Universidade Federal do Rio Grande do Sul

Instituto de Informática

Programa de Pós-Graduação em Computação

Material de apoio

Introdução aos Sistemas Multiagentes

Prof. Luis Otavio Alvares

Instituto de Informática - UFRGS

alvares@inf.ufrgs.br

Prof. Jaime Simão Sichman

Escola Politécnica - USP

jaime@pcs.usp.br

Publicado nas Jornadas de Atualização em Informática – XVI JAI

XVII Congresso da SBC, Brasília, 2-8 agosto 1997

SUMÁRIO

1. INTRODUÇÃO 4

2. INTELIGÊNCIA ARTIFICIAL DISTRIBUÍDA E SISTEMAS MULTIAGENTES 5

2.1 A METÁFORA DE INTELIGÊNCIA 5
2.2 CONCEITOS BÁSICOS 5
2.3 BREVE HISTÓRICO 6
2.4 RESOLUÇÃO DISTRIBUÍDA DE PROBLEMAS E SISTEMAS MULTIAGENTES 7
2.4.1 Resolução Distribuída de Problemas 7
2.4.2 Sistemas Multiagentes 8
2.4.3 Uma comparação entre as sub-áreas 10
2.5 PRINCIPAIS PROBLEMAS ABORDADOS EM IAD 10

3. SISTEMAS MULTIAGENTES REATIVOS 12

3.1 CARACTERÍSTICAS DOS AGENTES REATIVOS 12
3.2 O MODELO DA FUNCIONALIDADE EMERGENTE 13
3.2.1 Exemplo: Os robôs mineradores 14
3.3 O MODELO DA ECO-RESOLUÇÃO 16
3.3.1 Exemplo: Quebra-cabeça dos 8 17
3.4 O MODELO PACO 18
3.4.1 Exemplo: generalização cartográfica 20
3.5 COMPARAÇÃO ENTRE OS MODELOS 21
3.6 AMBIENTES DE DESENVOLVIMENTO 22
3.6.1 O Sistema SIEME 23
3.6.2 O Sistema SWARM 23
3.6.3 O sistema SIMULA 24

4. SISTEMAS MULTIAGENTES COGNITIVOS 28

4.1 CARACTERÍSTICAS BÁSICAS 28
4.2 ARQUITETURAS DE AGENTES 29
4.3 ORGANIZAÇÕES DE AGENTES 30
4.3.1 Modelos descendentes 30
4.3.2 Modelos baseados na utilidade 31
4.3.3 Modelos baseados na complementaridade 31
4.4 INTERAÇÃO ENTRE AGENTES 32
4.5 AMBIENTES DE DESENVOLVIMENTO E LINGUAGENS 33

5. CONCLUSÕES E PERSPECTIVAS 35

6. REFERÊNCIAS BIBLIOGRÁFICAS 37

1. Introdução

Sistemas Multiagentes é um domínio relativamente novo que tem obtido cada vez mais importância na pesquisa em Informática, seja em Inteligência Artificial, seja em Sistemas Distribuídos, em Robótica ou mesmo em Sistemas de Informação. Por ser uma área nova, é bastante difícil encontrar-se material introdutório, didático, especialmente em português. As obras disponíveis são teses, dissertações e artigos técnicos de difícil compreensão por um estudante ou um não especialista.

Este texto procura preencher esta lacuna propondo-se a fornecer uma visão geral do assunto e as bases conceituais da área, procurando facilitar a compreensão através de exemplos e indicando bibliografia suplementar para aprofundamento do estudo.

O texto está organizado em cinco capítulos. O Capítulo 2 introduz a Inteligência Artificial Distribuída (IAD) e os Sistemas Multiagentes, apresentando os conceitos básicos, um histórico e os principais problemas abordados. O Capítulo 3 trata dos sistemas multiagentes reativos, apresentando suas características principais, três modelos de sistemas multiagentes reativos e alguns ambientes de desenvolvimento. O Capítulo 4 aborda os sistemas multiagentes cognitivos, apresentando suas características básicas, a arquitetura dos agentes, formas de organização dos agentes, modos de interação entre os agentes e alguns ambientes de desenvolvimento. Finalmente, o Capítulo 4 apresenta conclusões e perspectivas da área.

2. Inteligência Artificial Distribuída e Sistemas Multiagentes

Neste capítulo, serão introduzidos os conceitos básicos da área de Inteligência Artificial Distribuída (IAD). Inicialmente, será feita uma comparação entre a IAD e os estudos clássicos de Inteligência Artificial (IA). A seguir, serão introduzidos alguns conceitos básicos, como *agentes, sociedades, ambientes, organizações* e *interações*, que serão utilizados nos capítulos subseqüentes. Um breve histórico da área será então apresentado. Serão introduzidas então as duas sub-áreas da IAD, a Resolução Distribuída de Problemas (RDP) e os Sistemas Multiagentes (SMA), onde serão explicitadas suas principais diferenças. Finalmente, os principais problemas de pesquisa da área serão então abordados.

2.1 A metáfora de inteligência

A IAD tornou-se nos últimos anos um domínio de pesquisa muito promissor. Enquanto estudos clássicos de IA tomam como modelo de inteligência o *comportamento individual humano*, cuja ênfase é colocada em representação de conhecimento e métodos de inferência, o modelo de inteligência utilizada em IAD é baseada no *comportamento social*, sendo que a ênfase é colocada em ações e interações entre agentes [Sichman et al. 92]. A metáfora utilizada em IA clássica é basicamente de origem psicológica: enquanto que aquela utilizada em IAD é de natureza sociológica/etológica.

Na vida real, podem-se observar várias situações onde costumamos atribuir um comportamento dito inteligente a entidades coletivas: uma empresa competitiva num determinado segmento de mercado, uma equipe de futebol que vence a maioria de seus adversários e até mesmo uma colônia de formigas em busca de alimento. Pode-se notar que neste último caso, tal comportamento é uma noção essencialmente ligada ao coletivo: poucos atribuiriam a uma única formiga a propriedade da inteligência.

Uma abordagem deste tipo torna-se desejável quando se deseja resolver problemas grandes e complexos, que requeiram conhecimento de vários domínios de conhecimento distintos, e que algumas vezes envolvam coleta de dados fisicamente distribuídos, como por exemplo em sistemas de controle de espaço aéreo. Do ponto de vista estritamente computacional, uma abordagem distribuída permite o controle da complexidade, uma operação degradada satisfatória e suporte para evoluções e eventuais replicações [Chandrasekaran 81].

2.2 Conceitos básicos

Em [Demazeau 95], encontra-se proposta uma decomposição básica de um sistema segundo uma metodologia de IAD. Esta decomposição, cujos principais conceitos serão introduzidos a seguir, será utilizada em todo o restante do texto, tendo sido inicialmente apresentada em [Sichman 95].

Dado um determinado sistema, denomina-se *agente* cada uma de suas entidades ativas. O conjunto de agentes forma uma *sociedade*. As entidades passivas serão designadas pelo termo *ambiente*. Um agente raciocina sobre o ambiente, sobre os outros agentes e decide racionalmente quais objetivos deve perseguir, quais ações deve tomar etc. [Pleiad 92]. A utilização do termo agente pressupõe portanto uma noção subjacente de controle [Boissier 93]. De fato, o termo ativo tem como objetivo sinalizar que o conceito de agente não corresponde a noções estáticas como módulos, conjunto de regras e bases de conhecimento, sem que a estas esteja intimamente relacionado um mecanismo de controle para a sua ativação.

Fazendo uma analogia grosseira com a área de sistemas distribuídos, um agente corresponderia à noção de processo, a sociedade corresponderia ao conjunto de processos e o ambiente ao conjunto de entidades do mundo, excluindo os outros processos, com os quais o sistema tem uma relação. Entretanto, quando se utiliza os termos agente e sociedade, a idéia é a de abordar um nível conceitual mais abstrato do que aquele tratado por processos. Na verdade, pode-se considerar que o nível de abstração tratado é aquele conhecido como nível de conhecimento ("knowledge level"), apresentado em [Newell 82].

O termo *interação* designa as trocas de informação que podem ocorrer entre os agentes. Como será visto mais tarde, tais trocas podem ser realizadas de modo direto (por comunicação explícita, por exemplo) ou de modo indireto (pela emissão de sinais através do ambiente, por exemplo). Cabe ressaltar entretanto que tal termo tem um significado diverso da noção de comunicação, tal como é utilizada em sistemas distribuídos. O interesse em IAD é o de definir que primitivas de comunicação podem exprimir a natureza de um trabalho cooperativo e como tais primitivas podem ser combinadas em protocolos de utilização genérica, não necessariamente ligados a uma aplicação particular.

Finalmente, o conceito de *organização* exprime as restrições aplicadas aos agentes pertencentes a uma sociedade, isto é, os meios através dos quais o projetista de tais sistemas (ou os próprios agentes) podem garantir que cada agente desejará realizar e efetivamente realizará o que deve ser feito e no bom momento. Várias alternativas de representação deste conceito já foram propostas e exploradas, como por exemplo estruturas de autoridade, táticas de controle, restrições sobre canais de comunicação, modelos do tipo pares (conhecimento, ação), etc.

Cabe ressaltar que devido ao fato da IAD ser uma disciplina relativamente recente, não existe ainda uma definição única do conceito de agentes que seja aceita por todos os pesquisadores da área. Para ilustrar este fato, encontram-se a seguir duas definições genéricas que abordam aspectos complementares deste conceito:

- Um agente é uma entidade *real* ou *virtual*, imersa num *ambiente* sobre o qual ela é capaz de agir, que dispõe de uma capacidade de *percepção* e de *representação* parcial deste ambiente, que pode se *comunicar* com outros agentes, e que possui um *comportamento autônomo*, conseqüência de suas observações, de seu conhecimento e da suas interações com os outros agentes [Ferber e Gasser 91].
- Um agente é uma entidade à qual nós podemos associar uma *identidade única,* e que é capaz de executar um processamento de cálculo. Um agente pode ser considerado como um *meio* que produz um certo número de ações, a partir de seu conhecimento e dos mecanismos internos que lhe são próprios [Gasser 92].

2.3 Breve histórico

As primeiras tentativas para resolver problemas de modo cooperativo ocorreram na década de 70 [Ferber e Gasser 91]. Uma destas primeiras tentativas foi o projeto HEARSAY-II [Erman et al. 80], um sistema para o reconhecimento de fala que originou o modelo de resolução de problemas conhecido como quadro-negro (“blackboard”) [Engelmore e Morgan 88].Nesta mesma época, surgem alguns modelos computacionais como os “beings” [Lenat 75] e os atores [Agha e Hewitt 85], desenvolvidos com o objetivo de tratar problemas relacionados com recursos compartilhados, estruturas de controle complexas e o aparecimento de comportamentos sofisticados a partir de interações muito simples.

No início da década de 80, nota-se pela primeira vez a importância da representação explícita, no seio de um agente, do conhecimento sobre outros agentes em [Konolige e Nilsson 80], num contexto de planejamento. A utilização de modelos organizacionais humanos em sistemas computacionais é discutida em [Fox 81], e surge o modelo da rede contratual [Smith 80], um protocolo de negociação baseado na noção de mercado econômico. Nesta época, as técnicas de orientação a objetos começam a tornar-se populares [Stefik e Bobrow 86]. Na área de robótica, são testados os chamados robôs reativos[Brooks 86], cuja idéia básica é a de utilizar uma arquitetura simples, baseada em comportamentos elementares, para projetar um robô autônomo. O conceito de sistema aberto também aparece pela primeira vez nesta época [Hewitt 86], crucial para sistemas Multiagentes. Finalmente, alguns ambientes computacionais foram desenvolvidos para testar e implementar estas idéias, como DVMT [Corkill e Lesser 93], DPSK [Cardozo 87] e MACE [Gasser et al. 87].

2.4 Resolução Distribuída de Problemas e Sistemas Multiagentes

Como já dito anteriormente, a IAD se divide em duas sub-áreas, a RDP e os SMA. A seguir, serão detalhadas as principais motivações científicas e características de cada uma delas, conforme apresentado inicialmente em [Sichman 95].

2.4.1 Resolução Distribuída de Problemas

A motivação inicial de uma resolução distribuída de problemas é um *problema inicial preciso* que deve ser solucionado. A estratégia de resolução apresenta as seguintes características:

- o problema é resolvido por um conjunto de agentes, fisicamente distribuídos em diversas máquinas conectadas via rede. Tais agentes são concebidos para solucionar um determinado problema particular;
- uma organização destes agentes é então concebida para restringir o comportamento destes agentes. Tal organização, na maioria dos casos, é completamente definida durante a fase de concepção do sistema;
- a interação entre os agentes é realizada seja por troca de mensagens, seja por meio do compartilhamento de dados comuns. A estrutura destas trocas é quase sempre definida completamente durante a fase de concepção do sistema, sendo intimamente relacionada ao modelo algorítmico sub-jacente (como por exemplo, o quadro-negro) e ao problema que o sistema deve resolver;
- os agentes são executados de modo concorrente, para aumentar a velocidade de resolução;
- os agentes cooperam, dividindo entre si as diversas partes do problemas original (sub-problemas, tarefas), ou podem até mesmo aplicar diferentes estratégias de resolução para uma mesma tarefa;

- existe a noção de um *controle global*, na maior parte dos casos implícito aos agentes, que garante um comportamento global coerente do sistema, conforme a organização inicialmente prevista. Tal controle pode ser implementado quer de modo centralizado (por exemplo, pela criação de um agente responsável pela gerência do sistema), quer de modo distribuído.

A abordagem RDP encontra-se representada na Figura 2.1 abaixo .Do ponto de vista de concepção do sistema, os agentes não existem a priori: sua concepção, bem como a de sua organização e de suas interações, é realizada face à existência de um problema que o sistema deve solucionar. Não existe preocupação quanto à reutilização dos agentes num outro contexto. De certo modo, podemos considerar os sistemas de RDP como um cruzamento de técnicas de sistemas distribuídos e de IA, que aplicam técnicas de coordenação e sincronização desenvolvidos na primeira área para integrar sistemas concebidos segundo a metodologia proveniente da segunda área.

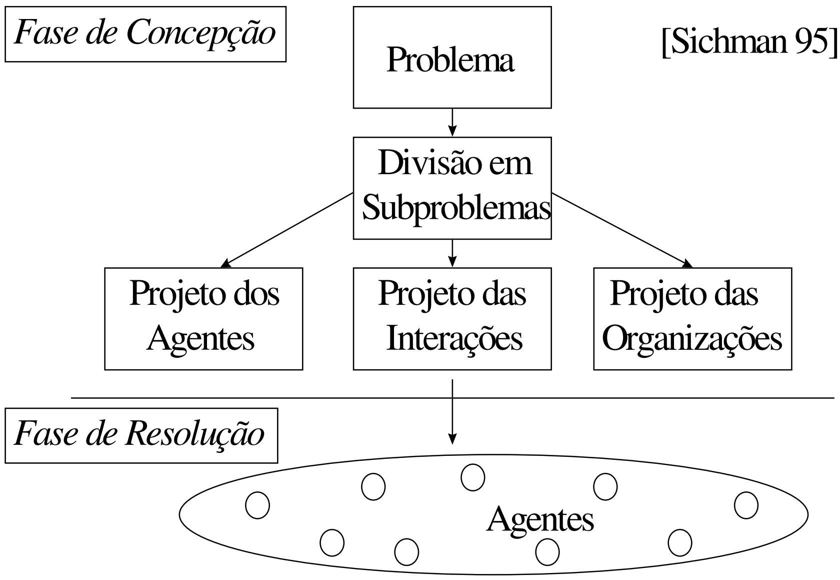


Figura 2.1 - A abordagem RDP

2.4.2 Sistemas Multiagentes

A problemática científica tratada em SMA é diversa da anterior. A área de SMA se interessa pelo estudo de agentes *autônomos* em um universo Multiagentes [Demazeau e Müller 90]. O termo autônomo designa aqui o fato de que os agentes têm uma existência própria, independente da existência de outros agentes. Como não existe um problema definido a priori que o sistema deve resolver, o objetivo da área é estudar modelos genéricos a partir dos quais podem-se conceber agentes, organizações e interações, de modo a poder instanciar tais conceitos quando se deseja resolver um problema particular. Dito de um outro modo, o objetivo é conceber os meios a partir dos quais pode-se assegurar que agentes desejem cooperar e efetivamente o façam, com o intuito de resolver um problema específico quando este for apresentado ao sistema.

Do ponto de vista de concepção do sistema, esta abordagem apresenta as seguintes características:

- os agentes são concebidos independentemente de um problema particular a ser resolvido. O projeto de um agente deve resultar numa entidade capaz de realizar um determinado processamento, e não numa entidade capaz de realizar este processamento exclusivamente no contexto de uma aplicação alvo particular;

- a concepção das interações também é realizada independentemente de uma aplicação alvo particular. Busca-se desenvolver protocolos de interação genéricos, que possam ser reutilizados em várias aplicações similares. Um exemplo de um tal protocolo seria, por exemplo, um protocolo de apresentação de um agente quando este ingressa numa sociedade. Obviamente, um protocolo deverá ser instanciado com dados do domínio do problema para poder ser efetivamente utilizado numa aplicação;

- a mesma filosofia anterior pode ser estendida ao projeto das organizações. Normalmente, se distingue as funcionalidades necessárias a uma resolução particular dos agentes que irão efetivamente implementar tais funcionalidades;

- durante a fase de resolução, os agentes utilizam suas representações locais dos protocolos de interação e das organizações para raciocinar e agir. Deste modo, não existe um controle global do sistema, este é implementado de forma totalmente decentralizada nos agentes.

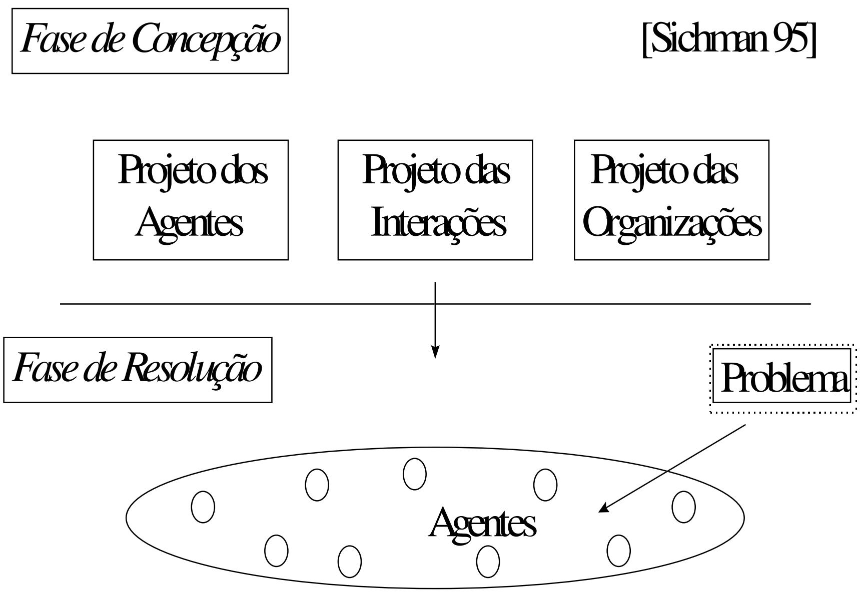


Figura 2.2 - A abordagem SMA

A abordagem SMA encontra-se representada na Figura 2.2. Do ponto de vista de concepção do sistema, os agentes, suas interações e sua organização no seio da sociedade são concebidas independentemente de um problema particular a ser resolvido. Torna-se possível, portanto, a reutilização de tais componentes quando se desejar projetar aplicações similares. Os agentes irão instanciar dinamicamente as organizações e interações quando um problema for apresentado ao sistema. De certo modo, pode-se considerar que certas propriedades globais do sistema que eram totalmente pré-definidas pelo projetista na abordagem RDP, e que eram também ligadas ao modelo algorítmico subjacente, são agora representadas de modo explícito e efetivamente utilizadas pelos próprios agentes do sistema.

2.4.3 Uma comparação entre as sub-áreas

Em [Durfee e Rosenchein 94], apresentam-se alternativamente 3 relações possíveis entre as sub-áreas de RDP e SMA:

- RDP é um *subconjunto* de SMA: pode-se considerar a RDP como um caso particular de SMA onde algumas hipóteses adicionais são válidas:
  - hipótese de *benevolência*: os agentes efetivamente desejam cooperar uns com os outros a priori, eles jamais se colocam a questão se devem ou não cooperar uns com os outros;
  - hipótese de *objetivo comum*: a hipótese anterior pode ser refinada considerando que todos os agentes têm um objetivo comum a priori, que é o de resolver o problema específico para o qual o sistema foi criado;
  - hipótese de *concepção centralizada*: tal noção é mais recente, e concerne o fato que um projetista (ou grupo de projetistas) concebe um sistema cujo objetivo é o de resolver um certo problema determinado;
- SMA *fornece uma base* para RDP: segundo este ponto de vista, a área de SMA se dedica ao estudo de certas propriedades internas aos agentes. Por exemplo, a decisão de cooperar ou não, a honestidade na comunicação de informações etc. Estas propriedades, quando implementadas de modo decentralizado nos agentes, podem fornecer uma base para a concepção de interações e organizações de agentes, num contexto de um sistema que deve resolver um problema particular, seguindo a abordagem de RDP;
- RDP e SMA são agendas de pesquisa complementares: mesmo considerando os pontos de vista anteriores, por vezes se torna difícil, senão impossível, a um observador externo de discernir se um dado sistema foi construído segundo a abordagem RDP ou SMA. Este fato indica que talvez possa ser mais conveniente diferenciar as sub-áreas não pelo comportamento global do sistema, mas sim pelas motivações científicas e técnicas dos projetistas e pela natureza das experimentações a serem realizadas.

2.5 Principais problemas abordados em IAD

Segundo [Bond e Gasser 88], os principais problemas abordados na área de IAD são os seguintes:

- *descrição, decomposição e alocação de tarefas*: como deve-se descrever e decompor uma tarefa complexa em sub-tarefas, de modo estático ou dinâmico, como estas sub-tarefas devem ser alocadas e em que ordem devem ser executadas;
- *interação, linguagem e comunicação*: quais primitivas básicas de comunicação devem ser utilizadas para exprimir os conceitos semânticos envolvidos num trabalho cooperativo e como estas primitivas podem ser combinadas em protocolos de resolução de problemas;
- *coordenação, controle e comportamento coerente*: como se pode garantir um comportamento global coerente numa coleção de agentes, cada qual com seus próprios objetivos e habilidades, e como um controle de tal sistema pode ser projetado;

- *conflito e incerteza*: considerando que os agentes não têm uma informação completa sobre o seu ambiente, como os conflitos que surgem a partir de visões locais podem ser detectados e resolvidos e como dados incertos e incompletos devem ser manipulados de modo a garantir resultados globais coerentes;
- *linguagens e ambientes de programação*: do ponto de vista computacional, que linguagens de programação devem ser utilizadas em tais sistemas e quais requisitos um ambiente de programação deve obedecer para possibilitar o desenvolvimento e teste de tais sistemas.

Obviamente, tais questões também têm sido investigadas por pesquisadores de outras áreas, como por exemplo em sistemas distribuídos e em lógica, há muito tempo. Pode-se considerar entretanto que a área de IAD consiste numa tentativa de resolver tais problemas de modo conjunto, num mesmo contexto científico.

3. Sistemas Multiagentes Reativos

A abordagem reativa foi introduzida por Brooks [Brooks 86] no domínio da robótica. Ante a falta de resultados satisfatórios dos robos baseados em sistemas cognitivos, principalmente em ambientes dinâmicos ou desconhecidos, ele propôs robôs bem mais simples, baseados em ações elementares ativadas diretamente em resposta a determinados estímulos (percepção), segundo uma prioridade pré-definida.

Neste capítulo serão apresentadas as principais características dos agentes reativos, três modelos de sistemas multiagentes reativos e três ambientes de desenvolvimento.

3.1 Características dos agentes reativos

Há necessidade de agentes complexos para realizar uma tarefa complexa? Os defensores dos sistemas multiagentes reativos dizem que não. O exemplo clássico é uma colônia de formigas. Cada formiga isoladamente é uma entidade bem simples; não se atribui inteligência a uma formiga. Entretanto, o trabalho realizado por uma colônia de formigas é bem complexo: procura de alimento, transporte do alimento até o formigueiro, cuidado com os ovos e larvas, defesa da colônia, etc. E elas têm tido êxito, adaptando-se às mudanças de condições, como os períodos glaciais, e sobrevivendo há muitos milhões de anos.

Através do exemplo das formigas, pode-se destacar as principais características dos agentes e dos sistemas multiagentes reativos:

- não há representação explícita de conhecimento - o conhecimento dos agentes é implícito e se manifesta através do seu comportamento;

- não há representação do ambiente - o seu comportamento se baseia no que é percebido a cada instante do ambiente, mas sem uma representação explícita deste;

- não há memória das ações - os agentes reativos não mantém um histórico de suas ações, de forma que o resultado de uma ação passada não exerce nenhuma influência sobre as suas ações futuras;

- organização etológica - a forma de organização dos agentes reativos é similar a dos animais, em oposição à organização social dos sistemas cognitivos;

- grande número de membros - os sistemas multiagentes reativos têm, em geral, um grande número de agentes, da ordem de dezenas, centenas ou mesmo milhões de agentes.

Portanto, os agentes reativos são muito simples e não possuem representação de seu ambiente. Desta forma, todas as informações relativas ao seu comportamento estão no ambiente e suas reações dependem unicamente de sua percepção deste ambiente. Eles não possuem registro de suas ações passadas nem podem antecipar (planejar) o futuro. Vivem no "aqui e agora". Mas, apesar desta simplicidade os tornar "indefesos" isoladamente, os agentes reativos devem sua força ao trabalho em grupo, que possibilita realizarem tarefas de que não seriam capazes individualmente.

Os métodos clássicos de resolução de problemas definem um problema de uma maneira global e métodos de resolução que se aplicam diretamente sobre esta definição. Os modelos de sistemas multiagentes reativos, ao contrário, concebem o problema como sendo um conjunto de agentes em interação, cada um deles com os seus próprios objetivos (comportamentos) individuais.

As próximas seções apresentam três modelos de sistemas multiagentes reativos: o modelo da Funcionalidade Emergente [Steels 90], a Eco-Resolução [Ferber e Jacopin 91] e o modelo PACO [Demazeau 90].

3.2 O Modelo da Funcionalidade Emergente

O modelo da Funcionalidade Emergente é apresentado em [Steels 90, Steels 91]. Este modelo utiliza como idéias principais a da arquitetura de subsunção ("subsumption architecture") [Brooks 86], que estabelece prioridades entre a execução dos comportamentos atribuídos aos agentes, e a da funcionalidade emergente, na qual um sistema com muitos componentes simples pode exibir um comportamento, como um todo, mais organizado do que o comportamento das partes individuais e onde novas funcionalidades não expressamente previstas podem “emergir” da interação entre os agentes.

A idéia básica da arquitetura de subsunção é a decomposição do controle em camadas ou módulos que correspondem a comportamentos elementares. Cada módulo é um autômato de estado finito e existe uma hierarquia de prioridade entre eles. A Figura 3.1 esquematiza a arquitetura de subsunção. Nesta figura, as setas de entrada nos módulos correspondem a informações obtidas através de sensores e as setas de saída correspondem a ações a serem executadas. A ação oriunda do módulo superior tem prioridade sobre a ação oriunda do módulo inferior.

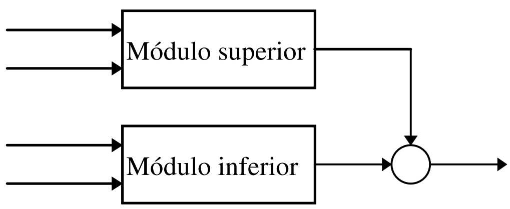


Figura 3.1: Arquitetura de subsunção

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.

Steels preconiza a construção de agentes capazes de realizar apenas alguns comportamentos elementares, mas não dá nenhuma indicação de como determinar estes comportamentos. Isto fica a cargo do projetista do sistema.

3.2.1 Exemplo: Os robôs mineradores

O exemplo clássico do modelo da funcionalidade emergente é o dos robôs mineradores. Este exemplo foi usado pela primeira vez por Luc Steels (Steels 90) mas tem sido desde então usado em várias áreas como na robótica, algoritmos genéticos e na simulação do comportamento de animais.

O objetivo é fazer um conjunto de robôs encontrar e levar para uma base central amostras de minerais. A localização das amostras e o ambiente são desconhecidos, mas tipicamente as amostras encontram-se agrupadas em "depósitos". A Figura 3.2 mostra os elementos do problema.

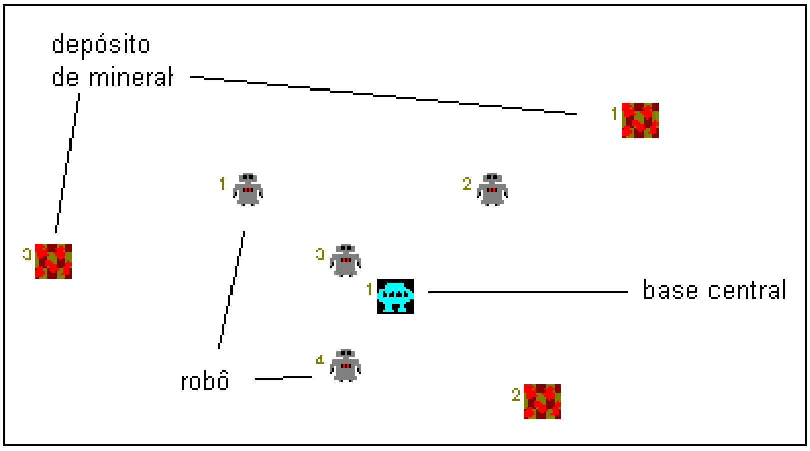


Figura 3.2: elementos do problema dos robôs coletores de mineral

Supõe-se que a base central emita um sinal decrescente com a distância (gradiente), que pode ser detectado pelos robôs e que indica o caminho de volta à base. Os robôs somente conseguem detectar os minerais a pequena distância.

Para resolver o problema proposto, Steels propõe robôs bem simples, capazes de realizar apenas cinco comportamentos elementares:

1 - evitar obstáculos

2 - se perceber um mineral e não estiver carregado (já com mineral), pegá-lo

3 - se perceber a base central e estiver carregado, descarregar

4 - se estiver carregado, seguir o sinal da base central (na direção do maior gradiente)

5 - realizar movimento randômico

A Figura 3.3 esquematiza a prioridade dos comportamentos.

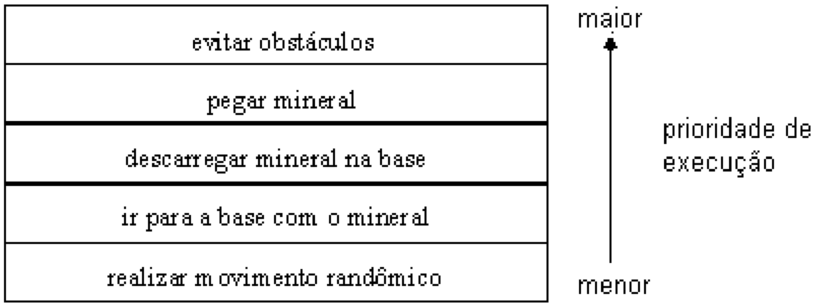


Figura 3.3: Arquitetura de subsunção do agente robô

As simulações realizadas mostram que estes robôs realizam a tarefa de levar todos os minerais para a base central. O problema é que quando um robô descarrega o mineral na base passa a procurar aleatoriamente mais minerais. Entretanto com uma pequena mudança no comportamento dos robôs consegue-se uma eficiência bem maior, como mostrado a seguir.

Altera-se o comportamento 4 e introduz-se um novo comportamento:

4 - se estiver carregado, seguir o sinal da base central (na direção do maior gradiente), deixando uma pista

6 - ao encontrar uma pista, se estiver descarregado seguir no sentido oposto ao sinal da base central (na direção do menor gradiente)

O novo esquema de prioridades passa a ser o indicado pela Figura 3.4.

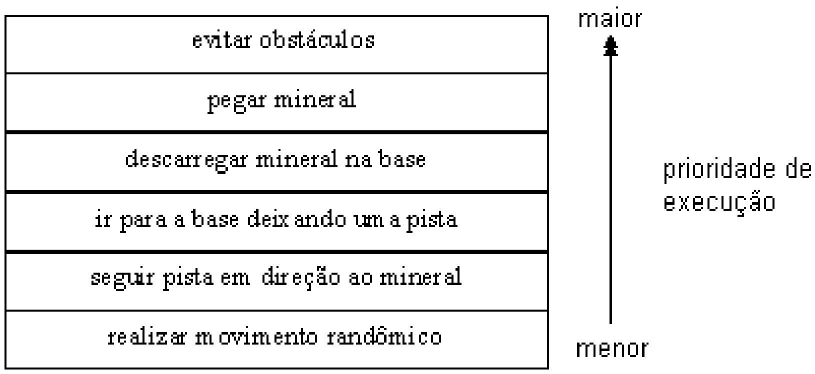


Figura 3.4: projeto do novo robô

As simulações realizadas mostram que estes robôs realizam a tarefa de forma bastante satisfatória, formando “trilhas”, representadas por bandeiras, entre os depósitos de mineral e a base central, como pode ser observado na Figura 3.5. [Drogoul 93] apresenta um estudo comparativo sobre a performance de vários tipos de robôs coletores de minerais.

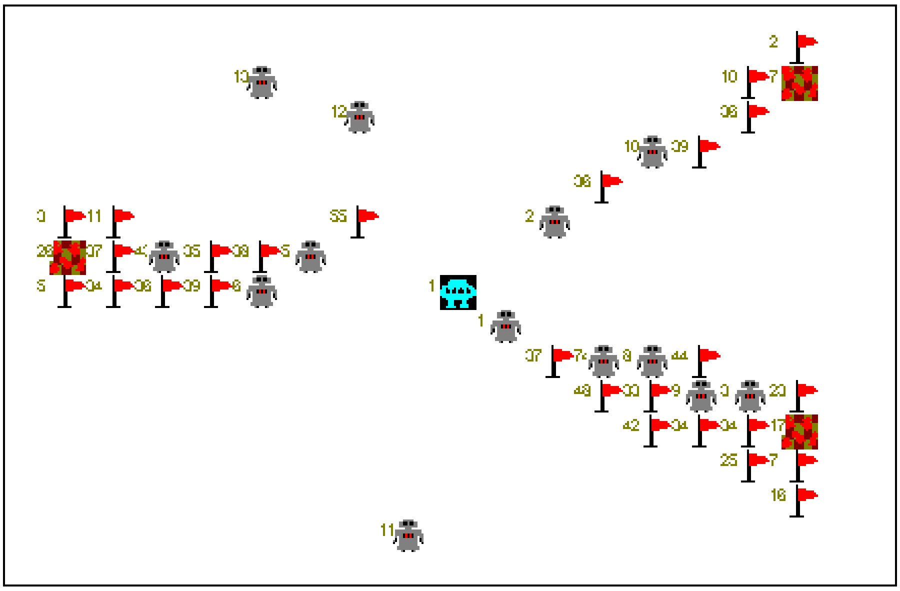


Figura 3.5: Exemplo de simulação dos robôs mineradores

3.3 O Modelo da Eco-Resolução

O modelo da Eco-Resolução [Ferber e Jacopin 91] define um problema como uma população de agentes cujo conjunto dos comportamentos tende a atingir um estado estável que é chamado de solução do problema.

Cada agente dispõe de um conjunto de comportamentos elementares predefinidos que o levam a procura de um estado de satisfação. Nesta procura por satisfação os agentes podem ser incomodados por outros agentes. Neste caso, eles agridem os que os estão incomodando, os quais são obrigados a fugir. Nesta fuga, eles podem ser levados a agredir outros agentes que os estão atrapalhando. Esta operação prossegue até que os agentes que estão incomodando se desloquem efetivamente.

Cada *eco-agente* pode ser caracterizado por:

- Um *objetivo*, que é atingir um estado de satisfação.
- Um *estado interno*: satisfeito, em busca de satisfação ou em fuga. A passagem de um estado a outro é dada por uma tabela de transição característica de um autômato de estado finito.
- *Ações elementares*, dependentes do domínio, que correspondem aos comportamentos de satisfação e de fuga dos agentes.
- Uma função de *percepção de quem o está incomodando*, isto é, o conjunto dos agentes que o está impedindo de ficar satisfeito ou de fugir.

Todo eco-agente possui dois comportamentos gerais:

- *Vontade de estar satisfeito*: O agente procura estar no seu estado de satisfação. Para isto ele executará seu comportamento de satisfação. Se ele não consegue atingir o estado de satisfação por estar impedido por outros agentes, ele os agride.

- *Obrigação de fugir*: Quando um eco-agente é atacado, ele tem obrigação de fugir, através do seu comportamento de fuga.

A especificação de eco-agentes para aplicações particulares corresponde à definição de seu estado de satisfação, de seu comportamento de satisfação e de seu comportamento de fuga.

3.3.1 Exemplo: Quebra-cabeça dos 8

O quebra-cabeça é um problema célebre em IA e em Resolução de Problemas porque é simples de modelar em termos de estados e de transições entre estados, mas muito difícil de solucionar em virtude do número de estados possíveis.

Consiste de um tabuleiro quadrado de dimensões n x n contendo n^2 - 1 peças e um lugar vazio chamado "branco". A única operação autorizada consiste em deslocar uma peça adjacente ao branco sobre o branco. A tarefa global consiste em rearranjar as peças até atingir uma configuração desejada, a partir de uma configuração inicial qualquer. A planificação clássica e a pesquisa heurística não parecem capazes, mesmo em máquinas poderosas, de resolver instâncias do problema de ordem superior a 13 (168 peças), em função principalmente do tamanho do espaço de estados gerados pelo planificador (algoritmo A* e variantes).

Mudando o enfoque, passa-se de uma abordagem tradicional *orientada a estados* para uma abordagem *orientada a agentes*, baseada na Eco-Resolução, descrita em [Drogoul e Dubreuil 92]. Neste caso, o agente peça tentará se satisfazer empurrando as peças que estão no seu caminho. E toda peça agredida procurará simplesmente deixar o caminho livre à peça que a está empurrando, se necessário agredindo por sua vez outra peça que a atrapalhe na fuga.

A escolha do movimento de uma peça (agente) será realizada com base em duas informações: a distância de seu objetivo e a distância do branco.

Para acelerar a convergência, estabeleceu-se uma ordem de resolução que corresponde a ordem em que as peças tentarão ocupar o lugar correspondente ao seu objetivo. Esta ordem é mostrada na Figura 3.6.

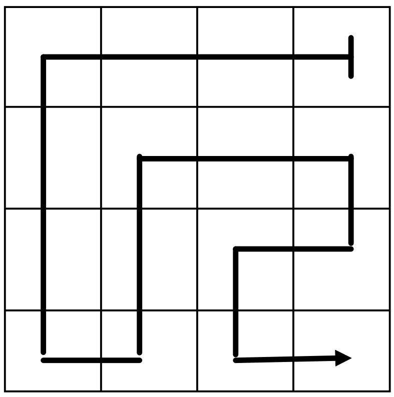


Figura 3.6: Ordem de posicionamento das peças

O estado de satisfação corresponde a estar na posição que é seu objetivo.

Os comportamentos elementares do agente peça são:

- *comportamento de satisfação*: Ir para a casa mais próxima do objetivo. Se houver duas casas eqüidistantes do objetivo, ir para a mais perto do branco.

- *comportamento de fuga*: Ir para o seu objetivo, se for adjacente. Senão, ir para a casa mais próxima do branco. Se houver duas casas eqüidistantes do branco, ir para a mais próxima de seu objetivo. Restrições: não ir para a casa que é o objetivo do agressor e não ir para a casa que é o objetivo do antecessor do agressor na ordem de preenchimento.

Um caso clássico na resolução do problema é o problema do canto. Com a solução adotada, este problema é resolvido automaticamente, como mostra a Figura 3.7.

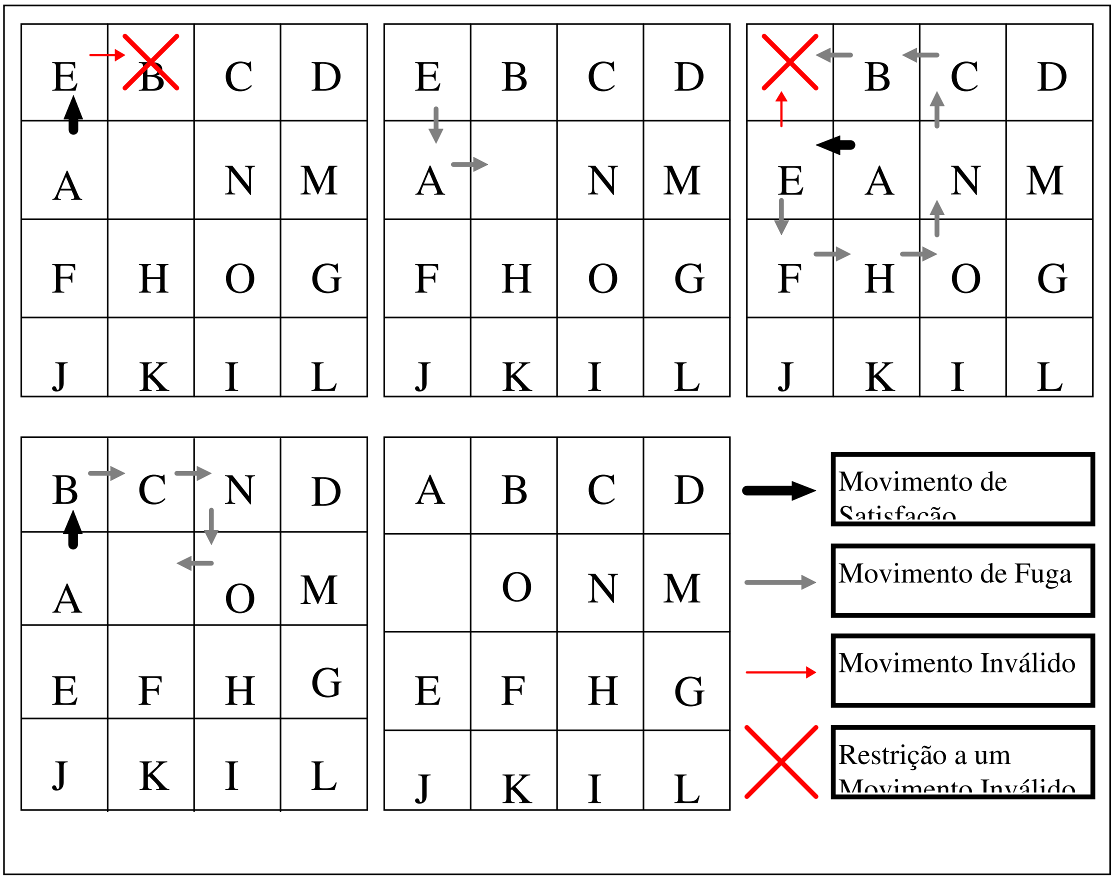


Figura 3.7: Problema do "canto".

Com a solução proposta, podem ocorrer *loops*. Por exemplo, uma peça em fuga (e que aguarda que o lugar escolhido para a fuga se libere) se encontra novamente agredida na continuação da agressão que ela realizou. Neste caso, todas as peças fazendo parte do laço gerado se agredirão indefinidamente, pois por ocasião da segunda agressão farão a mesma escolha de lugar para fugir. Uma maneira simples de solucionar este problema é considerar as peças em fuga temporariamente bloqueadas.

[Drogoul 93] apresenta uma tabela com os resultados experimentais do sistema que inclui configurações do jogo até 899 peças (30 x 30). Outra vantagem desta solução é a sua validade para qualquer tamanho do jogo e mesmo para tabuleiro retangular (número de linhas diferente do de colunas).

3.4 O Modelo PACO

O modelo PACO (COordination PAtterns) proposto em [Demazeau 90, Demazeau 93] tem sido utilizado principalmente em aplicações espacializadas como análise de imagens, cartografia e no controle de trajetória de robôs. Em lugar de considerar a procura de uma solução de um problema como uma minimização global de alguma função de energia, PACO propõe a busca dessa solução como a co-evolução de um conjunto finito de agentes (representando cada um uma solução parcial), interagindo entre si e com o ambiente (os dados do problema). A solução global do problema é dada pela posição do conjunto de agentes. O comportamento de um agente é caracterizado por uma combinação definida a priori de um conjunto de interações elementares independentes. Cada tipo de interação está associada à capacidade de o agente perceber um tipo de agente ou de objeto no ambiente. O interesse do agente, em um determinado instante, de interagir com outro agente ou objeto (o que se traduz pelo controle explícito pelo agente de seus campos de percepção e de comunicação) ocasiona (se o campo de ação do agente permite) uma ação do agente. As interações são modeladas sob a forma de forças, obrigando os agentes a se deslocar no ambiente

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 que o influenciarão na execução de uma ação.
- *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 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.

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

- *Regulagem e Aquisição*: nesta etapa são determinados, dinamicamente, 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 interação e quais são eles).

- *Tratamento mediante o valor dos campos*: nesta etapa realiza-se o cálculo das interações com o ambiente e entre os agentes 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.

3.4.1 Exemplo: generalização cartográfica

Generalização é o processo de abstração usado quando a escala de um mapa é reduzida. Envolve a modificação dos dados de forma que possam ser representados num espaço menor, preservando da melhor forma possível as suas características geométricas e descritivas. Um mapa é uma representação de uma realidade geográfica. Quanto menor a escala, mais esquemática é a representação.

A maioria dos mapas em pequenas e médias escalas são obtidos por generalização de grandes escalas. O mapa base, por sua vez, é geralmente obtido a partir de aerofotogrametria e as escalas menores a partir de sucessivas generalizações.

A dificuldade maior na generalização é *escolher* como representar um número suficiente de objetos geográficos numa superfície reduzida, usando símbolos que preservem a identificabilidade do objeto. Simplesmente reduzir o tamanho de todos os objetos é inaceitável porque muitos objetos tornam-se indecifráveis. Assim, numerosas modificações nos dados são necessárias em termos de definição de objetos e de forma geométrica. O processo de generalização é usualmente realizado de forma manual e requer muito tempo dos especialistas.

Tanto a abordagem algoritmica quanto a de sistemas baseados em conhecimento não chegaram a resultados satisfatórios porque não levam em consideração o contexto.

O sistema SIGMA, descrito em [Alvares et al. 96], é um sistema de generalização cartográfica que utiliza o modelo PACO. Os dados de entrada do sistema são os objetos geográficos que compõem o mapa a ser generalizado. Cada objeto geográfico é representado por um ou mais pontos referenciados por suas coordenadas geográficas. Cada ponto representado corresponde a um agente no sistema SIGMA.

No modelo foi introduzida a noção de *grupo*. Um grupo é um conjunto de agentes com propriedades geográficas ou topológicas comuns:

- *grupo natural*: contém todos os agentes que estão localizados na mesma posição no ambiente (e, portanto, no mapa). São formados automaticamente pelo sistema e são mantidos durante todo o processo, obrigando os seus componentes a realizarem os mesmos deslocamentos.
- *grupo artificial*: são grupos de agentes com uma topologia comum (por exemplo, prédios alinhados). Estes grupos são formados pelo usuário através da interface gráfica e devem respeitar esta topologia mesmo sob o efeito de diferentes interações.

Tais grupos podem ser considerados como organizações locais em uma sociedade de agentes.

Os agentes usam um conjunto de campos para determinar suas interações e controlar seu comportamento. Estes campos são definidos para cada agente da seguinte forma:

- *Campo de Percepção CPi*: corresponde ao conjunto de agentes que o agente i é capaz de perceber no ambiente. Este conjunto contém todos os agentes (de qualquer tipo) que estão situados em um disco de raio d.
- *Campo de Classe CCi*: contém todos os agentes que pertencem à mesma classe geográfica.
- *Campo de Objeto COi*: este campo reagrupa todos os agentes que fazem parte do mesmo objeto geográfico que o agente i.
- *Campo de Vizinhança CVi*: contém todos os agentes que estão a uma distância próxima da resolução definida pelo usuário. É um círculo de raio a.
- *Campo de Grupo CGi*: contém os agentes que pertencem a um mesmo grupo (natural ou artificial) que o agente i.

As interações no sistema são baseadas num modelo de forças eletrostáticas de atração e repulsão, resultando no deslocamento dos agentes no sistema.

- *Força de repulsão entre agentes*: esta força age no campo de vizinhança CVi. Cada agente tentará empurrar todos os outros agentes para fora deste disco, com o objetivo de aumentar sua visibilidade e, portanto, o espaço que ocupa no ambiente. A magnitude desta força depende da massa dos agentes envolvidos, da distância entre eles, do tamanho de seus símbolos e da resolução do meio de saída.
- *Força de acompanhamento integral*: esta força tem por objetivo manter os grupos naturais e artificiais, ocasionando um deslocamento igual para todos os integrantes de um grupo. Este deslocamento é definido pelo agente de maior autoridade (massa) do grupo.
- *Força de acompanhamento proporcional*: para evitar que a deformação de um objeto seja demasiadamente local em função das forças de repulsão a que um agente pode estar submetido, uma fração desta força age também sobre os demais agentes do mesmo objeto geográfico, de forma inversamente proporcional à distância.
- *Força de retorno à posição original*: para evitar um deslocamento muito grande em relação à sua posição ideal (posição correspondente às coordenadas geográficas originais do agente), existe uma força de atração do ponto original, diretamente proporcional à distância.
- *Troca de Simbologia*: quando um agente detecta que um conflito não pode ser solucionado localmente (quando detecta uma oscilação em seus movimentos), ele solicita a realização de uma mudança no símbolo usado pela sua classe (trocando para um menor), para ocupar menos espaço no ambiente e desta forma solucionar o conflito.

Os primeiros testes do sistema foram realizados com uma massa de dados representando um vilarejo, com cerca de 300 objetos geográficos que deram origem a aproximadamente 1800 agentes. Os mapas reais terão entre um milhão e cinco milhões de agentes.

3.5 Comparação entre os modelos

A Figura 3.8, retirada de [Frozza 97], resume as principais características dos modelos considerados nas seções anteriores.

	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

Figura 3.8: Principais características dos modelos [Frozza 97]

3.6 Ambientes de desenvolvimento

Esta seção apresenta sucintamente dois ambientes de desenvolvimento de sistemas multiagentes reativos encontrados na literatura e com mais detalhes um aambiente desenvolvido com fins didáticos na Universidade Federal do Rio Grande do Sul.

3.6.1 O Sistema SIEME

O sistema SIEME - SImulateur Evènementiel Multi-Entités [Magnin 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 para simulação em computador.

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 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 definição:

- das entidades envolvidas
  - criação da classe
  - atributos internos e externos
  - definições dos atributos
  - os sensores e acionadores
- da simulação
  - a definição da classe
  - a inicialização
- das regras
  - a parte condição
  - a parte ação
  - as regras externas
  - as regras internas
- do início da simulação

Toda a especificação é realizada utilizando a linguagem Smalltalk, empregando classes pré-definidas.

3.6.2 O Sistema SWARM

O sistema SWARM é uma plataforma de software multiagente para a simulação de sistemas adaptivos [Minar et al. 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*.

3.6.3 O sistema SIMULA

Os sistemas SIEME e Swarm, descritos nas seções anteriores exigem o conhecimento e o uso de uma linguagem de programação tradicional, respectivamente Smalltalk e C para a sua utilização. O ambiente SIMULA [Frozza 97] desenvolvido para fins didáticos, teve desde o seu projeto o objetivo de ser de fácil utilização, o que não impede sua utilização na simulação de problemas reais.

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.

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 3.9 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.

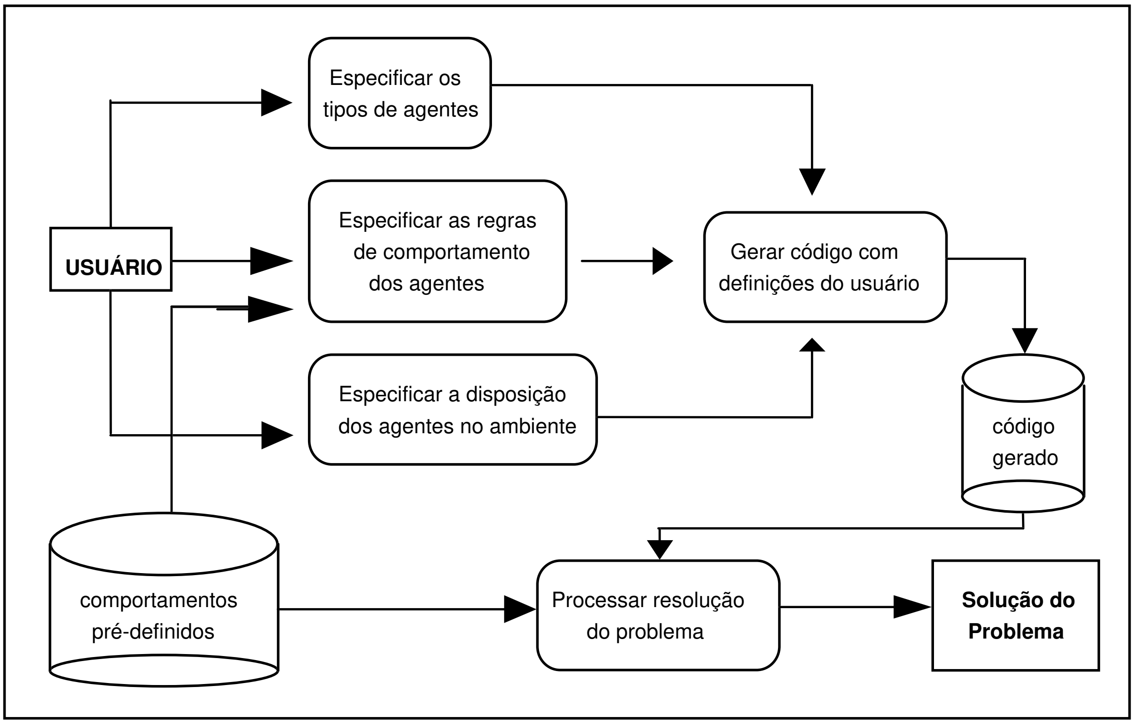


Figura 3.9 - Funcionalidade do Ambiente SIMULA [Frozza 97]

Como pode-se observar na figura, a definição de uma aplicação compõe-se de 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.

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 aplicação modelada.

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 no ambiente SIMULA e que são usados para montar as regras de comportamento dos agentes:

· *comportamento ativo*: ativa uma ação do agente em seu ambiente, causando uma mudança de estado desse agente;

· *comportamento passivo*: representa uma ação do agente, mas que não ocasiona, diretamente, uma mudança de posição do agente em seu ambiente;

· *comportamento de estado*: relacionado à percepção do agente e a seu estado atual, e que pode condicionar a execução de uma determinada ação pelo agente.

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. Os comportamentos equivalem à chamada de um procedimento que executará uma ação do agente.

A Figura 3.10 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 3.10 - Elementos de uma Regra de Comportamento

A seguir, são descritos os elementos que compõem as regras:

· *pré-condição* : são testes condicionais sobre o estado corrente do agente, que acionam ou não a execução da regra de comportamento associada ao agente;

· *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;

· *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;

· *pós-condição* : são os efeitos causados pela execução dos comportamentos. O usuário pode efetuar alterações ou atualizações de variáveis.

· *prioridade* : o usuário pode, através da prioridade, estabelecer a ordem de execução das regras, porque, a cada ciclo, o agente realiza apenas uma ação/comportamento.

A Figura 3.11 mostra a caixa de diálogo do ambiente para a definição das regras de comportamento.

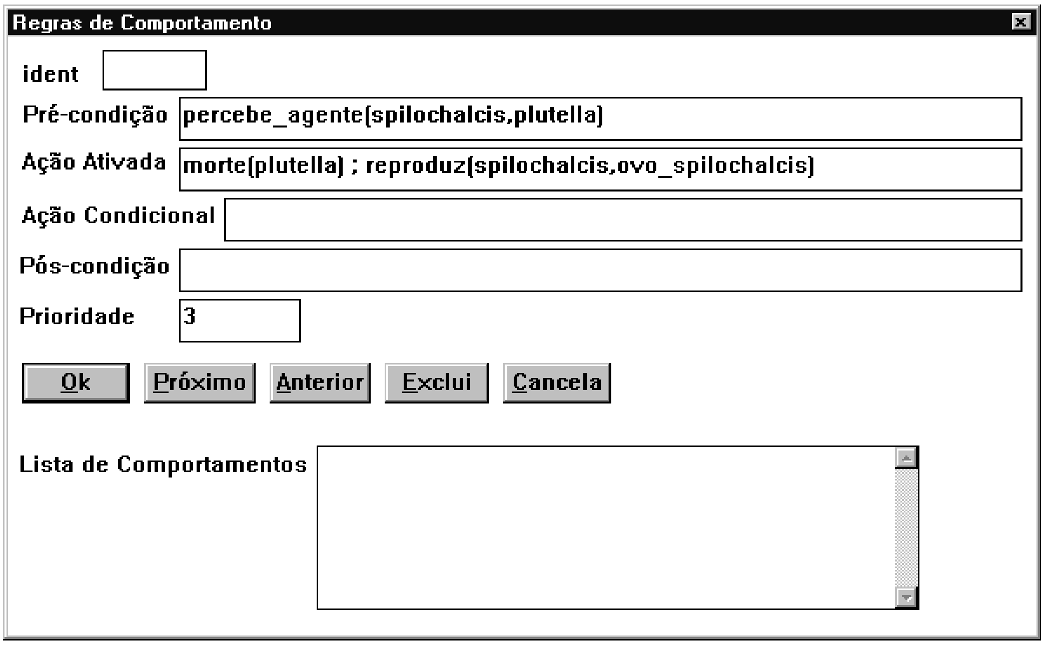


Figura 3.11 - Definição das Regras de Comportamento

4. Sistemas Multiagentes Cognitivos

Neste capítulo, serão apresentadas as principais propriedades dos chamados agentes cognitivos. Inicialmente, serão indicadas suas características básicas. A seguir, serão discutidos alguns aspectos relativos à sua arquitetura interna. Os principais tópicos referentes à organização e às interações entre tais agentes serão então apresentados. Finalmente, serão abordados alguns aspectos referentes a ambientes e linguagens de programação para o desenvolvimento de SMAs cognitivos.

4.1 Características básicas

Os SMAs cognitivos são baseados em modelos organizacionais humanos, como grupos, hierarquias e mercados. As principais características dos agentes cognitivos são as seguintes [Ferber e Gasser 91]:

- mantêm uma representação *explícita* de seu ambiente e dos outros agentes da sociedade;
- podem manter um *histórico* das interações e ações passadas, isto é, têm memória do passado;
- a comunicação entre os agentes é feita de modo *direto*, através do envio e recebimento de mensagens;
- seu mecanismo de controle é *deliberativo*, ou seja, tais agentes raciocinam e decidem sobre quais objetivos devem alcançar, que planos seguir e quais ações devem ser executadas num determinado momento;
- seu modelo de organização é baseado em modelos *sociológicos*, como as organizações humanas;
- uma sociedade contém tipicamente poucos agentes, na ordem de uma dezena.

Em [Demazeau 94], são apresentadas diversas classes de agentes cognitivos, de acordo com o aumento da complexidade de seu tratamento interno e de conceitos que se encontram explicitamente representados em seu interior. Tais classes, em ordem de complexidade decrescente e inclusiva, são as seguintes:

- agentes *organizados*: mantém perspectivas múltiplas sobre um determinado problema, obedecem a leis e regras sociais;
- agentes *negociantes*: contemplam um processo de resolução de conflitos por negociação;
- agentes *intencionais*: representam internamente noções como intenções, engajamentos, e planos parciais;
- agentes *cooperativos:* contém representações mútuas uns dos outros, existe um esquema de alocação de tarefas;
- módulos comunicantes: utilizam protocolos de comunicação;
- atores, processos: implementam primitivas de comunicação.

Cabe ressaltar que atualmente não se conhece nenhum sistema de agentes organizados que esteja efetivamente em modo operacional. Diversos modelos teóricos, entretanto, vêm sendo investigados com o intuito de se alcançar brevemente a consecução computacional de tais sistemas.

4.2 Arquiteturas de agentes

A abordagem clássica para construir agentes cognitivos pode ser considerada como um tipo especial de sistema baseado em conhecimento, utilizando a chamada linha simbólica de IA [Wooldridge e Jennings 95].Em [Sichman et al. 92], mostrou-se quais particularidades estes últimos deveriam contemplar de modo a consistir num agente. Basicamente, segundo a definição de agente apresentada em [Ferber e Gasser91], uma arquitetura de agentes deve contemplar ao menos módulos responsáveis pelos mecanismos de comunicação, percepção, raciocínio e decisão.

A título de exemplo, a Figura 4.1 abaixo mostra uma arquitetura de agente definida em [Sichman 95]. Tal arquitetura é baseada no modelo de agente ASIC [Boissier 93]. Este agente é composto por mecanismos de percepção e recepção (responsáveis pela entrada de dados), emissão e ação (responsáveis pela saída de dados), raciocínio e decisão (responsáveis por funções internas) e revisão (responsável pela revisão de crenças do agente). O agente raciocina sobre os outros agentes utilizando um mecanismo especial, denominado raciocínio social. Tal mecanismo utiliza as informações que o agente tem sobre os outros , armazenadas numa estrutura denominada descrição externa. Além dos mecanismos internos, o agente também possui diversos estados internos, como o estado de raciocínio (RS), de decisão (DS) e de engajamento (CS). Tais estados são modificados através da execução dos diversos mecanismos internos.

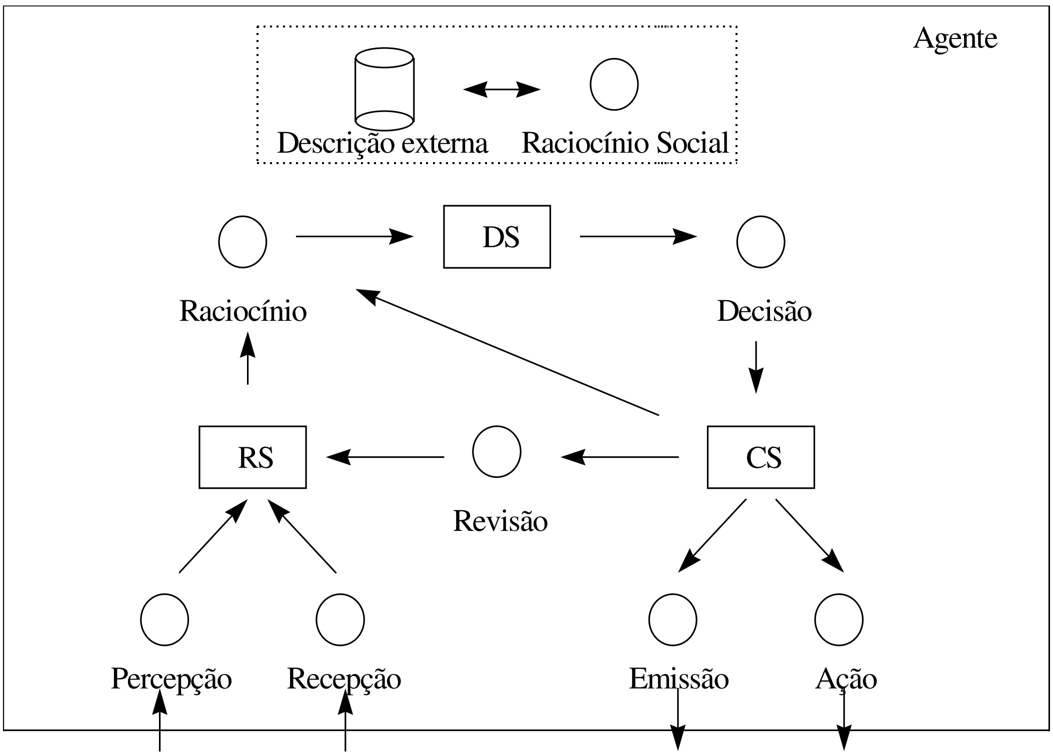


Figura 4.1 - Uma arquitetura de agente [Sichman 95]

4.3 Organizações de agentes

Do ponto de vista da psicologia social, as organizações de agentes devem ser analisadas através da ocorrência de interações sociais. A análise destas interações permite prever a formação dinâmica destas organizações. Por outro lado, caso existam restrições de comunicação entre os agentes (por exemplo fixadas por uma organização pré-estabelecida), estas podem limitar as interações sociais possíveis entre os diversos agentes do sistema.

Seguindo esta linha de raciocínio, duas classes de modelos para as interações sociais, e portanto para as organizações, são apresentadas em [Conte e Castelfranchi 92]:

- modelos *descendentes* ("top-down"): se considera, segundo uma abordagem RDP, que os agentes já têm a priori um problema a resolver. Neste caso, a cooperação é pré-estabelecida como uma hipótese de partida. As interações sociais são limitadas por uma organização pré-existente, que "guia" os agentes para atingir o objetivo para o qual o sistema foi concebido;
- modelos ascendentes ("bottom-up"): neste caso, os agentes não têm necessariamente um objetivo comum a atingir. As interações sociais, assim como as organizações, são estabelecidas dinamicamente como forma de atingirem seus próprios objetivos. Tais modelos são utilizados principalmente numa abordagem SMA.

  Os modelos ascendentes, por sua vez, podem ser decompostos em duas sub-classes, como mostrado em [Conte e Sichman 95]:

  - modelos baseados na *utilidade*: segundo tais modelos, dos quais a teoria dos jogos é um exemplo [Axelrod 84], o mundo social é considerado como um domínio de interferências entre agentes. Estes devem necessariamente coordenar suas ações para obter um comportamento global coerente. A existência de outros agentes *limita* a autonomia e poder de cada agente pertencente à sociedade;
  - modelos baseados na *complementaridade*: tais modelos levam em conta a possível existência de capacidades complementares entre os agentes, em relação às ações e recursos necessários para atingirem seus objetivos. Deste modo, a existência de outros agentes *aumenta* a autonomia e poder de cada agente, pois mesmo que cada um não possa atingir seus objetivos sozinho, ele pode alcançá-los pedindo auxílio aos outros.

4.3.1 Modelos descendentes

Nos modelos descendentes, a alocação de tarefas pode ser feita de modo estático (como na grande parte de sistemas distribuídos) ou de modo dinâmico, como no caso da rede contratual [Smith 80]. Os diversos papéis de resolução são, no entanto, definidos em tempo de concepção: a questão básica se resume portanto a atribuir os bons papéis aos bons agentes, que são considerados como benevolentes. Tais papéis de resolução são geralmente implícitos aos agentes e intimamente ligados à estrutura de implementação subjacente, como por exemplo nos modelos baseados em quadro negro [Engelmore e Morgan 88]

Como desvantagens de tal abordagem, podemos citar a ausência de uma perspectiva dinâmica, um certo subjetivismo social e uma ênfase na comunicação como motor das interações sociais [Conte e Castelfranchi 92]. Por outro lado, como mostrado em [Sichman 95], tais modelos mostram-se pouco adequados nos casos de sistemas Multiagentes abertos, onde não existe um controle global da entrada e saída de agentes da sociedade. Como vantagem deste modelo, obviamente, tem-se um ganho de performance, já que a fase de formação de organizações é pré-estabelecida no sistema.

Como exemplo de utilização deste modelo, podemos citar o projeto ESPRIT-II ARCHON [Jennings e Wittig 92], que reuniu vários institutos de pesquisa e indústrias européias. O objetivo deste projeto era o de "agentificar" sistemas industriais. A idéia básica era a de encapsular sistemas pré-existentes com uma camada responsável pela cooperação entre agentes. Tal camada era responsável, basicamente, pela definição dos papéis de resolução e pela comunicação entre agentes, servindo de interface para os módulos dependentes de domínio que seriam encapsulados.

4.3.2 Modelos baseados na utilidade

No caso dos modelos baseados na utilidade, toma-se como princípio da racionalidade o utilitarismo. Segundo tal paradigma, um agente é dito racional se ele sempre busca a maximização de sua utilidade esperada [Doyle 92]. Tal noção de racionalidade encontra-se presente na maior parte das teorias econômicas modernas. Como mostrado em [Sichman 95], os elementos de base do utilitarismo são a teoria da utilidade e a teoria da decisão.

A teoria da utilidade propõe um modelo para caracterizar as preferências de um agente entre diversos estados possíveis do mundo, refletindo seus desejos de alto nível. A teoria da decisão estende tal modelo, permitindo representar operações quantitativas sobre tais preferências (como por exemplo a soma) e também tornando possível o cálculo de tais preferências na presença de incertezas (utilidade esperada). A teoria dos jogos [Axelrod 84] é construída sobre estas duas teorias, se interessando sobre uma fonte particular de incertezas: aquela que leva em conta a ação de outros agentes, considerados igualmente como racionais [Gmytrasiewicz e Durfee 93].

Como inconvenientes de tal teoria, podemos citar o fato de se considerar os agentes como homogêneos e auto-suficientes, a incapacidade de explicar a ocorrência de cooperação (vide o chamado "dilema do prisioneiro"), a ausência da noção de objetivos e a visão da ação social como puramente estratégica [Conte e Sichman 95]. Por outro lado, a grande vantagem da utilização de tais modelos utilitários vem do fato de apresentarem um arcabouço matemático muito rigoroso e bem formalizado, o que facilita a obtenção de modelos e resultados teóricos um tanto quanto complexos.

Grande parte de trabalhos atualmente são fundados em modelos utilitários. Uma referência de base é sem dúvida [Rosenschein e Zlotkin 94].

4.3.3 Modelos baseados na complementaridade

Segundo tais modelos, as interações sociais são fundadas em relações estruturais objetivas (que existem na realidade) nas quais os agentes encontram-se mergulhados. Uma destas relações fundamentais é a relação de dependência social [Castelfranchi et al. 92]: agentes quase sempre necessitam uns dos outros para atingirem seus objetivos. Quando tais relações tornam-se subjetivas (pertencentes ao estado mental dos agentes), tais relações podem explicar porque agentes adotam os objetivos uns dos outros e porque algumas interações sociais como a cooperação surgem no seio de uma sociedade.

A idéia básica da teoria apresentada em [Castelfranchi et al. 92] e estendida em [Sichman 95] é a de que agentes não benevolentes adotam os objetivos uns dos outros de modo instrumental, como um meio para atingir seus próprios objetivos. Vamos supor que um agente *ag1* deseje alcançar um objetivo *g1* e para isto necessita realizar as ações *a1* e *a2*, mas tal agente não sabe executar a segunda ação. Suponhamos ainda que exista um outro agente *ag2*, complementar a *ag1*: tal agente sabe realizar *a2* e não *a1*, e também tem o objetivo *g1*. Quando ambos os agentes ficarem conscientes de sua dependência mútua, um deles irá propor ao outro a formação de uma coalisão (organização dinâmica) para atingir o objetivo. Nenhum deles pode atingir o objetivo sozinho, mas ambos o podem se cooperarem. Desta forma, pode-se explicar a formação dinâmica de organizações de agentes, baseado em suas complementaridades.

Em [Conte e Sichman 95] e [Sichman 95], são apresentados respectivamente dois sistemas, DEPNET e DEPINT, que ilustram a utilização de tais modelos em simulação social e formação de coalisões em sistemas Multiagentes abertos.

4.4 Interação entre agentes

Praticamente a totalidade dos trabalhos em interações entre agentes cognitivos se baseia na teoria de atos de fala ("speech act theory") [Searle 69]. Segundo tal teoria, comunicar é agir*:* trata-se de uma ação regular como qualquer outra, que deve ser gerada e processada pelos agentes.

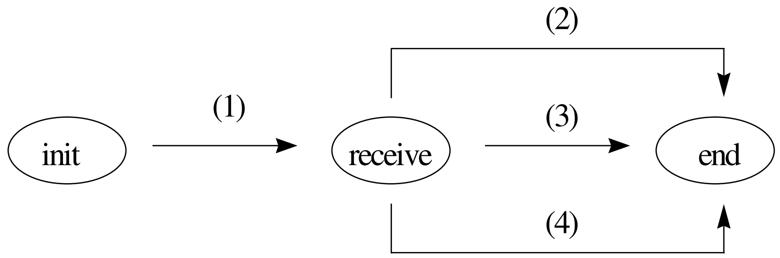
Tal teoria propõe uma categorização de primitivas de comunicação. Alguns exemplos são: *inform*, *ask-to-do*, *answer*, *promise*, *propose*, etc. Essas primitivas são associadas às suas conseqüências (ao menos as desejadas pelo agente que emitiu o ato de fala). Os atos de fala podem ser classificados segundo 3 aspectos:

- ato *locucionário*: emissão de palavras e sentenças com algum significado;
- ato *ilocucionário*: corresponde à *intenção* da emissão: request, order, apologize, assert;
- ato *perlocucionário*: corresponde ao resultado (desejado) da emissão: convince, insult, frighten.

Em [Searle 69], é proposta uma teoria para os atos ilocucionários. A maior parte dos SMAs que utilizam tal teoria representam portanto de modo explícito os conteúdos locucionários e ilocucionários nas mensagens enviadas. Como exemplo de algums atos ilocucionários, podem-se citar *action requesting*, *information seeking*, *information probing*, *information checking*, *instructing, informing*, *understanding event*, *advising*, *persuading*, etc.

Um outro ponto freqüentemente abordado na área é a estruturação de tais atos de fala em protocolos de interação. Tais protocolos servem para estruturar as trocas de mensagens entre agentes, e são concebidos com um caráter genérico. Deste modo, um determinado protocolo desenvolvido para uma aplicação pode ser utilizado em outras.aplicações.

Na Figura 4.2 a seguir, encontra-se representado um protocolo de formação de coalisões desenvolvido para o sistema DEPINT [Sichman 95]. Um agente envia uma proposição de coalisão para outro (1) que pode tomar 3 decisões distintas: avisar ao proponente que este tem uma crença incorreta sobre ele (2), recusar a oferta (3) ou aceitá-la (4).



1. (you) (request,bargaining,dec,depintproposition,receive)(proposition, data)

2. (if error) (you) (answer,warning,obs,depintproposition,end)(revision, data)

4. (if (!error & best_op))((you) (answer,informing,dec,depintproposition,end)(accept)

Figura 4.2 - Um protocolo de interação [Sichman 95]

4.5 Ambientes de desenvolvimento e linguagens

Basicamente, existem duas visões distintas a respeito do paradigma de agentes no que se refere ao nível de programação:

- visão de programação ponto pequeno ("programming-in-the-small"): a noção de agentes é suficientemente poderosa para gerar um novo paradigma de linguagens de programação, como os paradigmas procedural, funcional, lógico e de objetos. Deve-se portanto desenvolver linguagens de programação específicas, com os respectivos compiladores/interpretadores, que implementem diretamente os constructos computacionais adequados para o paradigma. Tais constructos consistiriam por exemplo na definição e tratamento de estados mentais como crenças, objetivos, planos, engajamentos, e na definição de primitivas de comunicação que se adequassem à teoria de atos de fala; [Searle 69]

- visão de programação ponto grande ("programming-in-the-large"): a maior contribuição da noção de agentes consiste numa prática de modelagem e integração de sistemas heterogêneos e abertos. Deste modo, não se deve impor uma linguagem de programação específica, e sim propiciar às diversas linguagens a possibilidade de implementar os constructos computacionais necessários ao paradigma através da construção de extensões (por exemplo, bibliotecas) que implementassem tais constructos.

Como exemplo da primeira visão, podem-se citar as linguagens AGENT0 [Shoham 91] e PLACA [Thomas 93]. O primeiro trabalho apresenta uma linguagem, implementada como uma extensão de LISP, onde cada agente tem 4 componentes:

- um conjunto de capacidades (coisas que o agente sabe fazer);
- um conjunto inicial de crenças
- um conjunto inicial de engajamentos (coisas que o agente deve fazer)
- um conjunto de regras de engajamento

O componente básico, que determina como o agente irá agir, é o conjunto de regras de engajamento. Cada uma destas regras contém uma condição de mensagem, uma condição mental e uma ação. Em cada ciclo de execução, as condições de mensagem são comparadas com as mensagens recebidas pelo agente, e a condição mental confrontada com as suas crenças correntes. Caso a regra seja disparada, o agente torna-se engajado com a ação. Ações podem ser internas ou externas, consistindo num envio de mensagem. As mensagens podem ser de três tipos distintos: "requests" para engajamento em ações, "unrequests" para desengajamento de ações e "informs" para comunicação de dados.

A linguagem PLACA [Thomas 93] é uma extensão de AGENT0, incorporando a noção de planos. Deste modo, no ciclo de um agente, além das atividades citadas acima, um agente pode refinar seus planos para atingir determinados objetivos.

Como exemplo da segunda abordagem, podemos citar o ambiente MASENV [Demazeau 95]. Tal ambiente utiliza como sistema distribuído subjacente o sistema DPSK+P, um núcleo para resolução distribuída de problemas orientado a objetos [Cardozo et al. 93]. O objetivo de tal ambiente é o de propiciar, através de bibliotecas escritas na linguagem C++, o desenvolvimento de protótipos de agentes, organizações e protocolos de interação. Um outro exemplo desta abordagem é a linguagem DAISY [Poggi 95].

Finalmente, cabe ressaltar que existem ainda linguagens destinadas exclusivamente à comunicação entre agentes. Tais linguagens definem standards para a troca de mensagens. O exemplo mais conhecido é resultado de um projeto do ARPA americano, num projeto denominado "knowledge sharing initiative" [Finin et al. 94]. Tal iniciativa compõe-se de duas partes:

- uma linguagem para manipulação e consulta de conhecimento (KQML);
- um formato para intercâmbio de conhecimento (KIF)

KQML é uma linguagem externa, que basicamente define performativas baseadas em atos de fala, como ask-if, tell perform, reply, etc. KIF é uma linguagem interna, que serve para expressar o conteúdo da mensagem. Trata-se de uma representação em ASCII de uma versão estendida de lógica de primeira ordem. Para poderem se comunicar, agentes devem ter estabelecido uma ontologia de termos comuns. Grande parte do esforço deste projeto consiste na definição de ontologias universais.

5. Conclusões e Perspectivas

Através deste texto procurou-se dar uma visão geral inicial sobre o estudo dos sistemas multiagentes, suprindo uma lacuna existente, principalmente em língua portuguesa. Como este é um tema relativamente novo, há muita dificuldade em se encontrar material didático, que possa ser empregado como livro texto em cursos introdutórios sobre o assunto.

O objetivo foi o de percorrer os diversos aspectos do estudo de sistemas multiagentes, dando um panorama dos problemas, pistas de soluções e potencialidades da área. O capítulo 1 apresentou os conceitos básicos, um breve histórico, os principais problemas e a diferenciação entre Resolução Distribuída de Problemas e Sistemas Multiagentes. O capítulo 2 abordou os sistemas multiagentes reativos, apresentando suas principais características, alguns modelos existentes e um ambiente de desenvolvimento. O capítulo 3 considerou os sistemas cognitivos, apresentando arquiteturas de agentes, modelos de organização, formas de interação, ambientes de desenvolvimento e linguagens.

A estruturação do texto separando os agentes cognitivos dos reativos reflete duas abordagens do estudo da inteligência: a Inteligência Artificial baseada em Conhecimento e a nova abordagem, a Inteligência Artificial baseada no Comportamento. A IA baseada em Conhecimento tradicionalmente privilegiou a modelagem e a construção de sistemas que "conhecem" algum domínio de aplicação. Estes sistemas modelam o domínio e podem responder a perguntas sobre este domínio, freqüentemente envolvendo extensamente resolução de problemas e raciocínio. A IA baseada no Comportamento, por outro lado, tem enfatizado a modelagem e a construção de sistemas que "se comportam" em algum domínio [Maes 93].

Independente da abordagem, os sistemas multiagentes estão iniciando uma verdadeira revolução na forma de resolver problemas, na forma de pensar os problemas, passando de uma visão global, de atitudes baseadas em uma abordagem centralizada, a uma visão local, a nível de agente (componente), onde o comportamento coletivo, a funcionalidade global, vai emergir das interações entre os componentes. Esta será uma revolução tranqüila, talvez mais uma evolução da forma de ver e pensar o mundo do que uma revolução, mas suas conseqüências serão enormes. Esta nova maneira de pensar os problemas permitirá que sistemas ainda mais complexos sejam projetados e desenvolvidos.

Esta mudança de visão começou com o advento da orientação a objetos, mas vai poder realmente se concretizar com os sistemas multiagentes pois permitirá a autonomia comportamental de seus componentes, possibilitando a realização de sistemas mais abertos, flexíveis, interativos e confiáveis.

Como reflexo desta nova visão que está se formando, muitas comunidades de pesquisa estão reinventando o conceito de agente: vida artificial, telecomunicações, recuperação de informações, linguagens de programação. Mesmo que em muitas delas o conceito de agente seja mais fraco que o empregado na comunidade de sistemas multiagentes, elas farão o conceito evoluir, de forma independente, podendo chegar a resultados não imaginados.

Um verdadeiro "caldo de cultura" para o desenvolvimento dos sistemas multiagentes é a Internet. Até agora ela tem sido usada principalmente para os chamados agentes de software, desenvolvidos essencialmente para a busca e filtragem de informações, mas é um campo excepcional para as pesquisas de sistemas multiagentes, especificamente para sistemas abertos. Além disso, devido à penetração que a Internet já tem, será uma grande vitrine para a divulgação e mesmo para o uso prático dos resultados das pesquisas em sistemas multiagentes. Dentro de poucos anos o comércio e a prestação de serviços através da Internet serão corriqueiros. Quando fizermos uma pesquisa de preço e condições de compra de algum produto ou marcarmos uma reunião de trabalho ou mesmo escolhermos um filme para assistir na televisão, estaremos fazendo uso da tecnologia dos sistemas multiagentes.

6. Referências Bibliográficas

[Agha e Hewitt 85] Agha, G. e Hewwit, C. Concurrent programming using ACTORs: exploiting large-scale parallelism. In: Bond, A.e Gasser, L. eds. *Readings in Distributed Artificial Intelligence*. San Mateo, Morgan Kaufmann, 1988.

[Alvares et al. 96] Alvares, L. O., Baeijs, C. e Demazeau, Y. *SIGMA: um Sistema Holístico de Generalização Cartográfica*. Anais GIS BRASIL 96, Curitiba, maio 1996.

[Axelrod 84] Axelrod, R. *The evolution of cooperation*. New York, Basic Books, 1984.

[Boissier 93] Boissier, O. *Problème du Contrôle dans un Système Integré de Vision. Utilisation d'un Système Multi-Agent.* Thèse de Doctorat de l'INPG, Grenoble, France, 1993.

[Bond e Gasser 88] Bond, A. e Gasser, L. eds. *Readings in Distributed Artificial Intelligence*. San Mateo, Morgan Kaufmann, 1988.

[Brooks 86] Brooks, R. A robust layered control system for a mobile robot. In*: IEEE Journal of Robotics and Automation*, v. 2, n. 1, p. 14-23, march 1986.

[Cardozo 87] Cardozo, E. *DPSK: a kernel for distributed problem solving*. Phd thesis, Carnegie Mellon University, 1987.

[Cardozo et al. 93] Cardozo, E., Sichman, J. S.e Demazeau, Y. Using the active object model to implement multi-agent systems. In: *Proc. 5th IEEE International Conference on Tools with Artificial Intelligence*, Boston, 1993.

[Castelfranchi et al. 92] Castelfranchi, C,. Micelli, M. e Cesta, A. Dependence relations among autonomous agents. In: Werner, E. e Demazeau, Y. eds. *Decentralized AI 3*, p. 215-227, Amsterdam, Elsevier, 1992.

[Chandrasekaran 81] Chandrasekaran, B. Natural and social system metaphors for distributed problem solving: introduction to the issue. In: *IEEE Transactions on Systems, Man and Cybernetics*, v. 11, n. 1, p. 1-5, Jan. 1981.

[Conte e Castelfranchi 92] Conte, R. e Castelfranchi, C. Mind is not enough: precognitive bases of social interactions. In*: Proc. of the 1st. Symposium on Simulating Societies*, Guildford, UK, 1992.

[Conte e Sichman 95] Conte, R. e Sichman, J.S. DEPNET: How to benefit from social dependence. In: Journal of Mathematical Sociology, v. 19, n. 2, 1995.

[Corkill e Lesser 93] Corkill, D. D. e Lesser, V. R. The distributed vehicle monitoring testbed: a tool for investigating distributed problem solving networks. In: *AI Magazine*, v. 4, n. 3, p. 15-33, Fall 1983.

[Demazeau e Müller 90] Demazeau, Y. e Müller, J. P. Decentralized artificial intelligence. In: Demazeau, Y. e Müller, J.P. eds. *Decentralized A I.*, Amsterdam, Elsevier, 1990.

[Demazeau 91] Demazeau, Yves. *Coordination Patterns in Multi-Agent World: Applications to Computer Vision and Robotics*. IEE Colloquium on Intelligent Agents, IEE, London, February, 1991.

[Demazeau 93] Demazeau, Yves. La Plate-forme PACO et ses applications. Anais Journée Nationale du PRC-IA sur les Systèmes Multi-Agents, 2., Montpellier, 1993.

[Demazeau 94] Demazeau, Y. *Distributed AI and Multi-Agent Systems*. Lecture notes, Lindo Center for Applied Mathematics, Odense, Denmark, 1994.

[Demazeau 95] Demazeau, Y. From interactions to collective behaviour agent-based systems. In: *Proc. of the 1st. European Conference on Cognitive Science*, St. Malo, France, 1995.

[Drogoul e Dubreuil 92] Drogoul, A e Dubreuil, C. Eco-Problem-Solving Model: Results of the N-Puzzle. In: WERNER, Eric; DEMAZEAU, Yves (Eds.). *Decentralized A.I. 3*. Amsterdam: North-Holland, 1992.

[Drogoul 93] Drogoul, A. *De La Simulation Multi-Agents à La Résolution Collective de Problèmes*. Thèse de L'Université Paris VI. Institut Blaise Pascal, LAFORIA TH93/05, França, 23 de novembre de 1993.

[Doyle 92] Doyle, J. Rationality and its role in reasoning. In: *Computational Intelligence*, v. 8, p.376-409, 1992.

[Durfee e Rosenchein 94] Durfee, E.H. e Rosenschein, J. S. Distributed problem solving and multi-agent systems: comparisons and exemples. In: *Proc. 13th. International Workshop on DAÍ*, Seattle, 1994.

[Engelmore e Morgan 88] Engelmore, R. e Morgan, T. eds. *Blackboard systems*. Reading, Addison-Wesley, 1988.

[Erman et al. 80] Erman, L.D. et al. The HEARSAY-II speech-understanding system: integrating knowledge to resolve uncertainty. In: *Computing Surveys*, v.12, n. 2, p. 213-253, June 1980.

[Ferber 91] Ferber, Jacques e Jacopin, Eric. The Framework of Eco-Problem Solving. In: Demazeau, Yves; Muller, Jean-Pierre (Eds.). Decentralized A.I..2, Amsterdam: North-Holland, 1991.

[Ferber e Gasser 91] Ferber, J. e Gasser, L. *Intelligence artificielle distribuée*. In: XI International Workshop on Expert Systems & their Applications, Avignon, France, 1991, Cours n. 9.

[Fini et al. 94] Finin, T., Fritzon, R., McKay, D. e McEntire, R. KQML as an agent communication language. In: *Proc. 3rd. international Conference on Information and Knowledge Management*, 1994.

[Fox 81] Fox, M.S. An organizational view of distributed systems. In: *IEEE Transactions on Systems, Man and Cybernetics*, v. 11, n. 1, p. 70-80, Jan. 1981.

[Frozza 97] Frozza, R. *SIMULA - um Ambiente para o Desenvolvimento de Sistemas Multiagentes Reativos*. Dissertação de Mestrado, CPGCC/UFRGS, Porto Alegre, 1997.

[Gasser et al. 87] Gasser, L. et al. MACE: a flexible testbed for distributed AI research. In: Hunhs, M. N. ed. *Distributed Artificial Intelligence*. Los Altos, Morgan Kaufmann, 1987.

[Gasser 92] Gasser, L. Boundaries, identity and aggregation: plurality issues in multi-agent systems. In: Werner, E. e Demazeau, Y. eds. *Decentralized AI 3*, Amsterdam, Elsevier, 1992.

[Gmytrasiewicz e Durfee 93] Gmytrasiewicz, P. J. e Durfee, E. H. Reasoning about other agents: philosophy, theory and implementation. In: Proc. 5th. European Workshop on Modelling Agents in a Multi-Agent World, Neuchâtel, Switzerland, 1993.

[Hewitt 86] Hewitt, C. Offices are open systems. In: Bond, A.e Gasser, L. eds. *Readings in Distributed Artificial Intelligence*. San Mateo, Morgan Kaufmann, 1988. P.321-329.

[Jennings e Wittig 92] Jennings, N. R. e Wittig, T. ARCHON: Theory e praxis. In: Avouris, N. M. e Gasser, L. eds. *DAÍ: Theory and Praxis*, Amsterdam, Kluwer, 1992.

[Konolige e Nilsson 80] Konolige, K. e Nilsson, N. Multi-agent planning systems. In: *Proc.1st. AAAI Conference*, Stanford, 1980.

[Lenat 75] Lenat, D.B. Beings: knowledge as interacting experts. In: Bond, A.e Gasser, L. eds. *Readings in Distributed Artificial Intelligence*. San Mateo, Morgan Kaufmann, 1988. P.161-168.

[Maes 93] Maes, P. Behavior-Based Artificial Intelligence. In: *From Animals to Animats 2*. Meyer, J-A. e Wilson S. (Eds). MIT Press, Cambridge, 1993.

[Magnin 96] Magnin, Laurent. *SIEME - Simulateur d'environnement pour systèmes multi-agents*. 1996. [http://www-laforia.ibp.fr/~magnin/these/sieme]

[Minar et al. 96] Minar, N.; Burkhart, R.; Langton, C.; Askenazi, M. *The SWARM Simulation System: A Toolkit for Building Multi-Agent Simulations*. Junnho 1996. [http://www.santafe.edu/projects/swarm/overview/overview]

[Newell 82] Newell, A.The knowledge level. In: *Artificial Intelligence*, v. 18, p. 87-127, 1982.

[Pleiad 92] Vers une taxinomie du vocabulaire pour les systèmes multi-agents. In: *Actes de la 1ère. Journée Systèmes Multi-Agents du PRC-GDR Intelligence Artificielle*, Nancy, France, 1992.

[Poggi 95] Poggi, A. DAISY: an object-oriented system for distributed artificial intelligence. In: Wooldridge, M. e Jennings, N. eds. *Intelligent Agents*, Springer-Verlag, LNAI 890, 1995.

[Rosenschein e Zlotkin 94] Rosenschein, J. e Zlotkin, G. *Rules of encounter: designing conventions for automated negotiation among computers*. Cambridge, MIT Press, 1994.

[Russel e Norvig 95] Russel, S. e Norvig, P. *Artificial Intelligence - a modern approach*. Prentice -Hall Inc, Upper Saddle River, 1995.

[Searle 69] Searle, J. *Speech acts*. Cambridge University Press, 1969.

[Sichman et al. 92] Sichman, J.S.; Demazeau, Y. e Boissier, O. When can knowledge-based systems be called agents? In: *Anais do IX SBIA*, p. 172-185, Rio de Janeiro, 1992.

[Sichman 95] *Du raisonnement social chez les agents: une approche fondée sur la théorie de la dépendance*. Thèse de Doctorat de l'INPG, Grenoble, France, 1995.

[Shoham 91] Shoham, Y. Agent oriented programming. In: *Artificial Intelligence*, v. 60, n. 1, p. 51-92, 1991.

[Smith 80] Smith, R. G. The contract net protocol: high-level communication and control in a distributed problem solver. In: *IEEE transactions on Computers*, v. 29, n. 12, p. 1104-1113, Dec. 1980.

[Steels 90] Steels, L. Cooperation between Distributed Agents through Self-Organization. In: *Decentralized A.I.* Demazeau, Y. e Muller, J-P. (Eds), North-Holland, Amsterdam, 1990.

[Steels 91] Steels, L. Towards a Theory of Emergent Functionality. In: *From Animals to Animats*. Meyer, J-A. e Wilson S. (Eds). MIT Press, Cambridge, 1991.

[Stefik e Bobrow 86] Stefik, M. e Bobrow, D. G. Object-oriented programming: themes and variations. In: *AI Magazine*, v. 6, n. 4, p. 40-62, 1986.

[Thomas 93] Thomas, R. *PLACA: an agent oriented programming language*. Phd Thesis, Stanford University, 1993.

[Wooldridge e Jennings 95] Wooldridge, M. e Jennings, N. *Agents theories, architectures and languages*. Tutorial notes, 1st International Conference on Multi-Agent Systems, San Francisco, 1995