UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

SIMULA - Ambiente Para Desenvolvimento de Sistemas Multiagentes Reativos

por

REJANE FROZZA

Dissertação submetida à avaliação, como requisito parcial
para a obtenção do grau de
Mestre em Ciência da Computação

Prof. Dr. Luis Otávio Campos Alvares
Orientador

Porto Alegre, março de 1997

CIP - CATALOGAÇÃO NA PUBLICAÇÃO

Frozza, Rejane

SIMULA - Ambiente Para Desenvolvimento de Sistemas Multiagentes Reativos / por Rejane Frozza. — Porto Alegre: CPGCC da UFRGS, 1997.

116 f.: il.

Dissertação (mestrado) — Universidade Federal do Rio Grande do Sul. Curso de Pós-Graduação em Ciência da Computação, Porto Alegre, BR-RS, 1997. Orientador: Alvares,

1. Inteligência Artificial Distribuída. 2. Agentes Inteligentes. 3. Sistemas Multiagentes. 4. Sistemas Multiagentes Reativos. 5. Modelos de Sistemas Multiagentes Reativos. I.Alvares, Luis Otávio Campos. II. Título.

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
Reitora: Profa. Wrana Panizzi
Pró-Reitor de Pós-Graduação: Prof. José Carlos Ferraz Hennemann
Diretor do Instituto de Informática: Prof. Roberto Tom Price
Coordenador do CPGCC: Prof. Flávio Rech Wagner
Bibliotecária-Chefe do Instituto de Informática: Zita Prates de Oliveira

Aos meus pais

Luiz e Angelina

Agradecimentos

Em especial, quero agradecer a meus amados pais, Luiz e Angelina, que me acompanharam nos momentos mais difíceis, importantes e alegres da minha vida. Sempre com palavras de carinho, apoio e estímulo, mostraram grande dedicação e amor a mim.

A minha irmã Nilse e ao meu cunhado Jorge pelas brincadeiras nos momentos de descontração e por serem tão alegres e queridos. A meu recém-nascido e lindo sobrinho Giovanni que me trouxe inspiração para o término deste trabalho.

Aos meus queridos e sempre amigos, Sílvia, Tanisi, Kika, Simone, Elton, Fabiane, Carlos Alba, Fernanda, Tiaraju, Vivi, Lisiane, Ana Carolina, Silvio, Rodrigo e Carla. Cada um com o seu jeito de ser e sempre com uma palavra de conforto e de carinho, conquistaram um lugar enorme no meu coração e jamais os esquecerei. Pelas horas de conversa, desabafo, apoio, compartilhando medos, dúvidas, mas também alegrias e etapas vencidas.

A Sílvia, Tanisi, Elton, Kika e Simone um agradecimento especial pelo grande apoio, carinho e amizade demonstrados. Tenho certeza que mesmo distantes estaremos sempre juntos. Adoro vocês.

Ao meu professor orientador, Luis Otávio Alvares, pelo companheirismo, orientação

Aos funcionários do CPGCC e, em especial, a Eliane, Carlos Alberto Júnior, Jorge, Mauríco, Ida, Fabiano e Margareth, pela amizade, ajuda e incentivo.

A Deus pela força que emite em nossos corações.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

IA	Inteligência Artificial
IAD	Inteligência Artificial Distribuída
RDP	Resolução Distribuída de Problemas
SMA	Sistema Multiagente

Resumo

Sistema multiagente é um tema de estudo em IAD, no qual um conjunto de agentes interage em um ambiente comum. A IAD baseia-se no comportamento social de agentes (humanos e artificiais), enfatizando as ações e as interações dos mesmos. Esses agentes podem ser cognitivos ou reativos.

Os sistemas multiagentes reativos têm sido usados em pesquisas e estudos ligados a campos importantes de aplicação, gerando o desenvolvimento de sistemas não apenas para a área acadêmica, mas também para atender às necessidades do mercado industrial.

Com o objetivo de abranger um ramo de pesquisas em sistemas multiagentes, este trabalho propôs a definição e a implementação de um protótipo de um ambiente de software que possibilita o desenvolvimento de aplicações em sistemas multiagentes reativos. Este ambiente tem a finalidade de facilitar a criação de tais aplicações com o uso de agentes, atingindo um resultado satisfatório.

O ambiente definido é o SIMULA, que possibilita ao usuário criar suas aplicações através de elementos de uma interface gráfica. O usuário, na interação com o ambiente, determina os agentes envolvidos no problema e como eles agirão no processo de resolução do mesmo. O usuário define a sua aplicação criando um modelo para ela.

Para definir as características do ambiente SIMULA, foram estudados e analisados três modelos de sistemas multiagentes reativos, encontrados em [STE 90], [DEM 93] e [FER 91], e estabelecido um quadro comparativo dos mesmos, segundo alguns critérios determinados. Esses modelos permitem que se faça a modelagem de aplicações nas quais o processo de resolução dos problemas, representados por tais aplicações, parece se adequar às características dos agentes reativos.

A validação do uso do ambiente SIMULA envolveu a modelagem de três aplicações: a atuação de robôs na busca de minerais, definido em [STE 90], o jogo PENGI, definido em [AGR 87] e [FER 91], e a atuação de parasitas no controle de pragas de plantações. Esta última sendo uma novidade para a resolução com agentes reativos.

Palavras-chave: Inteligência Artificial Distribuída, Agentes Inteligentes, Sistemas Multiagentes, Sistemas Multiagentes Reativos, Modelos de Sistemas Multiagentes Reativos

Title: "SIMULA - A Tool for Development of Reactive Multiagent Systems"

Abstract

Multiagent Systems is a subject of study in DAI (Distributed Artificial Intelligence) in which a group of agents interacts with the same tool . DAI is based on social behavior of agents (human and artificial ones) focus on actions and interactions of them . Those agents can be cognitive or reactive .

Reactive Multiagent Systems have been used in research and studies linked to important fields of use that generates the development of systems not only for academic areas but also to meet the needs of industrial market.

As we have the objective of ranging a research field of multiagent systems, this work comes up with a definition and implementation of a prototype of a software tool which enables the application development in reactive multiagent system . This tool has the purpose to ease the creation of such applications like the use of agents and consequently achieving a satisfactory result .

The tool is called SIMULA and it enables the user to create his own applications through elements from a graphic interface . The user who interacts with the tool determines the agents involved in the problem and how they will act in the process of solving this matter . Applications are developed based in models created by the user.

In order to characterize the tool SIMULA , three models of reactive multiagent systems found in [STE 90] , [DEM 93] and [FER 91] have been studied and analyzed. A comparative table has been made according to definite criteria. These models allow shaping the applications in which the process of problem solving represented by such applications seems adequate to the characteristics of reactive agents .

Validation of use of SIMULA tool involved modeling of three applications : performance of robots in mineral prospection defined in [STE 90] , PENGI game defined in [AGR 87] and [FER 91] and perfomance of parasite in controlling crop plagues that is being a novelty .

Keywords: Distributed Artificial Intelligence, Intelligent Agents, Multiagent Systems, Reactive Multiagent Systems, Models of Reactive Multiagent Systems

1 Introdução

Atualmente, as principais áreas de investigação da IA são os sistemas de aprendizagem, os sistemas de adaptação e a comunicação de agentes inteligentes, entre outras. O domínio de ação da IA se modifica com o avanço das pesquisas e com a passagem dos resultados para aplicações reais. Os avanços mais interessantes situaram-se na aprendizagem e na distribuição da inteligência [COE 93].

Pela classificação de [COE 94], a IA tem duas escolas de pensamento :

- Simbólica: visa a criação de programas capazes de imitar as características da inteligência humana, realizando tarefas que envolvam raciocínio, aprendizagem, planejamento e outras funções, através da manipulação de estruturas de símbolos. Algumas sub-áreas relacionadas são a resolução de problemas, o processamento do conhecimento, a prova de teoremas, os formalismos, os jogos, o reconhecimento da fala, a compreensão da língua natural, a composição musical e os tutores inteligentes;

- Conexionista: visa o desenvolvimento de máquinas inteligentes através da imitação dos modos como o cérebro humano é construído, realizando tarefas de reconhecimento de formas e padrões. Um exemplo são as redes neurais, que são constituídas por um conjunto de elementos de processamento, dispostos em camadas, e de um padrão de interconexão. Uma rede neural trabalha com estímulo e resposta, obtendo um resultado a partir de uma determinada configuração na entrada da rede e pela atuação dos neurônios e suas conexões.

Para situar-se os sistemas multiagentes reativos, utiliza-se a classificação de Pattie Maes, em [MAE 93], que distingue duas abordagens para a IA:

- IA baseada em conhecimento: enfatiza a modelagem e a construção de sistemas que tratam sobre um domínio de um problema. Estes sistemas modelam o domínio escolhido e podem responder questões sobre este domínio, freqüentemente envolvendo raciocínio. Produz trabalhos com sucesso, principalmente na área de sistemas especialistas, com a modelagem e o raciocínio sobre o domínio de um problema específico;

- IA baseada em comportamento: é vista como uma nova abordagem para o estudo da inteligência com o uso de sistemas de agentes reativos. Enfatiza a

modelagem e a construção de sistemas autônomos em ambientes dinâmicos que apresentam comportamentos no domínio de um problema. São sistemas com múltiplas atividades integradas, como por exemplo, para um robô as atividades seriam as de locomoção, navegação, coletagem de objetos e outras.

Estas duas abordagens diferem não apenas nos tipos de problemas estudados, mas também nas técnicas e soluções que exploram.

Nas organizações humanas, as atividades são realizadas por um grupo de pessoas que trabalham de modo cooperativo. Estas atividades em grupo envolvem a tomada de decisões e a distribuição do conhecimento. A partir disso, grupos de pesquisa em IAD começaram a estudar a concepção de sistemas compostos de agentes artificiais e agentes humanos participando na resolução de tarefas de grupo. Um objetivo da IAD, portanto, é construir sistemas capazes de solucionar problemas de forma cooperativa, com a utilização de SMAs.

A IA baseada em comportamento é uma nova abordagem para o estudo da inteligência, enfatizando a modelagem e a construção de sistemas para problemas que exibem alguma espécie de comportamento em seu domínio. Tem o objetivo de construir sistemas autônomos que operam em ambientes dinâmicos. Um exemplo para demonstrar a abordagem baseada em comportamento são as colônias de formigas que exibem comportamentos interessantes, atuando em grupo. Uma única formiga possui poucas capacidades, mas o comportamento da colônia de formigas, como um todo, é estruturado para realizar suas tarefas, como busca e transporte de comida.

Os últimos anos refletem um aumento do interesse na tecnologia orientada a agentes em várias áreas da ciência da computação, incluindo engenharia de software, redes de computadores e inteligência artificial. Agentes estão sendo usados em diversas aplicações como o gerenciamento de informação personalizada, o projeto de interfaces, os jogos de computadores e o gerenciamento de processos comerciais e industriais, entre outros.

1.1 Objetivos do Trabalho

As pesquisas em SMAs estão gerando o desenvolvimento de sistemas não apenas para a área acadêmica, mas também para atender às necessidades do mercado industrial.

A figura 1.1 tem o objetivo de facilitar a visualização e a compreensão do escopo deste trabalho.

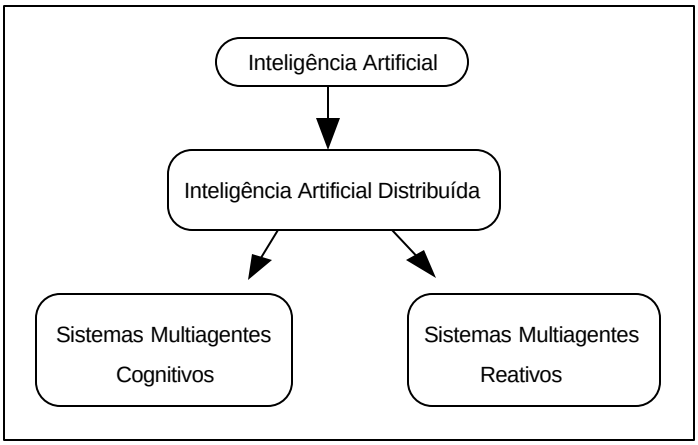


FIGURA 1.1 - Escopo do Trabalho

A inteligência artificial distribuída é uma ramificação da IA que enfatiza o trabalho conjunto de uma sociedade de agentes, os quais se organizam para solucionarem problemas. E a resolução de tais problemas é estruturada mais adequadamente através de modelos distribuídos. O comportamento e as características do grupo de agentes são pontos importantes na IAD para a construção de sistemas de agentes inteligentes.

Na IAD, encontram-se os sistemas multiagentes que podem ser classificados, para fins didáticos, em duas categorias: sistemas multiagentes cognitivos e sistemas multiagentes reativos. A essência deste trabalho envolve os agentes reativos.

Os sistemas multiagentes reativos têm sido usados em pesquisas e estudos ligados a campos importantes de aplicação, como simulação etológica (que refere-se ao comportamento dos animais e a sua acomodação às condições do ambiente), robótica, resolução de problemas e industrial.

A disponibilidade de um ambiente de desenvolvimento de sistemas reativos traria facilidades para a implementação das aplicações que estão sendo exploradas neste campo da computação. Com o crescimento do interesse pela utilização de sistemas reativos para a resolução de problemas, é importante que haja estímulo para a criação de sistemas reativos reais e que realmente funcionem com eficiência. A partir deste fato, este trabalho propôs a definição e a construção de um protótipo de um ambiente de desenvolvimento de aplicações em SMAs reativos, levando em conta características de modelos de SMAs reativos estudados.

Existem vários modelos de SMAs reativos, como os de [STE 90], [DRO 92] e [FER 91]. Estes modelos foram estudados e analisados para exemplificarem a realização de aplicações baseadas em sistemas reativos. Este estudo também contribuiu para uma

melhor definição das características que um ambiente deveria apresentar para facilitar o desenvolvimento de sistemas multiagentes reativos. Foi estabelecida também uma comparação das características dos modelos de SMAs reativos estudados com base em

A motivação para trabalhar com sistemas reativos deve-se a algumas características desses sistemas, que os tornam adequados a vários campos de aplicação:

- as soluções são robustas, no sentido que são mais tolerantes a falhas, ao mal funcionamento temporário de um ou mais agentes, ou a mudanças no ambiente;
- os agentes são simples, não havendo necessidade de elaboração de planos de ação e coordenação geral dos agentes;
- não é necessário haver um conhecimento do ambiente.

Portanto, um ambiente para SMAs reativos proporciona facilidade de desenvolvimento de aplicações com tais sistemas.

1.2 Organização do Texto

Nesta seção, apresenta-se uma descrição da organização deste trabalho que destaca os pontos importantes de cada capítulo.

O capítulo 2 enfatiza a relação e os objetivos da inteligência artificial distribuída com os sistemas multiagentes, destacando os principais problemas que ocorrem nesta abordagem.

O capítulo 3 aborda tópicos relacionados ao agente autônomo. Apresenta as características e a atuação de agentes em uma sociedade.

No capítulo 4, encontra-se a parte relacionada aos sistemas multiagentes. São apresentadas as categorias de sistemas multiagentes, que classificam os agentes em cognitivos e reativos, e uma descrição dos comportamentos sociais de agentes.

O capítulo 5 destaca as razões para a utilização de sistemas multiagentes na resolução de problemas, ilustrando um exemplo de resolução de uma tarefa sob a abordagem de sistemas multiagentes, além de mostrar aplicações desenvolvidas com esta abordagem.

No capítulo 6, descreve-se os modelos de sistemas multiagentes reativos estudados para o desenvolvimento deste trabalho, juntamente com exemplos de aplicações em cada um deles. Além disso, apresenta uma comparação entre estes modelos.

O capítulo 7 descreve o ambiente SIMULA. Este ambiente foi definido com o objetivo de possibilitar o desenvolvimento de aplicações em sistemas multiagentes reativos. Apresenta as características do ambiente e a sua funcionalidade.

O capítulo 8 descreve o protótipo do ambiente SIMULA e o capítulo 9 apresenta exemplos de modelagem de problemas com o uso deste ambiente.

No capítulo 10 são apresentadas as conclusões e os trabalhos futuros.

2 Inteligência Artificial Distribuída

Há classes de problemas que são naturalmente distribuídos e usam conhecimento distribuído para a sua resolução. A IAD fornece estratégias para a resolução de tais problemas.

A IAD está relacionada com situações de resolução de problemas, nas quais vários agentes cooperam para realizar um conjunto comum de objetivos. A idéia é usar agentes que trabalhem em conjunto para solucionar problemas que um único agente não é capaz de solucionar sozinho ou que a atuação de vários agentes se mostre mais eficaz. Estes agentes existem em um ambiente, interagindo com o mesmo e com os outros agentes.

Sob o aspecto de RDP, em IAD, uma tarefa global é inicialmente definida e o problema é projetar as entidades distribuídas, para permitir a execução desta tarefa global. O objetivo é estudar a distribuição e a resolução colaborativa de uma determinada tarefa.

A literatura, nesta área, mostra dois outros campos de estudo que derivam da IAD:

- a resolução distribuída de problemas : enfatiza o problema ou tarefa a ser executada e em como projetar um sistema composto de múltiplos agentes que agem em conjunto a fim de solucioná-la de forma eficiente.

- os sistemas multiagentes: enfatizam o agente e a sua interação com os demais agentes e com o ambiente. A existência dos agentes é independente de qualquer problema ou tarefa particular.

Em [DUR 94] encontra-se três visões diferentes relacionando Resolução Distribuída de Problemas e Sistemas Multiagentes. A primeira visão considera a Resolução Distribuída de Problemas como sendo um subconjunto de Sistemas Multiagentes. A segunda visão considera Sistemas Multiagentes como a base para Resolução Distribuída de Problemas. E a terceira visão considera Sistemas Multiagentes e Resolução Distribuída de Problemas complementares, não sendo possível classificar um sistema como um ou outro apenas pela observação de seu comportamento.

Segundo [LAB 93], pode-se classificar os trabalhos atuais que envolvem a IAD em:

- trabalhos teóricos sobre SMAs: organização de uma sociedade de agentes, compreensão do fenômeno da emergência, resolução de problemas em uma sociedade de agentes e outros;

- desenvolvimento de ferramentas: linguagens, metodologias de concepção de sistemas, plataformas de desenvolvimento de sistemas de IAD;
- utilização da arquitetura de IAD como uma tecnologia de desenvolvimento de sistemas de informática (por exemplo, industriais).

Usando o formalismo da IAD, um sistema pode ser descrito em dois níveis, segundo [WER 91]:

- Macro Nível : tem-se a construção de blocos do sistema como indivíduos, possuindo conhecimento, capacidade de resolução limitada e interação. Neste nível, considera-se o sistema como uma *Sociedade de Agentes*.
- Micro Nível : considera-se o sistema ou cada indivíduo como uma entidade separada, com um problema para solucionar, dado seu conhecimento e capacidade de resolução. Neste nível, a atenção focaliza-se no *Agente* .

2.1 Relação da IAD com os Sistemas Multiagentes

A IAD preocupa-se com a atividade de um agente autônomo em um ambiente multiagente. *Agente* é a palavra usada para designar uma entidade inteligente e
refere-se a que cada agente possui sua própria existência, a qual não é dependente da

A IAD baseia-se no *comportamento social*, com ênfase nas ações e nas interações dos agentes. Um comportamento social inteligente pode surgir no caso de membros *inteligentes* da sociedade (chamados Agentes Cognitivos) ou no caso de membros *não inteligentes* da sociedade (chamados Agentes Reativos). Uma das áreas de interesse, no campo da IAD, é a de Sistemas Multiagentes.

Os objetivos da IAD, em relação aos Sistemas Multiagentes, estão representados na figura 2.1 e podem ser descritos como:

- a identificação dos problemas que podem ser solucionados de forma eficiente com esta abordagem;
- a descrição dos problemas como agentes atuando em um ambiente multiagente, com o objetivo de atingirem a solução do problema;
- a compreensão do funcionamento de um sistema multiagente e de suas
- a análise do comportamento de um sistema multiagente e de sua importância
- a construção de sistemas multiagentes para aplicações em diversas áreas;

- a integração de sistemas multiagentes.

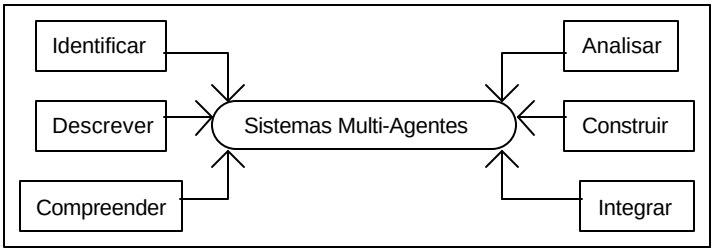


FIGURA 2.1 - Objetivo da IAD

2.2 Relação da IAD com a Resolução Distribuída de Problemas

Outra área de interesse no campo da IAD é a resolução distribuída de problemas. Nesta estratégia, utiliza-se um controle centralizado para atingir a solução de problemas. No controle tipo centralizado, um problema particular é decomposto em sub-problemas, que representarão sub-metas a serem alcançadas, e estes são distribuídos para um conjunto de agentes. Este controle pode ser exercido por um ou mais agentes, sendo responsável pela coordenação das atividades de solução de problemas pelos agentes. Os agentes podem compartilhar conhecimentos sobre o problema e sobre o desenvolvimento da solução e são criados para resolver o problema em particular apresentado. Esta abordagem pode ser representada como mostra a figura 2.2:

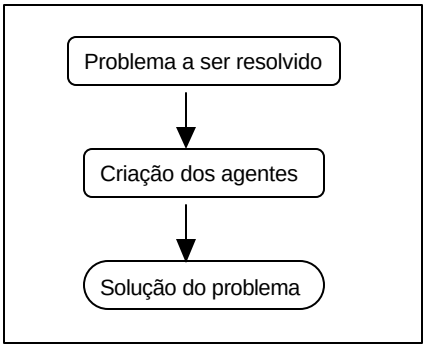


FIGURA 2.2 - Abordagem de RDP

Em RDP, os agentes são projetados para solucionar algum problema particular e não para solucionar quaisquer outros similares.

2.2.1 Exemplo de Resolução de uma Tarefa com a Abordagem de Resolução Distribuída de Problemas

A seguir, apresenta-se, a título de ilustração, um exemplo de resolução de problema, segundo a abordagem RDP, descrita em [SIC 92]:

"*A* deseja pintar a bandeira de um país na parede de seu escritório, mas ele não tem nem conhecimento nem habilidade para realizar esta tarefa. No entanto, há cinco pessoas na sala, com diferentes capacidades: *B* , que conhece as cores da bandeira do país e o seu formato; *C* , que é um pintor muito bom; *D* , que é um grande cantor; *E* , que é um bom artista e *F* , que é também um bom pintor. Todos têm capacidade de trabalhar juntos em uma mesma sociedade, podendo perceber seu ambiente (eles podem ver), comunicar-se uns com os outros (eles podem falar a mesma linguagem) e agir (eles podem usar suas mãos)."

No exemplo, *A* desempenhará, algumas vezes, a função de projetista do sistema e, algumas vezes, a função de usuário final, mas ele não participará das atividades de resolução do problema. Portanto, ele não deve ser considerado um agente.

A abordagem de RDP para solucionar o exemplo, apresentado anteriormente, é:

"*A* sabe que *B* é o único que sabe as cores e o formato da bandeira brasileira. Pediu-lhe, no entanto, que ensinasse a *E* como desenhar a bandeira. Uma vez que o formato estava desenhado, pediu a *C* e a *F* para começarem a pintar a bandeira, de acordo com as cores que *B* tinha falado a eles. *D* não realizou tarefa alguma. Quando *A* retornou, a parede estava pintada e ele sabia exatamente quem tinha feito o que e quando."

Pontos importantes nesta abordagem:

- embora trabalhando cooperativamente, não há necessidade dos agentes representarem explicitamente as capacidades e metas dos outros agentes participantes na solução. Isto é representado implicitamente pelo projetista;
- a descrição e decomposição da tarefa é, em muitos casos, feita pelo projetista;
- se a tarefa for adequadamente dividida pelo projetista, podem não surgir conflitos;
- não há necessidade de uma comunicação complexa entre os agentes para realizarem as suas metas;

- novos agentes não podem ser acrescentados dinamicamente à sociedade. Isto significa que um sistema desse tipo não pode ser considerado como um sistema aberto.

2.3 Problemas em Inteligência Artificial Distribuída

Cada um dos problemas, abaixo relacionados, aparecem nos domínios das

Os problemas estão divididos em seis aspectos [SIC 92]:

- Descrição, decomposição e alocação de tarefas: refere-se a como descrever e decompor, facilmente, uma tarefa complexa em subtarefas; como alocar estas subtarefas; qual a ordem em que elas devem ser executadas e como sintetizar resultados entre um grupo de agentes inteligentes;
- Interação, linguagens e comunicação: refere-se a como permitir que agentes se comuniquem e interajam; que linguagens de comunicação ou protocolos usar; o que e quando comunicar;
- Coordenação, controle e comportamento coerente: refere-se a como assegurar um comportamento global coerente em um conjunto de agentes, cada um com suas próprias habilidades e metas; como deve ser projetado o controle de tal sistema;
- Conflito e incerteza: refere-se a como reconhecer e adaptar pontos de vista diferentes e intenções conflitantes entre uma coleção de agentes, tentando coordenar suas ações, já que os agentes não têm informação total sobre o ambiente;
- Linguagens e ambientes de programação: refere-se, do ponto de vista computacional, a quais são as linguagens de programação que devem ser usadas e quais são os requisitos de um ambiente para a realização de testes;
- Como permitir que agentes individuais representem e raciocinem sobre ações, planos e conhecimento de outros agentes.

As soluções para estes problemas dependem das características dos domínios das aplicações em particular. Parte da tarefa das pesquisas em IAD envolve determinar quais as características dependentes do domínio e como elas afetam as respostas para estas

Quando o trabalho é feito por uma coleção de agentes, de forma coordenada, é importante a questão de divisão e organização deste trabalho: "Que agentes realizam quais tarefas e quando?". Uma distribuição de tarefas entre agentes exige que as mesmas sejam

formuladas e descritas. Tarefas que exigem mais conhecimento, do que um agente possui, devem ser decompostas.

Em um processo de decomposição, uma tarefa é decomposta em subtarefas menores, que exigem menos conhecimento para a sua solução. A escolha para a distribuição das tarefas depende da capacidade dos agentes para as executarem.

3 Agentes Autônomos

Os agentes são componentes chaves para desenvolver aplicações no domínio da IAD, como controle de tráfego aéreo, monitoração em redes de computadores e gerenciamento de informações que são filtradas de acordo com as necessidades dos usuários, entre outras. São entidades individuais que integram um sistema multiagente e que contribuem para atingir a resolução de um problema exposto a tal sistema.

Algumas características dos agentes humanos são aplicadas a agentes artificiais, no que se referem a estados mentais, tais como conhecimento, crença, intenção, compromisso, escolha e desejo.

Um agente age em um ambiente de acordo com suas próprias metas e conhecimento. O efeito das ações do agente é percebido pela produção de *eventos* , que correspondem a modificações do ambiente. Agentes são inteligentes se apresentarem a capacidade de serem flexíveis e adaptativos, compondo suas próprias metas (objetivos a serem atingidos) e realizando-as por ações eficientes [WER 91].

3.1 Características de um Agente

Um agente exibe as seguintes características :

- é uma entidade real ou virtual;
- está inserido em um ambiente: o ambiente pode mudar, como conseqüência das ações dos agentes. O ambiente é tudo o que é externo ao agente, podendo ser o mundo físico, um usuário via uma interface gráfica, uma coleção de outros agentes, a Internet e outros;
- percebe seu ambiente: a percepção é a troca de informações entre o agente e o seu ambiente. Um agente percebe o que está a sua volta e responde a trocas que ocorrem em tal ambiente e que influenciam as ações deste agente;
- possui autonomia: os agentes executam a maioria de suas tarefas de resolução de problemas sem a intervenção direta de humanos ou de outros agentes, possuindo uma existência própria, independente da existência de outros agentes. Possuem também controle sobre suas ações e seu estado interno. O comportamento autônomo dos agentes é a conseqüência das observações, do conhecimento e das interações desses agentes com os outros agentes e com o ambiente;

- possui a capacidade de agir no ambiente: os agentes podem executar ações que causam trocas no ambiente e/ou no estado interno do agente. Por exemplo, um agente pode passar de um *estado = parado* a um *estado = movimento* devido à necessidade de efetuar uma troca de posição no ambiente;

- pode comunicar-se com os outros agentes: a comunicação é a troca de informações entre os agentes. Essa comunicação pode ser intencional ou não-intencional. Na comunicação intencional, um agente envia uma mensagem específica a um outro agente, ou a vários outros agentes, com o objetivo de auxiliá-lo ou receber auxílio para a realização de alguma tarefa. Na comunicação não-intencional, o ambiente é a via para a troca de informações e a mensagem pode ser percebida por todos os agentes;

- tem capacidade social: interagem com outros agentes (humanos ou artificiais) e com o ambiente no qual se encontram, a fim de executar suas tarefas ou ajudar os outros em suas atividades;

- possui metas, as quais tenta atingir;

- pode agir em resposta ao seu ambiente ou pode guiar suas próprias metas (tomar iniciativas);

- possui adaptabilidade: capacidade do agente modificar seu comportamento em resposta a trocas ambientais ou aumento do conhecimento sobre a resolução do problema;

- pode ter mobilidade: capacidade do agente de trocar sua localização física no ambiente.

- possui conhecimento: o agente possui um conhecimento próprio e inicial, que pode ser visto como a representação do problema apresentado ao agente. Um agente pode, também, ampliar seu conhecimento através da percepção do ambiente e da comunicação com os outros agentes;

- pode ser capaz de raciocinar sobre as atividades, as ações e os planos de outros agentes, com o objetivo de guiar suas ações para a execução de tarefas. Desta forma, o agente pode decidir quando perceber, comunicar, planejar e agir.

3.2 Categorias de Agentes

Os agentes podem ser definidos como sendo:

- Componentes de um sistema: nesta categoria, os agentes atuam em uma parte do sistema auxiliando em tarefas específicas, como agente para monitoração de problemas em redes de computadores; agentes que agem como tutores, sugerindo os melhores caminhos para o usuário realizar suas tarefas; agentes que criam um modelo do usuário para filtrar as informações, como agentes de News e de Mails.

Ou seja, são agentes que interagem com o usuário a fim de realizar as tarefas para o mesmo.

- O próprio sistema (a solução): nesta categoria, os agentes trabalham em prol do usuário, sem haver necessidade de interação com o mesmo. Os agentes trabalham de forma mais autônoma, realizando a tarefa globalmente e não apenas parte dela.

3.3 Áreas de Estudo em Agentes

Segundo [WOO 95], as questões que envolvem agentes podem ser divididas em

⇒ Teorias de agentes.

⇒ Arquiteturas de agentes.

⇒ Linguagens de agentes.

3.3.1 Teorias de Agentes

Uma teoria de agente é vista como uma especificação para um agente, na qual desenvolve-se formalismos matemáticos para representar as propriedades dos agentes.

Um agente pode ser representado em termos de atitudes e pró-atitudes. As atitudes estão relacionadas à informação que um agente possui sobre o mundo no qual está inserido. As pró-atitudes guiam as ações dos agentes no ambiente.

A figura 3.1 mostra uma classificação de estados mentais quanto a atitudes e pró-atitudes de agentes.

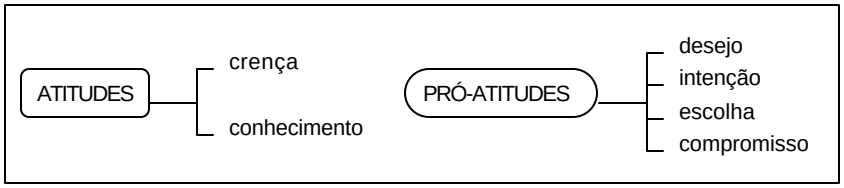


FIGURA 3.1 - Atitudes e Pró-Atitudes

Trabalhos que tratam sobre os elementos que compõem o estado cognitivo de um agente - as atitudes e pró-atitudes - podem ser encontrados em [COH 90], [COH 90a] e em [SHO 90].

3.3.2 Arquiteturas de Agentes

As arquiteturas são os modelos para a construção de sistemas de agentes. Pode-se classificar as arquiteturas de agentes em:

⇒ Arquiteturas deliberativas (abordagem clássica).

⇒ Arquiteturas reativas (abordagem alternativa).

⇒ Arquiteturas híbridas.

3.3.2.1 Arquiteturas Deliberativas

Nestas arquiteturas, o agente possui um modelo simbólico do mundo, representado explicitamente, no qual as decisões sobre que ações devem se executadas, por quais agentes e a que momento ocorrem através da montagem de planos de ações.

Referência relativa a este tipo de arquitetura envolve [JEN 94].

3.3.2.2 Arquiteturas Reativas (ou não-deliberativas)

São arquiteturas baseadas em comportamento. Utilizando esta abordagem, encontram-se trabalhos em linguagens de comportamento, com [BRO 91] e [BRO 91a], e na construção de agentes, como a arquitetura ANA (Agent Network Architecture) de [MAE 91].

Ao contrário das arquiteturas deliberativas, os agentes com arquiteturas reativas não possuem um modelo simbólico do mundo e nem trabalham com o planejamento de ações. A escolha de uma ação é determinada pela situação na qual o agente se encontra - o agente executa uma determinada ação, dada a ocorrência de uma condição.

3.3.2.3 Arquiteturas Híbridas

Uma arquitetura híbrida trabalha com as abordagens deliberativa e reativa em um sistema, com o objetivo de torná-lo mais adequado para a construção de agentes. A idéia é construir um agente atuante em dois subsistemas: o subsistema deliberativo, que contém um modelo simbólico do mundo, utilizando planejamento e tomada de decisões; o subsistema reativo, capaz de reagir a eventos que ocorrem no ambiente sem possuir capacidade de raciocínio.

Trabalhos envolvendo agentes neste tipo de arquitetura podem ser encontrados em [FEI 92], [MUL 94] e [MUL 94a].

3.3.3 Linguagens de Agentes

Linguagens de agentes são sistemas de software destinados à programação com agentes. Yoav Shohan, em [SHO 93], propõe a programação orientada a agentes como um novo paradigma de programação, na qual múltiplos agentes interagem entre si. A base desta linguagem de programação está na definição de agente, como uma entidade constituída de estados mentais (crenças, capacidades, escolhas e compromissos).

3.4 O Agente

Cada agente pode realizar suas próprias tarefas ou pode cooperar com os outros agentes para executar uma tarefa de caráter pessoal ou global.

Um agente apresenta duas partes principais, segundo [WER 91]:

- a parte estática, que define a arquitetura de um agente. Este aspecto é denominado *representação do conhecimento*, que trata da definição dos tipos de conhecimento do agente e como o mesmo é representado;
- a parte dinâmica, que corresponde aos *métodos de processamento* na arquitetura do agente.

A representação do conhecimento pode ser dividida em:

- representação do ambiente no qual o agente vive;
- capacidade de descrever o que um agente pode oferecer aos outros;
- representação dos problemas ou metas que o agente precisa solucionar;
- planos a serem executados;
- escolhas possíveis e decisões tomadas.

Os métodos de processamento podem ser divididos em:

- capacidade de raciocínio, incluindo planejamento da comunicação, detecção de incoerências, integração (combinação de dados), uso de dados, raciocínio sobre o conhecimento e o comportamento dos outros agentes;
- mecanismo de tomada de decisão.

3.5 Sociedade de Agentes

As sociedades de agentes são classificadas como [OLI 96]:

- homogêneas, quando os agentes são todos do mesmo tipo (mesma

- fechadas, quando os agentes são fixos no ambiente, ou abertas, quando há possibilidade de migração (entrada/saída de agentes);
- baseadas em leis (regras explícitas de comportamento, válidas para toda a

A figura 3.2 mostra que uma sociedade de agentes é composta pelo ambiente no qual os agentes estão inseridos, por um método de organização da sociedade e pelas interações entre os agentes e seu ambiente, envolvendo formas de comunicação entre os mesmos.

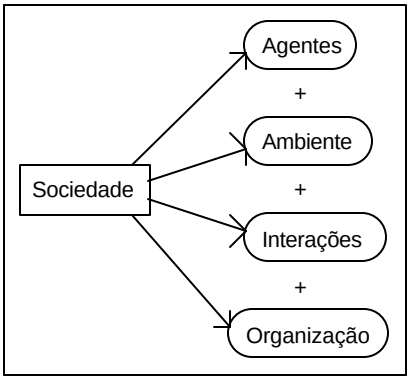


FIGURA 3.2 - Sociedade de Agentes

Em uma sociedade de agentes, pode-se definir o aspecto estático e o dinâmico. O *regras* , que regem a sociedade, ou o que pode ser feito dentro da sociedade. O aspecto dinâmico define o modo e o momento de agir na sociedade, dadas as regras que foram estabelecidas.

A sociedade de agentes é organizada de acordo com uma *Rede* (que pode ser hierárquica), que determina os *links de comunicação* entre os agentes. Usando estes *links* , ocorre a interação básica entre os agentes, que podem trocar entre si : conhecimentos, metas,

Dadas as interações básicas dentro da rede, os *links* entre os agentes são usados para a *comunicação* e para o *controle* .

O controle refere-se a estabelecer e a regular a troca de dados na sociedade. Este controle pode ser distribuído entre os agentes ou pode ser centralizado em um único agente.

3.5.1 Arquiteturas de Comunicação em uma Sociedade de Agentes

As interações e os processos de comunicação são pontos importantes a serem definidos em uma sociedade de agentes. A comunicação entre os agentes de uma sociedade tem duas formas:

- Comunicação direta: quando os agentes de uma sociedade conhecem uns aos outros e trocam dados. Neste tipo de comunicação, encontram-se as arquiteturas baseadas em troca de mensagens;
- Comunicação indireta: quando os agentes comunicam-se sem conhecer uns aos outros, pelo envio e recebimento de dados, de acordo com características pré-definidas. Neste tipo de comunicação, encontram-se as arquiteturas de quadro-negro (*blackboard*).

3.5.1.1 Arquiteturas de Quadro-Negro

Em uma arquitetura deste tipo, não há comunicação propriamente dita entre os agentes, mas as interações ocorrem através do quadro-negro. Quadro-negro é uma estrutura de dados dividida em regiões ou níveis e os agentes podem escrever e ler em um ou mais desses níveis, sob a supervisão de um mecanismo de escalonamento. Como referência, pode-se citar o trabalho de [ROT 84].

3.5.1.2 Arquiteturas de Troca de Mensagens

Em uma arquitetura deste tipo, os agentes comunicam-se diretamente entre si através de mensagens. Cada agente possuirá um nome, explicitamente conhecido, e uma representação dos outros agentes, incluindo capacidades, objetivos, conhecimentos e crenças. A organização das interações é feita com base em protocolos, que definem os passos de diálogo a serem executados pelos agentes para cada tipo de interação possível na sociedade. Um dos exemplos mais conhecidos é o protocolo de redes de contrato (*contract net*), utilizado para regular processos de negociação [SMI 88]. Quando um agente, chamado gerente, necessita que uma tarefa seja realizada, mas não possui os recursos necessários (como tempo e conhecimento), divulga uma mensagem na sociedade solicitando propostas. Os agentes (contratantes) que possuírem condições para realizar a tarefa submetem propostas ao gerente, que selecionará um ou mais agentes para a execução da tarefa. A seleção é comunicada aos proponentes através de uma mensagem de divulgação de resultados.

3.5.2 Estruturas de Organização

Há duas estruturas de organização para as sociedades de agentes [LAB 93]:

- Estrutura Horizontal: neste tipo de estrutura, todos os agentes da sociedade estão no mesmo nível, não há agentes mestres e agentes escravos (arquitetura na qual os agentes escravos apenas agem quando solicitados pelos agentes mestres). Um exemplo desta estrutura é um grupo de agentes de especialidades diferentes, trabalhando para resolver um mesmo problema;

- Estrutura Vertical: neste tipo de estrutura, os agentes estão estruturados por nível em uma arquitetura vertical. Em um mesmo nível, encontra-se, localmente, uma estrutura horizontal. O funcionamento dos agentes, nesta sociedade, é o seguinte: o problema a ser resolvido, de um agente que é superior na hierarquia, é decomposto em: sub-problemas que podem ser resolvidos localmente; sub-problemas que podem ser resolvidos pela cooperação com outros agentes do mesmo nível; sub-problemas a serem resolvidos por agentes de nível inferior na hierarquia.

3.5.3 O Controle das Atividades

Um agente pode executar atividades como:

- perceber o ambiente que o envolve;
- comunicar-se com os outros agentes;
- planejar como realizar uma tarefa;
- executar atividade de resolução de problema.

Para que estas atividades ocorram de forma coordenada, é necessário que haja controle sobre as mesmas. O controle pode ser *centralizado* ou *descentralizado* . Um controle centralizado poderia ser uma situação quando um agente, que tem conhecimento sobre um determinado problema, atribui a todos os outros agentes o que fazer e quando. Por outro lado, um controle descentralizado significa que qualquer membro da sociedade pode ajudar a estabelecer regras, que os agentes podem seguir.

Existem dois tipos de controle:

- Controle de Agente: refere-se a como um agente deve organizar internamente suas atividades;

- Controle de Sociedade: refere-se a como organizar o conjunto de agentes e a como controlar suas interações.

3.6 Arquitetura Funcional de um Agente Cognitivo

Um agente tem a possibilidade de adquirir conhecimento sobre o ambiente externo (capacidade de percepção do agente) e de interagir com os outros agentes (capacidade de comunicação do agente). Em função do conhecimento e das crenças que o agente possui e

conhecimento preciso, pois não submetem-se a constantes alterações do ambiente. No entanto, o conhecimento que provém da comunicação com os outros agentes é considerado como *conhecimento incerto*, pois está sempre em evolução.

3.6.2 Planejamento de Ações

Executar uma ação causa a passagem de um estado do agente e do ambiente para outro estado. A especificação de um problema contém um conjunto de metas, um conjunto de ações e uma descrição do estado inicial do ambiente. O objetivo é, então, encontrar uma

seqüência de ações que transformarão o ambiente de um estado inicial para um estado que satisfaça a descrição das metas. E isto pode ser feito através de planejamento das ações. O agente deve decidir qual a meta a ser satisfeita em primeiro lugar e, após, a ordem de execução de cada uma das próximas metas.

Pela intervenção de outros agentes, um plano pode ser adiado ou desfeito. O agente deve, por isso, alternar planejamento e execução, revisando as partes de seu plano.

Em um sistema multiagente, o planejamento pode ser distribuído. Neste caso, não há um plano global e cada agente elabora seu próprio plano, em concordância com os outros (no caso de agentes cooperativos). Algumas vezes, um agente necessita colaborar com outros agentes para construir planos complexos ou para realizar tarefas que ele não pode executar sozinho. Quando agentes querem realizar uma meta juntos, cada agente propõe um plano individual, incompleto e possível, sobre suas próprias crenças e capacidades. Então, os agentes usam seus planos individuais para, mutuamente, construírem um plano colaborativo para realizar a meta. Uma atividade colaborativa pode ser uma ação, envolvendo partes a serem feitas por um único agente, partes a serem feitas concorrentemente pelos agentes e partes a serem feitas juntamente entre os agentes.

Certos sistemas de IAD apresentam um planejamento centralizado, no qual um agente encarrega-se dos conflitos e da elaboração de um plano global.

3.6.3 Conflito e Cooperação

Conflito e cooperação são duas formas de relações entre agentes que possibilitam a execução de ações simultâneas por parte dos mesmos. Para ilustrar quando estas relações ocorrem, um exemplo são dois agentes empurrando um cubo, com o objetivo de deslocá-lo. O conflito surge se os dois agentes empurrarem o cubo em sentido contrário. Cada um dos agentes está evitando que o objetivo do outro seja cumprido - o de deslocar o cubo. Uma cooperação pode ser necessária se o cubo for muito pesado para um único agente conseguir deslocá-lo e, então, precisariam dois agentes para empurrarem o cubo no mesmo sentido. A figura 3.4 ilustra o conflito e a cooperação no deslocamento do cubo.

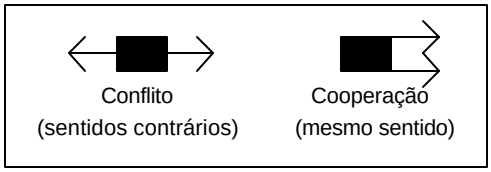


FIGURA 3.4 - Relações de conflito e cooperação no deslocamento de um cubo

O conflito é uma situação difícil dentro de um SMA, que acontece quando os interesses de um agente são afetados pelas decisões de outros agentes. Para evitar situações

conflitantes e atingir a solução de um problema, é preciso coordenar as atividades e negociar

Os conflitos podem ser locais ou globais. Um conflito local envolve um pequeno número de agentes, enquanto que um conflito global envolve a maioria dos agentes da sociedade.

Alguns conflitos que podem surgir, segundo [BER 95] são:

- o conflito de metas (objetivo a ser atingido) entre dois agentes, tal que a realização de uma meta por um impede o outro de atingir a sua meta, mesmo que parcialmente;
- os conflitos de resultado, quando vários agentes fornecem resultados diferentes para uma mesma informação solicitada;
- os conflitos de recursos, quando os agentes utilizam recursos comuns cujo número não é suficiente para permitir realizar suas metas inteira ou parcialmente.

3.6.3.1 Coordenação das Atividades

A coordenação de ações está ligada ao planejamento e à resolução de conflitos. Esta coordenação de ações do agente pode organizar-se de duas formas:

- uma coordenação através de um sistema capaz de determinar e de planejar (globalmente) as ações de diferentes agentes;
- ao contrário da anterior, dar autonomia total aos agentes, que identificariam os conflitos para resolvê-los localmente.

3.6.3.2 Negociação das Ações

Em certos domínios de aplicação, quando um grupo de agentes interage e coopera para atingir um objetivo, podem ocorrer conflitos que afetam a satisfação desse objetivo. Para resolver estes conflitos necessita-se de um mecanismo adicional, chamado de negociação. A negociação é uma técnica empregada para ajudar na solução de conflitos entre agentes e para atingir um acordo sobre a maneira de trabalhar em conjunto.

A negociação é caracterizada por:

- um número pequeno de agentes que participam do processo;
- um número pequeno de ações, por exemplo, propor, avaliar, modificar e

Para que o processo de negociação tenha êxito, é necessário seguir um protocolo que facilite a convergência para uma solução.

Exemplo de uma estrutura de negociação entre dois agentes *A* e *B* [LAB 93]:

1. o agente *A* faz uma proposição;

2. o agente *B* avalia a proposição apresentada pelo agente *A* ;

3. se o agente *B* estiver satisfeito, então interrompe a negociação, senão elabora uma contra-proposição e apresenta seus argumentos;

4. se o agente *A* considerar os argumentos do agente *B* , então a negociação é aceita, senão ocorre a intervenção de um terceiro agente, conforme o princípio da

.

O princípio da *Dinâmica de Substituição* consiste de ordenar as intervenções dos agentes e conhecer os diferentes pontos de vista dos mesmos, de maneira a que o grupo chegue a uma solução.

4 Sistemas Multiagentes

Um sistema multiagente é um conjunto de agentes que interagem em um ambiente comum, com o objetivo de realizarem suas tarefas.

Múltiplos agentes cooperam e comunicam-se para resolverem metas comuns, podendo estes agentes serem humanos ou artificiais. Uma vantagem do uso de agentes é que estes exibem, coletivamente, um comportamento emergente, sendo o comportamento da população de agentes como um todo maior do que a soma dos comportamentos individuais. O comportamento emergente é obtido através das interações entre os agentes, cada um possuindo apenas um conhecimento local sobre o estado do ambiente. Tal comportamento também é comum no campo da computação neural, modelada como uma coleção de neurônios biológicos, no qual as redes neurais do tipo *back-propagation* atuam através das propriedades emergentes de suas matrizes de conexões de pesos.

O comportamento de um agente, em um SMA, é função da percepção que ele possui de seu próprio estado, das suas interações com os outros agentes e de seu conhecimento. Segundo [STE 90], o comportamento de um grupo de agentes é emergente do comportamento individual dos mesmos.

SMAs envolvem a coordenação do comportamento inteligente em uma sociedade de agentes autônomos. A coordenação do comportamento relaciona-se com o compartilhamento de conhecimento, metas, capacidades e planos para agir ou solucionar problemas. Agente autônomo é aquele que tem sua própria existência, independente do problema a ser resolvido pela sociedade. Nesta abordagem, a sociedade de agentes já existe e o problema é apresentado a esta sociedade para a sua resolução. Pode ser representada da seguinte forma [HUB 95], como mostra a figura 4.1 :

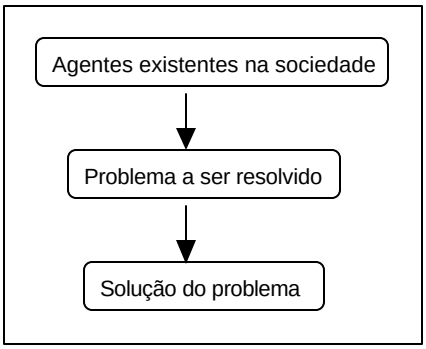


FIGURA 4.1 - Abordagem de SMAs

4.1 Auto-Organização de uma Sociedade de Agentes

Em uma sociedade humana, um problema complexo pode ser resolvido de forma eficiente por um grupo de indivíduos competentes e cuja organização evolui dinamicamente e de maneira autônoma, a fim de elaborar a resolução do problema. Criar uma organização que evolui dinamicamente e de maneira autônoma equivale a elaborar a auto-organização de um

A auto-organização de uma sociedade de agentes pode ser considerada como a modificação da topologia dos agentes que ocorre de forma autônoma, permitindo a estes adaptarem-se ao seu ambiente. Esta reorganização dinâmica da sociedade ocorre devido às mudanças nas interações entre os agentes e o ambiente e no estado interno dos mesmos.

4.2 Categorias de Sistemas Multiagentes

A capacidade de resolução de problemas por parte do agente e a arquitetura do agente são fatores que identificam duas grandes categorias de agentes, como mostra a figura 4.2:

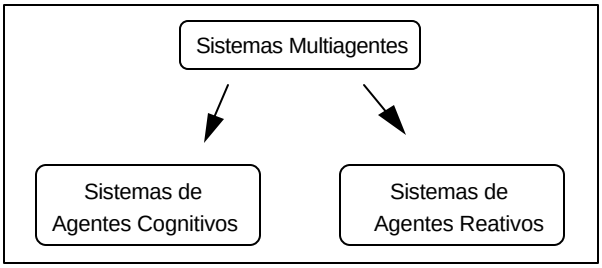


FIGURA 4.2 - Categorias de Sistemas Multiagentes

Um sistema multiagente pode ser constituído por agentes cognitivos ou por agentes reativos ou por ambos, sendo, neste último caso, considerado um sistema híbrido, no qual o comportamento e as ações dos agentes cognitivos se diferenciam dos reativos.

A nível didático, a divisão entre sistemas de agentes cognitivos e reativos parece ser bem acentuada, mas, na prática, é possível o desenvolvimento de sistemas que utilizem características de agentes cognitivos e reativos em um mesmo ambiente. E até mesmo um único agente pode agir de maneira cognitiva ou reativa de acordo com as circunstâncias na qual se encontra.

Nas seções seguintes, serão apresentadas as características marcantes de cada uma das categorias de sistemas multiagentes.

4.2.1 Sistemas de Agentes Cognitivos

O modelo cognitivo de agentes está associado à noção de estados mentais, como intenção, crença, desejo, compromisso, escolha e capacidades, análogos ou similares aos humanos [SHO 93].

A tupla (intenção, crença, ação) é o suporte lógico para modelar agentes cognitivos. A intenção refere-se ao compromisso de um agente, sendo necessária para a realização de metas e planos de ação. A crença caracteriza a visão que um agente possui do ambiente no qual está inserido e dos estados dos outros agentes. Um agente precisa revisar suas crenças quando ocorrem mudanças no ambiente. Através das ações, um agente executa tarefas,

Os agentes cognitivos são inteligentes e agem de acordo com o seu conhecimento, porque dispõem de uma capacidade de raciocínio sobre uma base de conhecimento e aptidões para tratar de informações diversas. Tais informações estão ligadas ao domínio da aplicação e são relativas às interações entre os agentes e entre os agentes e seu ambiente.

Como características principais de sistemas de agentes cognitivos, pode-se citar:

- Baseiam-se em modelos de organização social, como ocorre nas sociedades humanas;
- possuem uma representação explícita do conhecimento sobre o ambiente e sobre os outros agentes;
- planejam suas ações: como os agentes cognitivos são dotados de raciocínio, podem definir suas próprias metas e decidir quais ações devem ser executadas para atingir seu objetivo;
- possuem capacidade de percepção e comunicação;
- apresentam estados mentais, como desejos, intenções, crenças e compromissos: esses agentes possuem memória das suas ações realizadas no passado e, devido a isso, podem raciocinar sobre as mesmas, planejando as ações a serem realizadas no futuro;
- a sociedade de agentes cognitivos tem, geralmente, um pequeno número de membros.

4.2.1.1 Estrutura de um Agente Cognitivo

A estrutura genérica de um agente cognitivo, segundo [DEM 91], pode ser vista na figura 4.3.

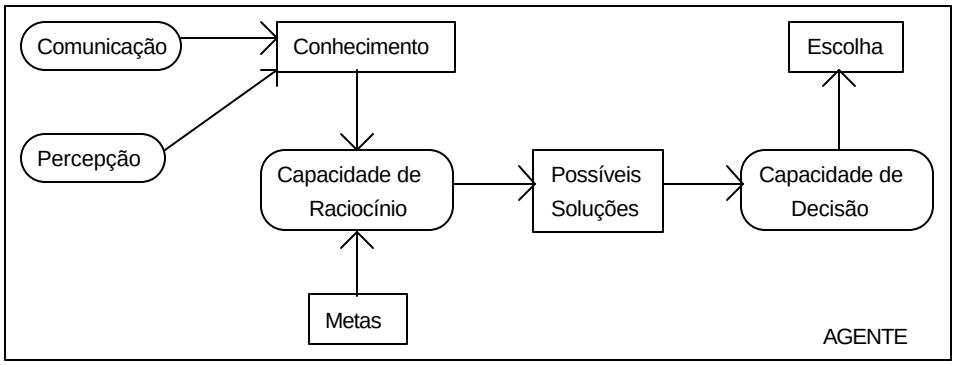


FIGURA 4.3 - Modelo Genérico de um Agente Cognitivo

O agente precisa obter representação do mundo ou do problema que ele tem que solucionar. Esta representação é o *conhecimento* , que pode ser adquirido através da percepção ou da comunicação com os outros agentes.

A partir da observação do comportamento de agentes e da comunicação com os outros agentes, é possível extrair *metas* para os mesmos.

Um agente pode considerar um conjunto de planos para realizar suas metas. Este conjunto é chamado de *possíveis soluções* . Não é preciso que um agente seja capaz de derivar todas as possíveis soluções, mas apenas uma parte delas, dependendo da *capacidade de raciocínio* do agente.

Quando várias soluções são aplicáveis, uma decisão deve ser tomada, escolhendo a melhor, do ponto de vista do agente. Isto é chamado de *escolha* , introduzindo um modelo de *capacidade de decisão* do agente.

4.2.1.2 Arquitetura de um Agente Cognitivo

A figura 4.4 descreve a arquitetura geral de um agente cognitivo.

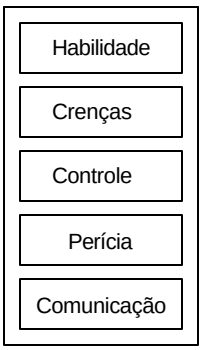


FIGURA 4.4 - Arquitetura Geral de um Agente Cognitivo

Apresenta-se, a seguir, uma explicação de cada uma das partes que compõem a arquitetura de um agente cognitivo [LAB 93]:

- Habilidade: é a declaração do conhecimento e das capacidades do agente;
- Crenças: em um ambiente multiagente, cada agente possui conhecimento sobre si mesmo e sobre os outros. Esse conhecimento não é, necessariamente, objetivo, então, fala-se de crenças de um agente. Crença é o conhecimento considerado como certo pelo agente.
- Controle: o conhecimento do controle em um agente é representado pelas metas, intenções, planos e tarefas que ele possui.
- Perícia: é o conhecimento sobre a resolução do problema. Por exemplo, para um sistema especialista, que utiliza regras, esse conhecimento corresponde a sua base de regras.
- Comunicação: o agente deve possuir um protocolo de comunicação que o permita interagir com os outros agentes, para uma boa cooperação e coordenação

4.2.2 Sistemas de Agentes Reativos

Os agentes reativos são considerados entidades mais simples que os cognitivos devido as suas características, principalmente de não apresentarem estados mentais (desejos, intenções, crenças e outros) e, portanto, não possuírem capacidade de raciocínio e planejamento.

A idéia principal, em um sistema multiagente reativo, é que um comportamento global inteligente possa ser alcançado a partir do comportamento individual dos agentes. Em um SMA, não é necessário que cada agente seja individualmente inteligente para alcançar um comportamento global inteligente. Para ilustrar essa idéia, utiliza-se o exemplo de uma colônia de formigas: mesmo que uma única formiga apresente poucas capacidades, pareça não ser muito inteligente, o comportamento de uma colônia de formigas, como um todo, é bem estruturado e fez com que as formigas já tenham sobrevivido por milhões de anos.

Alexis Drogoul, em sua tese [DRO 93], apresenta a modelagem e a simulação de um sistema multiagente reativo aplicado ao comportamento de uma colônia de formigas.

Tudo que um agente reativo sabe sobre as ações e os comportamentos dos outros membros da sociedade é percebido pelas mudanças no ambiente. Seguindo o exemplo das formigas, estas, usualmente, seguem um rastro químico quando buscam por comida, mas não existe um modelo de comunicação direta entre elas.

Como características principais de sistemas de agentes reativos, pode-se citar:

- baseiam-se em modelos de organização etológica, relacionados ao comportamento dos animais. Um exemplo clássico de uma simulação, utilizando agentes, é o de uma sociedade de formigas, que trabalham em conjunto na busca e transporte de alimento;
- possuem uma representação implícita do conhecimento sobre o ambiente e sobre os outros agentes: o conhecimento é representado pelo comportamento dos agentes reativos no sistema multiagente;
- são agentes baseados em comportamento: cada agente, individualmente, exibe comportamentos de acordo com a situação na qual se encontram no ambiente de solução de algum problema. Quando o ambiente altera-se, os agentes reativos mudam seu comportamento;
- o seu modo de funcionamento é por estímulo e resposta: a relação do agente reativo com seu ambiente ocorre através de respostas a estímulos recebidos. O agente executa uma determinada ação quando uma certa condição for satisfeita;
- possuem capacidade de percepção e comunicação: a comunicação ocorre através do ambiente, o qual é utilizado como meio de transmissão de mensagens. Tal comunicação não é dirigida a um determinado agente, mas é propagada a todos pelo ambiente, sem o uso de algum modelo de comunicação para o envio de mensagens;
- não possuem raciocínio: os agentes reativos não possuem memória das suas ações executadas no passado nem qualquer previsão das ações a serem executadas no futuro. Como não possuem capacidade de raciocínio, não planejam suas ações e,
- a sociedade de agentes reativos tem, geralmente, um grande número de membros.

4.3 Dimensões de Sistemas Multiagentes

O modelo de um agente, em um ambiente multiagente, estrutura a descrição do mesmo sob várias dimensões. Estas dimensões caracterizam sistemas baseados em múltiplos agentes [DEM 91].

A primeira dimensão refere-se à resolução distribuída de problemas, na qual as tarefas são inicialmente especificadas e, então, distribuídas entre os vários agentes.

A segunda dimensão refere-se à capacidade de um agente realizar uma tarefa (agentes autônomos ou especializados) e à localidade de uma tarefa (tarefa individual ou distribuída). Visto que um agente pode sofrer alterações, passando de completamente autônomo a especializado, e que as tarefas podem estender-se de individuais a distribuídas, quatro tipos de comportamento social são verificados:

- Co-habitação(autônoma, individual): é a capacidade que o agente apresenta de realizar uma tarefa com sucesso e sozinho. É uma questão típica de problemas clássicos de inteligência artificial em um ambiente mono-agente. Um agente não é obrigado a cooperar com os outros, porque está no mesmo ambiente;

- Cooperação(autônoma, distribuída): é a necessidade de um agente cooperar com outros agentes, para a execução de uma tarefa pessoal. Esta cooperação pode ser necessária ou porque o agente não é capaz de realizar a tarefa sozinho (soluções possivelmente restritas) ou porque os outros agentes podem realizá-la de forma mais eficiente (em um período de tempo menor);

- Colaboração(especializada, individual): refere-se a metas globais, que podem envolver todos os agentes e que podem ser realizadas individualmente por vários agentes. O principal problema é selecionar o agente para executar a tarefa;

- Distribuição(especializada, distribuída): refere-se a metas globais, que apenas podem ser realizadas por vários agentes coletivamente. O principal problema é dividir e distribuir a tarefa para os vários agentes.

A outra dimensão indica o tipo de informação trocada entre os agentes - é o modelo do agente, desde uma abordagem baseada em processos até um modelo cognitivo. A abordagem baseada em processos é melhor caracterizada por modelos de agentes puramente reativos. O modelo cognitivo abstrai a noção de conhecimento, crenças, desejos e intenções do agente.

Há dois critérios, segundo [SIC 92], que classificam o comportamento social de um agente:

- Localidade de Tarefa: uma tarefa pode ser global ou local. A tarefa global refere-se a todos os agentes da sociedade e a tarefa local refere-se a apenas um agente da sociedade;

- Capacidade de execução da tarefa: um agente pode ser capaz ou não de executar uma tarefa. Um agente é capaz de executar uma tarefa, se ele tem as habilidades exigidas para realizá-la.

4.3.1 Ambiente do Agente Autônomo

Cada agente possui um conhecimento incompleto, incerto e parcial do ambiente, no qual está inserido, e dos outros agentes que o cercam. Como conseqüência, não é possível, para um agente, planejar ações seguras ou encontrar soluções aplicáveis por um longo período. A incerteza sobre as escolhas de outros agentes exige que um agente, cuidadosamente, monitore a execução de planos e, freqüentemente, atualize seu conhecimento, suas metas e suas escolhas.

Pode-se distinguir três tipos de trocas de informação entre agentes:

- Conhecimento do agente: usualmente, é um modelo do ambiente, mas pode incluir um modelo do próprio agente. Dois agentes podem ter descrições *complementares* ou *conflitantes* de uma situação compartilhada, devido às diferenças na percepção do ambiente por ambos. Uma descrição é complementar quando o raciocínio sobre o conhecimento de outros agentes pode ser útil para a troca de informações. Uma descrição é conflitante quando surgem descrições contraditórias sobre uma mesma situação do ambiente;

- Planos ou caminhos possíveis para uma solução: a troca de soluções possíveis ocorre quando dois ou mais agentes concordam com uma solução ou plano comum. Tal processo pode ser feito encontrando-se a intersecção das soluções de cada agente. A intersecção pode ser vazia, devido às diferentes *capacidades de raciocínio* dos agentes.

- Escolha de um plano: após ser encontrado um conjunto de possíveis soluções, uma delas tem que ser de escolha comum, quando a cooperação for necessária. Para realizar esta escolha, duas possibilidades são oferecidas: escolher a primeira solução emitida da intersecção do conjunto de possíveis soluções ou exigir de cada agente.

A necessidade de escolha também ocorre quando um agente requisita que outro agente realize uma tarefa. No entanto, quando uma escolha não é aceita por outro agente, a

5 Resolução de Problemas com Sistemas Multiagentes

Os agentes interconectados, em um sistema multiagente, podem cooperar na resolução de problemas de forma distribuída e inteligente.

Pode-se citar, como razões para o interesse na distribuição da inteligência, para a resolução de problemas, os seguintes itens [BON 88]:

- adaptabilidade: a distribuição da inteligência permite, a um sistema de IAD, um poder adaptativo maior;
- redução de custos;
- facilidade de desenvolvimento e gerenciamento das tarefas: se um sistema inteligente pode ser construído de uma forma distribuída, cada parte pode ser desenvolvida separadamente por um especialista em um tipo particular de conhecimento ou domínio;
- aumento da eficiência (velocidade), devido à possibilidade de execução paralela;
- aumento da confiabilidade na resolução das tarefas;
- especialização: o conhecimento ou as ações podem ser coletadas de especialistas;
- limitações de recurso: os agentes computacionais têm recursos limitados para a resolução de problemas, necessitando de cooperação e coordenação.

Problemas são, algumas vezes, muito complexos ou grandes para serem solucionados por processos únicos, devido à representação semântica e ao poder computacional. Estes problemas podem requerer conhecimento de vários domínios diferentes e podem envolver gerenciamento de dados distribuídos fisicamente. Por isto, a distribuição de tarefas (resolução de problemas) entre os agentes, em um sistema multiagente, facilita e contribui para uma melhor conclusão das mesmas.

Um agente pode não ter conhecimento suficiente para solucionar um problema, já que cada agente possui um conhecimento incompleto do ambiente que o cerca. Desta forma, os agentes possuem habilidade na troca de informações com os outros agentes, facilitando a resolução de problemas de forma cooperativa.

Quando uma tarefa é atribuída a um grupo de agentes, cada agente inicia seu trabalho em direção a uma solução, baseado em seu conhecimento local. No entanto, desde

que as tarefas são atribuídas a agentes, e cada agente tem um conhecimento não completo, qualquer um destes agentes pode descobrir que não é capaz de solucionar a tarefa sozinho, sem adquirir mais conhecimento. Deste modo, os agentes, com diferentes conhecimento e capacidades, trabalham juntos para atingir a resolução da tarefa.

5.1 Exemplo de Resolução de uma Tarefa com a Abordagem de Sistemas Multiagentes

A seguir, apresenta-se, a título de ilustração, um exemplo de resolução de problema, segundo a abordagem de sistemas multiagentes, descrita em [SIC 92]:

" *A* deseja pintar a bandeira de um país na parede de seu escritório, mas ele não tem nem conhecimento nem habilidade para realizar esta tarefa. No entanto, há cinco pessoas na sala, com diferentes capacidades: *B* , que conhece as cores da bandeira do país e o seu formato; *C* , que é um pintor muito bom; *D* , que é um grande cantor; *E* , que é um bom artista e *F* , que é também um bom pintor. Todos têm capacidade de trabalhar juntos em uma mesma sociedade, podendo perceber seu ambiente (eles podem ver), comunicar-se uns com os outros (eles podem falar a mesma linguagem) e agir (eles podem usar suas mãos)."

No exemplo, *A* desempenhará, algumas vezes, a função de projetista do sistema e, algumas vezes, a função de usuário final, mas ele não participará das atividades de resolução do problema. Portanto, ele não deve ser considerado um agente.

Os agentes coexistem em um ambiente comum e cada um deles pode colaborar com os outros para realizar uma meta comum.

A abordagem de sistemas multiagentes (usando um modelo de agente cognitivo) soluciona o exemplo, anteriormente apresentado, da seguinte forma :

" *A* apresenta a tarefa para todos que estão na sala. Então, as pessoas, na sala, começam a se comunicar para estabelecer a divisão da tarefa. Uma vez decidido, iniciam seu trabalho. Quando *A* retorna, a parede está pintada, mas ele não tem idéia sobre a execução da tarefa. "

Pontos importantes nesta abordagem:

- A decomposição da tarefa é feita pelos agentes e não pelo usuário ou projetista. Pode ocorrer uma reorganização dinâmica, na qual os agentes podem decidir se eles devem trocar seu comportamento para melhor realizarem suas tarefas;

- Os agentes são autônomos, possuindo suas próprias metas locais. No exemplo, o agente *C* pode recusar em cooperar com os outros, se ele achar que é mais importante para ele dedicar-se a outra tarefa;

- Se o agente *D* possui algum conhecimento sobre grupos de dança americanos e os outros podem cantar e tocar algum instrumento musical, os próprios agentes serão capazes de formar um grupo de dança. No entanto, agentes são capazes de

- O ambiente pode mudar e os agentes devem incorporar essas mudanças em seu modelo interno do ambiente.

5.2 Simulação

Muitas aplicações de SMAs são desenvolvidas para simular alguma situação da realidade. *Simulação* é a tentativa científica que consiste em realizar uma reprodução artificial, chamada *modelo*, de um fenômeno real que se deseja estudar e observar o comportamento [DRO 93].

A simulação pode ser dividida em três etapas:

- Etapa de modelagem: consiste em construir o modelo do fenômeno a ser estudado. No caso de SMAs, modelar o problema como um conjunto de agentes;
- Etapa de experimento: consiste em aplicar variações sobre o modelo construído, alterando parâmetros que influam no processo de resolução;
- Etapa de validação: consiste em comparar os dados experimentais, obtidos com o modelo, com a realidade. É a análise dos resultados.

A figura 5.1 ilustra as etapas de um processo de simulação. A partir da realidade, faz-se uma modelagem da situação/problema desejado, que pode envolver coleta de dados, e constrói-se um modelo. A construção do modelo é fundada sobre uma teoria. Com o modelo construído, parte-se para a simulação e após para a avaliação do modelo. A avaliação é feita com os resultados obtidos pelo modelo e com as observações da realidade.

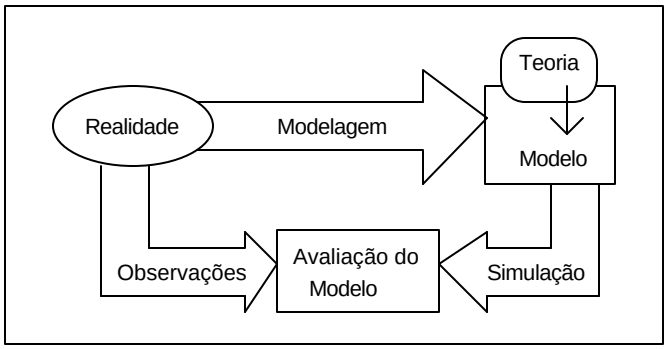


FIGURA 5.1 - Etapas de um Processo de Simulação

Na simulação multiagente, o fenômeno real é decomposto em um conjunto de elementos e em suas interações. Cada elemento é modelado como um agente e o modelo geral é o resultado das interações entre estes agentes.

5.2.1 Abordagem Reativa para a Simulação Multiagente em Etologia

A etologia é um domínio de aplicação que pode utilizar a modelagem multiagente reativa para simular seus processos. Tanto a etologia quanto a abordagem reativa baseiam-se no comportamento de suas entidades.

O comportamento dos animais é o núcleo de estudo da etologia, permitindo verificar as diferenças e analogias com o comportamento humano e estabelecer a relação do homem com a natureza.

A sociologia animal estuda as organizações de indivíduos capazes de adaptarem-se a seu ambiente, capazes de resolverem problemas e capazes de reagirem a modificações que ocorrem em seu ambiente. Modelar uma organização animal como um SMA é permitir a geração de sociedades artificiais de agentes. No caso da etologia, os agentes são os animais e os comportamentos dos animais são as ações comuns a todos os membros de uma mesma espécie, sob restrições de anomalias fisiológicas. Um agente recebe estímulos e age em resposta aos mesmos, que podem ser internos (provém do próprio agente, como hormônios) e externos (provém do ambiente e de outros agentes, como luz, humidade).

5.3 Algumas Aplicações em Sistemas Multiagentes

Esta seção visa mostrar alguns trabalhos desenvolvidos com a utilização de agentes atuando em conjunto em um ambiente. Os agentes envolvidos são cognitivos e reativos. O objetivo é ressaltar que cada vez um número maior de problemas, em diferentes áreas, está sendo resolvido com a modelagem das aplicações em SMAs.

5.3.1 O Sistema SODABOT

O sistema SODABOT (A Software Agent Environment and Construction system) é descrito em [COM 94] e trata com a criação de agentes de software. Os agentes de

software, neste sistema, são divididos em dois tipos: assistentes pessoais e agentes de aplicação. O objetivo da criação destes agentes é automatizar tarefas que consomem tempo e que são repetitivas. Os assistentes pessoais agem como secretárias eletrônicas, efetuando tarefas tais como: marcar reuniões, consultar compromissos, filtrar mensagens eletrônicas de acordo com as preferências de seu usuário, entre outras. Por outro lado, os agentes de aplicação coordenam a transferência e o processamento de informações entre pessoas e outros agentes, incluindo sistemas de processamento de textos e agentes recepcionistas. Este sistema dispõe de uma linguagem de programação agente (SodaBotL), usada para descrever as atividades dos agentes, que oferece ao usuário uma interface gráfica, e de um sistema operacional agente, o BSA (Basic Software Agent) que roda em background, atuando como um suporte computacional para implementar aplicações específicas de agentes.

5.3.2 Modelagem de Agentes

Alguns trabalhos ressaltam a necessidade de desenvolver um método de resolução de problemas no qual os agentes humanos e os artificiais distribuam responsabilidades em um ambiente multiagente. Em [MOU 95] é apresentada uma abordagem de modelagem de SMAs sob a forma de cenários, compostos de agentes reativos, destinados a suportar o trabalho colaborativo com a integração de agentes artificiais e humanos. Os agentes possuem diferentes responsabilidades, capacidades e funcionalidades. Os agentes artificiais são autônomos, realizando suas tarefas independentemente e gerando suas bases de dados locais. Quando precisam de ajuda, iniciam atividades de grupo, comunicando-se com outros agentes. Existe também uma base de dados, comum a todos os agentes, que representa o ambiente no qual eles agem. Os cenários são ambientes de trabalho em grupo, nos quais os agentes podem desempenhar várias funções e seus comportamentos dependem dessas funções.

Como exemplo, é citado um cenário que permite a um grupo de indivíduos determinar uma data de reunião: cada agente humano está associado a um agente artificial que desempenha a função de secretário, administrando a agenda de seu usuário e negociando com os outros agentes. O cenário global é um usuário que deseja realizar uma reunião com diversas pessoas. O usuário fornece a seu agente secretário os parâmetros da reunião (motivo da reunião, lista de pessoas a convidar para participar, duração da reunião, período a considerar para marcar a reunião). A partir disso, o agente artificial desempenha a função de secretário de modo autônomo, comunicando-se com os agentes secretários das pessoas a serem convidadas e esperando confirmação para iniciar a negociação com os agentes interessados, a fim de encontrar um horário adequado para a reunião. Este cenário, então, divide-se em quatro fases:

- Fase inicial: o agente inicial contata com os demais da lista de convidados para propor a participação na reunião. O agente inicial espera a resposta dos demais e cria uma lista de agentes convidados.
- Fase de determinação de funções: cada agente verifica o período proposto para a reunião e comunica suas restrições.

- Fase de negociação: nesta fase, o agente mais restrito propõe seus períodos de disponibilidade e de compromissos e é verificado se um horário livre é comum a

- Fase de término da negociação: o horário mais apropriado é comunicado ao agente inicial que fixa a reunião e avisa a cada um dos interessados.

O método baseado em cenário multiagente possui três fases:

- Analisar e modelar o domínio de aplicação e os componentes do SMA a um nível conceitual: envolve a descrição de cenários, a modelagem de agentes e a modelagem de conversação (interação entre os usuários e os agentes artificiais).
- Transformar os modelos do nível conceitual em especificações que detalham o comportamento do agente sob a forma de diagramas de transição. Estes diagramas correspondem a uma máquina de estados finita, que representa as trocas de estado realizadas no comportamento do agente.
- Testar e integrar os componentes no SMA.

5.3.3 Sistema Multiagente em Epidemiologia

As aplicações em SMAs estão expandindo-se para diversas áreas de estudo, como ciências, biologia, saúde, automação industrial e outras. Um modelo de SMA para aplicações na epidemiologia, que estuda as doenças e os fatores de saúde em uma população, é descrito em [DUR 95]. O modelo aborda um ramo da epidemiologia que é a epidemiologia operacional, estudando também os dispositivos de luta contra as doenças.

5.3.4 O Projeto MANTA : Simulação da Organização Social de uma

Este projeto é descrito em [DRO 93] e simula uma aplicação na área etológica que é o comportamento de uma colônia de formigas.

Inicialmente, foi feito um estudo sobre as formigas. As formigas apresentam dois traços predominantes:

- colonizam quase todas as áreas do planeta, com exceção de zonas glaciais e ambientes marinhos;
- são animais sociais (não existem formigas solitárias).

Distinguem-se duas grandes categorias de indivíduos em uma colônia de formigas:

- os indivíduos sexuados, machos e fêmeas (as formigas rainhas);
- as fêmeas não reprodutoras (as formigas soldados).

O projeto MANTA envolve dois domínios de pesquisa: a compreensão dos modos de organização de vida (etologia) e a síntese de organizações artificiais funcionais (computação). O interesse está na metodologia multiagente para efetuar simulações em etologia, biologia ou ecologia.

A primeira etapa da modelagem constituiu-se da decomposição de uma colônia de formigas em um conjunto de agentes, com uma representação gráfica e um nome, resultando

⇒ os ovos, as larvas e os casulos

⇒ a rainha, as operárias e os machos

⇒ os alimentos

Após, são descritos os comportamentos associados a cada um dos tipos de agentes da simulação. O ato de sair para procurar alimento é motivado pelas necessidades nutricionais das larvas, das formigas operárias e pela presença de alimento. Os estímulos podem ser próprios do agente ou provir do ambiente.

A plataforma de simulação é composta de duas partes : uma de modelagem e outra

5.3.5 ARCHON (Architecture for Cooperating Heterogeneos ON-line systems)

É uma arquitetura para controle de processos industriais, descrita em [ROD 90]. A motivação é a construção de uma aplicação industrial não limitada a um domínio específico. O sistema é composto por quatro partes:

- um módulo de comunicação;
- um módulo de coordenação;
- um módulo de cooperação;
- um módulo de controle.

Outras aplicações de SMAs na área industrial, como o GRATE, podem ser encontradas em [JEN 94].

6 Modelos de Sistemas Multiagentes Reativos

Os modelos de sistemas multiagentes reativos permitem que se faça a modelagem de aplicações nas quais o processo de resolução dos problemas, representados por tais aplicações, parece se adequar às características dos agentes reativos.

Neste trabalho são abordados três modelos de sistemas multiagentes reativos, que foram estudados e analisados com o objetivo de ressaltar as principais características de cada modelo. Estas características foram usadas para montar um quadro comparativo entre tais modelos (ver seção 6.4) e para determinar os elementos essenciais constituintes do protótipo de um ambiente de desenvolvimento de aplicações em SMA reativos.

A descrição de cada modelo e a comparação entre eles são apresentadas nas seções seguintes. Os exemplos de aplicações descritos, usando os modelos, foram implementados, com exceção do Modelo PACO, visando uma melhor compreensão do funcionamento dos mesmos.

6.1 Modelo da Funcionalidade Emergente

Este modelo é descrito em [STE 90] sob a forma de um problema, com as seguintes

" Um planeta distante possui depósitos de minerais os quais devem ser encontrados e os minerais levados a uma base central, também situada neste planeta. O ambiente é desconhecido e não há um mapa detalhado do terreno. A única informação presente é que o terreno possui vários obstáculos. Como solução para este problema, pensou-se em utilizar um conjunto de robôs móveis que explorassem o terreno e executassem as tarefas de busca e transporte dos minerais. "

O problema apresentado foi mapeado para um sistema multiagente reativo, através de uma representação analógica, na qual o ambiente (terreno) é representado por uma grade com duas dimensões, a posição do agente no ambiente é indicada pelo valor do par (linha,coluna) desta grade, e a base central, o mineral e o robô são os três tipos de agentes do sistema. A base central e os minerais são agentes fixos no ambiente (não trocam de posição), enquanto que os robôs são agentes reativos, dotados de movimento e baseados em comportamento. A figura 6.1 abaixo mostra a representação dos agentes neste modelo.

A. Comportamento de movimento

- escolher randomicamente uma direção para o movimento
- mover-se na direção escolhida

B. Comportamento de manuseio

- se perceber um mineral e não estiver carregando um, deve pegá-lo
- se perceber a base central e estiver carregando um mineral, deve descarregá-lo

C. Comportamento de evitar obstáculos

- se perceber um obstáculo à frente, efetuar um giro aleatório

Com isso, verifica-se as fases do comportamento de um robô como sendo:

2. Coleta de mineral
3. Retorno à base

4. Procura aleatória

Estas fases representam uma solução em que os robôs encontrarão os minerais e retornarão à base central, mas o tempo gasto para isso (tempo de busca do mineral + tempo de transporte do mineral até a base) pode ser muito grande. Os robôs não dispõem de nenhum mecanismo que lhes permita ter uma memória, para retornarem ao lugar da sua última recuperação de mineral. Desta forma, os robôs não apresentam uma funcionalidade emergente como uma ação coletiva eficiente, porque não há circulação de informação sobre a presença de mineral entre os membros do grupo de agentes.

Em uma segunda solução, estabelecida para o problema, o ambiente é utilizado como um artifício de memória para a recuperação de minerais. O robô, após carregar um mineral, retorna à base central, deixando marcas para constituir um caminho, como fazem as formigas que deixam um rastro químico. Todo robô poderá, acidentalmente, cruzar este caminho e segui-lo até chegar ao depósito de minerais, reduzindo seu tempo de busca. No caso de robôs, este caminho pode ser composto de pequenos fragmentos, detectados por um sensor. Isso é conhecido como um mecanismo de amplificação da informação [DRO 93].

As novas fases do comportamento de um robô, nesta solução, são:

2. Coleta de mineral

3. Retorno à base

4. Retorno ao mineral

Foi utilizado o conceito de gradiente associado à base central, servindo como guia para os robôs retornarem a mesma. O gradiente é um sinal de intensidade decrescente com a distância, emitido pela base central e que é percebido pelos robôs. A cada posição distante da base central, o valor é menor de uma unidade. Os robôs podem estar em um de dois modos:

1. Modo de Exploração : neste modo o robô está a procura de mineral e afasta-se da fonte do sinal (base central), seguindo na direção do menor gradiente.

2. Modo de Retorno : neste modo o robô volta para a base central, levando o mineral e seguindo na direção do maior valor do sinal.

Estes modos determinam o seguinte comportamento aos robôs : quando um robô estiver carregando um mineral, ele estará em modo de retorno e seguirá a direção do sinal crescente para chegar diretamente à base. Neste modo, o robô também deixará uma marca em cada posição, na qual vai passando, para traçar um caminho de retorno ao depósito de minerais encontrado. Quando o robô descarregar o mineral na base central, ele entrará em modo de exploração e seguirá as marcas deixadas até chegar novamente ao depósito de minerais.

Esta segunda solução melhora o desempenho de resolução do problema. Esta melhora ocorre em relação ao tempo de busca de minerais, com o uso de marcas para estabelecer um caminho, e em relação ao tempo de retorno à base, com o uso do gradiente.

Os robôs continuam sendo agentes simples, sem necessidade de adquirirem características cognitivas (como raciocínio e planejamento).

6.1.1 “Subsumption Architecture”

A "subsumption architecture" de Brooks, segundo [STE 90], estabelece prioridades entre a execução dos comportamentos atribuídos aos agentes. Esta arquitetura é usada, neste modelo, para hierarquizar os comportamentos de movimento e de evitar obstáculos. A figura 6.2 de uma "subsumption architecture" indica que, quando há um obstáculo, o comportamento de evitar obstáculos (que trata de colisões) precede o de movimento randômico.

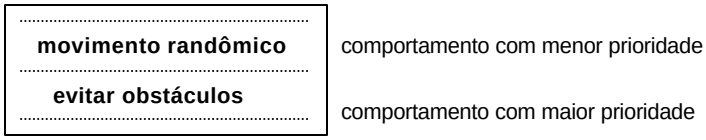


FIGURA 6.2 - "Subsumption Architecture" com dois comportamentos

Acrescentando-se os novos comportamentos aos agentes, pela segunda solução do problema descrito anteriormente, a "subsumption architecture" entre os comportamentos de movimento é mostrada na figura 6.3.

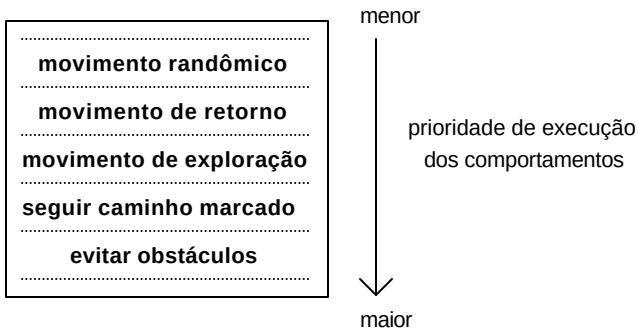


FIGURA 6.3 - "Subsumption Architecture" entre Comportamentos de Movimento

A arquitetura acima estabelece que o movimento de evitar obstáculos tem maior prioridade sobre todos os outros. Após, se houver uma trilha já determinada, o agente deverá seguir pela mesma, efetuando o seu movimento de exploração. Quando o agente estiver de posse do mineral, realizará o seu movimento de retorno. O movimento randômico de busca é o de menor prioridade a ser executado pelo agente.

6.1.2 Funcionalidade Emergente

Este modelo ilustra o conceito de *funcionalidade emergente*, que exprime a passagem da ação a nível individual de cada agente reativo para uma ação coletiva. Esta ação coletiva é obtida através do resultado das atividades dos agentes e das interações entre os mesmos, evoluindo para um comportamento em grupo inteligente.

A idéia que a funcionalidade emergente expressa é que um sistema com muitos componentes simples pode exibir um comportamento, como um todo, mais organizado do que o comportamento das partes individuais.

[STE 90] coloca que a funcionalidade está relacionada com o conceito de auto-organização de uma sociedade. A auto-organização é a capacidade que uma população de entidades (neste caso, um grupo de agentes) possui de se estruturar espacial e temporalmente em um ambiente. É uma modificação na topologia de um grupo de agentes, realizada de modo autônomo, permitindo a estes agentes se adaptarem ao seu ambiente através de uma nova organização. Pode-se considerar a estrutura da sociedade de agentes como a sua organização, e a realização das tarefas, por parte dos agentes, como a sua funcionalidade.

Aplicando este conceito no exemplo dos robôs, a auto-organização aparece quando os robôs estabelecem um caminho entre os depósitos de minerais e a base central. Isto indica uma forma de interação entre os agentes que tem como resultado uma estrutura espacial (o caminho). A funcionalidade emergente aparece no comportamento dos agentes de acidentalmente cruzarem o caminho já estabelecido e segui-lo para a coleta de minerais, diminuindo, assim, o seu tempo de busca. A emergência pode ser considerada, segundo [DRO 93], como um mecanismo de amplificação da informação.

- Porque os robôs, neste exemplo, são considerados agentes reativos ?

O robô (agente) não tem conhecimento nem do ambiente nem da presença e características dos outros robôs. Ele apenas apresenta comportamentos que indicam a reação a determinados estímulos e que podem ser vistos como um conhecimento próprio. Além disso, cada robô trabalha independente dos outros e não elabora planos de ação para executar suas tarefas e atingir a meta desejada, porque não raciocina neste sentido. Não há intencionalidade nas ações dos agentes, os quais apenas reagem a determinados estímulos. A comunicação é estabelecida através do ambiente, sendo difundida para todos os agentes, e não através de um canal único entre determinados agentes.

- Como são representadas as características dos agentes reativos nos robôs ?

Percepção : os robôs têm capacidade de percepção. No exemplo, eles percebem o sinal que é emitido pela base central, visando facilitar a volta dos robôs para a mesma, após encontrarem o mineral, e percebem os caminhos, estabelecidos pelos próprios robôs, através de marcas, ligando a base central aos depósitos de minerais encontrados. Também percebem a presença de mineral, ao chegar próximo a este.

Planejamento : os robôs não têm capacidade de estabelecer planos para atingir os objetivos, porque eles simplesmente reagem de maneira pré-definida a certos estímulos e não raciocinam em função de uma situação específica.

Comunicação : os robôs que encontram o mineral deixam marcas que constituem um caminho de retorno e que facilitam a tarefa de busca dos robôs que ainda procuram pelos depósitos de minerais. Isto é uma forma de comunicação, utilizando o ambiente como meio de transmissão da mensagem. Não é uma comunicação intencional por parte dos robôs, eles apenas são programados para agirem assim.

Estímulo-Resposta : os agentes reativos possuem a característica de agirem segundo um estímulo. O estímulo é uma conjugação de percepção externa - o que ele percebe do ambiente - e de percepção interna - o seu próprio estado - por exemplo, estar ou não carregado com mineral. A situação em que o robô se encontra representa o estímulo. A ação do robô de efetuar busca ou transporte de mineral representa a resposta.

6.2 Modelo PACO

PACO (Coordination Patterns) é um modelo, apresentado em [DEM 91] e [DEM 93], utilizado em aplicações espacializadas, como análise de imagens para a detecção de contornos e como a cartografia. Também se aplica à robótica para o controle de execução de

6.2.1 O Agente Reativo no Modelo PACO

O modelo envolve um conjunto de agentes reativos que atuarão conforme o domínio da aplicação. Cada agente deste conjunto assume comportamentos definidos a priori e representa uma solução parcial do problema. Tal solução corresponde à posição de cada agente no ambiente. A solução global do problema é dada pela posição do conjunto de agentes.

Uma solução para o problema é encontrada quando o conjunto de agentes atinge um estado de equilíbrio, que é a posição estável de cada um dos agentes no ambiente. Esta posição estável é alcançada quando os agentes não se movem mais ou se movem muito pouco, abaixo de um certo limiar aceitável pela aplicação.

A detecção de que uma solução foi encontrada é feita por um observador externo e não pelos próprios agentes, porque cada agente isoladamente não sabe se os outros agentes estão em uma posição estável ou não. Este observador externo pode ser o usuário do sistema ou um agente supervisor, com características diferentes dos outros agentes do sistema.

Neste modelo é introduzida a noção de campo que é o subconjunto do ambiente visível ao agente (a sua vizinhança). Para abordar tal noção, foram definidos três campos, para cada agente, que operacionalizam seu comportamento:

- Campo de percepção : guia as interações entre os agentes e o ambiente. É o subconjunto do ambiente que o agente pode perceber em um determinado momento. Por exemplo, o conjunto de agentes que o agente em questão sabe que existe.

- Campo de comunicação : guia as interações entre os agentes. É o subconjunto de agentes com quem o agente em questão pode se comunicar e que o influenciarão

- Campo de ação : guia as ações de deslocamento do agente no ambiente. É o conjunto de agentes que podem sofrer uma ação do agente em questão.

Os agentes não possuem representação de si mesmos, nem dos outros agentes nem do ambiente. O que eles possuem é percepção do ambiente e de seus agentes vizinhos. Tal percepção ocorre através de uma representação física e não de uma representação simbólica interna, como uma base de conhecimento do agente, contendo fatos que indicariam a existência de outros agentes, com tipos de problemas que eles resolveriam, e que descreveriam o ambiente. A representação física do ambiente é comum a todos os agentes. Por exemplo, em uma aplicação de análise de imagens, a própria imagem seria a representação do ambiente para os agentes e o que um agente perceberia, em relação aos outros, é se os pixels ao seu redor conteriam ou não um agente.

6.2.2 A Arquitetura do Modelo PACO

É uma arquitetura do tipo reativa, na qual o comportamento de um agente é caracterizado pelo conjunto de interações do mesmo. Tais interações, com o ambiente ou com os outros agentes, estão associadas com a capacidade de percepção do agente em um determinado momento.

Os elementos que compõem esta arquitetura são : o ambiente; os agentes, com uma possível organização inicial no ambiente; as interações entre agentes; as interações entre agentes e seu ambiente. A cada elemento do ambiente, chamado de entidade ambiental [DEM 93], estão associados valores que correspondem aos dados iniciais do problema. Por exemplo, em uma imagem tem-se valores iniciais, associados à cor de cada pixel desta imagem.

Os agentes são similares e variam o seu comportamento de acordo com o seu estado interno e de seu ambiente, buscando uma situação de equilíbrio. Cada agente possui atributos como massa, posição, velocidade e aceleração, já que as interações são baseadas em modelos físicos de forças. A massa representa a importância do agente no sistema e de sua solução parcial sobre a solução global do problema, que poderá provocar uma força de repulsão ou de atração entre os agentes. Os atributos posição, velocidade e aceleração referem-se à estabilidade dos agentes no ambiente para atingirem sua solução parcial.

[DEM 93] estabelece um Ciclo de Resolução que especifica o comportamento dos agentes. Este ciclo apresenta três etapas:

- Regulagem e Aquisição : nesta etapa são determinados os campos de percepção (por exemplo, o número de pixels considerados na análise de uma imagem) e de comunicação (por exemplo, o número de agentes envolvidos na

- Tratamento mediante o valor dos campos : nesta etapa realiza-se o cálculo das interações com o ambiente (percepção) e entre os agentes (comunicação) e o cálculo das forças exercidas sobre o agente.

- Regulagem e Ação : nesta etapa determina-se a ação de deslocamento e cálculo da nova posição do agente no ambiente, em função das forças calculadas.

Programar uma aplicação PACO consiste em instanciar o seguinte esqueleto:

- ambiente;
- agentes (número de agentes, atributos de cada agente e seus campos);
- organização da sociedade (opcional);
- interações com o ambiente (definir forças externas);
- interações entre agentes (definir forças internas);
- ações no ambiente;
- ciclo de resolução;
- exploração de resultados (como um observador externo pode extrair a

6.2.3 Exemplo de Aplicação

Uma aplicação de sistemas multiagentes reativos para a generalização cartográfica, utilizando a arquitetura PACO, é descrita em [BAE 95]. A generalização cartográfica consiste em criar um mapa a partir de dados geográficos, levando em conta a representação gráfica. Nesta aplicação, um objeto geográfico é representado por um conjunto de agentes e um único agente corresponde a um ponto geográfico neste objeto. O agente não pertence a

O exemplo seguinte explica o que é essa generalização cartográfica: considerando um mapa em uma escala X, se quer mudar a escala deste mapa para que ele seja representado na metade do seu tamanho original. Nesta transformação podem ocorrer problemas, por falta de espaço, para representar graficamente todos os elementos do mapa. Se no mapa houver um rio que passa ao lado de uma estrada, pode acontecer que, no mapa em escala menor, os dois elementos ficassem colados ou sobrepostos. Neste caso, poderia-se deslocar um pouco o rio ou a estrada da sua posição exata para melhorar a legibilidade do mapa.

Mapeando este exemplo para um sistema multiagente no modelo PACO, cada agente tem associado o atributo massa que é função da importância do agente no mapa. Se

dois agentes (objetos geográficos) estão muito próximos um do outro, de modo que suas representações gráficas no mapa ficariam em parte sobrepostas, ocorrerá uma força de repulsão entre estes agentes (esta força de repulsão modela um tipo de interação entre agentes), que os fará se deslocarem no mapa. Este deslocamento será realizado em função das massas dos objetos - o mais importante, por exemplo um rio ou uma estrada, se deslocará menos que um menos importante, como uma casa. Para isso, utiliza-se as fórmulas da física que tratam do deslocamento de corpos em função de sua massa.

6.3 Modelo Eco-Resolução

O modelo da Eco-Resolução, descrito em [FER 91], baseia-se em agentes com apenas dois tipos de comportamento: de satisfação e de fuga. Este modelo aplica-se à representação de um sistema multiagente reativo cujo domínio de aplicação é a etologia, que estuda o comportamento dos animais e as suas reações em relação às condições ambientais. [DRO 93], em sua tese, modelou um sistema que simula o comportamento de uma colônia de formigas, a partir de um modelo que ele chamou de Etho-Resolução, que pode ser visto como uma extensão do modelo da Eco-Resolução.

6.3.1 O Eco-Agente

Um eco-agente possui conhecimento local de seu próprio ambiente e comportamentos que modificam este conhecimento. O conhecimento local do agente depende das características do problema. Em algumas aplicações, o agente poderá ter mais informações sobre o ambiente do que em outras. Já o comportamento dos agentes é independente do domínio da aplicação.

Quanto ao conhecimento local, um eco-agente sempre conhece :

- o seu estado de satisfação: quando o agente atinge a sua meta é verdadeiro e,

- a dependência temporária de um agente em relação a outro (relacionamento mestre-escravo). A alocação de uma meta para um agente cria um relacionamento mestre-escravo entre o agente e a sua meta;

- os agentes que o impedem de agir.

A seguir, utiliza-se um exemplo, visto em [FER 91], para ilustrar esses conhecimentos do eco-agente. A figura 6.4 descreve um problema no qual o robô deve trocar sua posição, do lugar 1 para o lugar 4, e atingir sua meta - o café. Isto torna o robô escravo

Quanto aos comportamentos, os agentes possuem dois:

- Satisfação : descreve o comportamento de um agente que procura se satisfazer. No caso do exemplo acima, o robô deve atingir o café para estar satisfeito. Corresponderá ao estado final de um problema;
- Fuga : descreve o comportamento de um agente para 'fugir' de seu estado atual em função da pressão de outros agentes. É a resposta a uma agressão, fazendo o agente trocar a sua posição no ambiente para evitar conflitos com outros agentes.

O agente possui o estado de liberdade que é responsável por possibilitar a ação do agente. Este estado não troca a posição do agente no ambiente, porque um agente não age para obter sua liberdade. Se um agente não estiver livre para agir, os outros agentes que impedem a liberdade desse é que vão agir (realizar o comportamento de fuga).

6.3.2 Estados Internos do Eco-Agente

Os comportamentos (ações) são simples e baseados no estado interno dos agentes. O estado interno reflete o posicionamento do agente no ambiente, por exemplo, se ele está

O estado interno de um agente é representado pela tupla (s, f, l), onde 's' é o estado de satisfação, 'f' é o estado de fuga e 'l' é o estado de liberdade.

- s = 0, o agente não está satisfeito; s = 1, o agente está satisfeito

- f = 0, o agente não está em fuga; f = 1, o agente está em fuga
- l = 0, o agente não está livre; l = 1, o agente está livre

Por exemplo, a tupla (1,0,1) indica o estado interno no qual o agente está satisfeito, não está em fuga e está livre para agir. Um agente não pode estar em estado de fuga e de satisfação ao mesmo tempo. O estado inicial de um agente é representado pela tupla (0,0,0), que corresponde ao estado interno no qual o agente não está satisfeito, não está em fuga e não está livre para agir. Neste caso, o agente deve, primeiramente, obter o estado de liberdade para poder, então, agir.

O comportamento é uma ação sobre o estado interno do agente. Por exemplo, o 'estar satisfeito' passa a satisfação do estado 0 (zero) para o estado 1 (um). Portanto, uma ação sobre um estado interno gera outro estado interno.

6.3.3 O Eco-Problema

Um eco-problema é um problema que usa eco-agentes para atingir a sua solução. É composto por três unidades, representadas pela tupla (P, Si, Sf):

- 'P' é a população de agentes que agem para atingir a situação final do problema;
- 'Si' é a situação inicial do problema, que descreve as posições dos agentes no ambiente;
- 'Sf' é a situação final do problema, que cria os relacionamentos mestre-escravo entre os agentes e a meta que lhes é dada.

A solução de eco-problemas fundamenta-se nas interações entre os agentes. Os agentes devem interagir para se ajudarem a atingir o estado de satisfação, descrito na situação final do problema. Para exemplificar uma interação entre agentes, utiliza-se o exemplo do robô que deve chegar ao café para atingir seu estado de satisfação. Ocorre uma interação entre a porta e o robô - a porta tem que passar do seu estado de satisfação (fechada) para o estado de não satisfação (aberta) para que o robô esteja livre para atingir sua meta (passar

Uma aplicação que utiliza o modelo da eco-resolução para representar e resolver seu problema é descrita em [DRO 93a]. A aplicação é um jogo chamado PENGI.

6.3.4.1 Descrição do Problema

Um pingüim (controlado pelo jogador) move-se em um labirinto feito de cubos de gelo e diamantes e habitado por abelhas. O pingüim deve coletar todos os diamantes, enquanto evita ser picado por uma abelha ou esmagado por um cubo. O pingüim pode matar as abelhas, empurrando contra elas os cubos de gelo.

6.3.4.2 Modelagem do Problema com Eco-Resolução

A implementação baseia-se na solução de eco-problemas. Os agentes possuem uma percepção local de seu ambiente, um comportamento de satisfação para atingir sua própria meta individual e um comportamento de fuga, usado no caso de agressão. Os agentes que fazem parte do sistema são os diamantes, os cubos de gelo, as abelhas e o pingüim.

A descrição dos estados internos de cada agente segue abaixo:

- os diamantes estão satisfeitos e não podem fugir, portanto não possuem comportamento;
- os cubos de gelo não executam o comportamento de satisfação. Quando são atacados, por uma abelha ou por um pingüim, eles realizam o comportamento de fuga na direção oposta da agressão, ficando sua posição anterior livre. Durante esta fuga, um cubo pode entrar na área de percepção do pingüim e, então, atacá-lo ou pode esmagar uma abelha que entrou em sua posição anterior.
- as abelhas possuem uma área de percepção parametrizável (a distância na qual elas conseguem ver o pingüim). Seu comportamento de satisfação é duplo: se o pingüim mover-se na área de percepção da abelha, a abelha se moverá na direção do pingüim (que é sua meta); senão as abelhas se moverão randomicamente nesta área, escolhendo uma posição. As abelhas também possuem dois comportamentos agressivos: um contra os cubos de gelo e outro contra o pingüim.
- o comportamento de satisfação do pingüim consiste em atingir o diamante mais perto. Seu comportamento de fuga é gerado por uma agressão vinda de uma abelha ou de um cubo de gelo. Empurrar um cubo de gelo é o único comportamento

6.3.4.3 Comportamento do Sistema

Como o comportamento do pingüim é no sentido de mover-se na direção do diamante mais perto, ele tem que ter a informação da posição dos diamantes que ele ainda não atingiu. Esta informação é fornecida pelo gradiente, sinal gerado pelos diamantes. Estes sinais não representam uma estratégia, porque:

- apenas fornecem caminhos alternativos para chegar a cada diamante e não forçam o pingüim a seguir por estes caminhos;
- não coordenam as ações que o pingüim terá que executar para atingir um diamante. Por exemplo, não mostram como o pingüim deve agir quando encontrar uma abelha no caminho.
- apenas fornecem ao pingüim uma indicação do diamante mais próximo.

6.4 Taxonomia dos Modelos

Para estabelecer uma comparação entre os modelos de sistemas multiagentes reativos estudados, foram ressaltadas características de cada um, mostrando suas diferenças e similaridades. A tabela 6.1 mostra o quadro de comparação entre os modelos.

Algumas características apenas se diferenciam a nível conceitual, podendo ser modeladas da mesma forma. Isto pode ser percebido entre os modelos da Funcionalidade Emergente e da Eco-Resolução. Para ilustrar a relação entre as características desses dois modelos, descreve-se, a seguir, como seria a representação do problema dos robôs, abordado no modelo da Funcionalidade Emergente (seção 6.1), através dos conceitos utilizados no modelo da Eco-Resolução.

A representação do problema pela eco-resolução deve ser composta por uma:

- população de agentes: robôs, base central e minerais;
- situação inicial: posição dos robôs, da base central e dos minerais no ambiente;
- situação final: os robôs terem encontrado e carregado todos os minerais para a base central.

As outras características são relacionadas como:

- estado de satisfação: é quando o agente atinge sua meta. Para atingir sua meta, o agente realiza comportamento de movimento. O robô quando está em modo de exploração e atinge um mineral (sua meta), muda para modo de retorno (levar o
- estado de fuga: neste estado o agente troca de posição no ambiente. É, portanto, um comportamento de movimento realizado pelo agente;
- agentes que impedem a ação de outros agentes: são os obstáculos dos quais os agentes devem desviar, buscando sua liberdade para agir. Os robôs desviam de obstáculos quando movimentam-se no ambiente;
- dependência (relacionamento mestre-escravo entre o agente e sua meta): não existe explicitamente nos robôs, mas o fato do robô ter que encontrar o mineral é uma dependência, como o pingüim que quer chegar aos diamantes, no exemplo

O modelo PACO utiliza alguns conceitos diferentes em relação às características dos agentes e de seus comportamentos. Os agentes devem possuir atributos físicos para que as interações sejam calculadas. Além disso, deve ser definido para cada agente seus campos de percepção, comunicação e ação, para que os mesmos atuem conforme o Ciclo de Resolução determinado pelo modelo.

TABELA 6.1 - Comparação entre Modelos de Sistemas Multiagentes Reativos

	MODELOS DE SISTEMAS MULTIAGENTES REATIVOS		
Características	**Funcionalidade Emergente**	**PACO**	**Eco-Resolução**
Agentes	. são reativos . possuem estados internos . se auto-organizam em seu ambiente . suas atividades emergem para uma ação coletiva	. são reativos . possuem estados internos . podem ter uma organização inicial . possuem atributos (massa, posição, ...)	. são reativos . são baseados no comportamento de animais . possuem estados internos em relação a sua satisfação, fuga e liberdade
Ambiente dos Agentes	. local de atuação dos agentes determinado pela aplicação . possui uma representação física	. local de atuação dos agentes determinado pela aplicação . possui uma representação física	. local de atuação dos agentes determinado pela aplicação . possui uma representação física
Interações	. entre agentes . entre agentes e ambiente . associadas à percepção do agente	. entre agentes . entre agentes e ambiente . associadas à percepção do agente . baseadas em modelos físicos de forças	. entre agentes . entre agentes e ambiente . associadas à percepção do agente
Comportamentos	. representam ações dos agentes . variam conforme o estado interno do agente . ligados à percepção dos agentes . baseados na *Subsumption architecture*	. representam ações dos agentes . variam conforme o estado interno do agente . caracterizados pelas interações dos agentes . utilizam a noção de campos (vizinhança) dos agentes: de percepção, ação e comunicação . especificados por um Ciclo de Resolução	. representam ações dos agentes . variam conforme o estado interno do agente . possui os comportamentos de satisfação e de fuga
Solução do Problema	. é a situação final do problema, quando os agentes realizaram suas tarefas	. é encontrada quando os agentes atingem um estado de equilíbrio . gerada pelas soluções parciais de cada agente, que correspondem às posições de cada agente no ambiente	. é a situação final do problema, quando os agentes atingem o seu estado de satisfação

7 Um Ambiente para Desenvolvimento de Sistemas Multiagentes Reativos

Este capítulo apresenta uma descrição dos sistemas SIEME e SWARM, utilizados para o desenvolvimento de sistemas multiagentes, seguido da proposta de um novo ambiente para o desenvolvimento de sistemas multiagentes reativos, denominado SIMULA.

7.1 Exemplos de Ambientes de Desenvolvimento de Sistemas Multiagentes

Nas seções seguintes serão apresentadas as principais características de dois ambientes projetados para desenvolver sistemas multiagentes reativos.

7.1.1 O Sistema SIEME

O sistema SIEME - SImulateur Evènementiel Multi-Entités [MAG 96], é um simulador para sistemas multiagentes, desenvolvido por Laurent Magnin, do laboratório Laforia da Universidade Paris 6. O objetivo deste sistema é facilitar a descrição de um modelo multiagente ou mais precisamente o ambiente e as propriedades físicas dos agentes

O sistema SIEME é totalmente integrado ao sistema de programação Smalltalk 80, de modo que uma aplicação SIEME pode ser executada sem modificação em várias plataformas (Macintosh, Sun/Unix, PC, ...). O sistema foi utilizado no projeto Microb [MAG 95] que visa estudar as interações entre robôs móveis.

A especificação do problema (o modelo da simulação) é realizada de forma declarativa, representando os dados sob uma forma matemática. Esta especificação contém a

- das entidades envolvidas
  - criação da classe
  - atributos internos e externos
  - definições dos atributos
  - os sensores e acionadores

```
SimulSIEME subclass: #SimulationPerso
        instanceVariableNames: ''
        classVariableNames: ''
        poolDictionaries: ''
        category: 'MaSimulation'
```

As figuras 7.1 e 7.2 apresentam, respectivamente, o *Browser Smalltalk*, utilizado pelo sistema, e um exemplo de janela de simulação.

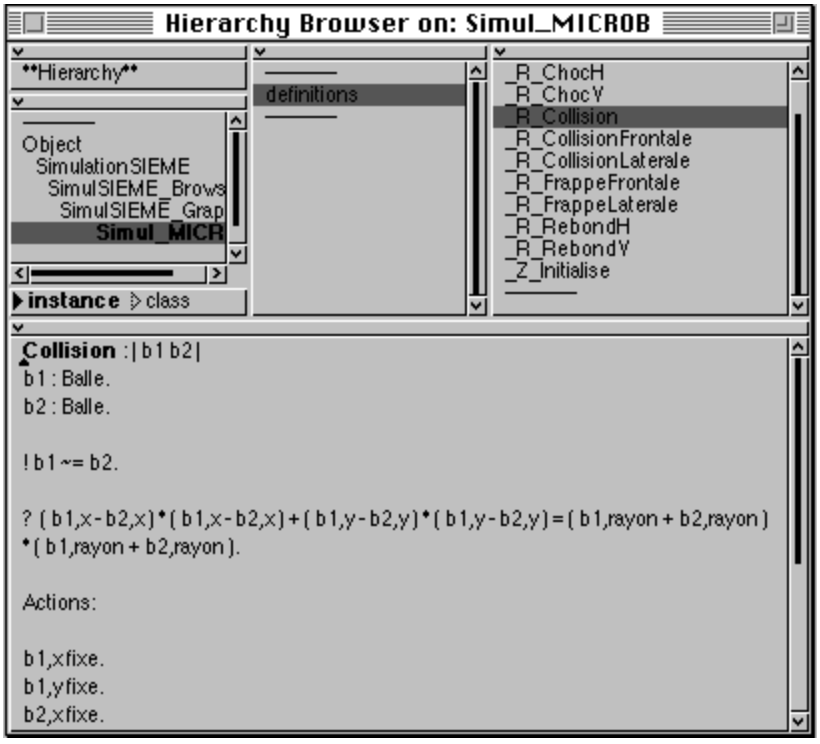


FIGURA 7.1 - Browser Smalltalk do Sistema SIEME

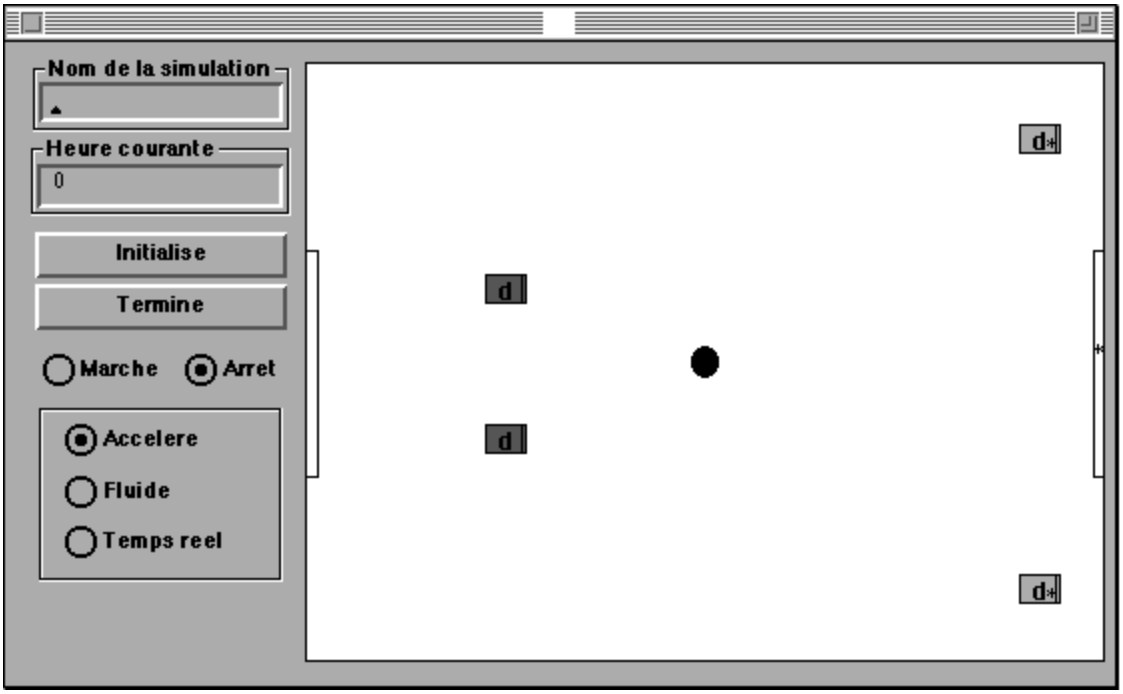


FIGURA 7.2 - Janela de Simulação do Sistema SIEME

7.1.2 O Sistema SWARM

O sistema SWARM é uma plataforma de software multiagente para a simulação de sistemas adaptivos [MIE 96]. Fornece uma biblioteca orientada a objetos de componentes reutilizáveis para a construção de modelos e para analisar experimentos sobre o mesmo. Requer o compilador GNU C, Unix e X Windows.

A unidade básica de uma simulação *swarm* é o agente e o componente fundamental que organiza os agentes de um modelo criado é um objeto chamado *swarm*, que é uma coleção de agentes independentes, interagindo através de eventos com um *schedule* de eventos. Um agente é qualquer entidade que pode gerar eventos que afetam a ele próprio e a outros agentes.

Um *schedule* é uma estrutura de dados que combina as ações dos agentes em uma ordem específica na qual devem ser executadas. Por exemplo, uma simulação, envolvendo coiotes, coelhos e cenouras, deve ter três ações: "coelhos comem cenouras", "coelhos escondem-se dos coiotes" e "coiotes comem coelhos". Cada ação é um evento discreto e o *schedule* combina as três em uma ordem específica: "cada dia, os coelhos comem cenouras, então eles escondem-se dos coiotes e, então, os coiotes tentam comer os coelhos".

A estrutura de uma simulação *swarm* consiste da criação de um modelo, composto por um grupo de agentes e por um *schedule* de atividade para os agentes. Os agentes são implementados como objetos e são criados a partir de uma classe da biblioteca *swarm*.

Uma vez que o usuário definiu os agentes e estabeleceu seus relacionamentos, o último passo, na construção do modelo, é escrever um *schedule* de atividades para os agentes, através da biblioteca *activity*.

Durante a execução do modelo, agentes observadores registram o que acontece na simulação. Estes agentes são, também, objetos *swarm*.

Exemplo de um trecho do programa principal de uma aplicação no sistema SWARM:

```
#import <defobj.h>
#import <tkobjc.h>
#import <simtools.h>
#import "MarketObserverSwarm.h"

int
main (int argc, char **argv)
```

```
{
        marketObserverSwarm *observerSwarm;
        initSwarm(argc, argv);
        observerSwarm = [MarketObserverSwarm create:globalzone];
        [observerSwarm buildObjects];
        [observerSwarm buildActions];
        [observerSwarm activateIn:nil];
        [observerSwarm go];
        return 0;
}
```

7.2 O Ambiente Proposto

Este ambiente possibilita, ao usuário, criar aplicações em sistemas multiagentes reativos, através de elementos de uma interface gráfica, os quais permitem a este usuário determinar os agentes envolvidos no problema e como eles agirão no processo de resolução do mesmo. O usuário define a sua aplicação criando um modelo para ela.

A modelagem consiste em representar um problema ou uma situação real, utilizando, para isso, um grupo de agentes reativos que interagem entre si e com o ambiente no qual estão inseridos, visando atingir uma solução.

Portanto, a função do usuário é definir a situação inicial do seu problema e determinar como os agentes agirão. Após estas difinições, o ambiente se encarrega de executar a simulação e apresentar uma situação final, atingida pela atuação dos agentes.

Os modelos de sistemas multiagentes reativos, descritos no capítulo 6, foram estudados e analisados com o objetivo de determinar os elementos e as características que uma ferramenta de desenvolvimento de aplicações, em tais tipos de sistemas, deveria apresentar.

Após o estudo e a análise desses modelos, com a implementação de um exemplo de aplicação, utilizando os modelos de [FER 91] e de [STE 90], contemplaram-se, para a proposta desta ferramenta, os modelos da Funcionalidade Emergente e da Eco-Resolução. O modelo PACO não foi considerado para a definição e a construção do ambiente por apresentar características que se diferenciam um pouco das características dos outros

O problema que o usuário desejar modelar deve se adequar às características de sistemas multiagentes reativos suportadas pela ferramenta de desenvolvimento.

7.2.1 Características do Ambiente de Desenvolvimento SIMULA

O ambiente projetado possui as seguintes características:

- apresenta uma interface gráfica, baseada em janelas e menus, que permite definir um problema ou uma situação como um sistema multiagente reativo;
- permite descrever os tipos de agentes envolvidos no problema ou na situação, que agirão em busca de uma solução;
- permite descrever as regras de comportamento dos agentes definidos: as regras de comportamento determinam como os agentes agirão (quais são as suas ações), para atingir uma solucão, e quando (em que situação) a regra será ativada. O comportamento dos agentes é modelado através de regras;
- permite situar os agentes em seu ambiente, determinando a posição inicial de cada um;
- possui comportamentos pré-definidos, usados para montar as regras de comportamento dos agentes;
- é limitado quanto às aplicações: o ambiente é adequado para problemas ou situações que possam ser modelados como um sistema de agentes e que se adaptem às características dos modelos da Funcionalidade Emergente e da Eco-Resolução, considerados para a construção do ambiente;
- a apresentação do processo de resolução do problema ou da situação, com a atividade dos agentes em seu ambiente, é feita de forma gráfica.

7.2.2 Funcionalidade do Ambiente SIMULA

O ambiente SIMULA destina-se a atender usuários que tenham conhecimento da tecnologia do uso de agentes para a construção de sistemas reais, científicos e simulações. E que tenham, também, um embasamento do que são agentes e de como eles podem atuar em sistemas.

O objetivo do SIMULA é diminuir o esforço de programação do usuário para criar suas aplicações, estimulando o mesmo a projetar novos sistemas com o uso de agentes reativos. Para isso, o ambiente oferece facilidades para a definição de problemas.

A figura 7.3 apresenta um esquema de como o usuário do ambiente SIMULA deve proceder para desenvolver suas aplicações em sistemas multiagentes reativos e de quais são as etapas internas de execução até ser atingida uma solução (situação final) para a aplicação descrita.

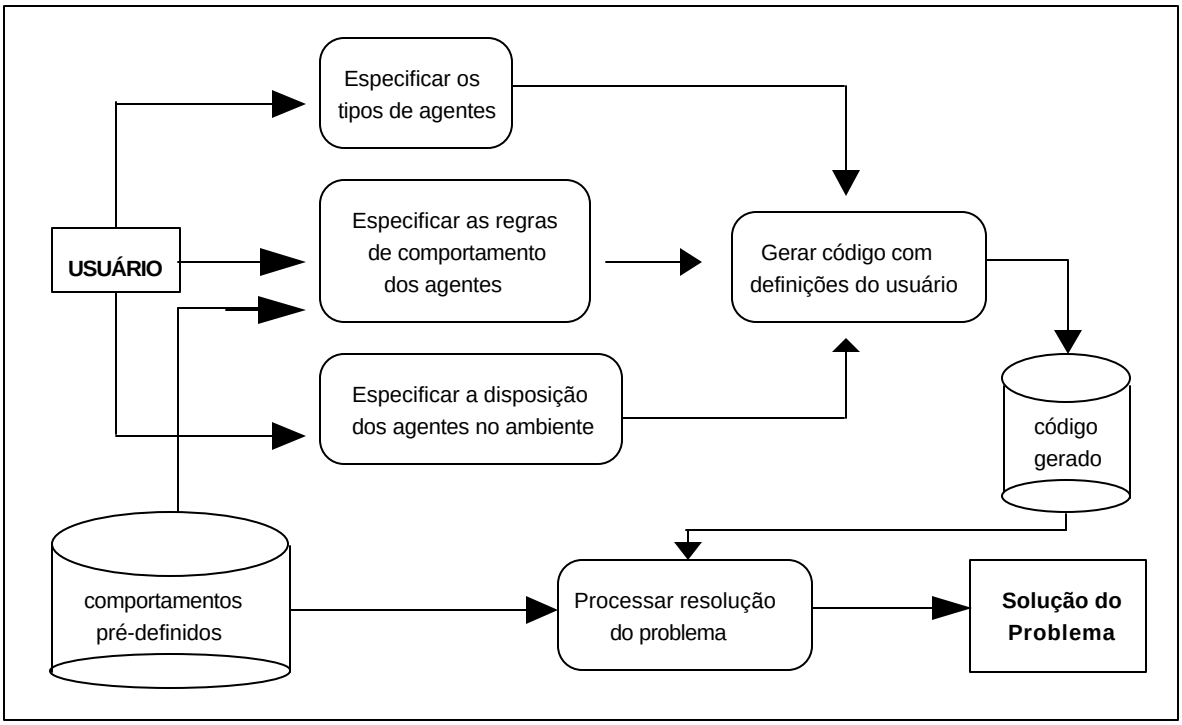


FIGURA 7.3 - Funcionalidade do Ambiente SIMULA

O ambiente foi estruturado em dois módulos:

- o módulo de definição;
- o módulo de execução.

7.2.2.1 Módulo de Definição do Ambiente SIMULA

O usuário interage com o ambiente SIMULA através do módulo de definição. Nesta etapa, o usuário deverá descrever o problema (aplicação) sob a forma de uma população de agentes, atribuindo comportamentos aos mesmos. Os comportamentos representam o processo de ação dos agentes, no sentido de atingirem uma situação final, vista pelo usuário, como uma solução ao seu problema.

O módulo de definição divide-se em três partes:

- especificação dos tipos de agentes que farão parte do sistema: o usuário especifica quais serão os agentes envolvidos no processo, associando um nome

para cada tipo de agente. Agentes do mesmo tipo possuem as mesmas características e as mesmas ações. Também é definida a área de percepção do agente, que refere-se à área na qual o agente consegue perceber outros agentes. Essa área de percepção corresponde a posições no ambiente do agente. Por exemplo, se um agente *A* tiver uma área de percepção igual a 2, significa que qualquer agente que estiver a duas posições do ambiente, ao redor do agente *A*, pode ser percebido por este;

- especificação das regras de comportamento dos agentes: as regras de comportamento são especificadas para cada tipo de agente e são elas que guiam as ações dos agentes no processo de resolução de problemas. Para especificar estas regras, o usuário utiliza os comportamentos pré-definidos do ambiente e também pode utilizar variáveis criadas por ele;
- especificação da disposição dos agentes no ambiente: é oferecida, ao usuário, a possibilidade de determinar a posição inicial de cada agente no ambiente no qual estes atuarão. É possível, também, determinar as dimensões do ambiente de atuação dos agentes, que é representado por uma matriz, na qual cada célula corresponde a uma posição que pode ser ocupada pelos agentes. Nesta etapa, o usuário define o número de agentes de cada tipo, especificados anteriormente.

7.2.2.2 Módulo de Execução do Ambiente SIMULA

Após serem realizadas todas as definições necessárias, o usuário poderá acionar o módulo de execução do ambiente SIMULA. Este módulo divide-se em duas partes:

- geração de código: este processo utilizará as definições feitas pelo usuário para a sua aplicação, juntamente com os comportamentos pré-definidos do ambiente para, então, gerar um código (programa em linguagem de programação C++), que será responsável pelo processo de resolução da aplicação;
- execução: corresponde à execução do código gerado e representa a ativação do processo de resolução da aplicação definida. Este processo é mostrado de forma gráfica, através de uma simulação da atuação dos agentes e de seus movimentos/deslocamentos no ambiente. Cada agente é representado por uma letra do nome do agente com uma cor, definida pelo usuário. O término do processo indica que os agentes atingiram uma situação final, que pode ser considerada como uma solução. O usuário acompanha a atuação dos agentes e

fica a seu cargo analisar se o processo gerado, com as suas definições, equivale à

7.2.3 Os Comportamentos Pré-Definidos do Ambiente SIMULA

Com o objetivo de facilitar o desenvolvimento de sistemas multiagentes reativos, o ambiente proposto possui implementados alguns comportamentos de base, que podem ser utilizados em muitas aplicações. As seções seguintes detalham os comportamentos pré-definidos. Este conjunto de comportamentos não é suficiente para a execução de todos os comportamentos possíveis, mas permite o desenvolvimento de um conjunto de aplicações.

Comportamentos não disponíveis nestes comportamentos elementares devem ser implementados através de programação convencional, obrigando conhecimento de detalhes de implementação do sistema, como estruturas de dados.

Os comportamentos são associados a cada um dos tipos de agentes definidos pelo usuário. Todos os agentes do mesmo tipo agem da mesma forma, possuindo os mesmos comportamentos.

Há três classes de comportamentos pré-definidos que o ambiente SIMULA oferece e os quais são usados para montar as regras de comportamento dos agentes:

- comportamentos ativos;
- comportamento passivo;
- comportamentos de estado.

7.2.3.1 Comportamentos Ativos

São comportamentos que ativam uma ação do agente em seu ambiente, causando uma mudança de estado desse agente. A seguir, são listados estes comportamentos e os parâmetros que devem ser fornecidos pelo usuário no momento da definição:

- realiza movimento randômico (agente a realizar o movimento)

_____: este comportamento refere-se a um agente realizar um movimento em seu ambiente para qualquer direção. Esta direção é determinada por uma função

_____: nome do tipo de agente que deve realizar este comportamento.

- deixa pista (agente que deixa a pista)

Descrição: este comportamento refere-se a um agente movimentar-se em seu ambiente, mas deixando uma marca (pista) por onde passa. A este comportamento

_____: nome do tipo de agente que deve seguir a pista.

- segue maior gradiente (agente que segue, agente que emite)

Descrição: este comportamento refere-se a um agente movimentar-se no sentido do maior gradiente. Gradiente é um sinal emitido por um agente e que pode ser percebido pelos outros agentes. Este sinal é representado pela distância entre o agente que emite o gradiente e os que o percebem. Para cada direção que o agente pode seguir (frente, trás, esquerda, direita), é calculado o gradiente, em relação ao agente que o emite, para determinar para qual lado o agente que segue o maior gradiente deve movimentar-se.

Parâmetros: nome do tipo de agente que deve seguir o gradiente e nome do tipo de agente que deve emitir o gradiente.

- segue menor gradiente (agente que segue, agente que emite)

Descrição: este comportamento refere-se a um agente movimentar-se no sentido do menor gradiente. Para cada direção que o agente pode seguir (frente, trás, esquerda, direita), é calculado o gradiente, em relação ao agente que o emite, para determinar para qual lado o agente que segue o menor gradiente deve movimentar-se.

Parâmetros: nome do tipo de agente que deve seguir o gradiente e nome do tipo de agente que deve emitir o gradiente.

- segue agente (agente que segue, agente seguido)

Descrição: este comportamento refere-se a um agente movimentar-se no sentido de perseguir outro agente, que também movimenta-se no ambiente. Para um agente A perseguir um agente B, o agente A deve perceber a presença do agente B.

_____: nome do tipo de agente que deve perseguir e nome do tipo de agente que será perseguido.

_____: nome do tipo de agente que deve morrer.

- reproduz (agente que reproduz, agente reproduzido, quantidade)

Descrição: este comportamento refere-se a um agente originar outros de sua espécie.

Parâmetros: nome do tipo de agente que efetuará a reprodução, nome do tipo de agente gerado e a quantidade de agentes gerados.

- transforma (agente a ser transformado, novo agente)

Descrição: este comportamento refere-se a um agente assumir um novo papel em seu ambiente. Os novos agentes passam, então, a fazer parte da simulação. A transformação de agentes significa uma ampliação do número de novos agentes e uma redução do número de agentes que se transformaram.

_____: nome do tipo de agente que deve ser transformado em outro, nome do tipo do novo agente.

7.2.3.2 Comportamentos Passivos

São comportamentos que representam uma ação do agente, mas que não ocasionam, diretamente, uma mudança de posição do agente em seu ambiente. O comportamento deste tipo pré-definido é:

- evita obstaculo ( )

Descrição: os obstáculos referem-se aos outros agentes existentes no ambiente. Quando um agente movimenta-se no ambiente, pode ser associado, ao comportamento de movimento, o comportamento de evitar obstáculos. Neste caso, o agente não poderá deslocar-se para uma posição já ocupada por outro agente e terá que desviar, movimentando-se em outra direção. O movimento de evitar obstáculos ocasiona a execução de outro comportamento de movimentação do agente. Este comportamento não possui parâmetros por estar associado a um comportamento de movimento.

7.2.3.3 Comportamentos de Estado

São comportamentos relacionados à percepção do agente e ao seu estado atual, e que podem condicionar a execução de uma determinada ação pelo agente. Como resultado, tais comportamentos retornam os estados verdadeiro ou falso.

Estes comportamentos são listados a seguir, juntamente com os parâmetros que devem ser fornecidos pelo usuário no momento da definição:

- percebe agente (agente que percebe, agente a ser percebido)

Descrição: este comportamento refere-se a um agente perceber a presença de outro agente. Se um agente A perceber um agente B, significa que o agente B está localizado dentro da área de percepção do agente A.

_____: nome do tipo de agente que deve verificar se outro agente encontra-se em sua área de percepção e nome do tipo de agente que deve ser percebido.

- percebe pista (agente que percebe)

Descrição: este comportamento refere-se a um agente perceber uma pista/trilha no ambiente. Um agente percebe a pista se a mesma estiver localizada na área de

Parâmetro: nome do tipo de agente que deve verificar se uma pista encontra-se em sua área de percepção.

- atinge agente (agente que atinge, agente a ser atingido)

Descrição: este comportamento refere-se a um agente atacar outro agente. Se um agente A atinge um agente B, significa que o agente A chega até a posição do agente B, ocupando-a, e fica de posse do agente B.

Parâmetros: nome do tipo de agente que deve atingir outro agente e nome do tipo de agente que será atingido.

- atinge tempo vida (agente, tempo máximo de vida do agente)

_____: este comportamento refere-se a um agente atingir o seu tempo máximo de vida. Um agente atingir o seu tempo máximo de vida pode acarretar o comportamento de morte deste agente.

Parâmetros: nome do tipo de agente e o tempo máximo de vida deste agente em dias.

- tempo fertilidade (agente que reproduz, agente reproduzido,

tempo de vida fértil do agente)

Descrição: este comportamento refere-se ao tempo de vida fértil do agente - durante quantos dias o agente reproduz.

Parâmetros: nome do tipo de agente que efetua a reprodução, nome do tipo de agente reproduzido e o tempo de vida fértil deste agente em dias.

- atinge vida adulta (agente, tempo para tornar-se adulto)

Descrição: este comportamento refere-se a um agente atingir o seu estágio de vida adulta para, então, poder atuar em seu ambiente. Um agente atingir o seu estágio de vida adulta pode acarretar o comportamento de transformação deste agente.

_____: nome do tipo de agente que deve atingir o estágio de vida adulta e o tempo, em dias, para um agente tornar-se adulto.

- tipo sexo (sexo, taxa de nascimento)

Descrição: este comportamento refere-se a um agente ser macho ou fêmea.

_____: o tipo do sexo do agente (macho ou fêmea) e a taxa de nascimento do sexo especificado em percentagem (0 a 100).

- taxa de sucesso (taxa)

Descrição: retorna "verdadeiro", na proporção do parâmetro "taxa" informado.

_____: taxa em percentagem (0 a 100).

7.2.4 Regras de Comportamento

A atuação dos agentes em um ambiente é modelada através de regras, que determinarão quais comportamentos dos agentes devem ser executados e sob que condições.

Como os agentes reativos são baseados em comportamento, a etapa de criação das regras, que ativarão a ação/comportamento dos agentes durante o processo de resolução do problema, é de grande importância.

Os comportamentos equivalem à chamada de um procedimento que executará uma

A figura 7.4 mostra o que deve ser especificado em cada regra de comportamento. Cada tipo de agente pode ter várias regras de comportamento.

REGRAS DE COMPORTAMENTO		
pré-condição	:	< condições para execução do comportamento >
ação-ativada	:	< comportamento a ser executado >
ação-condicional	:	< comportamento embutido na ação ativada >
pós-condição	:	< atualizações de variáveis >
prioridade	:	< ordem para a execução das regras >

FIGURA 7.4 - Elementos de uma Regra de Comportamento

- *pré-condição* : são testes condicionais sobre o estado corrente do agente, que acionam ou não a execução da regra de comportamento associada ao agente. O usuário pode utilizar, na pré-condição, expressões de comparação, com o uso de variáveis lógicas ou aritméticas, definidas por ele próprio, os comportamentos de estado pré-definidos e os conetivos lógicos AND, OR e NOT;
- *ação-ativada* : representa o(s) comportamento(s) a ser(em) executado(s), se a pré-condição for verdadeira. O usuário pode utilizar os comportamentos ativos pré-definidos. Se houver mais do que um comportamento a ser ativado, separá-

- *ação-condicional* : é um comportamento que representa uma condição dentro da execução da ação ativada. Por exemplo, define-se que o agente deve realizar um movimento randômico, mas antes de realizar este comportamento, define-se que o mesmo deve verificar se há obstáculos em seu caminho. O usuário pode utilizar os comportamentos ativos e passivo pré-definidos;
- *pós-condição* : são os efeitos causados pela execução dos comportamentos. O usuário pode efetuar alterações ou atualizações de variáveis. São permitidos comandos de atribuição e expressões aritméticas.

- *prioridade* : o usuário pode, através da prioridade, estabelecer a ordem de execução das regras, porque, a cada ciclo, a agente realiza apenas uma ação/comportamento. Se o agente puder executar mais do que um comportamento no mesmo ciclo, a prioridade estabelecerá qual será executado entre os possíveis. A prioridade é determinada por um valor numérico. Quanto menor o valor, maior a prioridade. Por exemplo, uma regra com prioridade = 1 é executada antes do que uma regra com prioridade = 3.

8 O Protótipo do Ambiente SIMULA

Desenvolveu-se um protótipo do ambiente definido, com o objetivo de verificar se as características contempladas eram suficientes e satisfatórias para a construção de sistemas, envolvendo a atuação de agentes reativos na solução de problemas e na simulação de

O protótipo foi desenvolvido na linguagem de programação C++ 3.1 da Borland para rodar em ambiente Windows e na plataforma PC (Personal Computer). A interface do protótipo foi criada no Resource Workshop da Borland.

8.1 A Interface do Protótipo

A figura 8.1 mostra como é a interface do protótipo desenvolvido.

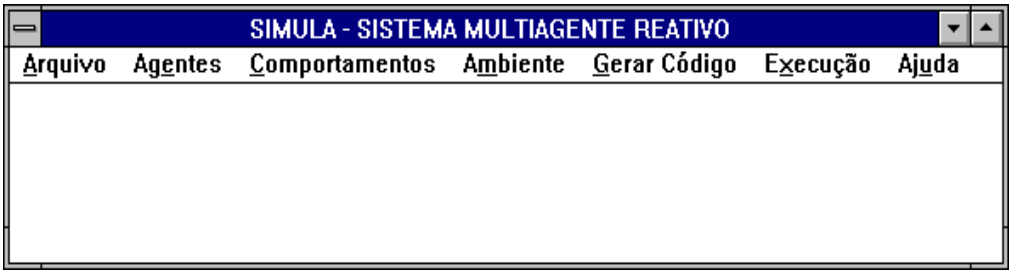


FIGURA 8.1 - Interface do Protótipo

A interface é constituída por um menu principal com sete opções, através das quais o usuário realiza as definições e a simulação da sua aplicação modelada.

Nas seções seguintes, são descritos cada um dos menus.

8.1.1 Menu ARQUIVO

Através das opções deste menu, o usuário consegue executar as funções padrão de manipulação de arquivos, como mostra a figura 8.2. Todas as definições feitas no ambiente podem ser salvas em um arquivo para posteriores alterações. As informações salvas são

referentes à definição dos tipos de agentes, suas regras de comportamento e suas posições iniciais no ambiente em que devem atuar.

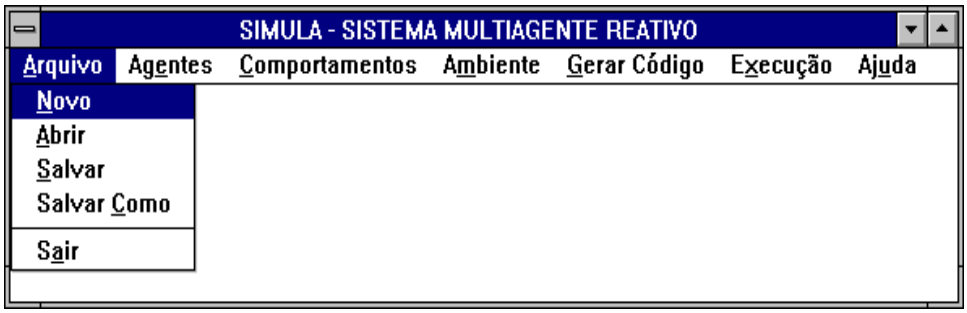


FIGURA 8.2 - Menu ARQUIVO

8.1.2 Menu AGENTES

Este menu permite a definição dos tipos de agentes que irão compor o ambiente de solução do problema. A figura 8.3 mostra a caixa de diálogo do ambiente para a definição dos agentes.

Nesta etapa, o usuário define o nome do tipo de agente e sua área de percepção. A área de percepção deve ser um valor inteiro, correspondente ao número de posições ao redor do agente no qual este consegue perceber outros agentes. O identificador é um valor fornecido pelo sistema.

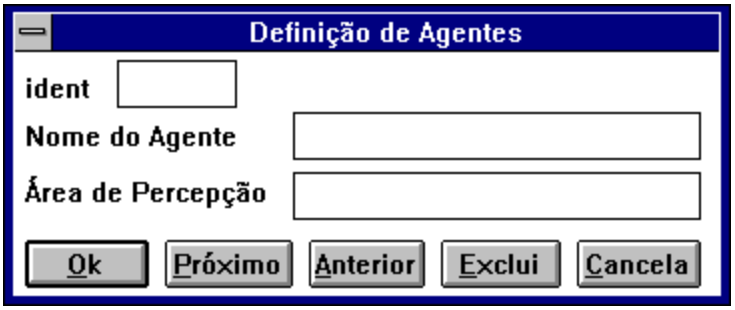


FIGURA 8.3 - Definição de Agentes

Os botões oferecem as seguintes funcionalidades:

- Botão Ok: é o botão de confirmação das definições efetuadas pelo usuário;
- Botão Próximo: visualiza a próxima definição de agente para alteração, inclusão

- Botão Anterior: visualiza a definição de agente anterior para alteração ou

- Botão Excluir: exclui a definição do agente;
- Botão Cancela: não considera as definições do agente efetuadas.

8.1.3 Menu COMPORTAMENTOS

Este menu contém as opções que permitem ao usuário definir as ações dos agentes através de regras de comportamento, definir as variáveis usadas em tais regras de comportamento e definir um critério de parada da execução do processo de simulação.

A figura 8.4 mostra a caixa de diálogo do ambiente para a definição das variáveis. O usuário define um nome para a variável, um tipo (que são os disponíveis na linguagem C) e um valor inicial.

FIGURA 8.4 - Definição de Variáveis

Os botões oferecem as seguintes funcionalidades:

- Botão Ok: é o botão de confirmação das definições efetuadas pelo usuário;
- Botão Próximo: visualiza a próxima definição de variável para alteração, inclusão

- Botão Anterior: visualiza a definição de variável anterior para alteração ou

- Botão Excluir: exclui a definição da variável corrente;
- Botão Cancela: não considera as definições de variáveis efetuadas.

A figura 8.5 mostra a caixa de diálogo do ambiente para a definição das regras de comportamento.

- Botão Anterior: visualiza a definição da regra de comportamento anterior para

- Botão Excluir: exclui a regra de comportamento;
- Botão Cancela: não considera as definições de regras de comportamento efetuadas.

A figura 8.6 mostra a caixa de diálogo do ambiente para a definição do critério de parada. O usuário define uma condição para finalizar a execução do processo. Pode utilizar variáveis definidas por ele próprio, com expressões lógicas e aritméticas, e utilizar os operadores lógicos AND, OR e NOT.

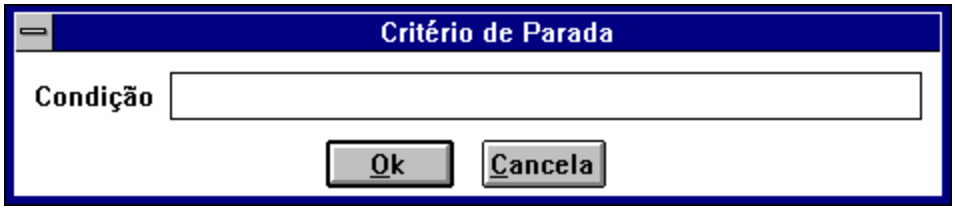


FIGURA 8.6 - Definição do Critério de Parada

8.1.4 Menu AMBIENTE

Este menu contém as opções que permitem ao usuário definir as dimensões do ambiente, no qual os agentes atuarão, e as posições iniciais dos agentes neste ambiente.

A figura 8.7 mostra a caixa de diálogo para a definição das dimensões do ambiente dos agentes. O ambiente de atuação dos agentes é representado por uma matriz e, assim, o usuário pode definir o número de linhas e de colunas dessa matriz. Se o usuário não fizer sua definição, o programa assumirá valores *default*.

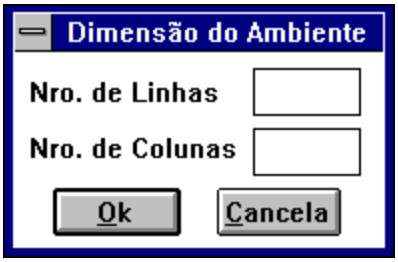


FIGURA 8.7 - Definição das Dimensões do Ambiente dos Agentes

O usuário pode optar entre posicionar os agentes inicialmente no ambiente ou não, através da escolha das opções Distribuição Definida ou Distribuição Aleatória, respectivamente. Na Distribuição Aleatória, o programa gera as posições iniciais para os agentes.

A figura 8.8 mostra a caixa de diálogo para a definição da distribuição dos agentes em seu ambiente.

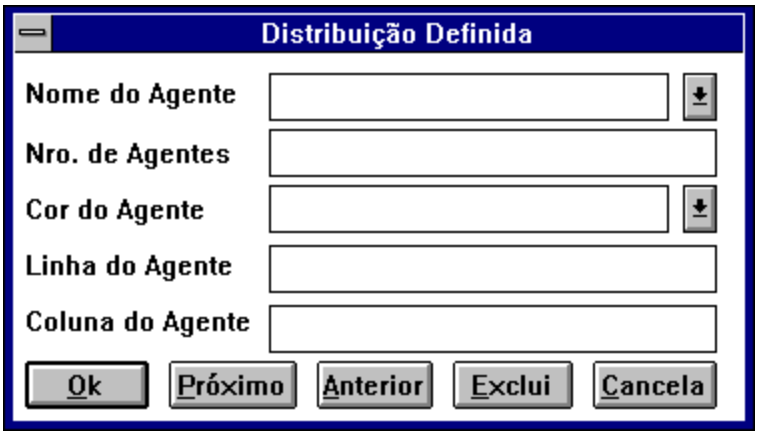


FIGURA 8.8 - Definição da Distribuição dos Agentes em seu Ambiente

Nesta etapa, o usuário associa o nome do tipo de agente, definido anteriormente, com novas características a serem especificadas. O usuário deve determinar o número de agentes de cada tipo de agente, podendo associar ao mesmo uma cor para identificá-lo durante o processo de simulação, e a posição de cada um dos agentes na matriz de ambiente (valor de linha e de coluna).

Os botões oferecem as seguintes funcionalidades:

- Botão Ok: é o botão de confirmação das definições efetuadas pelo usuário;
- Botão Próximo: visualiza a próxima definição de distribuição dos agentes para
- Botão Anterior: visualiza a definição de distribuição dos agentes anterior para
- Botão Excluir: exclui a definição de distribuição dos agentes;
- Botão Cancela: não considera as definições de distribuição dos agentes efetuadas.

8.1.5 Menu GERAR CÓDIGO

Esta opção representa um procedimento que faz parte do módulo de execução do sistema. Utiliza as definições feitas pelo usuário, juntamente com os comportamentos pré-definidos do ambiente, para montar um arquivo que será compilado no compilador C++ da Borland. Este arquivo será responsável pelo processo de simulação da aplicação definida pelo usuário. A máquina que rodará o ambiente SIMULA deve ter este compilador instalado.

Após o usuário efetuar todas as definições necessárias para a modelagem da sua aplicação, ele poderá ativar este menu.

O usuário tem acesso ao código gerado após as suas definições no ambiente, pois este código é um arquivo em linguagem C++. É possível, ao usuário, visualizar o arquivo de comportamentos pré-definidos, o arquivo com as definições de variáveis e o arquivo com a estrutura para o início do processo de simulação da aplicação definida.

8.1.6 Menu EXECUÇÃO

Opção que deve ser acionada após a geração do código. Dispara a execução do processo de resolução da aplicação definida.

8.1.7 Menu AJUDA

Contém informaçõe sobre o uso do ambiente, para o caso o usuário ter alguma

8.2 Estruturas de Implementação

Esta seção apresenta como os conceitos de sistemas multiagentes reativos foram abordados na programação e quais foram as principais estruturas de dados utilizadas para a implementação do protótipo do ambiente SIMULA.

A tabela 8.1 mostra a relação entre os conceitos envolvendo agentes e as estruturas de implementação que os representam.

TABELA 8.1 - Relação entre Conceitos e Estruturas de Implementação

CONCEITOS	ESTRUTURAS
agente	estrutura com características do agente
tipos de agentes	lista encadeada de estruturas de agentes
ambiente do agente	representação por matriz
disposição dos agentes no ambiente	posições (linha,coluna) da matriz
comportamento do agente	funções que determinam ações específicas
ativação do comportamento do agente	regras de comportamento
tipos de comportamento do agente	lista encadeada de regras de comportamento

A figura 8.9 mostra como as informações das definições feitas pelo usuário são armazenadas em forma de listas encadeadas. A lista principal é dos tipos de agentes e, a partir desta, tem-se acesso à lista dos agentes de determinado tipo e à lista de comportamento dos agentes do mesmo tipo.

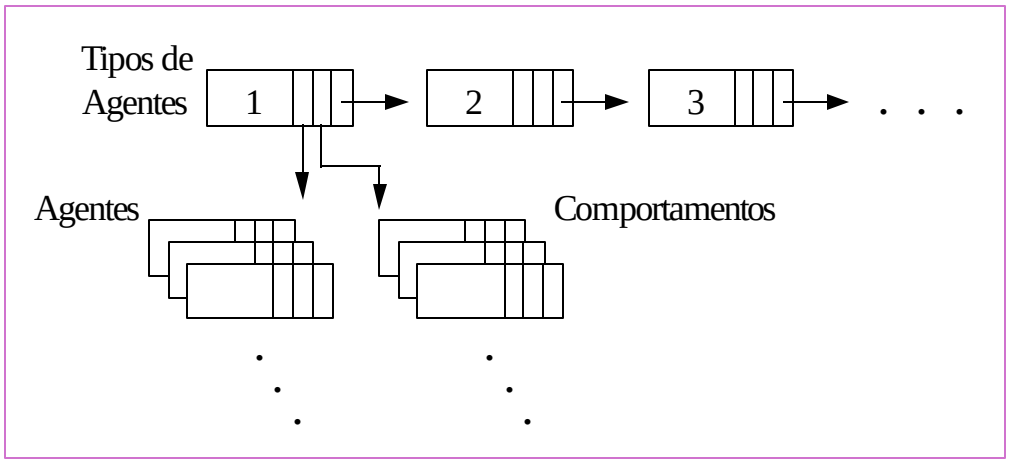


FIGURA 8.9 - Principais Estruturas de Listas Encadeadas

Cada nodo de uma lista encadeada representa uma estrutura com as informações referentes aos agentes.

As estruturas principais são as seguintes:

```
ESTRUTURA TIPO DE AGENTE
                    {
                        char *nome;       /* nome do tipo de agente */
                        int percepção,    /*área de percepção do agente em posições */
                            nro_agentes,  /* número de agentes do respectivo tipo */
                            cor;          /* cor do tipo de agente */
                        struct TIPO DE AGENTE *ptr;/*ponteiro para  estrutura*/
                        struct AGENTE *prox;  /* ponteiro para estrutura AGENTE */
                        struct COMPORTAMENTO *pont;/*ponteiro para estrutura */
                    }
```

```
ESTRUTURA  AGENTE
                    {
                        int pos_lin,   /*posição do agente na matriz do ambiente-linha */
                            pos_col,/*posição do agente na matriz do ambiente coluna*/
                            vida;      /* tempo de vida do agente */
                        struct AGENTE *ptr; /*ponteiro para própria estrutura*/
                    }

ESTRUTURA COMPORTAMENTO
                    {
                        char *pré_condição,      /* regras de comportamento */
                             *ação_ativada,
                             *ação_condicional,
                             *pós_condição;
                        int prioridade;       /* prioridade para execução das regras */
                        struct COMPORTAMENTO *ptr; /* ponteiro para estrutura */
                    }
```

8.3 Estrutura Principal do Módulo de Execução

Após ser gerado o código, com todas as definições feitas pelo usuário, este é compilado juntamente com os outros arquivos internos do sistema e, então, executado. O usuário, além de ter em sua máquina todos os arquivos que fazem parte do ambiente SIMULA, também deve ter instalado o compilador C++ da Borland.

Uma visão geral da estrutura da função principal pode ser vista a seguir:

```
Início
    Enquanto NOT (condição de parada)
        Para cada agente com comportamento
            Seleciona_Comportamento (comportamento)
            Executa_Comportamento (comportamento)
```

Fim

Os agentes atuam enquanto a condição de parada não for atingida. A cada ciclo, o agente só poderá efetuar um comportamento, portanto, o sistema faz uma seleção das regras de comportamento que o agente poderá realizar naquele momento e escolhe a de maior prioridade para ser executada.

8.3.1 Função *Seleciona_Comportamento*

Esta função tem a finalidade de verificar quais são os possíveis comportamentos que podem ser executados por um agente em um determinado momento. Entre os comportamentos possíveis, deve escolher o de maior prioridade e retornar, à função principal, qual o comportamento a ser executado.

Uma visão geral da estrutura da função *Seleciona_Comportamento* pode ser vista a seguir:

Função Seleciona_Comportamento (comportamento)

Início

Para cada regra de comportamento definida para o agente

Se pré_condição satisfeita

Então acrescenta comportamento na lista como possível execução

Entre os comportamentos da lista

Selecionar o de maior prioridade para execução

Retornar como parâmetro o comportamento a ser executado

Fim

8.3.2 Função *Executa_Comportamento*

Esta função tem a finalidade de ativar a execução do comportamento selecionado para o agente, através da chamada da função correspondente ao comportamento pré-definido do sistema. Também executa as pós-condições da regra de comportamento.

Uma visão geral da estrutura da função *Executa_Comportamento* pode ser vista a seguir:

Função Executa_Comportamento (comportamento)

Início

Executa o comportamento selecionado

Executa as pós-condições da regra de comportamento selecionado

Fim

9 Exemplos de Modelagem no Ambiente SIMULA

Para validar o uso do ambiente SIMULA, três aplicações foram definidas, conforme as características do ambiente. Duas delas já são conhecidas e foram descritas em seções anteriores: a atuação de robôs na busca de minerais [STE 90] e o jogo PENGI [AGR 87] e [FER 91]. A terceira aplicação envolve a área biológica, com a atuação de parasitas no controle de pragas de plantações.

Nas seções seguintes, são apresentadas as definições desses exemplos no ambiente SIMULA.

9.1 Modelagem dos Robôs

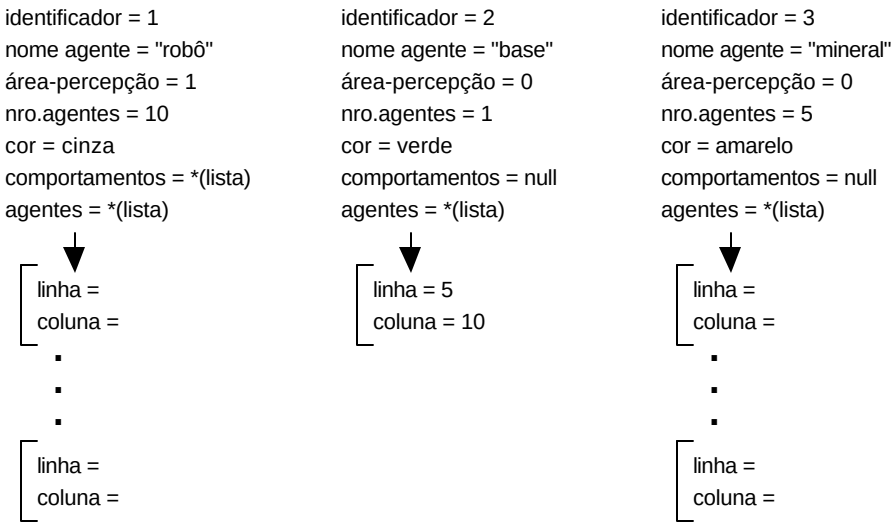
- Problema

Coleta de minerais por robôs em um terreno desconhecido. Após encontrados, os minerais devem ser levados à base central (descrição completa ver seção 6.1).

- Definições no ambiente SIMULA

Definição dos tipos de agentes:

Nesta simulação há três tipos de agentes: , *Base Central* e *Minerais*.



Há agentes que não possuem comportamentos, representado por "null" agentes que não possuem uma área de percepção, representada pelo valor zero.

Nesta simulação foram definidas duas variáveis: a variável "com_mineral" indica quando um agente consegue encontrar um mineral e a variável "qtd_mineral" controla a quantidade de minerais já encontrados pelos robôs.

nome da variável = com_mineral
tipo da variável = int
valor inicial = FALSE

nome da variável = qtd_mineral
tipo da variável = int
valor inicial = 5

Definição do critério de parada:

O critério da parada, definido para esta simulação, envolve a variável "qtd_mineral". Quando a variável atingir o valor zero, significando que todos os minerais já foram encontrados, a simulação termina.

condição : qtd_mineral = 0

Definição das regras de comportamento dos tipos de agentes:

Para esta simulação, foram definidas cinco regras de comportamento. As cinco regras referem-se ao comportamento dos robôs. A base central e os minerais não possuem comportamentos.

Regra 1: Se o robô não estiver com mineral, não perceber a presença de mineral e não perceber uma pista, então o robô deve realizar um movimento randômico pelo ambiente, evitando obstáculos, com o objetivo de encontrar minerais.

```
pré-condição : ( NOT(com_mineral)) AND (NOT(percebe_agente(robo,mineral)) AND
                    (NOT(percebe_pista(robo))
ação-ativada : realiza_movimento_randomico(robo)
ação-condicional : evita_obstaculo( )
pós-condição :
prioridade : 1
```

Regra 2: Se o robô não estiver com mineral e perceber a presença de mineral, então ocorre a atualização das variáveis definidas. A variável "com_mineral" recebe o valor TRUE, indicando que o robô está de posse de um mineral, e a variável "qtd_mineral" é decrescida de um, indicando que mais um mineral foi encontrado. Neste momento, o robô entra em modo de retorno, no qual deve levar o mineral encontrado até a base central.

```
pré-condição : ( NOT(com_mineral)) AND (percebe_agente(robo,mineral))
ação-ativada :
ação-condicional :
pós-condição : com_mineral = TRUE ; qtd_mineral = qtd_mineral - 1
prioridade : 3
```

Regra 3: Se o robô estiver com mineral e não perceber a presença da base central, então o robô deve realizar um movimento pelo ambiente seguindo na direção do maior gradiente, emitido pela base central, e evitando obstáculos. Enquanto o robô segue o caminho em direção à base central, ele deve deixar uma pista.

```
pré-condição : (com_mineral) AND (NOT(percebe_agente(robo,base)))
ação-ativada : segue_maior_gradiente(robo,base) ; deixa_pista(robo)
ação-condicional : evita_obstaculo( )
pós-condição :
prioridade :4
```

Regra 4: Se o robô estiver com mineral e perceber a presença da base central, então o robô descarrega o seu mineral e entra novamente em modo de exploração, para encontrar outros minerais. A variável "com_mineral" recebe o valor FALSE, indicando que o robô não está de posse de um mineral.

pré-condição : (com_mineral) AND (percebe_agente(robo,base))
ação-ativada :
ação-condicional :
pós-condição : com_mineral = FALSE
prioridade : 5

Regra 5: Se o robô não estiver com mineral, não perceber a presença de mineral e perceber uma pista, então o robô deve seguir a pista encontrada na direção do menor gradiente, com o objetivo de chegar a algum mineral.

pré-condição : (NOT(com_mineral)) AND (NOT(percebe_agente(robo,mineral)) AND
(percebe_pista(robo))
ação-ativada : segue_pista(robo)
ação-condicional : segue_menor_gradiente(robo, base)
pós-condição :
prioridade : 2

Definição da matriz do ambiente:

A simulação da atividade dos agentes ocorre em um ambiente determinado por uma matriz de 25 linhas por 25 colunas, por exemplo.

9.2 Modelagem do Jogo PENGI

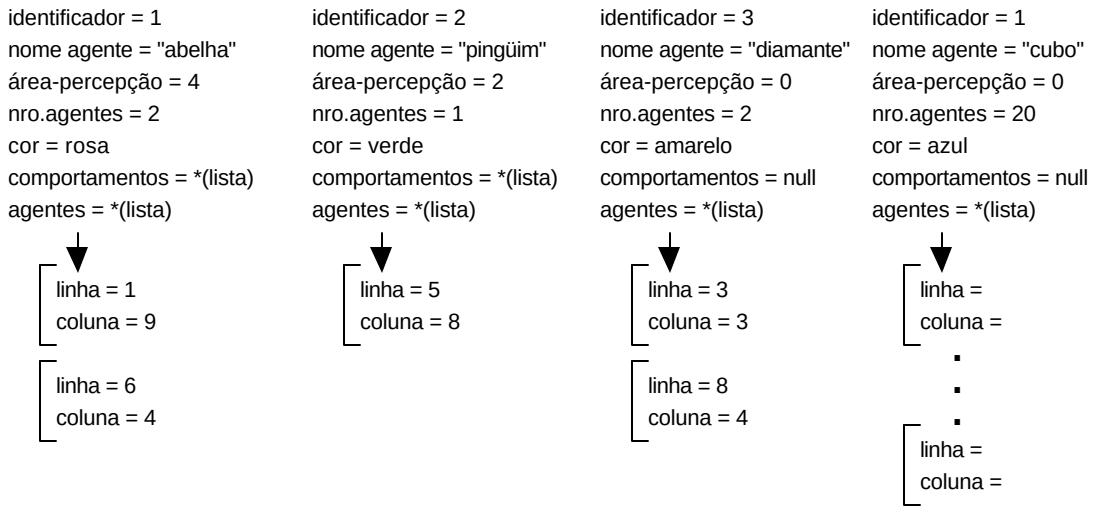
- Problema

Um pingüim move-se em um labirinto feito de cubos de gelo e diamantes e habitado por abelhas. O pingüim deve coletar todos os diamantes, enquanto evita ser picado por uma abelha (descrição completa ver seção 6.3.4).

- Definições no ambiente SIMULA

Definição dos tipos de agentes:

Nesta simulação há quatro tipos de agentes: *Abelhas*, *Pingüim*, *Cubos* e *Diamantes*.



Definição das variáveis:

Nesta simulação foi definida apenas uma variável: a variável "qtd_diamante" que controla a quantidade de diamantes já encontrados pelo pingüim.

nome da variável = qtd_diamante
tipo da variável = int
valor inicial = 2

Definição do critério de parada:

O critério da parada, definido para esta simulação, envolve a variável "qtd_diamante". Quando a variável atingir o valor zero, significando que todos os diamantes já foram encontrados, a simulação termina. Se o pingüim for pego pela abelha, a simulação

Regra 3 adiante).

condição : qtd_diamante = 0

Definição das regras de comportamento dos tipos de agentes:

Para esta simulação, foram definidas três regras de comportamento para o pingüim e três regras de comportamento para as abelhas. As regras 1, 2 e 3 referem-se ao comportamento das abelhas e as regras 4, 5 e 6 referem-se ao comportamento dos pingüins. Os cubos e os diamantes não possuem comportamentos.

Regra 1: Se a abelha perceber a presença do pingüim, então a abelha deve seguir o pingüim pelo ambiente, evitando obstáculos. A abelha consegue perceber a presença do pingüim antes que ele consiga perceber a presença da abelha, porque a abelha possui uma área de percepção maior que a do pingüim.

pré-condição : percebe_agente(abelha,pingüim)
ação-ativada : segue_agente(abelha,pingüim)
ação-condicional : evita_obstaculo( )
pós-condição :
prioridade : 2

Regra 2: Se a abelha não perceber a presença do pingüim, então a abelha deve realizar um movimento randômico pelo ambiente, evitando obstáculos, com o objetivo de

pré-condição : (NOT(percebe_agente(abelha,pingüim)))
ação-ativada : realiza_movimento_randômico(abelha)
ação-condicional : evita_obstaculo( )
pós-condição :
prioridade : 1

Regra 3: Se a abelha perceber a presença do pingüim e atingir o pingüim (conseguir matá-lo), então a simulação termina. A variável "qtd_mineral" recebe o valor zero para que a simulação pare pelo critério de parada.

pré-condição : (percebe_agente(abelha,pingüim)) AND (atinge_agente(abelha,pingüim))
ação-ativada :
ação-condicional :
pós-condição : qtd_mineral = 0
prioridade : 3

Regra 4: Se o pingüim atingir um diamante (encontrá-lo), então ocorre a atualização da variável definida. A variável "qtd_diamante" é decrescida de um, indicando que mais um diamante foi encontrado pelo pingüim.

pré-condição : atinge_agente(pingüim,diamante)
ação-ativada :
ação-condicional :
pós-condição : qtd_diamante = qtd_diamante -1
prioridade : 3

Regra 5: Se o pingüim não perceber a presença da abelha e não atingir (encontrar) algum diamante, então o pingüim deve realizar um movimento pelo ambiente seguindo na direção do menor gradiente, emitido pelos diamantes, e evitando obstáculos.

pré-condição : (NOT(percebe_agente(pingüim,abelha))) AND (NOT(atinge_agente(pingüim,diamante)))
ação-ativada : segue_menor_gradiente(pingüim,diamante)
ação-condicional : evita_obstaculo( )
pós-condição :
prioridade : 1

Regra 6: Se o pingüim perceber a presença da abelha, então o pingüim deve fugir da abelha, evitando obstáculos, com o objetivo de sair da área de percepção da abelha.

pré-condição : percebe_agente(pingüim,abelha)
ação-ativada : foge_de_agente(pingüim,abelha)
ação-condicional : evita_obstaculo( )
pós-condição :
prioridade : 2

Definição da matriz do ambiente:

da atividade dos agentes acorre em um ambiente determinado por uma matriz de 10 linhas por 10 colunas, por exemplo.

9.3 Modelagem da Atuação de Parasitas no Controle de Pragas de Plantações

- Problema

O parasita estudado é o *Spilochalcis Pseudofulvovariegata* que pode atacar a *Plutella Xylostella*. A *Plutella* é uma praga causadora de danos em culturas de couve, repolhos e outras no Brasil, sendo atacada, pelo *Spilochalcis,* em seu estágio de pupa (metamorfose da *Plutella*).

O objetivo do *Spilochalcis* é ovopositar nos hospedeiros (no caso, as *Plutellas*) para reprodução. O encontro do parasitóide com o hospedeiro é dirigido pelo odor da planta hospedeira (couve, repolho). Os hospedeiros (pragas) que abrigam os ovos do *Spilochalcis* morrem.

Estudos, visando o conhecimento da biologia deste parasitóide (*Spilochalcis*) e o seu comportamento reprodutivo, são de fundamental importância no controle biológico das pragas de cultura.

As *Plutellas* são pragas que parasitam em plantas e não possuem comportamento de movimento (são fixos no ambiente). As plantas emitem um odor que, se for percebido pelo *Spilochalcis*, o estimula a atacar a planta, ou seja, hospedar-se na *Plutella*. Após ovopositar

em seu hospedeiro, o *Spilochalcis* segue em movimento aleatório na procura de outros hospedeiros. O movimento do parasitóide *Spilochalcis*, nesta simulação, desconsidera a ação de fatores abióticos como o vento.

Sempre que a *Plutella* for parasitada, ela morre, servindo de alimento para o ovo do *Spilochalcis*. Se o ovo for fecundado, o agente *Plutella* transforma-se em um novo agente *Spilochalcis*, assumindo todos os comportamentos deste agente.

Para esta simulação foram utilizados as seguintes informações:

1. As *Plutellas* fêmeas efetuam reprodução;

2. As Plutellas macho podem ser parasitadas também;

3. Tempo de fertilidade da *Plutella* = 8 dias;

4. Média de ovos colocados por dia pela *Plutella* = 18 ovos;

5. Taxa de eclosão dos ovos da *Plutella* = 66%;

6. Tempo até o ovo da *Plutella* tornar-se adulto = 15 dias;

7. Tempo de vida adulta da *Plutella* = 20 dias;

8. Proporção de nascimento de *Plutellas* fêmeas = 50%;

9. O *Spilochalcis* coloca um ovo em cada *Plutella* parasitada;

10. Tempo até o ovo do *Spilochalcis* tornar-se adulto = 13 dias;

11. Tempo de vida adulta do *Spilochalcis* = 77 dias;

12. Taxa de eclosão dos ovos do *Spilochalcis* = 100%;

13. O *Spilochalcis* reproduz durante todo o seu tempo de vida;

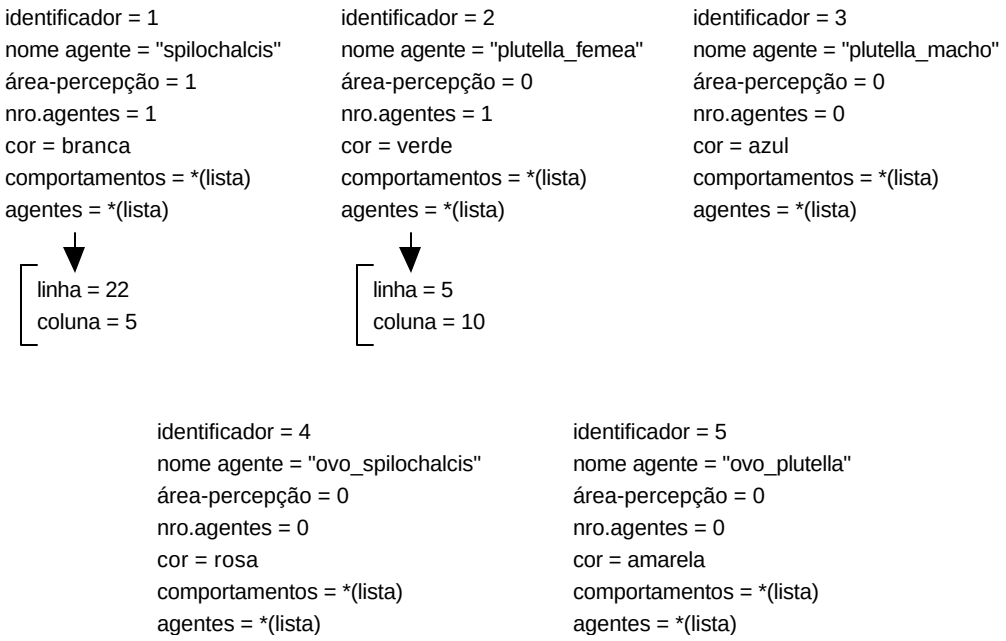
14. Cada 10 ciclos do processo correspondem a 1 dia de vida para os agentes.

Quando os ovos de *Plutellas* e *Spilochalcis* atingem o seu estágio adulto, começam a participar da simulação, assumindo os comportamentos de seus agentes originários.

- Definições no ambiente SIMULA

Definição dos tipos de agentes:

Nesta simulação há cinco tipos de agentes: *Spilochalcis*, *Plutella Fêmea*, *Plutella Macho, Ovos do Spilochalcis* e *Ovos da Plutella*.



Definição das variáveis:

Nesta simulação não foram definidas variáveis.

Definição do critério de parada:

O critério da parada, para esta simulação, é o *default* do sistema. Quando o usuário desejar terminar a simulação, deve pressionar qualquer tecla.

condição : (! kbhit ( ))

Definição das regras de comportamento dos tipos de agentes:

Para esta simulação, foram definidas quatro regras de comportamento para o *Spilochalcis*, duas regras de comportamento para as *Plutella_Femea*, uma regra de comportamento para as *Plutella_Macho*, uma regra de comportamento para os *Ovo_Spilochalcis* e duas regras de comportamento para os *Ovo_Plutella*. As regras 1, 2, 3 e 4 referem-se ao comportamento dos *Spilochalcis*, as regras 5 e 6 referem-se ao comportamento das *Plutellas_Femea,* a regra 7 refere-se ao comportamento das *Plutellas_Macho,* a regra 8 refere-se ao comportamento dos *Ovo_Spilochalcis* e as regras 9 e 10 referem-se ao comportamento dos *Ovo_Plutella*.

Regra 1: Se o *Spilochalcis* não perceber a presença de uma *Plutella* fêmea e não perceber a presença de uma *Plutella* macho, então o *Spilochalcis* deve realizar um movimento randômico pelo ambiente, evitando obstáculos, com o objetivo de parasitar e ovopositar em *Plutellas*.

pré-condição : (NOT(percebe_agente(spilochalcis,plutella_femea))) AND
(NOT(percebe_agente(spilochalcis,plutella_macho)))
ação-ativada : realiza_movimento_randomico(spilochalcis)
ação-condicional : evita_obstaculo( )
pós-condição :
prioridade : 1

Regra 2: Se o *Spilochalcis* perceber a presença de uma *Plutella* fêmea, então o *Spilochalcis* deve parasitar nesta *Plutella*, ocasionando a morte da mesma, e deve efetuar a reprodução (colocação de seu ovo).

pré-condição : percebe_agente(spilochalcis,plutella_femea)
ação-ativada : morte(plutella_femea) ; reproduz(spilochalcis,ovo_spilochalcis,1)
ação-condicional :
pós-condição :
prioridade : 2

Regra 3: Se o *Spilochalcis* perceber a presença de uma *Plutella* macho, então o *Spilochalcis* deve parasitar nesta *Plutella*, ocasionando a morte da mesma, e deve efetuar a reprodução (colocação de seu ovo).

pré-condição : percebe_agente(spilochalcis,plutella_macho)
ação-ativada : morte(plutella_macho) ; reproduz(spilochalcis,ovo_spilochalcis,1)
ação-condicional :
pós-condição :
prioridade : 3

Regra 4: Se o *Spilochalcis* atingir o seu tempo máximo de vida, então o *Spilochalcis* deve morrer.

pré-condição : atinge_tempo_vida(spilochalcis,77)
ação-ativada : morte(spilochalcis)
ação-condicional :
pós-condição :
prioridade : 4

Regra 5: Se a *Plutella* fêmea atingir o seu tempo máximo de vida, então a *Plutella* fêmea deve morrer.

pré-condição : atinge_tempo_vida(plutella_femea,20)
ação-ativada : morte(plutella_femea)
ação-condicional :
pós-condição :
prioridade : 2

Regra 6: Se a *Plutella* fêmea estiver no seu período de fertilidade, então a *Plutella* fêmea deve reproduzir seus ovos. Na reprodução, o agente *Plutella* coloca uma certa quantidade de ovos, neste caso 18 ovos, que farão parte da simulação como inertes no ambiente, até que atinjam o estágio adulto.

pré-condição : tempo_fertilidade(plutella_femea,8)
ação-ativada : reproduz(plutella_femea,ovo_plutella,18)
ação-condicional :
pós-condição :
prioridade : 1

Regra 7: Se a *Plutella* macho atingir o seu tempo máximo de vida, então a *Plutella* macho deve morrer.

pré-condição : atinge_tempo_vida(plutella_macho,20)
ação-ativada : morte(plutella_macho)
ação-condicional :
pós-condição :
prioridade : 1

Regra 8: Se o *Ovo_Spilochalcis* atingir a sua vida adulta, que neste caso demora 13 dias, então o *Ovo_Spilochalcis* transforma-se em um agente do tipo *Spilochalcis* e assume todos os comportamentos desse tipo de agente.

pré-condição : atinge_vida_adulta(ovo_spilochalcis,13)
ação-ativada : transforma(ovo_spilochalcis,spilochalcis)
ação-condicional :
pós-condição :
prioridade : 1

Regra 9: Se o *Ovo_Plutella* atingir a sua vida adulta, que neste caso demora 15 dias, a taxa de eclosão indicar que o ovo vai emergir e for o nascimento de uma fêmea, então

o *Ovo_Plutella* transforma-se em um agente do tipo *Plutella_Femea* e assume todos os comportamentos desse tipo de agente.

```
pré-condição : (atinge_vida_adulta(ovo_plutella,15)) AND (taxa_de_sucesso(66)) AND (tipo_sexo(femea,50))
ação-ativada : transforma(ovo_plutella,plutella_femea)
ação-condicional :
pós-condição :
prioridade : 1
```

Regra 10: Se o *Ovo_Plutella* atingir a sua vida adulta, que neste caso demora 15 dias, a taxa de eclosão indicar que o ovo vai emergir e for o nascimento de um macho, então o *Ovo_Plutella* transforma-se em um agente do tipo *Plutella_Macho* e assume todos os comportamentos desse tipo de agente.

```
pré-condição : (atinge_vida_adulta(ovo_plutella,15)) AND (taxa_de_sucesso(66)) AND (tipo_sexo(macho,50))
ação-ativada : transforma(ovo_plutella,plutella_macho)
ação-condicional :
pós-condição :
prioridade : 1
```

Definição da matriz do ambiente:

A simulação da atividade dos agentes acorre em um ambiente determinado por uma matriz de 25 linhas por 25 colunas, por exemplo.

10 Conclusões e Trabalhos Futuros

O crescimento da utilização da abordagem de agentes, nas mais diversas áreas de aplicações, mostra o interesse nas pesquisas sobre sistemas multiagentes, atingindo um grau satisfatório nos resultados obtidos.

Este interesse surgiu da necessidade de aplicar novas técnicas e conceitos para a construção de sistemas e para auxiliar no desenvolvimento dos mesmos. Neste sentido, os agentes satisfazem às expectativas, não sendo apenas utilizados para a solução de problemas acadêmicos, mas também de sistemas reais.

Este trabalho abordou estudos sobre agentes autônomos, descrevendo características gerais e uma visão das áreas de pesquisa envolvidas. Apresentou questões sobre os sistemas multiagentes, enfatizando o seu uso na resolução de problemas.

O ponto fundamental deste trabalho envolveu os agentes reativos e, por este motivo, foram estudados e analisados alguns modelos de sistemas multiagentes reativos. Nestes modelos, foram observadas as características dos agentes reativos e os comportamentos dos mesmos na resolução de tarefas. Também foi feita uma taxonomia entre os modelos estudados.

A partir de todos os estudos realizados, envolvendo os agentes reativos, foi possível atingir o objetivo do trabalho que era propor um ambiente para o desenvolvimento de aplicações em sistemas multiagentes reativos.

Os objetivos, para a construção de tal ambiente, foram:

- incentivar cada vez mais os usuários a modelarem seus problemas para serem resolvidos com o uso de agentes reativos;
- facilitar o trabalho dos usuários, livrando-os da tarefa de programação e envolvendo-os apenas na definição de seus problemas como um conjunto de agentes possuindo comportamentos que regem suas ações.

A escolha de trabalhar com agentes reativos foi pela interessante maneira com que atuam: cada agente possui um conjunto de comportamentos, que são ativados por estímulos no ambiente onde estão inseridos, não se preocupando em planejar nem em negociar suas ações, mas agindo de forma autônoma.

O ambiente SIMULA, em comparação com outros ambientes para desenvolvimento de sistemas multiagentes, descritos neste trabalho, oferece maior conforto ao usuário, porque não exige que o mesmo conheça integralmente uma linguagem de programação para desenvolver suas aplicações com o uso de agentes. Isto acontece com os outros ambientes estudados, nos quais o usuário modela e programa seu problema em uma linguagem de

- o usuário não precisa conhecer uma linguagem de programação específica para desenvolver sua aplicação. A tarefa do usuário é modelar a sua aplicação através de agentes, definir estes agentes e montar as regras de comportamento dos mesmos.

Livrar o usuário da tarefa de programação levou a definir, no ambiente SIMULA, uma biblioteca de comportamentos pré-definidos para os agentes. O usuário utiliza estes comportamentos para definir as regras de comportamento dos agentes da sua aplicação. Caso não existisse esta biblioteca, o usuário teria que programar os comportamentos dos agentes como funções na linguagem de programação utilizada no desenvolvimento do ambiente.

Foram feitas três modelagens de problemas no ambiente SIMULA, que permitiram mostrar a sua adequação e facilidade de uso. O enfoque maior foi para o problema do emprego de parasitas no controle de pragas de plantações por se tratar de um problema novo e real, enquanto os dois outros eram casos já descritos na bibliografia. A descrição e os dados para esta simulação foram fornecidos pelo professor Rocco A. Di Mare da Universidade Federal de Santa Maria que estuda o comportamento de parasitóides. Segundo este professor, é importante o interesse em desenvolver programas e simulações em relação ao comportamento reprodutivo de populações, neste caso os parasitóides, pois julga de grande ajuda em previsões e testes de hipóteses de trabalho.

Todo trabalho desenvolvido sempre pode apresentar continuidade, com novas idéias e pesquisas, objetivando melhorias. Para o ambiente de desenvolvimento de sistemas multiagentes reativos SIMULA também propõem-se alguns trabalhos futuros, identificados a seguir:

- melhorar a parte gráfica do processo de resolução do problema, com a possibilidade de identificar os agentes envolvidos através de figuras. A idéia é oferecer ao usuário uma espécie de editor de figuras para que ele desenhe os

agentes da sua simulação ou possa escolher figuras já prontas, disponíveis neste editor;

- após serem definidas as regras de comportamento dos agentes, efetuar uma análise sintática, verificando se a sintaxe utilizada pelo usuário está adequada. No protótipo atual, não ocorre esta análise e a geração do código é feita diretamente pelas definições do usuário, podendo aparecer algum erro apenas quando o código gerado for compilado no compilador C++;

- possibilidade do usuário incorporar novos comportamentos ao ambiente, não ficando restrito aos comportamentos pré-definidos. Mas, para isso, o usuário teria que conhecer as estruturas de dados utilizadas pelo sistema e programar os novos comportamentos na linguagem C++, para incorporá-los à biblioteca de comportamentos pré-definidos. Além disso, teria que tornar este comportamento reconhecido no sistema para a utilização do mesmo em outras definições;

- o sistema oferecer a possibilidade de gerar algumas estatísticas de acordo com as informações resultantes do processo de execução. No SIMULA, o usuário é responsável pela tarefa de analisar o processo de resolução do problema;

- definir uma metodologia para obter a descrição dos agentes e respectivos comportamentos, com vistas ao uso do ambiente SIMULA;

- estabelecer as classes de problemas para os quais o ambiente proposto é mais adequado.

Bibliografia

[AGR 87] AGRE, Philip; CHAPMAN, David. Pengi: An Implementation of the theory of activity. In: NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE, AAAI, 1987. **Proceedings…** [S.l. : s.n.], 1987. p. 268-272.

[ANT 92] ANTOINE, J. Y.; BAUJARD, O. et al. Vers une Taxinomie du Vocabulaire pour les Systèmes Multi-Agents. In: JOURNÉE NATIONALE DU PRC-IA SUR LES SYSTÈMES MULTI-AGENTS, 1., 1992, Nancy, FR. **Annales…** [S.l. : s.n.], 1992.

[BAE 95] BAEIJS, Christof; DEMAZEAU, Yves; ALVARES, Luis. Application des Systèmes Multi-Agents à la Généralisation Cartographique. In: JOURNÉES FRANCOPHONES SUR L'INTELLIGENCE ARTIFICIELLE DISTRIBUÉE ET LES SYSTÈMES MULTI-AGENTS, 3., 1995, Chambéry, FR. **Annales...** [S.l. : s.n.], 1995.

[BAL 95] BALUARD, Christine Piquemal; TROUILHET, Sylvie. Déterminer global d'une société d'agents: modèle et exemple. In: JOURNÉES FRANCOPHONES SUR L'INTELLIGENCE ARTIFICIELLE DISTRIBUÉE ET LES SYSTÈMES MULTI-AGENTS, 3., 1995, Chambéry-St-Baldoph, Savoie, FR. **Annales...** [S.l. : s.n.], 1995.

[BAR 95] BARRIL, Patrick; BRETTE, Jean-François. Agents situés, environnement et observation dans um SMA. In: JOURNÉES FRANCOPHONES SUR L'INTELLIGENCE ARTIFICIELLE DISTRIBUÉE ET LES SYSTÈMES MULTI-AGENTS, 3., 1995, Chambéry-St-Baldoph, Savoie, FR. **Annales...** [S.l. : s.n.], 1995.

[BAU 91] BAUJARD, Olivier. Un Environnement pour le Développement D'Applications Réparties en Intelligence Artificielle. In: JOURNÉES UNIX DE GRENOBLE, 1991, Grenoble, FR. **Actes…** [S.l. : s.n.], 1991.

[BEA 95] BEALE, Russell; WOOD, Andrew. Agent-Based Interation. In: PEOPLE AND COMPUTERS, 9., 1994, Glasgow, UK. **Proceedings…** [S.l. : s.n.], 1994. p. 239-245.

[BER 95] BERON, Frédéric; CARPUAT, Bernard et al. La résolution de conflit sans négociation: modèle et évaluation. In: JOURNÉES FRANCOPHONES SUR L'INTELLIGENCE ARTIFICIELLE DISTRIBUÉE ET LES SYSTÈMES MULTI-AGENTS, 3., 1995, Chambéry-St-Baldoph, Savoie, FR. **Annales...** [S.l. : s.n.], 1995.

[BOI 92] BOISSIER, Olivier; DEMAZEAU, Yves. A Distributed Artificial Intelligence View on General Purpose Vision Systems. In: WERNER, Eric; DEMAZEAU, Yves (Eds.). **Decentralized A.I.3**. Amsterdam: North-Holland, 1992.

[BON 88] BOND, Alan H.; GASSER, Les. An Analysis of Problems and Research in DAI. In: BOND, Alan H.; GASSER, Les (Eds.). **Readings in Distributed Artificial Intelligence**. San Mateo, Califórnia: Morgan Kaufman, 1988.

[BOR 94] BORDINI, Rafael Heitor. **Suporte Linguístico para Migração de Agentes**. Porto Alegre: CPGCC da UFRGS, 1994. Dissertação de Mestrado.

[BRA 96] BRAZIER, F.M.T. et al. Modelling Distributed Industrial Processes in a Multi-Agent Framework. In: KIRN, S.; O'HARE, G.M.P. (Eds.). **Cooperative Knowledge Processing**. [S.l.] : Springer-Verlag, 1996.

[BRI 90] BRIDGELAND, David Murray; HUHNS, Michael N. Distributed Truth Maintenance. In: NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE - AAAI, 8., 1990, Boston. **Proceedings...** Cambridge: The MIT Press, 1990. v.1.

[BRO 91] BROOKS, Rodney A. Intelligence Without Reason. In: INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE (IJICAI-91), 12., 1991, Sydney. **Proceedings...** Austrália : [s.n.], 1991.

[BRO 91a] BROOKS, Rodney A. Intelligence Without Representation. **Artificial Intelligence**, Amsterdam, v. 47, p 139-159, 1991.

[COE 93] COELHO, Helder. **Inteligência Artificial:** Estado da Arte. Lisboa: INESC e UTL/ISEG, 1993.

[COE 94] COELHO, Helder. **Fundamentos de Inteligência Artificial.** Lisboa : Instituto Superior de Economia e Gestão, Universidade Técnica de Lisboa, 1994.

[COH 90] COHEN, P.R.; LEVESQUE, H.J. Intention is choice with commitment. **Artificial Intelligence**, Amsterdam, v. 42, p 213-261, 1990.

[COH 90a] COHEN, P.R.; LEVESQUE, H.J. Rational interaction as the basis for communication. In: COHEN, P.R.; MORGAN, J.; POLLACK, M.E. (Eds.). **Intentions in Communications**. Cambridge:The MIT Press, 1990. p. 221-256,

[COM 94] COEN, Michael H. **SodaBot: A Software Agent Environment and Construction System**. [S.l.] : MIT Artificial Intelligence Laboratory, 1994. (A.I. Technical Report 1493).

[CON 90] CONRY, S. E.; INTOSH, D. J. Mac; MEYER, R. A. DARES: A Distributed Automated Reasoning System. In: NATIONAL

CONFERENCE ON ARTIFICIAL INTELLIGENCE - AAAI, 8., 1990, Boston. **Proceedings...** Cambridge: The MIT Press, 1990. v.1.

[DEM 90] DEMAZEAU, Yves; MÜLLER, Jean-Pierre. Decentralized Artificial Intelligence. In: DEMAZEAU, Yves; MULLER, J.P. (Eds.). **Decentralized A.I.** Amsterdam: North-Holland, 1990.

[DEM 91] DEMAZEAU, Yves. **Coordination Patterns in Multi-Agent Worlds:** Applications to Computer Vision and Robotics. London : IEE, 1991.

[DEM 91a] DEMAZEAU, Yves; MÜLLER, Jean-Pierre. From Reactive to Intentional Agents. In: DEMAZEAU, Yves; MULLER, Jean-Pierre (Eds.). **Decentralized A.I. 2**. Amsterdam: North-Holland, 1991.

[DEM 93] DEMAZEAU, Yves. La Plate-forme PACO et ses applications. In: JOURNÉE NATIONALE DU PRC-IA SUR LES SYSTÈMES MULTI-AGENTS, 2., 1993, Montpellier, FR. **Annales...** [S.l. : s.n.], 1993.

[DEM 93a] DEMAZEAU, Yves. Distributed Artificial Intelligence & Multi-Agent Systems. In: SIMPÓSIO BRASILEIRO DE INTELIGÊNCIA ARTIFICIAL, 10., 1993, Porto Alegre. **Anais...** Porto Alegre: SBC, 1993.

[DRO 91] DROGOUL, Alexis. From Animals to Animats**.** In: MEYER, J. A.; WILSON, S.W. (Eds.). In: INTERNATIONAL CONFERENCE ON THE SIMULATION OF ADAPTIVE BEHAVIOR, 1., 1991. **Proceedings...** [S.l.] : MIT Press, 1991.

[DRO 92] DROGOUL, Alexis; DUBREUIL, Christophe. Eco-Problem-Solving Model: Results of the N-Puzzle. In: WERNER, Eric; DEMAZEAU, Yves (Eds.). **Decentralized A.I. 3**. Amsterdam: North-Holland, 1992.

Annales... [S.l. : s.n.], 1995.

[FEI 92] FERGUSON, I.A. Towards an architecture for adaptive, rational, mobile agents. In: WERNER, Eric; DEMAZEAU, Yves (Eds.). **Decentralized A.I.3**. Amsterdam: North-Holland, 1992.

[FER 91] FERBER, Jacques; JACOPIN, Eric. The Framework of Eco-Problem Solving. In: DEMAZEAU, Yves; MULLER, Jean-Pierre (Eds.). **Decentralized A.I. 2**, Amsterdam: North-Holland, 1991.

[FER 92] FERBER, J. e DROGOUL, Alexis. Using Reactif Multi-Agent Systems in Simulation and Problem Solving. In: **Distributed Artificial Intelligence**: Theory and Praxis. Bruxelas : ECSC-EEC-EAEC, 1992.

[GAS 91] GASSER, Les. Social Conceptions of Knowledge and actions: DAI Foundations and Open Systems Semantics. **Artificial Intelligence**, Amsterdam, v.47, n. 1-3, Jan. 1991.

[GMY 91] GMYTRASIEWICZ, Piotr J.; DURFEE, Edmund H.; WEHE David K.. The Utility of Communication in Coordinating Intelligent Agents. In: NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE - AAAI, 9., 1991. **Proceedings...** Menlo Park: AAAI Press, 1991. v.1.

[HIL 96] HILLIS, W. Daniel. **Intelligence as an Emergent Behaviour, or "The Songs of Eden".** Disponível por WWW em http://http3.brunel.ac.uk:8080/deptos/AI/sophia/al-hilli.html (20 de outubro de 1996).

[HUB 95] HUBNER, Jomi Fred. **Migração de Agentes em Sistemas Multi-Agentes Abertos.** Porto Alegre: CPGCC da UFRGS, 1995. Dissertação de Mestrado.

[INT 95] INTERNATIONAL CONFERENCE ON MULTI-AGENT SYSTEMS - ICMAS, 1., 1995, San Francisco. **Proceedings...** Menlo Park: AAAI Press, 1995.

[JAE 95] JAEGER, Trent; PRAKASH, Atul. Representation and Adaptation of Organization Coordination Knowledge for Autonomous Agent Systems. In: CONFERENCE ON SOFTWARE ENGINEERING AND KNOWLEDGE ENGINEERING, 7., 1995, Rockville. **Proceedings...** [S.l. : s.n.], 1995.

[JEN 94] JENNINGS, Nick R. **Cooperation in Industrial Multi-Agent Systems**. Singapore: World Scientifica, c1994. (World Scientific Series in Computer Science, v.43).

[JEN 95] JENNINGS, Nick R. Agent Software. In: UNICON SEMINAR ON AGENT SOFTWARE, 1995, London, UK. **Proceedings…** [S.l. : s.n], 1995. p. 12-27.

[JEN 95a] JENNINGS, N. R.; CORERA, J. M.; LARESGOITI, I. Developing Industrial Multi-Agent Systems. In: INTERNATIONAL

CONFERENCE ON MULTI-AGENT SYSTEMS - ICMAS, 1., 1995, San Francisco. **Proceedings...** Menlo Park: AAAI Press, 1995.

[LAB 93] LABIDI, S.; LEJOUAD, W. **De L'Intelligence Artificielle Distribuée aux Systèmes Multi-Agents**. [S.l.] : Institut National de Recherche en Informatique et en Automatique - INRIA, 1993.

[MAE 91] MAES, Pattie. The agent network architecture (ANA). **SIGART Bulletin**, [S.l.], p 115-120, 1991.

[MAE 93] MAES, Pattie. Behavior-Based Artificial Intelligence. In: INTERNATIONAL CONFERENCE ON THE SIMULATION OF ADAPTIVE BEHAVIOR, 2., 1993. **Proceedings...** Cambridge: MIT Press, 1993.

[MAE 93a] MAES, Pattie. **Designing Autonomous Agents**: Theory and Practice From Biology to Engineering and Back. Cambridge: The MIT Press, 1993.

[MAG 95] MAGNIN, Laurent. **Etude d'organisations chez les robots footballeurs** (projet Microb). Disponível por WWW em http://www-laforia.ibp.fr/~magnin/these/publis/pub_Marcia9.95.html (22 de setembro de 1995).

[MAG 96] MAGNIN, Laurent. **SIEME - Simulateur d'environnement pour systèmes multi-agents**. Disponível por WWW em http://www-laforia.ibp.fr/~magnin/these/sieme.html (18 de novembro de 1996).

[MIE 96] MINAR, Nelson et al. **The SWARM Simulation System: A Toolkit for Building Multi-AgentSimulations**. Disponível por WWW em http://www.santafe.edu/projects/swarm/overview/overview.html (18 de novembro de 1996).

[MIN 88] MINSKY, Marvin. **The Society of Mind**. New York: A Touchstone book, 1988.

[MOU 95] MOULIN, Bernard; CLOUTIER, Louis. Une méthode de conception de systèmes multiagents réactifs basée sur la notion de scénario. In: JOURNÉES FRANCOPHONES SUR L'INTELLIGENCE ARTIFICIELLE DISTRIBUÉE ET LES SYSTÈMES MULTI-**Annales...** [S.l. : s.n.], 1995.

[MUL 94] MULLER, J.P.; PISCHEL, M.; THIEL, M. A pragmatic approach to modelling autonomous interacting systems. In: WOOLDRIDGE, M.; JENNINGS, N.R. (Eds.). WORKSHOP ON AGENT THEORIES, ARCHITECTURES AND LANGUAGES, 1994, Amsterdam. **Proceedings...** [S.l. : s.n.], 1994.

[MUL 94a] MULLER, J.P.; PISCHEL, M. Modelling interacting agents in dynamic environments. In: EUROPEAN CONFERENCE ON ARTIFICIAL INTELLIGENCE (ECAI-94), 11., 1994, Amsterdam. **Proceedings...** [S.l. : s.n.], 1994.

[OLI 96] OLIVEIRA, Flávio Moreira de. Inteligência Artificial Distribuída. In: ESCOLA REGIONAL DE INFORMÁTICA, 4., 1996, Canoas, RS; **Anais...** Canoas: SBC, 1996.

[REG 95] REGNIER, S.; DUHAUT, D. Une approche multi-agents pour la resolution de problèmes robotiques. In: JOURNÉES FRANCOPHONES SUR L'INTELLIGENCE ARTIFICIELLE DISTRIBUÉE ET LES SYSTÈMES MULTI-AGENTS, 3., 1995, Chambéry, FR. **Annales...** [S.l. : s.n.], 1995.

[ROD 90] RODA, C.; JENNINGS, N.R.; MANDANI, E.H. ARCHON: A Cooperation Framework for Industrial Process Control. In: WORKING CONFERENCE ON COOPERATING KNOWLEDGE BASED SYSTEMS, 1990. **Proceedings...** [S.l.] : Springer-Verlag, 1990.

[ROT 84] ROTH, B. Hayes. **A Blackboard Model of Control**. Stanford, CA : Stanford University, 1984. (Technical Report HPP83-38).

[SHO 90] SHOHAM, Yoav. **Agent-oriented Programming**. Stanford, CA : Computer Science Department, Stanford University, 1990.

[SHO 93] SHOHAM, Yoav. Agent-oriented Programming. **Artificial Intelligence**, Amsterdam, v.60, p. 51-92, 1993.

[SIC 92] SICHMAN, Jaime; DEMAZEAU, Yves; BOISSIER, Olivier. When Can Knowledge-Based Systems Be Called Agents ? In: BRAZILIAN SYMPOSIUM ON ARTIFICIAL INTELLIGENCE, 9., 1992. **Proceedings...** Rio de Janeiro : [s.n.], 1992.

[SMI 88] SMITH, R.G. The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver. In: BOND, A.H.; GASSER, L. (Eds.). **Readings in Distributed Artificial Intelligence**. San Mateo: Morgan Kaufmann, 1988. p. 357-366.

[STE 90] STEELS, Luc. Cooperating Between Distributed Agents Through Self-Organisation. In: DEMAZEAU, Yves; MULLER, J.P. (Eds.). **Decentralized A.I**. Amsterdam: North-Holland, 1990.

[STE 91] STEELS, Luc. Towards a Theory of Emergent Functionality. In: INTERNATIONAL CONFERENCE ON THE SIMULATION OF ADAPTIVE BEHAVIOR, 1., 1991. **Proceedings...** MIT Press, 1991.

[TRA 94] TRAVERS, Michael. Recursive Interfaces for Reactive Objects. In: CHI'94, 1994, Boston. **Proceedings...** Massachusetts: MIT Media Laboratory, 1994.

[VIC 93] VICCARI, Rosa Maria. **Fundamentos da Inteligência Artificial**. Porto Alegre: CPGCC da UFRGS, 1993. (RP-215).

[WAI 93] WAINER, Jacques. Introspection and Projection in Reasoning about Other Agents. In: SIMPÓSIO BRASILEIRO DE INTELIGÊNCIA

ARTIFICIAL, 10., 1993, Porto Alegre. **Anais...** Porto Alegre: SBC, 1993.

[WAL 95] WALKER, A.; WOOLDRIDGE, M. Understanding the Emergence of Conventions in Multi-Agent Systems**.** In: INTERNATIONAL CONFERENCE ON MULTI-AGENT SYSTEMS, 1., 1995, San Francisco. **Proceedings...** [S.l. : s.n.], 1995.

[WAV 92] WAVISH, Peter. Exploiting Emergent Behaviour in Multi-Agent Systems. In: WERNER, Eric; DEMAZEAU, Yves (Eds.). **Decentralized A.I.3**. Amsterdam: North-Holland, 1992.

[WER 91] WERNER, Eric; DEMAZEAU, Yves. In: EUROPEAN WORKSHOP ON MODELLING AUTONOMOUS AGENTS IN A MULTI-AGENT WORLD, 3., 1991, Kaiserslautern. **Proceedings...** Germany: [s.n.], 1991.

[WOO 94] WOOLDRIDGE, Michael; JENNINGS, Nicholas R.. Towards a Theory of Cooperative Problem Solving**.** In: MODELLING AUTONOMOUS AGENTS IN A MULTI-AGENT WORLD - MAAMAW-94, 1994, Odense, Denmark. **Proceedings...** [S.l. : s.n.] , 1994.

[WOO 94a] WOOLDRIDGE, Michael; JENNINGS, Nicholas R.. Formalizing the Cooperative Problem Solving Process**.** In: INTERNATIONAL WORKSHOP ON DISTRIBUTED ARTIFICIAL INTELLIGENCE, 13., 1994, Lake Quinhalt. **Proceedings...** [S.l. : s.n.], 1994.

[WOO 95] WOOLDRIDGE, Michael; JENNINGS, Nicholas R. **Intelligent Agents:** Theory and Practice. [S.l. : s.n.], 1995.