Murilo Cerone do Nascimento
Tássio Naia dos Santos

Arquitetura *BDI* para agentes e *Blackboard*: um estudo de caso de técnicas de IA aplicadas a jogos

São Paulo
2010

Murilo Cerone do Nascimento
Tássio Naia dos Santos

Arquitetura *BDI* para agentes e *Blackboard*: um estudo de caso de técnicas de IA aplicadas a jogos

Monografia apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Bacharel em Engenharia.

Área de Concentração:
Engenharia de Computação

São Paulo
2010

Murilo Cerone do Nascimento
Tássio Naia dos Santos

Arquitetura *BDI* para agentes e *Blackboard*: um estudo de caso de técnicas de IA aplicadas a jogos

Monografia apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Bacharel em Engenharia.

Área de Concentração:
Engenharia de Computação

Orientadores:
Ricardo Nakamura
Roberto Cezar Bianchini

São Paulo
2010

. . . no amor e para graduar!

“O resultado de qualquer pesquisa séria só pode ser fazer duas questões crescerem onde antes apenas crescia uma.”

Thorstein Veblen
Sociólogo e economista
noruego-estadunidense
(1853 – 1929)

Resumo

Falar da motivação para uso de técnicas de IA em jogos, e do potencial que o BDI e o quadro-negro têm de agregar ao realismo da experiência dos jogos. (Citar artigos falando do emprego dessas técnicas.)

Falar do intuito do estudo de caso, do que se pretende (e como) quantificar. Mencionar as maiores escolhas e como imaginamos que elas delinearam os resultados. (E instigar o leitor à leitura do restante da tese!)

Palavras-chave: Engenharia. Engenharia da Computação. Inteligência Artificial. Arquitetura BDI. Sistema Blackboard. Jogos.

Abstract

Same text (*ibidem idem* previous section), but now in English.

Keywords: Engineering. Computer Engineering. Artificial Intelligence. BDI Architecture. Blackboard System. Games.

Lista de Figuras

Sumário

Capítulo 1

Introdução

Motivação e referências básicas do trabalho.

Quem são os orientadores, e como viemos a escolher o tema do trabalho. Falar aqui dos cursos de Computação Gráfica, Design e Programação de Games, SBGames.

Capítulo 2

Aspectos Conceituais

2.1 Inteligência Artificial

A Inteligência Artificial (IA) é uma área de pesquisa da ciência da computação e Engenharia da Computação, dedicada a buscar métodos ou dispositivos computacionais que possuam ou simulem a capacidade racional de resolver problemas, pensar ou, de forma ampla, ser inteligente.

Apenas recentemente, com o surgimento do computador moderno, é que a inteligência artificial ganhou meios e massa crítica para se estabelecer como ciência integral, com problemáticas e metodologias próprias. Desde então, seu desenvolvimento tem extrapolado os clássicos programas de xadrez ou de conversão e envolvido áreas como visão computacional, análise e síntese da voz, lógica difusa, redes neurais artificiais e muitas outras. Inicialmente a IA visava reproduzir o pensamento humano. A Inteligência Artificial abraçou a idéia de reproduzir faculdades humanas como criatividade, auto-aperfeiçoamento e uso da linguagem. Porém, o conceito de inteligência artificial é bastante difícil de se definir. Por essa razão, Inteligência Artificial foi (e continua sendo) uma noção que dispõe de múltiplas interpretações, não raro conflitantes ou circulares.

2.1.1 História

O primeiro trabalho a respeito de IA foi realizado por Warren McCulloch e Walter Pitts em 1943, em que demonstrava um modelo de neurônios artificiais, em que cada neurônio poderia estar “ligado” ou “desligado”, isso dependeria dos estados em que os neurônios vizinhos estariam. (RUSSEL & NORVIG, 2004: 18)

Donald Hebb demonstrou em 1949 uma regra de atualização simples para definir a intensidade de conexão entre neurônios, essa regra é influente ainda nos dias de hoje (RUSSEL & NORVIG, 2004: 18).

Em 1950 Alan Turing publicou o artigo *“Computing Machinery and Intelligency”*. Neste artigo ele apresentou diversos algoritmos de IA e seu famoso teste, o “Teste de Turing”. (Idem, 18).

Em 1951, no departamento de matemática de Princeton, Marvin Minsk e Dean Edmound apresentaram o primeiro computador de rede neural, o SNARC era composto por 3000 válvulas e simulava uma rede de 40 neurônios. (Op. Cit.,18)

Segundo RUSSEL & NORVIG (2004: 18-19), cinco anos após o SNARC John McCarthy realizaram um seminário em Dartmouth, reunindo todos os grandes pesquisadores do conhecimento. Nesse seminário Allen Newell e Herbert Simon do Carnegie Tech apresentaram o *Logic Theorist* o primeiro programa de raciocínio. Nesse seminário John McCarthy juntamente com os outros seminaristas percebeu que a IA devia ser um campo em separado e não uma ramificação da matemática nem mesmo estar sob o nome de pesquisa operacional ou teoria de controle, então eles definiram o nome de Inteligência Artificial e, após este seminário a IA foi definida como o campo em que tenta construir máquinas que funcionem de forma autônoma e em ambientes mutáveis.

Pesquisas sobre inteligência artificial foram intensamente custeadas na década de 1980 pela Agência de Projetos de Pesquisas Avançadas sobre Defesa (*"Defense Advanced Research Projects Agency"*), nos Estados Unidos, e pelo Projeto da Quinta Geração (*"Fifth Generation Project"*), no Japão. O trabalho subsidiado fracassou no sentido de produzir resultados imediatos, a despeito das promessas grandiosas de alguns praticantes de IA, o que levou proporcionalmente a grandes cortes de verbas de agências governamentais no final dos anos 80, e em conseqüência a um arrefecimento da atividade no setor, fase conhecida como O inverno da IA. No decorrer da década seguinte, muitos pesquisadores de IA mudaram para áreas relacionadas com metas mais modestas, tais como aprendizado de máquinas, robótica e visão computacional, muito embora pesquisas sobre IA pura continuaram em níveis reduzidos.

Os fatos acima listados marcam o nascimento e desenvolvimento da inteligência artificial, daquela época até os dias de hoje ocorreram grandes avanços neste ramo da computaçao.

Atualmente a IA está mais madura e não possui mais aquele ar de pioneirismo do inicio. O pesquisador de IA deve provar suas teorias com diversos experimentos empíricos e a demonstração dos resultados, e com o surgimento da Internet os pesquisadores foram capazes de compartilhar as evoluções de suas pesquisas e acelerar os processos de desenvolvimento.

Encorajados pela resolução dos subproblemas da IA, os pesquisadores começaram a examinar o problema do "agente como um todo". O movimento estabelecido tem como objetivo entender o funcionamento interno de agentes incorporados a ambientes reais com entradas sensoriais continuas. Um dos ambientes mais importantes para os agentes inteligentes é a Internet, diversas ferramentas utilizam a Inteligência Artificial na Internet como mecanismo de pesquisa, sistemas de recomendação e sistemas de construção de web sites. (RUSSEL & NORVIG, 2004:27-28) Na tentativa de se construir agentes

percebeu-se que subcampos que anteriormente eram isolados da IA, precisavam de uma reorganização para o melhor aproveitamento de seus resultados. Outra conseqüência foi que a IA se aproximou de outras áreas, como a teoria de controle e a economia, áreas que também lidam com agentes. (RUSSEL & NORVIG, 2004:28) No século XXI há um grande interesse em *Agentes Inteligentes* uma nova ramificação que pretende reunir todas as subáreas da IA, pois se percebeu que todas juntas poderiam criar um Agente perfeito realizando o principal objetivo da IA, entender o pensamento humano.

Entre os teóricos que estudam o que é possível fazer com a IA existe uma discussão onde se consideram duas propostas básicas: uma conhecida como “forte” e outra conhecida como “fraca”. A investigação em Inteligência Artificial Forte aborda a criação da forma de inteligência baseada em computador que consiga raciocinar e resolver problemas uma, sendo assim a forma de IA forte é classificada como autoconsciente. A Inteligência Artificial Fraca centra a sua investigação na criação de inteligência artificial que não é capaz de verdadeiramente raciocinar e resolver problemas. Uma tal máquina com esta característica de inteligência agiria como se fosse inteligente, mas não tem autoconsciência ou noção de si.

2.1.2 Áreas de aplicação

Enquanto que o progresso direcionado ao objetivo final de uma inteligência similar à humana tem sido lento, muitas derivações surgiram no processo. Exemplos notáveis incluem as linguagens LISP e Prolog, as quais foram desenvolvidas para pesquisa em IA, mas agora possuem funções não-IA. A cultura Hacker surgiu primeiramente em laboratórios de IA, em particular no MIT AI Lab, lar várias vezes de celebridades tais como McCarthy, Minsky, Seymour Papert (que desenvolveu a linguagem Logo), Terry Winograd (que abandonou IA depois de desenvolver SHRDLU). Muitos outros sistemas úteis têm sido construídos usando tecnologias que ao menos uma vez eram áreas ativas em pesquisa de IA. Alguns exemplos incluem:

- Planejamento automatizado e escalonamento: a uma centena de milhões de quilômetros da Terra, o programa Remote Agent da NASA se tornou o primeiro programa de planejamento automatizado (autônomo) de bordo a controlar o escalonamento de operações de uma nave espacial.

- Jogos: O Deep Blue da IBM se tornou o primeiro programa de computador a derrotar o campeão mundial em uma partida de xadrez, ao vencer Garry Kasparov.

- Controle autônomo: O sistema de visão de computador ALVINN foi treinado para dirigir um automóvel, mantendo-o na pista.
- Robótica: Muitos cirurgiões agora utilizam robôs assistentes em microcirurgias.
- Lógica incerta: Técnica para raciocinar dentro de incertezas, tem sido amplamento usada em sistemas de controles industriais.
- Redes Neurais: Vêm sendo usadas em uma larga variedade de tarefas, desde sistemas de detecção de intrusos a jogos de computadores.
- Aplicações utilizando Vida Artificial são utilizados na indústria de entretenimento e no desenvolvimento da Computação Gráfica.
- Sistemas baseados na idéia de agentes artificiais, denominados Sistemas Multiagentes, têm se tornado comuns para a resolução de problemas complexos.

Atualmente existem muitas aplicações práticas que envolvem conceitos de inteligência artificial. Dentre todas as áreas que, de alguma forma implementam conceitos de IA, os jogos eletrônicos tem ganhado grande destaque nos últimos anos.

2.2 Jogos digitais

Após pesados investimentos em novas tecnologias, os jogos digitais atuais possuem gráficos avançadíssimos que beiram a realidade, no entanto, pouco foi investido em seu conteúdo. Houve uma preocupação enorme por parte das empresas desenvolvedoras de jogos em acompanhar a evolução tecnológica dos últimos tempos, com isso nasceram novos consoles poderosos como o XBOX da Microsoft, o Playstation 3 da Sony e o Nintendo Wii, porém, pouco foi feito com relação ao conteúdo dos jogos. É muito fácil observar este fenômeno, basta notar que atualmente os jogos que mais são lançados são continuações de jogos antigos com gráficos mais bonitos. Os jogos que mais provam esta teoria são os RPGs (Role-Playing Game) e os jogos de aventura onde o jogador controla um personagem dentro de um universo e interage com outros jogadores ou NPCs (Non-Player Character). Apesar de toda qualidade gráfica oferecida por estes jogos, todos possuem uma lógica parecida, o usuário deverá realizar as mesmas tarefas para completar o jogo, no melhor caso existe mais de uma maneira de realizar uma mesma tarefa, no entanto, o jogo será sempre o mesmo, os NPCs sempre dirão as mesmas falas e farão as mesmas coisas não importa quantas vezes o usuário jogue. Um jogo que representa bem este fato

é o jogo da Nintendo “Legend of Zelda: Ocarina of Time”, nele o jogador ontrola um personagem que vive uma aventura para salvar uma princesa, na época em que foi lançado e até hoje o jogo ainda é um grande sucesso da Nintendo, no entanto, não importa quantas vezes seja jogado, a história é sempre a mesma, os diálogos com os NPCs são sempre os mesmos diálogos limitados, os poucos momentos em que o jogador pode “falar” com um NPC é para dizer “sim” ou “não” e em alguns casos é obrigado a escolher uma resposta pois a outra faz com que o NPC apenas repita a pergunta! Isso prejudica o jogo, pois uma vez que o usuário o termina, não existe mais a sensação do desafio a ser superado se ele jogar novamente, ou seja, a diversão associada ao jogo diminui. Talvez com um pouco de esforço e dedicação seja possível desenvolver jogos que não sejam tão simples e superficiais, onde os NPCs tenham “vontades” e possam dialogar com o usuário de forma mais parecida com a realidade humana. Outro ponto interessante seria se o NPC mudasse seu perfil toda vez que o jogador recomeçasse o jogo, tornando-o imprevisível.

2.3 Inteligência em jogos

O principal objetivo do uso de técnicas de inteligência artificial em jogos eletrônicos é a diversão. Por causa disso, o conceito de IA acabou recebendo outra interpretação por parte dos desenvolvedores de jogos, surgiu o conceito de *Game AI*.

Diferentemente da IA acadêmica que busca solucionar problemas extramamente difíceis, como imitar o reconhecimento que os humanos são capazes de realizar (reconhecimento facial e de imagens e objetos, por exemplo), ou mesmo entender e construir agentes inteligentes, a IA para jogos se preocupa com os resultados que o sistema irá gerar, e não como o sistema chega até os resultados. Isso se deve ao fato que jogos eletrônicos são negócios e os consumidores desses produtos os compram em busca de diversão, e não lhes interessa como a inteligência de um personagem no jogo foi criada, desde que ela transforme o jogo divertido e desafiador, além, claro, de tomar decisões coerentes com o contexto do jogo.

Um exemplo que deixa claro a utilidade da IA para jogos são os *shooters*. Nos jogos de tiros existem várias técnicas de IA que podem ser aplicadas, por exemplo, é possível fazer com que os *Non-Player Characters* (NPCs) se comuniquem e criem estratégias para tentar cercar um inimigo, fato que pode tornar o jogo ainda mais interessante, no entanto, também é possível utilizar técnicas de IA para fazer com que os NPCs acertem todos os disparos na cabeça de seus inimigos, fato que passa longe da realidade humana, e que poderia prejudicar a qualidade do jogo do ponto de vista de um usuário.

No começo do desenvolvimento de jogos eletrônicos, a programação de IA era mais usualmente conhecida por *programação de jogabilidade*, pois não havia nada de inteligente sobre os comportamentos exibidos pelos personagens controlados pelo computador. A figura abaixo **(figura IA jogos (SCHWAB, 2004) SCHWAB, Brian. AI Game Engine Programming. Hingham: Charles River Media. 2004.)**contém alguns exemplos de como a IA foi utilizada em jogos com o passar do tempo.

2.3.1 Crença, desejo e intenção — a arquitetura BDI

O modelo BDI (*Beliefs Desires Intentions*) foi originalmente proposto por Bratman como uma teoria filosófica do raciocínio prático, propondo uma análise do comportamento humano que seria baseado em crenças, desejos e intenções. Basicamente supõe-se que as ações são derivadas a partir de um processo chamado raciocínio prático. Este processo é constituído por duas etapas, na primeira, deliberação, o agente seleciona um conjunto de desejos que devem ser alcançados, de acordo com a situação atual das crenças do mesmo. Na segunda etapa ocorre a determinação de como os desejos produzidos no passo anterior podem ser atingidos através do uso dos meios disponíveis ao agente **[4] Michael Wooldridge, Reasoning about Rational Agents. Cambridge, MA: The M. I. T. Presss 2000.**. A seguir explicaremos melhor o que são crenças, desejos e intenções.

- **Crenças (Beliefs)**: Representam as características do ambiente e são atualizadas constantemente. Podem ser vistas como a componente informativa do sistema.
- **Desejos (Desires)**: Representam os objetivos a serem alcançados. Podem ser vistos como motivações do sistema.
- **Intenções (Intentions)**: Representam o atual plano de ações escolhido.

A partir deste modelo nasceu a arquitetura BDI para agentes. Agentes BDI são sistemas localizados em um ambiente (em nosso caso o ambiente será virtual) sujeito a variações, além disso, percebem o estado deste ambiente constantemente e podem atuar sobre o mesmo para tentar alterar o estado atual. Abaixo vemos o processo de raciocínio prático de um agente BDI.

Figura 2 - Diagrama de uma arquitetura BDI genérica [4]

Como se pode observar existe sete elementos básicos que compõem um agente BDI, são eles **[5] Ingrid Oliveira de Nunes, Implementação do modelo e da arquitetura BDI**:

- Um conjunto de crenças (*Desires*) atuais que representam as informações que o agente tem do ambiente.

- Uma função de revisão de crenças (*Belief Revision Function*), a qual determina um novo conjunto de crenças a partir da percepção da entrada e das crenças do agente.
- Uma função de geração de opções (*Option Generation Function*), a qual determina as opções disponíveis ao agente (seus desejos), com base nas suas crenças sobre seu ambiente e nas suas intenções.
- Um conjunto de opções (*desires*) corrente que representa os possíveis planos de ações disponíveis ao agente.
- Uma função de filtro (*filter*), a qual representa o processo de deliberação do agente, que determina as intenções do agente com base nas suas crenças, desejos e intenções atuais.
- Um conjunto de intenções (*Intentions*) atual, que representa o foco atual do agente, isto é, aqueles estados que o agente está determinado a alcançar.
- Uma função de seleção de ação (*Action Selection Function*), a qual determina uma ação a ser executada com base nas suas intenções atuais.

2.3.2 Quadro-negro — blackboard

A forma mais simples de apresentar o conceito de blackboard é através de uma metáfora que propõem a seguinte situação **(Daniel D. Corkill, Blackboard Systems. Blackboard Technology Group, Inc.)**: “Imagine um grupo de cientistas reunidos em uma sala trabalhando de forma cooperativa para resolver um problema. Para chegar a uma solução é utilizado um quadro negro (*blackboard*). A resolução do problema começa quando o mesmo é escrito no quadro negro juntamente com informações iniciais. Os cientistas verificam o conteúdo do quadro e aguardam uma oportunidade para aplicar seus conhecimentos visando solucionar o problema. Quando um cientista encontra informações suficientes para fazer uma contribuição, o mesmo coloca sua contribuição no quadro negro, e com sorte, este processo ativará outro cientista e assim por diante até chegarem à solução do problema.” Da metáfora acima se pode concluir que um blackboard possui uma base de dados comum a diversos sistemas que estão tentando solucionar um problema e utilizam e atualizam as informações existentes nesta base de dados. A seguir apresentaremos algumas características desta técnica:

- **Independência de conhecimento**: No exemplo acima, supomos que cada cientista adquiriu seus conhecimentos de maneira independente dos outros, ou seja, num

blackboard cada sistema que participa da solução de um problema tem seus próprios níveis de conhecimento, independentemente dos outros sistemas.

- **Diversidade nas técnicas de solução de problemas**: Uma das grandes vantagens do blackboard é que não interessa como os sistemas que estão trabalhando na resolução do problema funcionam, ou seja, um sistema pode utilizar redes neurais enquanto outro pode utilizar simulações, que para o blackboard estas "fontes de conhecimento" são caixas pretas que fazem suas contribuições.

- **Representação da informação é flexível**: Não existe nenhuma definição de como deve ser a informação, dando liberdade a quem implementa de definir como representar os dados.

- **Linguagem comum entre os sistemas**: Ao mesmo tempo em que existe a flexibilidade na representação da informação, é necessário, por outro lado, que todos os sistemas envolvidos sejam capazes de entender tais informações.

- **Liberdade de organização dos dados**: A organização dos dados é livre, porém, deve ser feita de tal forma que, no caso da base de dados possuir muita informação, seja fácil para um sistema encontrar informações específicas de forma rápida e simples.

- **Ativações baseadas em eventos**: Os sistemas que estão trabalhando na solução do problema não se comunicam diretamente, ao invés disto, todos "observam" a base de dados comum e utilizando as informações da mesma buscam gerar novos dados que são colocados na mesma base, tais dados podem ativar outro sistema que fará a mesma coisa até que o problema seja completamente resolvido.

- **Necessidade de controle**: É preciso haver um "órgão" para controlar todos os sistemas e decidir quem deve e quem não deve poder alterar os dados do quadro negro, assim não existe o risco de mais de um sistema alterar uma mesma área de dados simultaneamente.

- **Geração de solução incremental**: É evidente que a solução de um problema acontece de passo em passo, ou seja, um sistema faz uma contribuição com novos dados, a seguir, outro sistema utiliza tais dados e faz uma nova contribuição e de ciclo em ciclo o problema tende a ser resolvido.

Abaixo temos uma visão geral de como funciona o blackboard. Basicamente existem fontes de conhecimento (sistemas que estão trabalhando na resolução do problema) um

sistema de controle que concede permissões às fontes de conhecimento para alterarem os dados do quadro negro, este por sua vez, é uma base de dados que contem informações que ficam disponíveis a todas as fontes de conhecimento.

Figura 1- Visão geral de um sistema blackboard

Neste projeto utilizaremos a técnica blackboard como parte de um jogo computacional que será desenvolvido.

2.3.3 O que avaliar

O que o estudo de caso vai avaliar, e como. Decisões de projeto que validam o estudo (C++, *data-driven* design, prototipação, engenharia de software), isto é, porque o projeto permite extrapolar conclusões para jogos na indústria.

Capítulo 3

Especificação

Nesta seção descrevemos os critérios de aceitação do projeto:

- descrição das telas,
- opções esperadas dos menus,
- comportamento esperado dos NPCs

O Game Design, que é um dos anexos da monografia, complementa a especificação do projeto.

Capítulo 4

Metodologia

O sucesso de um projeto depende em grande medida da capacidade de adaptação dos envolvidos. Nesta seção tratamos do método de planejamento de atividades to trabalho, e de sua evolução no decorrer do projeto. Assim, em certa medida trataremos das mudanças que se operaram nas expectativas e previsões que fizemos ao longo do tempo.

4.1 Evolução

1. Levantamento de atividades e assuntos a pesquisar, cronograma preliminar (de pesquisas), criação de diário de bordo.
2. Decisões de projeto iniciais (condições de contorno): ferramentas e tecnologias de desenvolvimento:
   - escolha de linguagem de presença expressiva na indústria: C++;
   - definição de licensa livre para o projeto;
   - opção por jogo 2D;
   - pesquisa de simuladores BDI;
   - identificação de necessidades de renderização;
   - pesquisa de engines livres.
3. Divisão do projeto em escopos de conhecimento. C++, Java, BDI, AgentSpeak, SDL, Arquitetura de software para jogos.
4. Cronograma refinado (com um mês de folga):
   - escrita do documento de especificação game-design document (GDD);
   - previsão de tempos de implementação, busca de conteúdo artístico, e modelagem;
   - especificação de testes.
5. Metodologia de desenvolvimento: controle de versão, práticas de engenharia de software (referências), prototipação das funcionalidades básicas.

4.1.1 Percalços e reajustes

O que não ocorreu como previsto, e os ajustes que foram feitos. (Falar em termos mais genéricos, o específico será dito na história de evolução do projeto.)

Complexidade da modelagem de agentes (?)

Capítulo 5

Implementação

5.1 Visão global

Como as coisas funcionam. O que faz o quê (macroscopicamente).

Uma explicação sucinta da estrutura do programa, como ele inicia (forking) e como age em regime (responsabilidades java vs C++, e o protocolo).

5.2 Arquitetura

5.3 Protocolo de comunicação

5.4 Estrutura de pastas

O projeto foi desenvolvido com vistas a se prestar à experimentação no design. Assim, algumas ferramentas para a escrita de *scripts* de diálogos ou pequenas cenas foram desenvolvidas.

Descrevemos a seguir a organização dos arquivos do projeto. É importante que essa estrutura seja clara e intuitiva, já que é nossa intenção que alguns desses arquivos (que configuram o comportamento do jogo) sejam alterados primariamente por designers do jogo. Assim, é preciso que haja critério na complexidade que se expõe a esses "usuários".

5.5 Arquivos auxiliares

Uma série de arquivos de texto auxiliares são usados pelo jogo. Eles especificam

- diálogos,
- agentes,
- estados de agentes,
- tipos de estímulos a agentes,

Explicamos a seguir a função de cada um dos tipos de arquivo.

Os *diálogos* são scripts que codificam possíveis dizeres que NPCs ou o jogador podem optar por dizer. Carregam informação sobre o sentimento que expressam ou a impressão que causam. Seguem a gramática descrita no apêndice A.

Agentes são especificados em *AgentSpeak*, a linguagem interpretada pela ferramenta *Jason*. Cada arquivo contém a modelagem de um tipo de agente.

Os três últimos itens são arquivos de interface, usados para leitura pelos processos de renderização e de interpretação do sistema BDI. Optou-se por essa solução porque a comunicação se dá pelo envio de números, que, nesse caso, indicam qual a linha do arquivo de interface em que a mensagem está contida. Obviamente, essa abordagem só é possível porque as mensagens são conhecidas *a priori*.

Para comunicar o estímulo que uma determinada fala no diálogo provoca no agente, a lógica do jogo envia ao sistema BDI um inteiro indicando a linha em que está escrita a string que define o estado. Na prática, a lógica do jogo faz uma chamada para enviar o inteiro correspondente ao estímulo que se deseja comunicar, e uma busca é feita no arquivo que contém os estímulos possíveis para encontrar o número da linha correspondente. Esse número é, por fim, enviado ao sistema BDI, que fará a sua própria busca para identificar qual o estímulo recebido.

No caso de funções, o arquivo de interface deve ser gerado automaticamente pelo código Java, para que seja consultado durante a execução pela lógica do jogo.

Capítulo 6

Testes

Capítulo 7

Conclusão

Sucesso do estudo de caso?

7.1 Trabalhos futuros

Próximo SBGames?

7.2 *Post Mortem*

7.2.1 *"Lessons learned"*

Apêndice A

Gramática dos *scripts* de diálogo

Descrevemos a seguir a gramática da linguagem de especificação de diálogos.