Modelo Formal de Sistemas Multiagente com Mobilidade

Projeto parcialmente financiado por PIBIC/CNPq

5 de dezembro de 2005

Aluno: Alvaro Heiji Miyazawa
Orientadora: Prof. Dra. Ana Cristina Vieira de Melo

Resumo

Monografia apresentada em cumprimento às condições para a aprovação na disciplina *Trabalho de Formatura Supervisionado*, visando à conclusão do curso de Bacharelado em Ciência da Computação no Instituto de Matemática e Estatística da Universidade de São Paulo.

Sumário

Parte I
Parte técnica

1 Introdução

O objetivo principal deste trabalho é fornecer uma definição de mobilidade no contexto de Sistemas Multiagente. Para tanto foi utilizado um arcabouço conceitual chamado SMART[1], que forneceu definições formais de conceitos relacionados a sistemas multiagente. Este trabalho é apresentado como uma extensão desse arcabouço.

O arcabouço SMART foi especificado em Z, um notação formal baseada em lógica de primeira ordem e teoria dos conjuntos, e as definições foram construidas de forma que a aplicabilidade do arcabouço não ficasse restrita a sistemas computacionais. Por exemplo, pode-se descrever a organização de uma firma, a hierarquia, os relacionamentos entre os funcionário, etc. Para mais informações sobre aplicações não computacionais do arcabouço SMART veja [Sil04].

Note que a mobilidade de um agente poderia ser abordada como uma capacidade implícita do agente, porém o objetivo deste trabalho é tratá-la explicitamente, devido à sua importância em sistemas multiagente modernos.

Para não reduzir o escope do arcabouço, a definição de mobilidade deveria ser o mais abstrata possível. Por essa razão, foi utilizada a noção de mobilidade apresentada por R. Milner em [Mil99].

2 Conceitos e Tecnologias Estudadas

2.1 SMART

A especificação SMART consiste de quatro elementos principais:

- Entidade;
- Objeto;
- Agente;
- Agente Autônomo.

Um elemento do arcabouço é definido por quatro conjuntos:

[1]Structured and Modular Agent and Relationship Types

- atributos;
- capacidades;
- objetivos;
- motivações.

De acordo com [dL01a] um atributo é uma característica perceptível, por exemplo, cor, textura, sabor etc. Uma ação é um evento discreto que pode mudar o estado do ambiente, por exemplo, levanta, troca roda, anda, guarda material etc. Um objetivo é um estado particular a ser alcançado no ambiente, por exemplo, trocar as quatro rodas de um carro. E uma motivação é qualquer desejo ou preferência que possa causar a geração ou adoção de objetivos e que afete o resultado das tarefas planejadas para satisfazer esses objetivos, por exemplo, manter o carro sempre funcionando bem

Cada um dos quatro elementos do arcabouço é definido através de restrições sobre esses conjuntos, de forma que, forma-se uma hierarquia entre os elementos.

Uma entidade é definida pela restrição de que o conjunto de atributos seja não vazio, isto é, uma entidade precisa necessariamente ser caracterizável, apresentar atributos que a definam. Por exemplo, uma xicara azul de porcelana deve ter atributos como cor azul, material porcelana, formato côncavo etc.

Um objeto é uma entidade com a restrição adicional de que o conjunto de capacidades seja não vazio, isto é, um objeto deve ser capaz de fazer algo. Por exemplo, uma xícara pode armazenar café, por isso ela tem como capacidades (ações) "armazenar café".

Um agente é um objeto com a restrição adicional de que o conjunto de objetivos seja não vazio, isto é, um agente deve ter objetivos a partir dos quais direcionará suas ações. Por exemplo, um robô que troca pneus numa oficina tem o objetivo de trocar os pneus de algum automóvel.

Um agente autônomo é um agente com a restrição adicional de que o conjunto de motivações seja não vazio, isto é, uma agente autônomo deve ter motivações a partir das quais gera objetivos a serem atingidos. Por exemplo, o presidente de uma empresa visa ao sucesso da empresa.

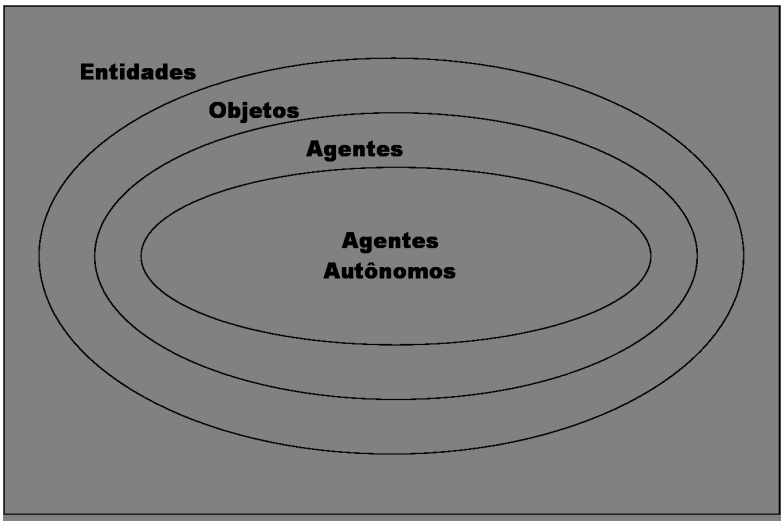


Fig. 1. Hierarquia de entidades do modelo SMART.

O ambiente consiste de um conjunto de entidades (note que o conjunto de entidades inclue objetos, agentes e agentes autônomos) e um conjunto não vazio de atributos, que inclue todos is atributos das entidades nesse ambiente.

Cada um dos elementos citados acima (entidade, objeto, agente e agente autônomo) possui um estado associado[2] que contém informações que não são especificadas pelas características inerentes a cada elemento (atributos, capacidades, objetivos e motivações). Por exemplo, uma entidade tem um estado associado que contém um conjunto de atributos chamado situação (esse conjunto define o estado da entidade). Um estado associado é a manifestação do elemento correspondente quando inserido em um ambiente.

Cada um dos elementos que possuem capacidades (objeto, agente e agente autônomo) possui um esquema[3] de ação associado, esse esquema associado provê meios de seleção de ações. Aqueles elementos também possuem uma descrição do que ocorre quando há uma interação, isto é, uma mudanção no ambiente por meio de um conjunto de ações. Note que as ações de um objeto podem potencialmente afetar seu estado.

[2]No arcabouço os quatro elementos são denominados entity, object, agent e autonomous agente e seus estados associados são respectivamente entity state, object state, agent state e autonomous agente state

[3]um esquema é uma estrutura da linguagem de especificação Z que provê mecanismos de definição, para mais informações sobre esquemas veja a subseção seguinte e as referências [Jac96], [Spi89] e [WD96]

Cada um dos elementos que possue objetivos (agente e agente autônomo) possui um esquema de percepção associado que descreve como as características do elemento correspondente afetam suas percepções. Note que a percepção de um agente pode afetar sua linha de ação.

A diferença entre um agente e um agente autônomo é que um agente é orientado por objetivos, enquanto um agente autônomo é motivado. Isso significa que um agente tem uma agenda fixa particular, já um agente autônomo pode gerar seus objetivos dinamicamente.

Para mais informações sobre o arcabouço SMART veja [dL01a]. Para a especificação do SMART veja o apêndice A.

2.2 Z

Z é uma notação baseada em lógica de primeira ordem e teoria dos conjuntos usada para descrever sistemas computacionais. Um sistema é modelado em Z, gerlamente, representando seu estado e algumas operações que podem mudar esse estado.

Alguns conceitos são importantes para a compreensão da especificação e serão apresentados aqui, um glossário da notação será apresentado no apêndice B.

2.2.1 Conjuntos dados (Givend Sets)

São tipos de dados primitivos sobre os quais nada é especificado. Por exemplo:

$$[CHAR]$$

2.2.2 Descrições Axiomáticas

Usamos descrições axiomáticas para definir constantes, funções, [4] Por exemplo:

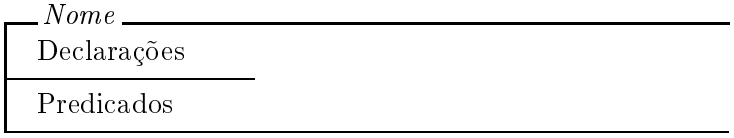
$$fat : \mathbb{N} \rightarrow \mathbb{N}$$

$$fat(0) = 1$$
$$fat(n) = n * fat(n-1)$$

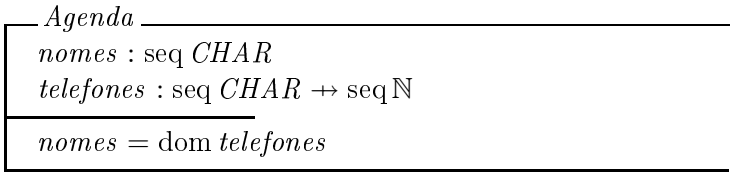
[4] em Z fun ções como fatorial são um tipo de contante [?].

2.2.3 Esquemas

Os esquemas são usados para estruturar e compor especificações, promovendo encapsulamente, reúso e outras características interessantes. Um esquema tem a seguinte estrutura:

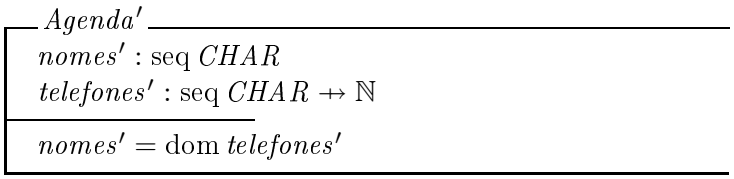


Por exemplo:

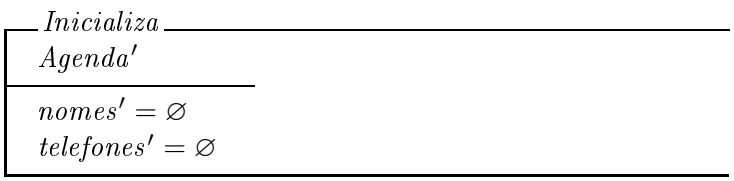


Note que um esquema pode representar o estado de um sistema, por exemplo, uma agenda de telefones, como acima. Para descrever uma operação sobre um esquema, usam-se duas cópias desse esquema, uma representando o estado antes da opração e outra representando o estado depois da operação, esta última será marcada com um apóstrofe no final do nome do esquema.

Por exemplo, o esquema Agenda após uma operação.



Alguns exemplos de operações:



Adiciona

$\Delta Agenda$
$nome? : \text{seq } CHAR$
$numero? : \text{seq } CHAR$

$nome? \notin nomes$
$nomes' = nomes \cup \{nome?\}$
$telefones' = telefones \cup \{nome? \mapsto numero?\}$

O $\Delta Agenda$[5] significa que estamos incluindo os esquemas Agenda e Agenda'. As variáveis seguidas de ? são variáveis de entrada e as variáveis de saída são indicadas por !.

2.3 Mobilidade

Os termos mobilidade e agente são usados em domínios diferentes com significados diferentes. Citando R. Milner em [Mil99], "... a full analysis of such possibilities ... would have been vague and inconclusive[6]". A noção de mobilidade escolhida como base para este trabalho foi aquela do π-calculus: mobilidade como mudança de conectividade. Essa idéia foi escolhida por duas razões principais, porque ela é bastante conveniente para o propósito de representar mobilidade de software e porque ela é uma possível ligação entre este arcabouço baseado em estados e o π-calculus[7].

[5]Existe outra convenção quanto à inclusão de esquamas, quando incluimos $\Delta Agenda$ e queremos garantir que as variáveis não são alteradas usamos o símbolo $\Xi Agenda$, desta forma todas as igualdades entre variáves antes e depois ficam implícitas.

[6]... uma análise completa de tais possibilidades ... teria sido vaga e inconclusiva.

[7]π-calculus é baseado em comunicação.

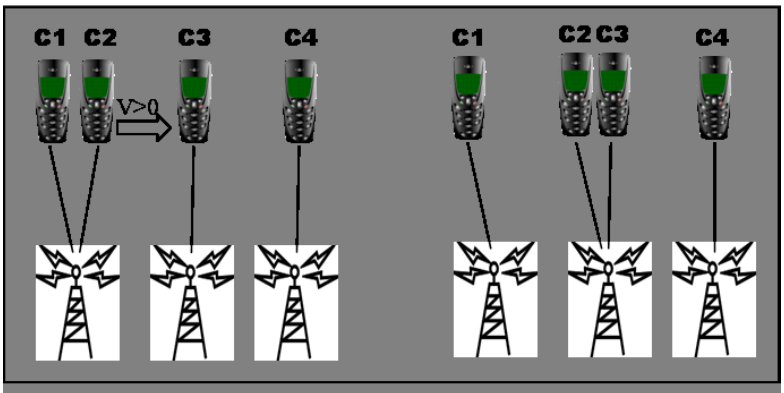


Fig. 2. Exemplo de mobilidade, o celular 2 se movimenta e muda de antena.

3 Resultados e Produtos Obtidos

3.1 Especificação

Um agente móvel é um agente com capacidade explícita de comunicação e a habilidade de mudar sua conectividade[8].

Essa capacidade de comunicação é especificada como um conjunto de canais através dos quais o agente pode se comunicar. Um canal é uma interface de comunicação não especificada[9]

$$[CHANNEL]$$

MobileAgent

$$Agente$$
$$channels : \mathbb{P}\, CHANNEL$$

$$channels \neq \varnothing$$

[8]Esse trabalho não especifica o tipo agente móvel autônomo, porém ele pode ser facilmente derivado da especificação de agente móvel.

[9]Esse tipo é especificado como um conjunto dado (given set), veja a subseção anterior.

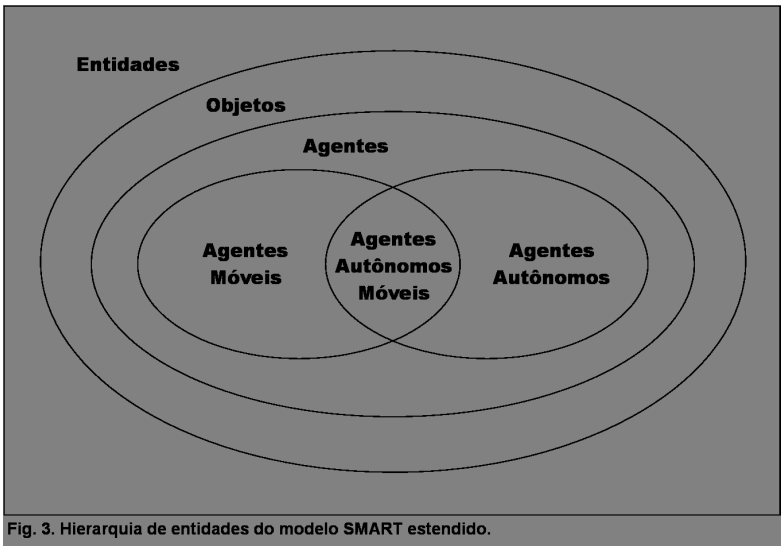


Fig. 3. Hierarquia de entidades do modelo SMART estendido.

A capacidade de comunicação de um agente por afetar suas percepções, pois ao mudar sua capacidade de comunicação um agente passa a ter contato com outros agentes que ele possivelmente não conhecia, portanto novas informações estão disponíveis. A função mobilewillperceive é uma função de seleção de percepções, através da qual o agente decide qual partição de todas as percepções levar em conta.

MobileAgentPerception

$$
\begin{array}{l}
MobileAgent \\
AgentPerception \\
mobilewillperceive : \mathbb{P}\, CHANNEL \rightarrow \mathbb{P}\, Motivation \\
\qquad \rightarrow \mathbb{P}\, Goal \rightarrow Environment \rightarrow View \\
\hline
\mathrm{dom}\, mobilewillperceive = channels
\end{array}
$$

A capacidade de comunicação de um agente pode também afetar sua conduta, o agente pode levar em consideração os agentes com quem pode se comunicar para decidir que ações executar. A função mobileactions é uma função de seleção de ações, através da qual o agente decide o que fazer dadas as informações que tem.

$$
\begin{array}{|l}
\hline
\textit{MobileAgentAction} \\
\textit{MobileAgent} \\
\textit{AgentAction} \\
\textit{mobileactions} : \mathbb{P}\, \textit{CHANNEL} \rightarrow \mathbb{P}\, \textit{Motivation} \\
\qquad \rightarrow \mathbb{P}\, \textit{Goal} \rightarrow \textit{View} \rightarrow \textit{Environment} \rightarrow \mathbb{P}\, \textit{Action} \\
\hline
\mathrm{dom}\, \textit{mobileactions} = \textit{channels} \\
\hline
\end{array}
$$

O conjunto de ações a serem executadas por um agente móvel é descrito como a aplicação da função de seleção de ações aos conjuntos de canais, motivações e objetivos, às percepções reais do ambiente e ao próprio ambiente.

$$
\begin{array}{|l}
\hline
\textit{MobileAgentState} \\
\textit{MobileAgent} \\
\textit{MobileAgentPerception} \\
\textit{MobileAgentAction} \\
\hline
\textit{willdo} = \textit{mobileactions}\ \textit{channels}\ \textit{motivations}\ \textit{goals} \\
\qquad \textit{actualpercepts}\ \textit{environment} \\
\hline
\end{array}
$$

Este esquema descreve o que permanece estático quando atuamos sobre o estado do agente, isto é, quando ocorre uma interação. De acordo com o esquema abaixo nem o esquema de percepções nem o esquema de ações mudam, note que isso não significa que uma mudança de estado não afeta as percepções e ações dos agentes, significa que as funções de seleção de percepções e de seleção de ações não é afetada, mas como mudamos os conjuntos sobre os quais aplicamos essa funções para obter as percepções e ações, essas são realmente afetadas pela mudança de estado.

$$
\begin{array}{|l}
\hline
\Delta \textit{MobileAgentState} \\
\textit{MobileAgentState} \\
\textit{MobileAgentState}' \\
\Delta \textit{AgentState} \\
\Xi \textit{MobileAgentPerception} \\
\Xi \textit{MobileAgentAction} \\
\hline
\end{array}
$$

Quando um agente móvel atua em um ambiente, o ambiente muda e também suas percepções e ações (veja parágrafo acima). Isso é descrito pela aplicação das funções:

- canperceive sobre o ambiente e as ações de percepção.
- mobilewillperceive sobre canais, motivações, objetivos e percepções possíveis.
- mobileactions sobre canais, motivações, objetivos, percepções reais[10] e o ambiente.

MobileAgentInteracts

$$\Delta MobileAgentState$$
$$AgentInteracts$$

$$possiblepercepts' = canperceive\ environment'$$
$$\quad perceivingactions$$
$$actualpercepts' = mobilewillperceive\ channels'$$
$$\quad motivations'\ goals'\ possiblepercepts'$$
$$willdo' = mobileactions\ channels'\ motivations'$$
$$\quad goals'\ actualpercepts'\ environment'$$

Um sistema multiagente com mobilidade nada mais é do que um sistemas multiagente com um conjunto de agentes móveis e algumas restrições, são elas:

- O conjunto de agentes móveis é um subconjunto do conjunto dos agentes;
- O conjunto de agentes é composto pelos agentes autônomos, agentes servidores e agentes móveis.
- Cada agente móvel do sistema se comunica com ao menos um outro agente móvel do sistema[11].

MultiAgentSystemWithMobility

$$MultiAgentSystem$$
$$mobileagents : \mathbb{P}\ MobileAgent$$

$$mobileagents \subset agents$$
$$agents = autonomousagents \cup serveragents \cup$$
$$\quad mobileagents$$
$$\forall m : mobileagents \bullet \exists n : mobileagente \mid$$
$$\quad m \neq n \bullet m.channels \cap n.channels \neq \varnothing$$

[10] Percepções reais são uma parte das percepções possíveis, pois o agente pode muitas vezes desconsiderar parte das percepções, por exemplo, numa situação em que algum sensor não está funcionando corretamente.

[11] Essa condição garante que o grafo de comunicação não seja desconexo.

4 Conclusões

O principal produto deste trabalho é a especificação formal do modelo SMART estendido com o conceito de mobilidade. Essa especificação pode servir como base para a implementação de um sistema consistente com o modelo proposto.

Note que existe um modelo matemático para agentes móveis chamado π-calculus, esse modelo se baseia na comunicação entre os agentes, permitindo a análise de características como deadlock, *starving* etc. O modelo proposto utiliza a notação Z que se baseia em estados, dessa forma podemos fazer verificações de consistência, aderência de implementação, etc e também refinar a especificação obtendo um sistema funcional.

Um dos objetivos deste trabalho era descobrir se especificar esse modelo seria viável. Agora possíveis continuações para esse projeto seriam:

- Realizar mais verificações, inclusive sobre a especificação do SMART;
- Implementar um sistema consistente com o modelo;

Referências

[CG98] L. Cardelli and A. Gordon. *Mobile ambients.* In *Foundations of Software Science and Computation Structures: First International Conference*, FOSSACS '98, Springer-Verlag, 1998.

[dL01a] M. d'Inverno and M. Luck. *Understanding Agent Systems.* Springer-Verlag 2001.

[dL01b] M. d'Inverno and M. Luck. *Formal Agent Development: Framework to System.* In J. Rash, C. Rouff, W. Truszkowski, D. Gordon, and M. Hinchey, editors, in *Formal Approaches to Agent-Based Systems: First International Workshop*, FAABS 2000, LNAI 1871, pages 133-147. Springer, 2001.

[dKLW88] M. d'Inverno, D. Kinny, M. Luck and M. Wooldridge. *A Formal Specification of dMARS.* In Intelligent Agents IV In Proceedings of the Fourth International Workshop on Agent Theories, Architectures and Languages, Singh, Rao and Wooldridge (eds.), Lecture Notes in Artificial Intelligence, 1365, Springer-Verlag, 1998.

[LP04] H. R. Lewis and C. H. Papadimitriou. *Elementos de Teoria da Computação*. Bookman, segunda ediçao, 2004.

[Jac96] J. Jacky. *The Way of Z*. Cambridge University Press, 1996.

[Hsi02] Hsi-Kuo Li. *Software Agent*. WWW: Knowledge Systems Institute, http://www.ksi.edu/thesis/danli/danli.pdf, 2002.

[MM02] L. Mariani and E. Merelli. *Agent Reactive Component for SMART*. Third International Symposium *From Agent Theory to Agent Implementation*, Vienna, 2002.

[Mil95] R. Milner. *Communication and Concurrency*. Prentice Hall, 1995.

[Mil99] R. Milner. *Communicating and Mobile Systems: the π-calculus*. Cambridge University Press, 1999.

[Sil04] L. P. da Silva. *Um Modelo Formal para a Quinta Disciplina*, tese de doutorado, IME-USP, 2004.

[Spi89] J. M. Spivey. *Understanding Z - A Specification Language and its Formal Semantics*. Cambridge University Press, 1989.

[WD96] J. Woodcock, J. Davies. *Using Z: Specification, Refinement, and Proof*. Prentice Hall, 1996.

[Wor96] J. B. Wordsworth. *Software Engineering with B*. Addison-Wesley, 1996.

Parte II
Parte subjetiva

5 Desafios e Frustrações Encontrados

No início da iniciação científica tive contato com alguns métodos formais como Z e B, depois li um livro e alguns artigos sobre CCS e π-calculus. Após esse contato inicial estudei mais a fundo Z e o modelo SMART.

Apesar de muito interessante o estudo do modelo SMART foi um pouco frustrante, pois o livro muitas vezes ficava bastante confuso, além disso parte da especificação do SMART apresentava erros e algumas vezes partes inteiras estavam faltando, para resolver isso tive

que buscar partes da especificação em outros trabalhos e nos artigos originais dos autores.

Após as principais decisões terem sido tomadas, iniciar a especificação se mostrou bastante difícil e saber quando parar se mostrou mais difícil ainda, pois sempre que eu lia a especificação pensava em algo para mudar. Após várias mudanças decidimos por um versão final.

Outra frustração foi que a maioria das ferramentas para Z são comerciais, e as ferramentas livres, em geral, não são muito intuitivas, por isso gastei um tempo considerável procurando ferramentas e tentando aprender a usá-las. Sistemas multiagente implementados apresentaram o mesmo problema, embora existam alguns livres, parte deles está descontinuada, e grande parte têm pouca documentação.

6 Disciplinas Relevantes

As disciplinas básicas,

- MAC0110 - Introdução à Computação
- MAC0122 - Princípios de Desenvolvimento de Algoritmos
- MAT0138 - Álgebra I para Computação
- MAT0139 - Álgebra Linear para Computação

foram de suma importância, embora não se relacionem diretamente com os assuntos estudados durante a iniciação me deram maturidade para lidar com certos aspectos dos temas estudados.

As disciplinas de maior relevância para a iniciação científica foram:

- MAC0329 - Álgebra Booleana e Aplicações
- MAC0239 - Métodos Formais em Programação
- MAC0425 - Inteligência Artificial
  Relevantes principalmente para fixar os conceitos de lógica.
- MAC0332 - Engenharia de Software
  Tive contato com especificação formal (Z, UML etc) e métodos de desenvolvimento.
- MAC0435 - Métodos Formais para Especificação e Construção de Programas
  Tive maior experiência com especificação em Z.
- MAC0340 - Laboratório de Engenharia de Software
  Tive maior experiência com especificações e métodos de desenvolvimento.

7 Interação com a orientadora

Durante a maior parte da minha iniciação minha orientadora promoveu um seminário semanal com todos os orientandos dela, onde a cada reunião algum aluno apresentava o trabalho que estava desenvolvendo, possibilitando dessa forma grande troca de informações e experiências. Além disso, ela sempre se mostrou muito disposta a conversar, tirar dúvidas etc.

8 Passos que tomaria se fosse continuar atuando na área

Se fosse continuar atuando na área em que fiz iniciação científica cursaria mais disciplinas nas áreas de engenharia de software, inteligência artificial, sistemas distribuidos e lógica. Como disse em outra seção, uma possivel continuação deste trabalho seria implementar um sistema consistente com o modelo.

9 Agradecimentos

Gostaria de agradecer especialmente à professora Ana Cristina Vieira de Melo, pela paciência, apoio e disponibilidade. Agradeço também aos amigos Paulo Roberto de Araújo França Nunes e Paulo Salem com quem diversas vezes pude discutir sobre os mais variados temas.

Parte III
Apêndices

A Especificação do SMART

$[Attribute]$
$Environment == \mathbb{P}_1\ Attribute$
$[Action]$
$Goal == \mathbb{P}_1\ Attribute$
$[Motivation]$

Entity

$attributes : \mathbb{P}\, Attribute$
$capabilities : \mathbb{P}\, Action$
$goals : \mathbb{P}\, Goal$
$motivations : \mathbb{P}\, Motivation$

$attributes \neq \varnothing$

Env

$environment : Environment$
$entities : \mathbb{P}\, Entity$

$environment \neq \varnothing$
$\bigcup \{e : entities \bullet e.attributes\} \subseteq environment$

EntityState

$Entity$
Env
$situation : \mathbb{P}\, Attribute$

$situation \neq \varnothing$
$attributes \cap situation = \varnothing$
$attributes \cup situation \subset environtment$

$\Delta EntityState$

$\Xi Entity$
$EntityState$
$EntityState'$

$situation' \subset environment'$

Object

$Entity$

$capabilities \neq \varnothing$

ObjectAction

$Object$

$objectactions : Environment \rightarrow \mathbb{P}\, Action$

$\forall\, environment : Environment \bullet$
$\quad objectactions\ \ environment) \subseteq capabilities$

ObjectState

$EntityState$

$ObjectAction$

$willdo : \mathbb{P}\, Action$

$willdo = objectactions\ \ environment$

$willdo \subseteq capabilities$

$\Delta ObjectState$

$ObjectState$

$ObjectState'$

$\Delta EntityState$

$\Xi ObjectAction$

$effectinteraction :$
$\quad Environment \rightarrow \mathbb{P}\, Action \nrightarrow Environment$

ObjectInteracts

$\Delta ObjectState$

$environment' =$
$\quad effectinteraction\ \ environment\ \ willdo$

$willdo' = objectactions\ \ environment'$

Agent

$Object$

$goals \neq \varnothing$

AgentPerception

$$
\begin{array}{l}
Agentperceivingactions : \mathbb{P}\, Action \\
canperceive : Environment \rightarrow \mathbb{P}\, Action \nrightarrow View \\
willperceive : \mathbb{P}\, Goal \rightarrow View \rightarrow View \\
\hline
perceivingactions \subseteq capabilities \\
\forall\, env : Environment;\ as : \mathbb{P}\, Action \bullet \\
\qquad as \in \mathrm{dom}(canperceive\ env) \Rightarrow \\
\qquad as = perceivingactions \\
\mathrm{dom}\, willperceive = goals
\end{array}
$$

AgentAction

$$
\begin{array}{l}
Agent \\
ObjectAction \\
agentactions : \mathbb{P}\, Goal \rightarrow View \rightarrow Environment \rightarrow \mathbb{P}\, Action \\
\hline
\forall\, gs : \mathbb{P}\, Goal;\ v : View;\ env : Environment \bullet \\
\qquad (agentactions\ gs\ v\ env) \subseteq capabilities \\
\mathrm{dom}\, agentactions = goals
\end{array}
$$

AgentState

$$
\begin{array}{l}
AgentPerception \\
AgentAction \\
ObjectState \\
possiblepercepts, actualpercepts : View \\
\hline
actualpercepts \subseteq possiblepercepts \\
possiblepercepts = canperceive\ environment\ perceivingactions \\
actualpercepts = willperceive\ goals\ possiblepercepts \\
perceivingactions = \varnothing \Rightarrow posspercepts = \varnothing \\
willdo = agentactions\ goals\ actualpercepts\ environment
\end{array}
$$

Δ*AgentState*

$$
\begin{array}{l}
AgentState \\
AgentState' \\
\Delta ObjectState \\
\Xi AgentAction \\
\Xi AgentPerception
\end{array}
$$

AgentInteracts

$\Delta AgentState$

$ObjectInteracts$

$possiblepercepts' =$
$\quad canperceive\ environment'\ perceivingactions$
$actualpercepts' =$
$\quad willperceive\ goals\ possiblepercepts'$
$willdo' =$
$\quad agentactions\ goals\ actualpercepts'\ environment'$

AutonomousAgent

$Agent$

$motivations \neq \varnothing$

AutonomousAgentPerception

$AutonomousAgent$
$AgentPerception$
$autowillperceive :$
$\quad \mathbb{P}\, Motivation \rightarrow \mathbb{P}\, Goal \rightarrow Environment \rightarrow View$

$\mathrm{dom}\ autowillperceive = motivations$

AutonomousAgentAction

$AutonomousAgent$
$AgentAction$
$autoactions : \mathbb{P}\, Motivation \rightarrow \mathbb{P}\, Goal \rightarrow$
$\quad View \rightarrow Environment \rightarrow \mathbb{P}\, Action$

$\mathrm{dom}\ autoactions = motivations$

AutonomousAgentState

$AgentState$
$AutonomousAgentPerception$
$AutonomousAgentAction$

$willdo =$
$\quad autoactions\ motivations\ goals$
$\quad actualpercepts\ environment$

$\Delta AutonomousAgentState$

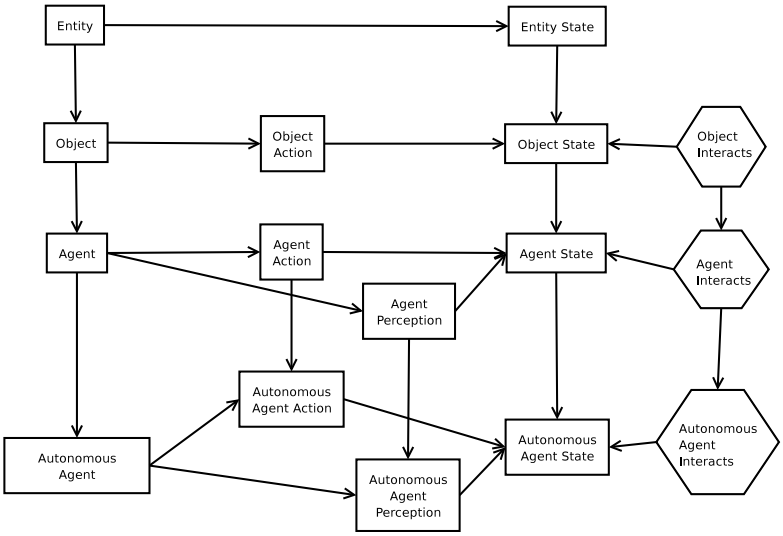
$AutonomousAgentState$
$AutonomousAgentState'$
$\Delta AgentState$

$autowillperceive' = autowillperceive$
$autoactions' = autoactions$

$AutonomousAgentInteraction$

$\Delta AutonomousAgentState$
$AgentInteracts$

$possiblepercepts' = canperceive\ \ environment'$
$\quad perceivingactions$
$actualpercepts' = autowillperceive\ \ motivations'$
$\quad goals'\ \ possiblepercepts'$
$willdo' = autoactions\ \ motivations'$
$\quad goals'\ \ actualpercepts'\ environment'$



$NeutralObject$

$Object$

$goals = \varnothing$
$motivations = \varnothing$

ServerAgent

$Agent$

$motivations = \varnothing$

MultiAgentSystem

$entities : \mathbb{P}\, Entity$
$objects : \mathbb{P}\, Object$
$agents : \mathbb{P}\, Agent$
$autonomousagents : \mathbb{P}\, AutonomousAgent$
$neutralobjects : \mathbb{P}\, NeutralObject$
$serveragents : \mathbb{P}\, ServerAgent$

$autonomousagents \subseteq agents \subseteq object \subseteq entities$
$agents = autonomousagents \cup serveragents$
$object = agents \cup neutralobjects$
$\#agents \geq 2$
$\#autonomousagents \geq 1$
$\exists\, aa1, aa2 : agents \bullet aa1.goals \cap aa2.goals \neq \varnothing$

B Glossário de notação

B.1 Definições

Abreviação

$a == x$

Tipo livre

$a ::= b \mid c \mid ...$

Tipo primitivo

$[a]$

Operador prefixo

$a\_$

Operador posfixo

$\_a$

Operador infixo

$$\_a\_$$

B.2 Lógica

Constante lógica verdadeiros

$$true$$

Constante lógica falsa

$$false$$

Negação lógica

$$\neg\, p$$

Conjunção lógica

$$p \wedge q$$

Disjunção lógica

$$p \vee q$$

Implicação lógica

$$p \Rightarrow q$$

Equivalência lógica

$$p \Leftrightarrow q$$

Quantificado universal

$$\forall\, X \bullet q$$

Quantificador existencial

$$\exists\, X \bullet q$$

Definição local

$$(\textbf{let}\ a == x;\ ... \bullet p)$$

B.3 Conjuntos e Expressões

Igualdade

$x = y$

Desigualdade

$x \neq y$

Pertence

$x \in A$

Não pertence

$x \notin A$

Conjunto vazio

$\varnothing$

Subconjunto

$A \subseteq B$

Subconjunto próprio

$A \subset B$

Definição de conjunto (set comprehention)

$X \bullet x$

Expressão lambda

$(\lambda X \bullet x)$

Tupla

$(x, y, ...)$

Par

(x, y)

Produto Cartesiano

$A \times B$

Conjunto potência

$\mathbb{P}\, A$

Interseçao de conjuntos

$A \cap B$

União de conjuntos

$A \cup B$

Diferença de conjuntos

$A \setminus B$

Número de elementos de um conjunto

$\# A$

B.4 Relações

Relação binária

$A \leftrightarrow B$

Mapeamento

$a \mapsto b$

Domínio de uma relação

$\mathrm{dom}\, R$

Imagem de uma relação

$\mathrm{ran}\, R$

Composição de relações

$R \fatsemi S$

Composição inversa de relações

$$R \circ S$$

Restrição de domínio

$$R \lhd S$$

Antirestrição de domínio

$$R \lhd\!\!\!\!- S$$

Restrição de imagem

$$R \rhd S$$

Antirestrição de imagem

$$R \rhd\!\!\!\!- S$$

Inversa

$$R^{\sim}$$

Imagem relacional

$$R(\!| S |\!)$$

Sobreposição

$$R \oplus S$$

B.5 Funções

Função parcial

$$X \nrightarrow Y$$

Função total

$$X \rightarrow Y$$

Função injetora parcial

$$X \rightarrowtail Y$$

Função injetora total

$X \rightarrowtail Y$

Função sobrejetora parcial

$X \twoheadrightarrow Y$

Função sobrejetora total

$X \rightarrow\!\!\!\!\!\rightarrow Y$

Função bijetora

$X \rightarrowtail\!\!\!\!\!\rightarrow Y$

B.6 Seqüências

Seqüência finita

$\mathrm{seq}\, X$

Seqüência finita não vazia

$\mathrm{seq}_1\, X$

Seqüência injetora

$\mathrm{iseq}\, X$

Concatenação

$s \frown t$

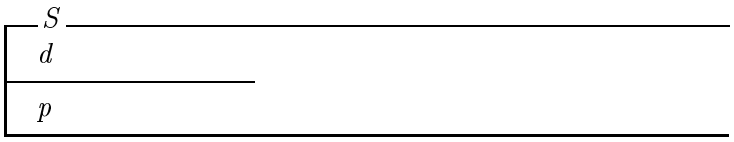
Primeiro elemento da seqüência

$head\ S$

Tudo exceto o primeiro elemento

$tail\ S$

B.7 Esquema



B.8 Definição Axiomática

$$d$$
$$p$$

B.9 Definição Genérica

$$[X]$$
$$d$$
$$p$$

B.10 Cálculo de Esquemas

Esquema horizontal

$$S == T$$

Inclusão de esquema

$$[T;\ ... \mid ...]$$

Seleção de componente

$$z.a$$

Binding

$$\theta S$$

Negação de esquema

$$\neg\, S$$

Conjunção de esuqemas

$$S \land T$$

Disjunção de esquemas

$$S \lor T$$

Composição de esquemas

$$S \fatsemi T$$

B.11 Convenções

Variável de entrada de uma operação

$a?$

Variável de saída de uma operação

$a!$

Variável de estado antes de operação

a

Variável de estado depois de operação

a'

Esquema de estado antes de operação

S

Esquema de estado depois de operação

S'

Mudança de estado

ΔS

Manutenção de estado

ΞS