha apresenta enormes potencialidades para aumentar o poder computacional das FSM. É uma abordagem muito superior mesmo se comparada com máquinas hierárquicas, pois torna a implementação mais simples e transparente. Esta solução facilitou a implementação do sistema. O modelo trata ações diversas, bem como comportamentos de manobras. Para tal, foram usados recursos de *path-planning* e tratamento de colisões, implementados por meio de uma estrutura de *waypoints* presentes em todos os objetos distribuídos no cenário. Outro ponto a salientar é a possibilidade de incorporar caminhos ao modelo através da geração progressiva de caminhos, onde possíveis obstáculos são tratados como ações intermediárias do tipo `Walk`, que são manipuladas do mesmo modo que as demais ações. Mostra-se também que o uso de uma pilha com apenas duas posições parece ser uma boa solução para este problema, independente da complexidade das ações em execução; e mesmo quando os agentes incorporam mecanismos de drives e emoções.

Estudo e análise da incorporação de drives e emoções. Apesar de não terem sido implementados, fez-se um estudo aprofundado ao longo desta tese relativo à incorporação de drives e emoções ao modelo, pois apresentam enormes potencialidades para tornar conteúdo e representação das histórias mais realistas e interessantes. Esta tese apresenta e discute pontos intrigantes que surgem quando se consideram causas e efeitos, tanto na geração como na representação das histórias, quando drives e emoções estão presentes. Uma questão, que ainda deve ser estudada em maior profundidade, refere-se à questão da influência que drives e emoções podem ter no direcionamento da história, especialmente quando tratados durante a representação gráfica, visto que, atualmente, o sistema proposto deixa somente a cargo do IPG processos referentes à geração do conteúdo. No Capítulo 5 desta tese, apresenta-se um esquema

conceitual para a incorporação de drives e emoções em nível de representação da história, que neste caso produz efeito apenas sobre ações intermediárias e a representação gráfica, mantendo assim a direção da história previamente simulada. Este estudo é original em se tratando de sistemas de *storytelling*.

Especificação de um modelo de câmera automática. No ambiente de *storytelling* apresentado nesta tese, a qualquer momento podem ocorrer mudanças do personagem-foco, bem como da ação corrente. Deste modo, o modelo de câmera virtual proposto faz uso de um repositório de três câmeras, que são configuradas para selecionar e capturar cada novo evento em exibição. Esta configuração permite que existam alternâncias entre câmeras ativas, segundo funções heurísticas que selecionam, a cada momento, a câmera mais adequada. A câmera também está integrada com a estrutura de *waypoints* do cenário, de modo que possa facilmente se posicionar de modo a capturar o evento em exibição. Na Seção 8.3 é apresentada a proposta de uma pequena mudança de paradigma, que dá à câmera maior controle sobre as ações processadas pelos atores virtuais, possibilitando assim melhor contextualização das cenas, em caso de cortes bruscos, entre duas tomadas consecutivas.

Modelo para incorporação de histórias em TV interativa. Mediante um estudo detalhado das tecnologias de iTV, pôde-se melhor compreender os requisitos necessários para levar conteúdo personalizado com interatividade parametrizável a cada usuário. Como resultado desta pesquisa, o Capítulo 6 apresenta uma arquitetura para a incorporação do sistema de histórias interativas proposto nesta tese em um provável ambiente de iTV. A arquitetura proposta mostra-se adequada para o propósito em questão, pois permite que cada usuário possa manipular o conteúdo individualizado, sem sobrecarregar as linhas de dados e processamento do servidor. Além disso, por ser multiplataforma, pode ser facilmente exportada para diferentes arquiteturas de set-top boxes. Acredita-se também que a incorporação de histórias interativas, em iTV, seja uma proposta inédita, pois não foi encontrado na literatura nenhum sistema que se proponha a tal fim. Além disso, as características que o sistema apresenta, em especial a interatividade parametrizável, se mostra como uma ótima solução para a questão do nível de interatividade

que o usuário/telespectador espera encontrar quanto interagindo com programas interativos.

8.3 Trabalhos Futuros

No decorrer do desenvolvimento desta tese, muitas idéias foram consideradas. Entretanto, dada a necessidade de atender aos aspectos de maior prioridade, nem todas puderam ser implementadas. A seguir, são apresentadas sugestões de pesquisas futuras.

Gerenciador de Enredos: O Gerenciador de Enredos é uma poderosa ferramenta de interação com o usuário, que, da mesma forma como ocorre com o IPG, opera sobre qualquer tipo de evento. Seus mecanismos permitem uma interação mais transparente com o IPG, mesmo para usuários que não têm afinidade com a linguagem e notação utilizadas. Entretanto, atualmente, os mecanismos de inserção de eventos e objetivos são baseados em parâmetros estáticos. Para tornar esta ferramenta mais genérica, deve-se incorporar mecanismos para consultar a base Prolog e, em função dos predicados nela estabelecido, montar as listas de argumentos que cada operação pode disponibilizar. Outro ponto a ser analisado consiste na extensão do Gerenciador de Enredos para permitir a definição do contexto das histórias através de uma interface gráfica intuitiva.

Câmera virtual: Atualmente, a câmera tem a função de descobrir, em tempo real, a ação a exibir e se configurar em função dela. Esta abordagem é a utilizada por Charles [29], como forma de capturar as ações que resultam da interação entre os personagens. Pelo que foi observado, este modelo de câmera tem maior aplicabilidade em sistemas que adotam a abordagem de geração de história baseada em personagem (*character-based*), visto que não se sabe de antemão, os eventos que irão ocorrer. No presente trabalho, faz-se uso de uma abordagem baseada em enredo, ou seja, o enredo comanda o andamento da visualização (*plot-based*). Como discutido no Capítulo 7, algumas tomadas iniciam com cortes bruscos, pois a câmera dá início à captura da nova cena quando ela já está ocorrendo, dificultando assim o entendimento pelo usuário do novo evento.

Para contornar este problema, propõe-se como trabalho futuro uma mudança de paradigma. Em vez das ações serem enviadas diretamente

aos personagens, estas seriam enviadas inicialmente para a câmera, como mostrado na Figura 8.1. Uma vez que a câmera sabe qual ação vai ocorrer e com qual personagem, ela pode inicialmente se posicionar e contextualizar a nova cena, para somente então redirecionar a ação ao personagem específico. Este processo somente deve ser realizado para ações que envolvam personagens principais.

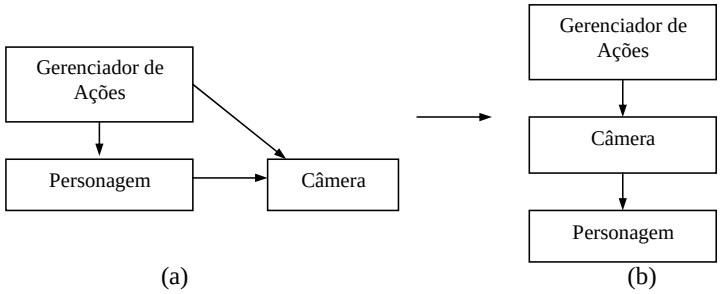

Figura 8.1: a) Modelo atual e b) novo modelo para implementação da câmera virtual.

Os recursos até então implementados na câmera estão muito aquém do necessário para tornar as tomadas tão realistas, como ocorre na dramatização atual em televisão. Para isso, recursos como transições mais suaves, movimentação da câmera durante tomadas, bem como efeitos para retratar situações específicas, como no caso de emoções, devem ser adicionados ao modelo. Estes problemas requerem a investigação de câmeras inteligentes que passem a comportar-se como diretores virtuais das histórias que estão sendo encenadas.

Variedade de Histórias: Vários aprimoramentos e a inclusão de novos recursos podem ser realizados no intuito de enriquecer as histórias, tanto sob os aspectos de variabilidade, como possibilidade de definição de eventos. A incorporação e o tratamento adequado de drives e emoções [37][16][65] mostram-se como uma ótima alternativa. Em especial, o efeito dos drives pode ser considerado tanto nos processos de simulação como de representação das histórias. Quando tratados a nível de representação, estes drives podem ser utilizados também como forma de direcionar o rumo da história. Nesse caso, mecanismos mais avançados de interação com o IPG fazem-se necessários. Esse esquema de geração do enredo se assemelha muito a uma abordagem baseada em personagens (*character-based*), onde a história emerge unicamente da interação entre personagens.

Alterações na ferramenta em mais alto nível incluem modificações nos algoritmos de planejamento do IPG tanto para a melhoria de desempenho quanto para obter um maior acoplamento com a visualização gráfica. Um exemplo disso é a possibilidade de tratamento dos eventos, já durante a fase de geração, como atividades com uma certa duração. Ao se aproximar os modelos lógicos e gráficos, é possível que o esforço para a obtenção de uma grande variedade de histórias seja reduzido.

Aspectos Audiovisuais: No que se refere a aspectos audiovisuais, propõe-se especialmente a incorporação de técnicas para a representação de expressões faciais, resultante das emoções que os personagens venham a exibir. Adicionalmente, os personagens podem também se expressar por meio de áudio, fazendo uso de um sistema integrado de animação facial com voz [92].

Para que os personagens possam expressar verbalmente ações ou diálogos, necessita-se inicialmente desenvolver ferramentas para a geração de textos que descrevam os encadeamentos de eventos que definem a história. A geração de textos para enredos gerados pelo IPG já vem sendo estudada [56], mas os textos gerados até o momento têm um caráter muito repetitivo. Pesquisas envolvendo linguagem natural podem eventualmente levar a textos de melhor qualidade.

Outro recurso que pode tornar as representações mais realistas é a geração de histórias paralelas entre personagens figurantes. Atualmente, os personagens figurantes limitam-se a guarnecer locais contra ataque de inimigos.

Nível de Interatividade: Está se investigando a possibilidade de implementação de dois novos recursos para tratar níveis mais avançados de interação: participação do usuário como personagem em uma história interativa e interação de vários usuários em uma única história interativa. No primeiro caso, o usuário assume o papel de um personagem (que atualmente é desempenhado por um agente), que deve desempenhar ações seqüenciais fornecidas pelo Gerenciador de Enredos. Já existem trabalhos na literatura que sugerem uma abordagem complementar: incorporação de histórias interativas em jogos [8][62].

Para o caso do usuário participar de sua própria história, algumas questões devem ser tratadas. Inicialmente, deve-se definir qual papel

o usuário vai assumir. Deve-se permitir que o usuário possa escolher, dentro do elenco de personagens, qual deseja vivenciar. Caso o usuário assuma um papel secundário, o fluxo da história pode ser mais facilmente garantido, visto que ações por ele realizadas não terão influência direta sobre os personagens principais. Caso o usuário escolha um personagem principal, deve existir algum mecanismo para garantir que as operações sejam executadas na ordem estipulada. Esta abordagem dá ao usuário maior liberdade de participação na história, cujo nível de interação pode ser semelhante à de jogos 3D em primeira pessoa.

Durante o desenrolar da história, o personagem controlado pelo usuário recebe as ações que deve realizar, estando a execução das mesmas sob responsabilidade do usuário. Para tornar a interação mais desafiadora, ações específicas como descobrir o local onde a ação deve ser executada, deslocar-se, lutar e procurar pelo inimigo, podem ser realizadas, em sua total funcionalidade, pelo usuário. O confronto entre a IA dos personagens autônomos com a esperteza do jogador pode dar à história um grau a mais de realismo. Esquemas de alternância entre a câmera em primeira pessoa com a câmera automática também devem ser pesquisados para esta situação.

Para o segundo caso, onde vários usuários interagem na mesma história, além dos problemas anteriormente mencionados, outros devem ser tratados. O principal se refere à estrutura cliente/servidor, proposta no Capítulo 6, para um ambiente de iTV. A representação da estrutura proposta é muito adequada quando existem vários usuários conectados ao servidor, interagindo individualmente com suas histórias. Entretanto, caso a mesma história seja compartilhada por mais de um usuário, aspectos de centralização dos processos de geração e controle da história devem ser devidamente cobertos.

Tratamento de colisão entre atores: Atualmente, o modelo de path-planning adotado para controle de movimentação dos personagens trata colisão apenas com objetos estáticos. Para uma representação mais realista, especialmente quanto existem muitos personagens próximos, deve-se incorporar ao modelo esquemas para fazer o tratamento de colisão entre personagens. Para isso, pode-se fazer uso de técnicas usadas em problemas que envolvem simulação de multidões, bem como algoritmos e estratégias utilizados em jogos de computador.

Bibliografia

[1] **FIPA ACL message structure specification**. Disponível em: <http://www.fipa.org/specs/fipa00061/>. Acesso em: 21 jan. 2005.

[2] AMANDI, A. A.. **Programação de agentes orientada a objetos**. PhD thesis, Porto Alegre: CPGCC da UFRGS, 1997.

[3] D'AMICO, C. B.. **Inteligência artificial: uma abordagem de agentes**. Technical report, Porto Alegre: CPGCC da UFRGS, 1995.

[4] ARAUJO, M. P.; CELES, W.. **Uma estratégia para a visualização do céu e seus elementos em jogos**. In: PROCEEDINGS OF THE 1ST BRAZILIAN WORKSHOP IN GAMES AND DIGITAL ENTERTAINMENT - WJOGOS'02, Fortaleza, October 2002.

[5] ARIJON, D.. **Grammar of the film language**. Focal Press, London, 1976.

[6] **Advanced television system commitee**. Disponível em: <http://www.atsc.org>. Acesso em: 17 nov. 2004.

[7] BAECKER, R. M.. **Readings in Groupware and Computer-Supported Cooperative Work**. Morgan Kaufmann Publishers, Inc, San Mateo, CA, 1993.

[8] BARROS, L. M.; MUSSE, S. R.. **Utilizando técnicas de interactive storytelling para geração dinâmica de enredos para jogos**. In: PROCEEDINGS OF THE BRAZILIAN SYMPOSIUM ON COMPUTER GAMES AND DIGITAL ENTERTAINMENT - WJOGOS'04, Curitiba, 2004.

[9] BATES, J.. **Virtual reality, art and entertainment**. Presence: Teleoperators and Virtual Environments, 1:133–138, 1992.

[10] BATES, J.; LOYALL, A. B. ; REILLY, W. S.. **An architecture for action, emotion, and social behavior**. Technical report CMU-CS-92-142, School of Computer Science, Carnegie-Mellon University, Pittsburg, PA, 1992.

[11] BATES, J.; LOYALL, A. B. ; REILLY, W. S.. **Integrating reactivity, goals, and emotions in a broad agent**. Technical report CMU-CS-92-144, School of Computer Science, Carnegie-Mellon University, Pittsburg, PA, 1992.

[12] BECKER, V.. **Infra-estrutura internet2 para desenvolvimento e teste de programas e ferramentas para TV interativa**. Disponível em: <http://www.i2tv.ufsc.br/conheca.htm>. Acesso em: 12 out. 2004.

[13] BECKER, V.; MORAES, A.. **Do conteúdo televisivo analógico para o digital: uma proposta de comercial para TV interativa**. In: SCPDI '03 - III SIMPóSIO CATARINENSE DE PROCESSAMENTO DIGITAL DE IMAGENS, p. 122–134, Florianópolis, Outubro 2003.

[14] BLINN, J.. **Where I am? what am I looking at?** IEEE Computer Graphics and Applications, 8(4), 1988.

[15] BOND, A. H.; GASSER, L.. **Readings in Distributed Artificial Intelligence**. Morgan Kaufmann Publishers, San Mateo, CA, August 1988.

[16] BREZEAL, C.. **A motivational system for regulating human-robot interaction**. In: AAAI-98: PROCEEDINGS OF THE FIFTEENTH NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE, p. 54–61, Madison, Wisconsin, 1998. American Association for Artificial Intelligence.

[17] BROOKS, R. A.. **Intelligence without reason**. Technical report a.i. memo 1293, Artificial Intelligence Laboratory, MIT, 1991.

[18] BROOKS, R. A.. **A robust layered control system for a mobile robot**. Technical report a. i. memo 864, Artificial Intelligence Laboratory, MIT, 1995.

[19] BURNS, D.; GOSLING, J.; MCGEADY, S. ; SHORT, R.. **Set-top boxes - the next platform (panel)**. In: PROCEEDINGS OF THE 22ND ANNUAL ACM CONFERENCE ON COMPUTER GRAPHICS. ACM SIGGRAPH, p. 479, Los Angeles, CA, September 1995. ACM Press.

[20] **Mheg-5 overview. Cabot Communications Limited**. <http://www.cabot.co.uk/resources/MHEG5overview.html>. Acesso em 25 de jan. de 2005.

[21] CAIN, T.. **Practical optimizations for A* path generation**. In: Rabin, S., editor, AI GAME PROGRAMMING WISDOM, p. 146–152. Charles River Media, Hingham, Massachusetts, 1st edition, 2002.

[22] CALDER, B.; COURTNEY, J.; FOOTE, B.; KYRNITSZKE, L.; RIVAS, D.; SAITO, C.; LOO, J. V. ; YE, T.. **Java TV API technical overview, v1.0**. Technical report, Sun MicroSystems, Palo Alto, CA, November 2000. Disponível em: <http://java.sun.com/products/javatv/jtv-1_0-spec_overview.pdf>.

[23] CAVAZZA, M.; EARNSHAW, R.; MAGNENAT-THALMANN, N. ; THALMANN, D.. **Motion control of virtual humans**. IEEE CG&A, 18(5):24–31, Sept/Oct 1998.

[24] CAVAZZA, M.; CHARLES, F. ; MEAD, S.. **Ai-based animation for interactive storytelling**. In: PROCEEDINGS OF COMPUTER ANIMATION, p. 113–120, Seoul, Korea, 2001. IEEE Computer Society Press.

[25] CAVAZZA, M.; CHARLES, F. ; MEAD, S.. **Character-based interactive storytelling**. IEEE Intelligent Systems, special issue on AI in Interactive Entertainment, 17(4):17–24, July 2002.

[26] CAVAZZA, M.; CHARLES, F. ; MEAD, S.. **Interacting with virtual agents in interactive storytelling**. In: ACM JOINT CONFERENCE ON AUTONOMOUS AGENTS AND MULTI-AGENT SYSTEMS, AAMAS'02, p. 318–325, 2002.

[27] CAVAZZA, M.; CHARLES, F. ; MEAD, S.. **Emergent situations in interactive storytelling**. In: SAC '02: PROCEEDINGS OF THE 2002 ACM SYMPOSIUM ON APPLIED COMPUTING, p. 1080–1085, Madrid, Spain, 2002. ACM Press.

[28] CHARLES, F.; CAVAZZA, M. ; MEAD, S.. **Character-driven story generation in interactive storytelling**. Technical report, VSMM, Berkeley, 2001.

[29] CHARLES, F.; LUGRIN, J.; CAVAZZA, M. ; MEAD, S.. **Real-time camera control for interactive storytelling**. In: GAME-ON, London, UK, 2002.

[30] CHIARIGLIONE, L.. **International organization for standardization - MPEG-2**. Disponível em: <http://www.chiariglione.org/mpeg/standards/mpeg-2/mpeg-2.htm>. Acesso em 25 de jan. de 2005.

[31] **ChorusOS 5.0 features and architecture overview**. Disponível em: <http://docs.sun.com/app/docs/doc/806-6897?q=chorusOS&a=expand>. Acesso em 25 de jan. de 2005.

[32] CHRISTIANSON, D.; ANDERSON, S.; HE, L.; SALESIN, D.; WELD, D. ; COHEN, M.. **Declarative camera control for automatic cinematography**. In: PROCEEDINGS OF THE AAAI-96, Portland, Oregon, August 1996.

[33] CIARLINI, A.. **Geração interativa de enredos**. PhD thesis, Departamento de Informática, PUC-Rio, Rio de Janeiro, 1999.

[34] CIARLINI, A.; VELOSO, P. ; FURTADO, A.. **A formal framework for modelling at the behavioural level**. In: PROC. OF THE 10TH EUROPEAN-JAPANESE CONFERENCE ON INFORMATION MODELLING AND KNOWLEDGE BASES, Saariselkä, Finland, 2000.

[35] CIARLINI, A.; FEIJÓ, B. ; FURTADO, A.. **An integrated tool for modelling, generating and exhibiting narratives**. In: AIS'2002: AI, SIMULATION AND PLANNING IN HIGH AUTONOMY SYSTEMS, p. 150–154, Lisboa, Portugal, April 2002.

[36] CIARLINI, A.; FURTADO, A.. **Understanding and simulating narratives in the context of information systems**. In: PROCEEDINGS OF THE 21ST INTERNATIONAL CONFERENCE ON CONCEPTUAL MODELING - ER'2002, p. 291–306, Tampere, Finland, October 2002.

[37] COSTA, N.; FEIJÓ, B.. **Agents with emotions in behavioral animation**. Computer & Graphics, 20(3):377–384, 1996.

[38] CRAWFORD, C.. **Assumptions underlying the erasmatron interactive storytelling engine**. In: Mateas, M.; Sengers, P., editors, PROCEEDINGS OF THE AAAI FALL SYMPOSIUM: NARRATIVE INTELLIGENCE, TECHNICAL REPORT, p. 112–114, Menlo Park, CA, 1999. AAAI Press.

[39] DAVENPORT, G.; AGAMANOLIS, S.; BARRY, B.; BRADLEY, B. ; BROOKS, K.. **Synergistic storyscapes and constructionist cinematic sharing**. IBM System Journal, 39(3,4):456–469, 2000.

[40] **Deep exploration**. Disponível em: <http://www.righthemisphere.com/products/dexp/>. Acesso em: 11 jan. 2005.

[41] DONATO, G.; BARTLETT, M. S.; HAGER, J. C.; EKMAN, P. ; SEJNOWSKI, T. J.. **Classifying facial actions**. IEEE Transactions on Pattern Analysis and Machine Intelligence, 21(10):974–989, 1999.

[42] DRISCOLL, G.. **The essential guide to digital set-top boxes and interactive TV**. Prentice Hall, Upper Saddle River, NJ, 1st edition, November 2000.

[43] DRUCKER, S.; GALYEAN, T. ; ZELTZER, D.. **Cinema: a system for procedural camera movements**. In: SI3D '92: PROCEEDINGS OF THE 1992 SYMPOSIUM ON INTERACTIVE 3D GRAPHICS, volumen 25, p. 67–70, New York, NY, March 1992. ACM Press.

[44] DRUCKER, S.. **Intelligent camera control for graphical environments**. PhD thesis, MIT Media Lab, 1994.

[45] DRUCKER, S. M.; ZELTZER, D.. **Camdroid: a system for implementing intelligent camera control**. In: SI3D '95: PROCEEDINGS OF THE 1995 SYMPOSIUM ON INTERACTIVE 3D GRAPHICS, volumen 28, p. 139–144. ACM Press, April 1995.

[46] DUCHAINEAU, M.; WOLINSKI, M.; SIGETI, D.; MILLER, M.; ALDRICH, C. ; MINEEV-WEINSTEIN, M.. **ROAMing terrain: Real-time, optimally adapting meshes**. In: PROCEEDINGS OF THE CONFERENCE ON VISUALIZATION '97, p. 81–88, 1997.

[47] **Digital video broadcasting project**. Disponível em: <http://www.dvb.org>. Acesso em: 17 nov. 2004.

[48] **The Sims**. Electronic Arts. Disponível em: <http://thesims.ea.com/us/index.html>. Acesso em: 04 jun. 2004.

[49] EKMAN, P.. **An argument for basic emotions**. Cognition and Emotion, 6:169–200, 1992.

[50] **Devmaster.net**. Disponível em: <http://www.devmaster.net/engines/list.php>. Acesso em: 10 fev. 2005.

[51] **Macromedia flash mx webpage**. Disponível em: <http://www.macromedia.com/software/flash>. Acesso em: 22 abr. 2003.

[52] ULRICH, T.. **Fly3d homepage**. 2003. Disponível em: <http://www.fly3d.com.br/>. Acesso em: 27 jan. 2005.

[53] PETER FORMAN, R. W. S. J.. **The future of digital entertainment/creating convergence**. Scientific American, 283(5):7, November 2000. Special Report.

[54] FU, D.; HOULETTE, R.. **The ultimate guide to FSMs in games**. In: Rabin, S., editor, AI GAME PROGRAMMING WISDOM 2, p. 283–302. Charles River Media, Hingham, Massachusetts, 2nd edition, 2004.

[55] FURHT, B.. **Interactive television systems**. In: PROCEEDINGS OF THE 1996 ACM SYMPOSIUM ON APPLIED COMPUTING, p. 7–11, Philadelphia, Pennsylvania, February 1996. ACM Press.

[56] FURTADO, A.; CIARLINI, A.. **Generating narratives from plots using schema information**. In: NLDB'2000: 5TH INTERNATIONAL CONFERENCE ON APPLICATIONS OF NATURAL LANGUAGE TO INFORMATION SYSTEMS, Versalhes, França, June 2000.

[57] FURTADO, A. L.; CIARLINI, A. E. M.. **Cognitive and affective motivation in conceptual modelling**. Revista Colombiana de Computación, 3(2), 2002. Disponível em: <http://www.unab.edu.co/editorialunab/revistas/rcc/rev32.htm>.

[58] GARLAND, M.; HECKBERT, P. S.. **Surface simplification using quadric error metrics**. In: PROCEEDINGS OF THE 24TH ANNUAL CONFERENCE ON COMPUTER GRAPHICS AND INTERACTIVE TECHNIQUES - SIGGRAPH'97, p. 209–216. ACM Press/Addison-Wesley Publishing Co., August 1997.

[59] **Software engineering for large-scale multi-agent systems, international conference on software engineering**. In: GARCIA, A. F.; LUCENA, C.; CASTRO, J.; OMICINI, A. ; ZAMBONELLI, F., editors, SELMAS 2002, volumen 1 de **1**, p. 653–653, Orlando, 2002.

[60] GIBBS, R. A.; HOCH, M.; GONG, H. ; WANG, S.. **Enabling custom enhancements in digital sports broadcasts**. In: PROCEEDINGS OF THE 2001 CONFERENCE ON 3D TECHNOLOGIES FOR THE WORLD WIDE WEB - WEB3D '01, p. 101–107, Paderbon, Germany, February 2001. ACM Press.

[61] GIRAULT, A.; LEE, B. ; LEE, E.. **Hierarchical finite state machines with multiple concurrency models**. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 18(6):742–760, June 1999.

[62] GLASSNER, A.. **Interactive Storytelling: Techniques for 21st Century Fiction**. AK Peters Ltd, 2004.

[63] GLEICHER, M.; WITKIN, A.. **Through-the-lens camera control**. In: Catmull, E. E., editor, COMPUTER GRAPHICS PROCEEDINGS. ANNUAL CONFERENCE SERIES. ACM SIGGRAPH, volumen 26(2), p. 331–340, July 1992.

[64] GRASBON, D.; BRAUN, N.. **A morphological approach to interactive storytelling**. In: Fleischmann, M.; Strauss, W., editors, PROCEEDINGS: CAST01, LIVING IN MIXED REALITIES. SPECIAL ISSUE OF NETZSPANNUNG.ORG/JOURNAL, THE MAGAZINE FOR MEDIA PRODUCTION AND INTER-MEDIA RESEARCH, p. 337–340, Sankt Augustin, Germany, 2001.

[65] GRATCH, J.; MARSELLA, S.. **Tears and fears: modeling emotions and emotional behaviors in synthetic agents**. In: AGENTS '01: PROCEEDINGS OF THE 5TH INTERNATIONAL CONFERENCE ON AUTONOMOUS AGENTS, p. 278–285, Montreal, Canada, June 2001. ACM Press.

[66] HAREL, D.. **Statecharts: A visual formalism for complex systems**. Science of Computer Programming, (8):231–274, 1987.

[67] O'HARE, G.; JENNINGS, N. R.. **Foundations of distributed artificial intelligence**. John Wiley and Sons, New York, NY, March 1996.

[68] HAWKINS, B.. **Creating an event-driven cinematic camera, part one**. 2003. Gamasutra article. Disponível em: <http://www.gamasutra.com/features/20030108/hawkins_01.htm>. Acesso em: 01 abr. 2004.

[69] HE, L.; COHEN, M. F. ; SALESIN, D. H.. **The virtual cinematographer: a paradigm for automatic real-time camera control and directing**. In: PROCEEDINGS OF THE 23RD ANNUAL CONFERENCE ON COMPUTER GRAPHICS AND INTERACTIVE TECHNIQUES. ACM SIGGRAPH, volumen 30, p. 217–224, August 1996.

[70] HENRY, D.. **The quake II's MD2 file format**. December 2002. Disponível em: <http://tfc.duke.free.fr/old/models/md2.htm>. Acesso em: 27 jan. 2005.

[71] HOINEFF, N.. **Produção de conteúdo, eis a questão**. Julho 2003. Disponível em: <http://www.observatoriodaimprensa.com.br/artigos/qtv010720031.htm>. Acesso em: 26 out. 2004.

[72] HOULETTE, R.; FU, D. ; ROSS, D.. **Towards an AI behavior toolkit for games**. In: AAAI SPRING SYMPOSIUM ON AI AND INTERACTIVE ENTERTAINMENT, March 2001.

[73] HUMPHREY, B.. **3DS file loader**. Disponível em: <http://www.gametutorials.com/gtstore/pc-72-1-3ds-file-loader.aspx>. Acesso em: 27 jan. 2005.

[74] POZZER, C.. **Storytelling project**. ICAD/IGames/VISIONLAB. Disponível em: <http://www.icad.puc-rio.br/~pozzer/tese/>. Acesso em: 10 mar. 2005.

[75] **International telecommunication union - ITU, study group 6 - broadcasting services**. Disponível em: <http://www.itu.int/ITU-R/study-groups/rsg6/>. Acesso em: 10 fev. 2005.

[76] **The online iTV dictionary.** What is digital TV (HDTV)? Disponível em: <http://www.itvdictionary.com/hdtv.html>. Acesso em: 16 jan. 2003.

[77] **The online iTV dictionary.** What are electronic program guides & interactive program guides? Disponível em: <http://www.itvdictionary.com/epg_ipg.html>. Acesso em: 15 jan. 2003.

[78] **The online iTV dictionary.** What are set-top boxes? Disponível em: <http://www.itvdictionary.com/set-top_box.html>. Acesso em: 16 jan. 2003.

[79] **The online iTV dictionary.** Examples of interactive television. Disponível em: <http://www.itvdictionary.com/examples_of_itv.html>. Acesso em: 02 mar. 2003.

[80] IZARD, C. E.. **The psychology of emotions**. New York: Plenum Press, New York, 1991.

[81] IZARD, C. E.. **Four systems for emotion activation: cognitive and noncognitive processes**. Psychological Review, 100(1):68–90, 1993.

[82] **JavaOS - an overview of the operating system architecture**. Disponível em: <http://java.sun.com/developer/products/JavaOS/OverView/index.html>. Acesso em 25 de jan. de 2005.

[83] JENNINGS, N. R.. **Coordination techniques for distributed artificial intelligence**. In: O'Hare, G.; Jennings, N. R., editors, FOUNDATIONS OF DISTRIBUTED ARTIFICIAL INTELLIGENCE, p. 187–210. John Wiley & Sons, New York, NY, March 1996.

[84] **JNI (Java Native Interface)- Sun Microsystems, Inc**. Disponível em: <http://java.sun.com/j2se/1.3/docs/guide/jni/>. Acesso em: 23 jul. 2004.

[85] JUCHEM, M.; BASTOS, R. M.. **Engenharia de sistemas multiagentes: uma investigação sobre o estado da arte**. Technical report series, PUC-RS, Abril 2001.

[86] KREBS, P.; KINDSCHI, C. ; HAMMERQUIST, J.. **Building interactive entertainment and e-commerce content for microsoft TV**. Microsoft Press, Redmond, Washington, 1st edition, February 2000.

[87] LIDÉN, L.. **Strategic and tactical reasoning with waypoints**. In: Rabin, S., editor, AI GAME PROGRAMMING WISDOM, p. 211–220. Charles River Media, Hingham, Massachusetts, 1st edition, 2002.

[88] LINDSTROM, P.; KOLLER, D.; RIBARSKY, W.; HODGES, L.; FAUST, N. ; TURNER, G.. **Real-time, continuous level of detail rendering of height fields**. In: COMPUTER GRAPHICS PROCEEDINGS. ANNUAL CONFERENCE SERIES. ACM SIGGRAPH, p. 109–118, 1996.

[89] LINDSTROM, P.; PASCUCCI, V.. **Terrain simplification simplified: A general framework for view-dependent out-of-core visualization**. IEEE Transaction on Visualization and Computer Graphics, 8(3):239–254, 2002.

[90] LOSASSO, F.; HOPPE, H.. **Geometry clipmaps: Terrain rendering using nested regular grids**. In: COMPUTER GRAPHICS

PROCEEDINGS. ANNUAL CONFERENCE SERIES. ACM SIGGRAPH, p. 769–776, 2004.

[91] LOYALL, A. B.; BATES, J.. **Hap: A reactive, adaptive architecture for agents**. Technical report CMU-CS-91-147, School of Computer Science, Carnegie-Mellon University, Pittsburg, PA, 1991.

[92] LUCENA, P. S.. **Expressive talking heads: um estudo de fala e expressão facial em personagens virtuais**. Master's thesis, Departamento de Informática, PUC-Rio, 2002.

[93] LUNA, F. D.. **Introduction to 3D Game Programming with DirectX 9.0**. Wordware Publishing, Inc., 1st edition, June 2003.

[94] MARRIOT; STUCKEY, P.. **Programming with Constraints**. MIT Press, 1998.

[95] MARRIN, C.; MYERS, R.; KENT, J. ; BROADWELL, P.. **Steerable media: interactive television via video synthesis**. In: PROCEEDINGS OF THE 2001 CONFERENCE ON 3D TECHNOLOGIES FOR THE WORLD WIDE WEB - WEB3D '01, p. 7–15, Paderbon, Germany, February 2001. ACM Press.

[96] MATEAS, M.. **An oz-centric review of interactive drama and believable agents**. Technical report, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, April 1997.

[97] MATEAS, M.; SENGERS, P.. **Narrative intelligence**. In: AAAI FALL SYMPOSIUM 1999, TECHNICAL REPORT, p. 1–10, Cape Cod, MA, November 1999. AAAI Press.

[98] MATEAS, M.; STERN, A.. **Towards integrating plot and character for interactive drama**. In: Dautenhahn, K., editor, SOCIALLY INTELLIGENT AGENTS: THE HUMAN IN THE LOOP, AAAI FALL SYMPOSIUM, TECHNICAL REPORT, p. 113–118, Menlo Park, CA, 2000. AAAI Press.

[99] MATEAS, M.; STERN, A.. **Architecture, authorial idioms and early observations of the interactive drama façade**. Technical report, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, December 2002.

[100] MATEAS, M.; STERN, A.. **Façade: An experiment in building a fully-realized interactive drama**. In: GAME DEVELOPERS CONFERENCE, GAME DESIGN TRACK, San Jose, CA, March 2003.

[101] MINSKY, M.. **The society of Mind**. Pan Books, London, 1987.

[102] MURRAY, J. H.. **Hamlet no Holodeck: o futuro da narrativa no ciberespaco**. Itaú Cultura: Unesp, São Paulo, 1st edition, 2003.

[103] NAREYEK, A.. **AI center homepage**. Disponível em: <http://www.ai-center.com/home/alex/publications.html>. Acesso em: 10 abr. 2004.

[104] NELSON, M. N.; LINTON, M.. **A highly available, scalable iTV system**. In: PROCEEDINGS OF THE FIFTEENTH ACM SYMPOSIUM ON OPERATING SYSTEMS PRINCIPLES, volumen 29(5), p. 54–67, Copper Mountain, Colorado, December 1995.

[105] NOVAK, J.; FLEISCHMANN, M.; STRAUSS, W.; VALLE, C.; PERANOVIC, P. ; SEIBERT, C.. **From memoria futura to i2TV: A technological framework and two models for new forms of cultural participation and production in mixed realities**. In: ICHIM '01, INTERNATIONAL CULTURAL HERITAGE INFORMATICS MEETING, volumen 1, p. 27–41, Milano, Italy, September 2001.

[106] **OpenTV datasheet.** EN2 set-top box software. Disponível em: <http://www.opentv.com/utilities/product-sheets/en2_datasheet.pdf>. Acesso em: 25 jun. 2003.

[107] ORKIN, J.. **Tips from the trenches**. In: Rabin, S., editor, AI GAME PROGRAMMING WISDOM, p. 29–36. Charles River Media, Hingham, Massachusetts, 2002.

[108] **Oz project homepage**. School of Computer Science, Carnegie Mellon University, 2002. Disponível em: <http://www-2.cs.cmu.edu/afs/cs.cmu.edu/project/oz/web/oz.html>. Acesso em: 10 fev. 2004.

[109] PARKE, F. I.; WATERS, K.. **Computer facial animation**. A K Peters Ltd, September 1996.

[110] PERLIN, K.. **Responsive face**. Technical report, Media Research Lab, New York University, 1997. Disponível em: <http://mrl.nyu.edu/~perlin/demox/Face.html>.

[111] PHILLIPS, C. B.; BADLER, N. I. ; GRANIERI, J.. **Automatic viewing control for 3D direct manipulation**. In: SI3D '92: PROCEEDINGS OF THE 1992 SYMPOSIUM ON INTERACTIVE 3D GRAPHICS, volumen 25, p. 71–74. ACM Press, March 1992.

[112] **PowerTV homepage**. Disponível em: <http://www.powertv.com/>. Acesso em 25 de jan. de 2005.

[113] POZZER, C. T.; DREUX, M. ; FEIJÓ, B.. **A real-time single-pass continuous LOD algorithm for regular height maps**. In: PROCEEDINGS OF THE BRAZILIAN SYMPOSIUM ON COMPUTER GAMES AND DIGITAL ENTERTAINMENT - WJOGOS'04, Curitiba, 2004.

[114] PROPP, V.. **Morphology of the folktale**. Journal of American Linguistics, 24(4), 1958.

[115] PROPP, V.. **Morphology of the Folktale (Laurence Scott, Trans.)**. University of Texas Press, Austin, Texas, 2nd edition, 1968.

[116] RABIN, S.. **Implementing a state machine language**. In: Rabin, S., editor, AI GAME PROGRAMMING WISDOM, p. 314–320. Charles River Media, Hingham, Massachusetts, 1st edition, 2002.

[117] RAO, A. S.; GEORGEFF, M. P.. **BDI agents: from theory to practice**. In: FIRST INTERNATIONAL CONFERENCE ON MULTIAGENT SYSTEMS, ICMAS-95, p. 312–319, San Francisco, June 1995.

[118] REILLY, W. S.; BATES, J.. **Building emotional agents**. Technical report CMU-CS-92-143, School of Computer Science, Carnegie-Mellon University, Pittsburg, PA, 1992.

[119] REYNOLDS, C.. **Flocks, herds and schools: a distributed behavioral model**. Computer Graphics, 21:25–34, 1987.

[120] REYNOLDS, C. W.. **Steering behaviors for autonomous characters**. In: PROCEEDINGS OF GAME DEVELOPERS CONFERENCE, p. 763–782, San Jose, California, 1999.

[121] RIBEIRO, L. A.; CAETANO, M. F.; SCHULTER, A.; BECKER, V.; MONTEZ, C.; MELO, E. ; FROHLICH, A.. **Infra-estrutura para recepção de TV interativa baseada em settopbox para o projeto I2TV**. In: BOLETIM BIMESTRAL SOBRE TECNOLOGIA DE

REDES PRODUZIDO E PUBLICADO PELA RNP, volumen 7(2,3), Julho 2003. Disponível em: <http://www.rnp.br/newsgen/0303/setbox.html>.

[122] SANTOS, G. L.. **Máquinas de estados hierárquicas em jogos eletrônicos**. Master's thesis, Departamento de Informática, PUC-Rio, Rio de Janeiro, RJ, 2004.

[123] SGOUROS, N.. **Dynamic generation, managing and resolution of interactive plots**. Artificial Intelligence, 107:29–62, 1999.

[124] SHOLAM, Y.. **Agent-oriented programming**. Artificial Intelligence, 60:51–92, 1993.

[125] **Sicstus Prolog homepage**. Disponível em: <http://www.sics.se/isl/sicstuswww/site/index.html>. Acesso em: 10 set. 2004.

[126] SILVA, J. Q.. **TV digital interativa**. Monografia de especialização em redes de computadores, Universidade do Vale do Rio dos Sinos - Unisinos, São Leopoldo, Abril 2003. Disponível em <http://geocities.yahoo.com.br/tvdigitalbr/jones_quadros_tv_digital.zip>.

[127] SILVA, C.; SEIXAS, R.. **Integração de agentes autônomos e SIG em uma arquitetura para simulação de confrontos**. In: SIMPóSIO DE DESENVOLVIMENTO E MANUTENçãO DE SOFTWARE DA MARINHA - SDMS, 2003.

[128] SOARES, L.; RODRIGUES, R. ; MUCHALUAT, D.. **Authoring and formatting hypermedia documents in the HyperProp System**. ACM Multimedia Systems Journal, 8(2):118–134, 2000.

[129] SPIERLING, U.; BRAUN, N.; IURGEL, I. ; GRASBON, D.. **Setting the scene: playing digital director in interactive storytelling and creation**. Computers & Graphics, 26:31–44, 2002.

[130] STERREN, W.. **Terrain reasoning for 3D action games**. 2001. Gamasutra article. Disponível em: <http://www.gamasutra.com/features/20010912/sterren_01.htm>. Acesso em: 11 abr. 2004.

[131] **Java technologies for interactive television**. 2001. Technical White Paper, SUN Microsystems. Disponível em: <http://java.sun.com/products/javatv/whitepapers/TechInterTV052101.pdf>.

[132] THALMANN, D.. **A new generation of synthetic actors: the real-time and interactive perceptive actors**. In: PROC. PACIPHIC GRAPHICS '96, p. 200–219, Taipeh, Taiwan, 1996.

[133] TOLEDO, E.. **Cronologia da televisão no brasil**. Disponível em: <http://www.paremasmaquinas.com.br/hist-tvbr.htm>. Acesso em: 02 fev. 2005.

[134] TOMLINSON, B.; BLUMBERG, B. ; NAIN, D.. **Expressive autonomous cinematography for interactive virtual environments**. In: AGENTS '00: PROCEEDINGS OF THE FOURTH INTERNATIONAL CONFERENCE ON AUTONOMOUS AGENTS, p. 317–324. ACM Press, 2000.

[135] TOZOUR, P.. **Stack-based finite-state machines**. In: Rabin, S., editor, AI GAME PROGRAMMING WISDOM 2, p. 303–306. Charles River Media, Hingham, Massachusetts, 2nd edition, 2004.

[136] TU, X.; TERZOPOLOUS, D.. **Artificial fishes: physics, locomotion, perception, behavior**. In: COMPUTER GRAPHICS PROCEEDINGS, ANNUAL CONFERENCE SERIES, ACM SIGGRAPH, p. 14–21, 1994.

[137] **Interactive TV today**. Disponível em: <http://www.itvt.com/etvwhitepaper.html>. Acesso em: 05 ago. 2003.

[138] **Interactive TV today**. Disponível em: <http://www.itvt.com/etvwhitepaper-2.html>. Acesso em: 05 ago. 2003.

[139] ULRICH, T.. **Continuous LOD terrain meshing using adaptive quadtrees**. Gamasutra article, 2000. Disponível em: <http://www.gamasutra.com/features/20000228/ulrich_01.htm>. Acesso em: 09 out. 2003.

[140] ULRICH, T.. **Chunked LOD: Rendering massive terrains using chunked level of detail control**. In: COMPUTER GRAPHICS PROCEEDINGS. ANNUAL CONFERENCE SERIES. ACM SIGGRAPH, 2000.

[141] **VxWorks operating system**. Disponível em: <http://www.windriver.com/products/device_technologies/os/>. Acesso em 25 de jan. de 2005.

[142] VELÁZQUEZ, J. D.. **Modeling emotions and other motivations in synthetic agents**. In: AAAI-97: PROCEEDINGS OF THE FOURTEENTH NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE, p. 10–15, Menlo Park, CA, 1997. AAAI Press.

[143] **The virtual reality modeling language**. International Standard ISO/IEC 14772-1, March 1997. Disponível em: <http://www.web3d.org/Specifications/VRML97>.

[144] **Microsoft webTV**. Disponível em: <http://www.webtv.com>. Acesso em: 13 ago. 2003.

[145] WILHELMS, J.; SKINNER, R.. **A 'notion' of interactive behavioral animation control**. IEEE Computer Graphics and Applications, 10:14–22, 1990.

[146] WOOLDRIDGE, M. J.; JENNINGS, N. R.. **Agent theories, architectures, and languages: a survey**. In: PROCEEDINGS OF THE ECAI94 WORKSHOP ON AGENT THEORIES, ARCHITECTURES AND LANGUAGES, p. 1–32, Amsterdam, The Netherlands, 1994.

[147] WOO, M.; NEIDER, J.; DAVIS, T. ; SHREINER, D.. **OpenGL Programming Guide**. Addison Wesley, Massachusetts, 3rd edition, 1997.

[148] WOOLDRIDGE, M.. **An Introduction to MultiAgent Systems**. John Wiley and Sons, 1st edition, June 2002.

[149] YANG, Q.; TENENBERG, J. ; WOODS, S.. **On the implementation and evaluation of abtweak**. Computational Intelligence Journal, 12(2):295–318, 1996.

[150] YOUNG, R.. **Creating interactive narrative structures: The potential for ai approaches**. In: AAAI SPRING SYMPOSIUM IN ARTIFICIAL INTELLIGENCE AND INTERACTIVE ENTERTAINMENT, Palo Alto, California, 2000. AAAI Press.

[151] YOUNG, R.. **An overview of the mimesis architecture: Integrating narrative control into a gaming environment**. In: AAAI SPRING SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND INTERACTIVE ENTERTAINMENT, AAAI TECHNICAL REPORT, p. 78–81, Stanford, CA, March 2001. AAAI Press.

[152] FININ, T.; FRITZSON, R.; MCKAY, D. ; MCENTIRE, R.. **KQML as an Agent Communication Language**. In: Adam, N.; Bhargava, B. ; Yesha, Y., editors, PROCEEDINGS OF THE 3RD INTERNATIONAL CONFERENCE ON INFORMATION AND KNOWLEDGE MANAGEMENT (CIKM'94), p. 456–463, Gaithersburg, MD, USA, 1994. ACM Press.

[153] **3D GameStudio**. Disponível em: <http://www.conitec.net/a4info.htm>. Acesso em: 04 jan. 2005.

A
Apêndices

A.1
Predicados

```
db(character(brian,1.0)).
db(character(hoel,1.0)).
db(character(marian,1.0)).
db(character(draco,-1.0)).

db(hero(brian,90.0)).
db(hero(hoel,90.0)).
db(victim(marian,90.0)).
db(villain(draco,90.0)).

db(place(princess_castle)).
db(place(dragon_castle)).
db(place(knight_castle)).
db(place(church)).

db(home(brian,knight_castle)).
db(home(hoel,princess_castle)).
db(home(marian,princess_castle)).
db(home(draco,dragon_castle)).

db(current_place(hoel,knight_castle)).
db(current_place(brian,knight_castle)).
db(current_place(marian,princess_castle)).
db(current_place(draco,dragon_castle)).

db(protection(princess_castle,1.0,70.0)).
db(protection(dragon_castle,-1.0,60.0)).
db(protection(knight_castle,1.0,0.0)).
db(protection(church,1.0,0.0)).

db(strength(hoel,40.0)).
db(strength(brian,80.0)).
db(strength(draco,45.0)).
db(strength(marian,10.0)).

db(affection(hoel,marian,100.0)).
db(affection(brian,marian,100.0)).
db(affection(marian,brian,0.0)).
db(affection(marian,hoel,0.0)).
db(alive(marian)).
db(alive(brian)).
db(alive(draco)).
db(alive(hoel)).
```

A.2 Operações

```
operator(1,
       go(CH,PL1),
       [
         alive(CH),
         not(kidnapped(_,CH)),
         not(kidnapped(CH,_)),
         current_place(CH,PL0),
         place(PL1),
         dif(PL0,PL1)
       ],
       [
         not(current_place(CH,PL0)),
         current_place(CH,PL1)
       ],
       10,
       [current_place(CH,PL1)],
       [],[]).

operator(2,
       reduce_protection(PL),
       [
         protection(PL,KIND,LPROT),
         { LPROT>0.0, LPROT1=LPROT-10.0 }
       ],
       [
         not(protection(PL,KIND,LPROT)),
         protection(PL,KIND,LPROT1)],
       10,
       [protection(PL,KIND,LPROT1)],
       [],[]).

operator(3,
       kidnap(VIL,VIC),
       [
         alive(VIC), alive(VIL),
         victim(VIC,VIC_L),
         character(VIC,KIND1),
         {VIC_L>80.0},
         not(kidnapped(VIC,_)),
         villain(VIL,VIL_L),
         {VIL_L>80.0},
         strength(VIC,VIC_S),
         current_place(VIC,PL),
         protection(PL,KIND2,LP),
         strength(VIL,VIL_S),
         current_place(VIL,PL),
         home(VIL,PL1),
         dif(PL,PL1),
         {VIL_S>VIC_S+LP*KIND1*KIND2}
       ],
       [
         kidnapped(VIC,VIL),
         not(current_place(VIC,PL)),
         not(current_place(VIL,PL)),
         current_place(VIC,PL1),
         current_place(VIL,PL1)
       ],
       10,
       [kidnapped(VIC,VIL)],
       [],[]).
```

```
operator(4,
       attack(CH,PL),
       [
         alive(CH),
          not(kidnapped(CH,_)),
          character(CH,KIND1),
         current_place(CH,PL),
         protection(PL,KIND2,L_PROT),
         dif(KIND1,KIND2),
         {
           L_PROT>0.0,
           L_PROT1 = L_PROT-30.0
         }
       ],
       [
         not(protection(PL,KIND2,L_PROT)),
         protection(PL,KIND2,L_PROT1)
       ],
       10,
       [protection(PL,KIND2,L_PROT1)],
       [],[]).

operator(5,
       fight(CH1,CH2),
       [
         alive(CH1), alive(CH2),
         not(kidnapped(CH1,_)),
         not(kidnapped(CH2,_)),
         dif(CH1,CH2),
         strength(CH1,LS1), strength(CH2,LS2),
         character(CH1, KIND1),
         character(CH2, KIND2),
         dif(KIND1,KIND2),
         {
           LS1>=10.0, LS2>=10.0
         },
         current_place(CH1,PL), current_place(CH2,PL),
         protection(PL,KIND3,L_PROT),
         {
           L_PROT=<0.0,
           NEW_LS1=LS1-LS2,
           NEW_LS2=LS2-LS1
         }
       ],
       [
         not(strength(CH1,LS1)), not(strength(CH2,LS2)),
         strength(CH1,NEW_LS1), strength(CH2,NEW_LS2)
       ],
       10,
       [strength(CH1,NEW_LS1), strength(CH2,NEW_LS2)],
       [],[]).

operator(7,
       free(HERO,VIC),
       [
         hero(HERO,_),victim(VIC,_),
         alive(HERO), alive(VIC),
         kidnapped(VIC,VIL), not(alive(VIL)),
         current_place(VIC,PL), current_place(HERO,PL),
         affection(VIC,HERO,LA)
       ],
       [
         not(kidnapped(VIC,VIL)), not(affection(VIC,HERO,LA)),
         affection(VIC,HERO,100.0)
       ],
       10,
       [not(kidnapped(VIC,VIL))],
       [],[]).
```

```
operator(6,
       kill(CH1,CH2),
       [
         victim(VIC,_),
         alive(CH1), alive(CH2),
         not(kidnapped(CH1,_)),
         dif(CH1,CH2),
         character(CH1, KIND1),
         character(CH2, KIND2),
         dif(KIND1,KIND2),
         strength(CH1,LS1), strength(CH2,LS2),
         current_place(CH1,PL), current_place(CH2,PL),
         protection(PL,KIND3,L_PROT),
         {
           L_PROT=<0.0,
           LS2<0.0, LS1>0.0
         }
       ],
       [
         not(alive(CH2))/*,
         not(affection(VIC,CH1,LA)),
         affection(VIC,CH1,100.0) */
       ],
       10,
       [not(alive(CH2))],
       [],[]).


operator(8,
       marry(CH1,CH2),
       [
         hero(CH1,_), victim(CH2,_),
         alive(CH1), alive(CH2),
         affection(CH1,CH2,L1),
         {L1>80.0},
         affection(CH2,CH1,L2),
         {L2>80.0},
         current_place(CH1,church),
         current_place(CH2,church),
         not(married(CH1,_)),
         not(married(CH2,_))
       ],
       [
         married(CH1,CH2), married(CH2,CH1)
       ],
       10,
       [married(CH1,CH2), married(CH2,CH1)],
       [],[]).

operator(9,
       get_stronger(CH1),
       [
         alive(CH1),
         strength(CH1,L1),
         {L2=L1+80}
       ],
       [
         not(strength(CH1,L1)),
         strength(CH1,L2)
       ],
       10,
       [strength(CH1,L2)],
       [],[]).
```

A.3
Regras que levam à geração dinâmica de objetivos

```
/*
 Se, no início da história, há uma vítima, ela vai realizar alguma ação que a
tornará frágil.
*/
rule(
    [
      e(i,victim(VIC,LEVEL)),
      e(i,character(VIC,KIND0)),
      e(i,current_place(VIC,PLACE)),
      e(i,protection(PLACE,KIND1,PROT))
    ],
    (
      [T],
      [
         h(T,current_place(VIC,PLACE1)),
        h(T,protection(PLACE1,KIND2,PROT1)),
         h({(KIND2*KIND0*PROT1)<(KIND1*KIND0*PROT)}),
         h(T>i)
      ],
       true
    )
).


/* Se a vítima  fica desprotegida, o vilão desejará raptá-la. */
rule(
    [
      e(i,victim(VIC,_)),
      e(i,character(VIC,KIND0)),
      e(i,current_place(VIC,PLACE1)),
      e(i,protection(PLACE1,KIND1,PROT1)),
      e(i,villain(VIL,_)),
      h(g,current_place(VIC,PLACE2)),
      h(g,protection(PLACE2,KIND2,PROT2)),
      h({(KIND2*KIND0*PROT2)<(KIND1*KIND0*PROT1)})
    ],
    (
      [T3],
       [
         h(T3,kidnapped(VIC,VIL))
       ],
       true
    )
).


/* Se a vítima foi raptada, o herói tentará salvá-la */
rule(
    [
      e(T1,kidnapped(VIC,VIL))
    ],
    (
      [T2],
       [
         h(T2,not(kidnapped(VIC,VIL))),
         h(T2>T1)
       ],
       true
    )
).
```

```
/*Se a afeição dos dois personagens é alta, eles desejarão casar*/
rule(
    [
      e(T,affection(CH1,CH2,L1)),
       h(T,affection(CH2,CH1,L2)),
      h(T,not(married(CH1,_))),
      h(T,not(married(CH2,_))),
      h({L2>95.0}), h({L1>95.0})
    ],
    (
      [T2],
       [
         h(T2,married(CH1,CH2)),
         h(T2>T)
       ],
       true
    )
).
```