Arquitetura Unificada Agente-Objeto: Usando Objetos da Tecnologia Orientada a Objetos para Modelar o Ambiente de Sistemas Multiagentes

Márcia Ito
Laboratório de Pesquisa em Ciência de Serviços (LaPCiS) – Programa de Pós-Graduação do Centro Paula Souza (POS-CEETEPS) – São Paulo – Brasil
marcia.ito@centropaulasouza.sp.gov.br

Leandro Ramos da Silva
Laboratório de Pesquisa em Ciência de Serviços (LaPCiS) – Programa de Pós-Graduação do Centro Paula Souza (POS-CEETEPS) – São Paulo – Brasil
lramos@gmail.com

Resumo – Hoje os sistemas além de prover informações devem auxiliar na tomada de decisão de seus usuários, aumentando desta forma a complexidade dos aplicativos. Assim é preciso desenvolver elementos que tenham a capacidade de resolver problemas, e neste ponto a tecnologia orientada a agentes parece ser uma das soluções possíveis. Para que estes elementos, os agentes, possam solucionar problemas é preciso que tenham a informação disponível. Estas informações, hoje, encontram-se implementadas com a tecnologia de orientação a objetos. Assim, é objetivo deste artigo apresentar uma arquitetura que possibilite o desenvolvimento de sistemas em que os agentes atuam como processadores e os objetos como elementos que provêem as informações aos agentes.

Palavras-chave: Arquitetura de Software, Tecnologia de Orientação a Agentes, Tecnologia Orientada a Objetos.

Abstract – Actually, systems in addition to providing information should assist its users in decision making, thereby increasing the complexity of applications. So you need to develop elements that have the ability to solve problems, and at this point the agent oriented technology seems to be a possible solution. For these elements, the agents, solve problems we need to have the information available. This information, today, are implemented with object oriented technology. It is therefore aim of this paper presents an architecture which enables the development of systems in which agents act as processors and objects as elements that provide information to agents.

Palavras-chave: Software Architecture, Agent Oriented Technology, Object Oriented Technology

Introdução

Com o aumento da complexidade dos sistemas em que se exige principalmente que os mesmos auxiliem na tomada de decisão a tecnologia de orientação a agentes vem despontando como uma alternativa para o desenvolvimento deste tipo de aplicação. A tecnologia orientada a agentes é indicada em sistemas que precisam ter a capacidade para resolver problemas e

responsabilidades. Além disso, neste tipo de sistema a solução não pode ser descrita do princípio ao fim, devido a diversas mudanças que podem ocorrer nas regras de análise dos resultados e definições de conduta. Nesta tecnologia os elementos de software, os agentes, são projetados para resolverem problemas de forma flexível por meio de coordenação, negociação e troca de informações entre seus elementos. Percebe-se que os sistemas complexos possuem estas características o que faz com que o uso da tecnologia orientada a agentes seja uma solução viável. [1] [2] [3]

Assim sendo, o principal elemento nessa tecnologia é o agente. De acordo com Woodridge [4] “Um agente é um sistema computacional que está situado em algum ambiente e que é capaz de ações autônomas neste ambiente de modo a alcançar os seus objetivos.” Assim, pode-se dizer que os Agentes são componentes de softwares sociáveis; ou seja, eles são capazes de interagir com os demais agentes de forma a caracterizar uma “organização”. Por outro lado, os agentes são capazes de racionalizar, pensar, acerca de algum assunto e produzir uma resposta direcionada por ações. E por último, os agentes estão inseridos em um ambiente que pode ser qualquer elemento externo ao agente capaz de interagir com ele.

Em termos conceituais o agente deriva da noção de agência. De acordo com esse conceito, o agente atua no lugar de alguém, pois ele tem a capacidade de se comunicar com outros para realizar a tarefa que lhe foi designada [5]. No software, os agentes representam papéis de pessoas do mundo real que têm as suas atividades modeladas no sistema. Da mesma forma que existem interações entre as pessoas para a execução de tarefas, nos sistemas desenvolvidos com a tecnologia de agentes há comunicação entre os agentes para realizar as suas atividades, ou em outras palavras, alcançar os seus objetivos. [6] [4]

Assim, o objetivo deste artigo é apresentar uma arquitetura que permita representar esta noção de agência, ao utilizar agentes como papéis de pessoas do mundo real e os objetos como as entidades do ambiente que esses agentes necessitam para realizar suas tarefas.

Para o desenvolvimento da arquitetura unificada agente-objeto fez-se um estudo bibliográfico dos conceitos da tecnologia de agentes e objetos relacionados com a evolução do desenvolvimento de software. A seguir a alternativa de modelagem do ambiente físico dos agentes por meio de objetos é sugerida. Finalmente a arquitetura é desenvolvida e avaliada.

Tecnologia Orientada a Agentes e o Desenvolvimento de Software

No início do desenvolvimento de sistemas tinha-se o conceito monolítico, no qual o sistema todo era concebido como um único bloco. Os programadores tinham total controle sobre os programas e seu estado. As chamadas de suas funcionalidades eram feitas pelo operador de computadores. À medida que os sistemas começaram a ficar mais complexos e a exigir mais memória e processadores mais poderosos, os programadores sentiram a necessidade de introduzir algum tipo de organização no código. Surgiu, então, o conceito de modularidade. Os programas passaram a ser organizados em fragmentos menores denominados subrotinas, utilizadas em várias situações, o que aumentou, a integridade local. As subrotinas, porém, não possuem controle sobre os estados das variáveis que manipulam e são executadas por inteiro quando chamadas. [7] [8] [9]

A orientação a objetos, por sua vez, acrescenta ao conceito modular uma das características que se obtém ao decompor os sistemas por componentes: a possibilidade de ter controle sobre o estado das variáveis que manipula. Nessa tecnologia, os objetos são passivos, pois não possuem controle sobre suas ações. As ações são executadas apenas quando solicitadas. O objeto que executa não pode, *a priori*, interagir com o objeto que solicitou a execução da ação. [8] [10]

Do mesmo modo, a orientação a agentes acrescenta à tecnologia anterior o conceito de autonomia. Os agentes possuem controle não somente do estado de suas variáveis, mas também sobre as suas ações. Eles têm as suas próprias regras e objetivos e isso faz com que pareçam "objetos ativos com iniciativa". Assim, ao contrário dos objetos que não interagem entre si de forma dinâmica, na tecnologia orientada a agentes isso é possível. Um agente pode questionar outro na execução de alguma ação, comunicá-lo de que há falta de dados para executá-la, pedir outras informações para complementá-la, ou até mesmo recusar a sua execução. Na prática, dois sistemas de fabricantes diferentes poderiam trocar dados com segurança e sem a necessidade de pertencerem à mesma plataforma. Por exemplo, o agente do aplicativo A somente fornece a informação caso o agente do aplicativo B tenha permissão e os requisitos necessários para isso. Com relação à utilização de componentes no sistema, ocorre uma diminuição, já que o agente encapsula toda a infra-estrutura de comunicação e manipulação das informações. Desse modo, o desenvolvedor preocupa-se apenas em projetar as responsabilidades e os objetivos dos agentes que farão parte do sistema. [11] [8] [4]

Desse modo, observa-se que, na verdade, o agente evolui de conceitos anteriores, conforme síntese da evolução no desenvolvimento de software apresentado na tabela 1.

Tabela 1 – Evolução dos conceitos para o desenvolvimento de software desde o seu início até os dias atuais Fonte: ODELL, J. Objects and Agents: How do they differ? [8]

Unidade\Conceito	Monolítico	Modular	Orientado a Objetos	Orientado a Agentes
Comportamento	Não modular	Modular	Modular	Modular
Estado	Externo	Externo	Interno	Interno
Invocação	Externo	Externo (chamada de função)	Externo (mensagem)	Interno (regras e objetivos)

Para operarem, os agentes precisam existir dentro de um ambiente que define as propriedades do mundo no qual atuam. Uma vez que os agentes precisam entender e conhecer o ambiente em que se encontram, é necessário criar uma estrutura adequada que os comporte e os represente. Por sua vez, o ambiente pode ser classificado em: físico e de comunicação. [12] [13]

O ambiente físico se refere ao meio em que o agente se encontra e no qual precisa interagir a fim de alcançar seus objetivos ou até decidir qual deve perseguir. Por exemplo, o agente pode interagir com o mundo por meio de sensores e câmeras. Nesse caso, o modelo do ambiente físico é mais simples, pois as interações que os agentes fazem com o ambiente são diretas. Entretanto, existem situações em que ele é totalmente representado por

software. Nessa situação, ele contém os princípios e os processos que governam e apóiam uma população de entidades. Os modelos propostos relacionam-se com o tipo de interação que o agente tem com o ambiente. [12] [13] [14]

O ambiente de comunicação é o meio pelo qual os agentes se inter-relacionam, ou seja, se comunicam. Ele contém os princípios e os processos que governam e apóiam a troca de idéias, conhecimentos, informações e dados. Além disso, possui funções e estruturas que são empregadas para permitir a comunicação e o protocolo de interação entre os agentes e os grupos de agentes.

Schwambach, Pezzin, Falbo [15], notando esta lacuna, propõem em seu trabalho a utilização da tecnologia de orientação a objetos para implementar o ambiente dos agentes. A partir dessa proposta, desenvolveu-se um framework na qual o ambiente físico é representado pelo paradigma da orientação a objetos.

A seguir demonstra-se como o uso da teoria de agentes e objetos pode representar o mundo real. O objetivo principal é apontar os elementos do ambiente físico dos agentes que são representados pelos objetos da tecnologia de orientação a objetos.

Arquitetura Unificada Agente-Objeto: Ambiente Físico representado pela Orientação a Objetos

A técnica de produção de modelos é antiga e utilizada em todas as áreas de conhecimento, quando se deseja simplificar a representação do sistema estudado. A redução da complexidade ocorre pela decomposição da realidade em elementos “fáceis de entender”. O modelo é uma abstração semanticamente encapsulada do sistema, demonstrando a organização e o comportamento dele. [10]

Na engenharia de software, os modelos comunicam a estrutura e o comportamento do software, permitem visualizar e controlar a arquitetura, gerenciam os riscos e possibilitam compreender melhor o aplicativo que se está construindo, expondo oportunidades de simplificação e reusabilidade. Aliás, a reusabilidade é muito útil no desenvolvimento de software, pois ter códigos prontos que possam ser utilizados novamente aumenta a produtividade. [10]

Outro aspecto da modelagem é que os melhores modelos são aqueles que refletem a realidade. Como os modelos são desenvolvidos para simplificar a realidade, é importante que eles não a distorçam. Para o desenvolvimento de software, o modelo adequado é aquele em que, dada uma situação real, seja possível modelar uma solução que permita facilmente transformá-lo em um sistema computacional. [10]

A orientação a objetos, nesse sentido, parece ser um modo natural de pensar sobre o mundo e de projetar programas de computador. Quando se pensa, por exemplo, em bola, tela, botão, etc., percebe-se que todos eles podem ser representados por objetos na orientação a objetos. [10] [16]

Todavia, a abstração na orientação a objetos é natural somente quando se modelam elementos estáticos que não possuem um comportamento complexo. Quando é preciso modelar elementos que controlam outros de forma

ativa, a dificuldade fica evidente. Por exemplo, ao desenvolver uma solução orientada a objetos para trocar o pneu de um carro, é relativamente fácil encontrar os objetos: pneu, carro, macaco e chave de roda. Porém, a afirmação de que esses objetos bastam para modelar a troca de pneu não soa natural, pois nesse caso não há a necessidade de uma pessoa que manipule os objetos. Eles por si só realizam as ações – o que confude as pessoas, já que não são atos que ocorrem no mundo real. Resumindo, percebe-se que modelar elementos estáticos é natural na orientação a objeto, enquanto que o comportamento dinâmico não é simples, pois exige uma capacidade de abstração maior.

Diferentemente, na orientação a agentes temos entidades autônomas e ativas que atuam num ambiente físico que é composto por elementos estáticos. A facilidade está em modelar os elementos ativos que controlam os outros e que são representados pelos agentes. No exemplo da troca de pneu, é fácil encontrar o elemento ativo: o mecânico ou o motorista que troca o pneu. Porém, como não há preocupação em representar o ambiente físico é difícil saber qual a representação para o carro, ou até saber se o macaco e a chave de roda são agentes ou não para a solução do problema.

Ao aliar as tecnologias de orientação a objetos e agentes, consegue-se uma forma de representar tanto os elementos ativos quanto os estáticos e, portanto, de obter modelos próximos à representação do mundo real.

No exemplo de cálculo da média final de um aluno que um professor realiza, o professor é um elemento ativo, com objetivos e critérios próprios, e melhor representado por um agente. Porém, o professor não consegue alcançar o seu objetivo sem acessar os elementos de seu ambiente, necessários para a execução da tarefa a ele designada. Em suma, o professor precisa da prova do aluno para olhar as notas e a matrícula, de uma calculadora para inserir as notas e calcular a média e do boletim para escrever a média final. No caso, olhar, inserir e escrever são interações do professor que ocorrem com elementos estáticos: provas, calculadora e boletim. A calculadora é um elemento estático, pois mesmo realizando a ação de calcular, não tem vontade própria; por exemplo, não recusa calcular a média de dois números. Por outro lado, de acordo com Wooldridge [4] e Russell [12], da mesma forma que os humanos têm olhos, ouvidos, mãos e outros órgãos para interagire com o ambiente, o agente também tem a sua forma de interação com o seu ambiente. Pode-se concluir que a prova, a calculadora e o boletim são elementos estáticos que representam o ambiente físico do agente professor.

Os elementos estáticos são melhor representados pela orientação a objetos. Afinal, segundo Booch, Rumbaugh, Jacobson [10], a orientação a objetos lida naturalmente com o conceito de objeto, que é uma abstração de elementos encontrados na realidade que se deseja modelar. Assim, é possível dizer que o ambiente físico pode ser representado por objetos da orientação a objetos.

Como o ambiente físico pode ser representado pelos objetos, é necessário definir a forma de interação entre os objetos e os agentes. A interação, nesse caso, representa o ato de “sentir”, “olhar”, “manipular”, entre outros que o agente exerce sobre o seu ambiente. Na orientação a objetos, a forma de ter acesso às operações dos objetos é enviando uma mensagem a eles. Dessa forma, a interação entre o ambiente físico e os agentes ocorre por meio de troca de mensagens entre o agente e o objeto.

O uso das teorias de agentes e objetos facilita a modelagem de sistemas computacionais, tornando-os mais próximos da realidade que automatizam.

No próximo capítulo apresenta-se a utilização desses conceitos no desenvolvimento de um módulo de identificação de pacientes para simular a interação entre os agentes e os objetos nesta arquitetura.

Implementação e Simulação da Arquitetura Unificada Agente-Objeto: O Módulo de Identificação do Paciente no Sistema TeleDM

O contexto para o desenvolvimento do sistema TeleDM [17] envolve o processo de monitoração do paciente diabético e para tanto é necessário desenvolver um módulo que faça a identificação do paciente. Este módulo é, também, utilizado para avaliar se a interação entre o agente o seu ambiente representado por objetos ocorre de forma transparente, sem que o usuário perceba ou que erros no resultado do sistema ocorram por representação inadequada do ambiente.

Inicialmente é preciso elaborar a arquitetura unificada agente-objeto. Neste caso o agente é o atendente que terá como objetivo de identificar o paciente. Em seguida o ambiente físico é modelado. Sendo o ambiente físico tudo aquilo que o agente manipula para realizar as suas atividades, no Sistema TeleDM ele é todo representado por software, por não existirem interações com câmeras ou sensores. Assim, neste módulo as entidades que representam o ambiente que os agentes precisam manipular são as fichas dos pacientes. Para implementar o ambiente físico com a tecnologia da orientação a objetos é preciso analisar a ficha do paciente e definir as classes que as representam. Em seguida, deve-se distribuir os atributos e operações da ficha do paciente nas respectivas classes. As classes encontradas são:

- Endereço – Representa o endereço completo da pessoa. Uma pessoa pode encontrar-se em mais de um endereço; um médico, por exemplo, que pode ter como referências a clínica e o hospital em que atende;
- Médico – Controla as informações relacionadas ao médico do paciente;
- Paciente – Controla as informações do paciente;
- Telefone – Representa os números de telefones para contato.

Após encontrar as classes, estabelecem-se os relacionamentos que existem entre elas e elabora-se o diagrama de classe. Esse diagrama representa a estrutura estática de classes do sistema, onde a estrutura descrita é sempre válida em qualquer ponto do ciclo de vida de seus objetos. [10] [16]

Em seguida, as permissões de acesso a cada classe devem ser estabelecidas. Para isso analisou-se a ação que o agente faz sobre o objeto – pegar, guardar (salvar), escrever (incluir/alterar), apagar (excluir) e olhar (consultar). Quando o agente não precisa dele para as suas atividades, o acesso é definido como “Não tem acesso”. Aqueles que têm a permissão de escrever são os responsáveis pela classe. No módulo de identificação do paciente tem-se o agente atendente que terá todas as permissões sobre as classes definidas.

Para esta validação não é necessário fazer a simulação de todas as situações, assim escolheu-se aquela em que o agente acessa vários objetos para conseguir alcançar o seu objetivo que é identificar o paciente ou encaminhá-lo para o setor de “acompanhamento de serviços”. A situação que

se encaixa neste caso é a que o paciente se enganou na matrícula, ou seja, a matrícula foi encontrada no sistema, porém não corresponde ao paciente que entrou em contato. Neste caso o paciente então é identificado pelo seu nome e o nome da mãe.

Ao iniciar a execução do software o agente atendente apresenta o menu principal. O usuário, ao escolher a opção “atender o contato”, faz com que o agente atendente selecione o plano (plano 1) para alcançar o seu objetivo. O plano 1 é aquele que identifica o contato pela matrícula. O agente percebe que consegue resolver sozinho o problema e inicia a interação com o usuário, questionando se o contato possui o número de matrícula. É a partir desse ponto que, dependendo da simulação, o agente prossegue com a execução do plano atual (plano 1), ou procura por um outro plano (plano 2). Para a situação em que o paciente enganou-se de matrícula, ele fornece uma matrícula que coincidentemente existe na base de dados, porém não pertence a ele. O número de matrícula fornecido é 2, e pertence ao paciente fictício “Pedro da Silva”. Esse fato obriga o agente a utilizar o plano 2 para conseguir alcançar o seu objetivo. O quadro 1 apresenta as ações executadas pelo agente atendente a partir do momento em que o usuário fornece o número de matrícula.

1. I'm asking the Identifier Number!
2. I'll search the patient with identifier: 2
3. I find the patient!
4. I'll do Perguntar_nome_telefone
5. I'm asking if Pedro da Silva and the telephone number, 9991-9999 are correct.
6. The patient's data isn't correct!
7. I've stored AtendenteCloset.amb.
8. The result of my work is: false
9. I'm Atendente, my role in this cicle is active and I'll find a work!
10. I'm Atendente, my role in this cicle is active and I'll find a work! So I'll choose a plan!
11. I have 2 plans to choose
12. I'm autonom for this goal!
13. The plan 0 choosed has these actions:
14. Perguntar_tipo_contato
15. Perguntar_nome_paciente
16. Perguntar_nome_mae
17. Procurar_paciente_e_mae
18. Mostrar_numero_matricula
19. Perguntar_fazer_o_que
20. I'm Atendente, my role in this cicle is active and I'll execute my plan!
21. I start to execute my plan!
22. I've restored my closet!
23. My list has 6 actions
24. I'll do Perguntar_tipo_contato
25. I'm interacting to user!
26. The contact is a patient!
27. I'll do Perguntar_nome_paciente
28. I'm asking the patient name!
29. The patient name is Joao da Silva
30. I'll do Perguntar_nome_mae

31.	I'm asking patient's mother name
32.	The patient's mother name is Maria da Silva
33.	I'll do Procurar_paciente_e_mae
34.	I'll search the patient Joao da Silva whose mother's name is Maria da Silva
35.	I find the patient!
36.	I'll do Mostrar_numero_matricula
37.	I'm showing the identifier number, 1

Quadro 1 – Relato das ações executadas pelo agente atendente após o usuário fornecer a matrícula que não pertence a ele

A terceira linha do quadro apresenta o momento em que o agente atendente instancia e acessa o objeto “Paciente” cujo identificador é 2. Na quarta linha do quadro o agente atendente acessa o objeto “Paciente” para ter a informação sobre o telefone do paciente. O objeto “Paciente” por sua vez acessa a informação no objeto telefone. Na linha 33 o agente atendente instancia e acessa o objeto “Paciente” cujo nome é João da Silva e tem a mãe com nome Maria da Silva. Na linha 36 o agente acessa o objeto “Paciente” para obter o identificador do paciente. Percebe-se que ao executar a sua tarefa o agente atendente conseguiu acessar e instanciar o objeto “Paciente” e o resultado da simulação apresenta o esperado.

Discussão e Conclusões

Por meio da simulação conclui-se que a interação entre o agente e o seu ambiente composto por objetos atingiu o resultado esperado. Os agentes acessavam o objeto de sua responsabilidade. A troca de mensagem entre o agente e o objeto ocorreu sem problemas e todas as informações necessárias para que o agente pudesse alcançar o seu objetivo foram obtidas. Caso tal fato não ocorresse, não teríamos sucesso na execução das simulações. Desta forma tem-se que a arquitetura agente-objeto é uma alternativa viável para o desenvolvimento de sistemas orientado a agentes em que o ambiente físico é totalmente representado por software.

Trabalhos futuros podem demonstrar que esta arquitetura otimiza o desempenho deste tipo de sistema, já que a comunicação é tende a ser menor entre os agentes, pois estes se comunica somente com os agentes executores e não com entidades passivas que são modeladas como agentes em arquitetura em que existam somente agentes como entidades de software.

Referências

[1] BERGENTI, F.; POGGI, A. A development environment for the realization of open and scalable multi-agent systems. In: GARIJO, F. J.; BOMAN, M. (eds.), **MultiAgent System Engineering**: Proceedings of the 9th European Workshop on Modelling Autonomous Agents in a Multi-Agent World. (Lecture Notes in Computer Science, v. 1647). London: Spring-Verlag, 1999, p. 52-62.

[2] JENNINGS, N. R. Agent-Oriented Software Engineering. In: GARIJO, F. J.; BOMAN, M. (eds.). **Multi agent system engineering**: proceedings of the

ninth european workshop on modelling autonomous agents in a multi-agent world, 1999. pp. 1-7. (Lectures Notes in Artificial Inteligence,1647).

[3] JUCHEN, M.; BASTOS, R. **Engenharia de sistemas multiagentes**: uma investigação sobre o estado da arte. Porto Alegre: Faculdade de Informática – PUCRS, 2001. 43 p. (Technical Report Series, n. 014)

[4] WOOLDRIDGE, M. **An introduction to multiagent systems**. 1ª. edição. (s. l.): John Wiley & Son., 2002. 348 p.

[5] TURBAN, E.; ARONSON, J. E. **Decision support systems and intelligent systems**, 5a. edição. (s. l.): Prentice Hall, 1998. 890 p.

[6] SICHMAN, J. S. **Raciocínio social e organizacional em sistemas multiagents**: avanços e perspectivas. 2003. 235 p. Livre Docência (Livre Docência em Engenharia Elétrica) – Escola Politécnica, Universidade de São Paulo, São Paulo, 2003.

[7] DIJKSTRA, E.W. **A discipline of programming**. 1ª. Edição. (Englewood Cliffs): Prentice Hall, 1976. 217 p.

[8] ODELL, J. **Objects and agents**: how do they differ? Draft 2.2. 1999. Disponível em: http://www.agent.ai/doc/upload/200302/odel00_5.pdf Acesso em 28 Jul 2010

[9] WIRTH, N. **Systematic programming**: an introduction. Englewood Cliffs, NJ: Prentice Hall, 1973. 169 p.

[10] BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. **The unified modeling language**: user guide. (s. l.): Addison-Wesley, 1999. 482 p.

[11] ITO, M. **Uma análise do fluxo de comunicação em organizações dinâmicas de agentes**. São Paulo, 1999. 141 p. Dissertação (Mestrado em Engenharia Elétrica). Escola Politécnica, Universidade de São Paulo, São Paulo, 1999.

[12] RUSSELL, S.; NORVIG P. **Artificial intelligence**: a modern approach. 2ª. edição, (s. l.): Prentice Hall, 2003, 1080 p.

[13] WEYNS, D.; PARUNAK, V. D.; MICHEL, F.; HOLVOET, T.; FERBER, J. Environments for multiagent systems state-of-the-art and research challegens. In: ENVIRONMENTS FOR MULTI-AGENT SYSTEMS INTERNATIONAL WORKSHOP, 1., 2004, New York. **Revised Selected Papers**. (s. l.): Spring-Verlag, Vol. 3374. 2005.

[14] ODELL, J.; PARUNAK, H. V. D.; FLEISCHER, M.; BREUCKNER, S. Modeling agents and their environment. In: GIUNCHIGLIA, F., ODELL, J., WEIS, G. (eds.). **Agent-Oriented Software Engineering III**. (s. l.): Springer-Verlag, 2002. (Lecture Notes in Computer Science, vol. 2585).

[15] SCHWAMBACH, M. M.; PEZZIN, J.; FALBO, R. A. OplA: uma metodologia para o desenvolvimento de sistemas baseados em agentes e objetos. In: Jornada Iberoamericanas de Ingeniería del Software e Ingeniería del Conocimiento, 4., 2004, Madrid. **Proceedings.**

[16] ERIKSSON, H.E.; PENKER, M.; LYONS, B.; FADO, D. UML 2 toolkit. Indianapolis: Wiley Publishing Inc, 2004. 511 p.

[17] ITO, M., SILVA, L. R., MARTINI, J. S. C., IOCHIDA, L. C. Um Sistema Multi-agente para a Monitoração de Pacientes Diabéticos com Base no Modelo GRPC In: Anais do XXIX Congresso da Sociedade Brasileira de Computação – IX Workshop de Informática Médica evento satélite do XXIX Congresso da Sociedade Brasileira de Computação, p.1935 – 1944, 2009, Bento Gonçalves, RS. **Proceedings.**

Contato

Márcia Ito
Laboratorio de Pesquisa em Ciencias de Servico (LaPCiS)
Unidade de Pos-Graduacao, Extensao e Pesquisa
Centro Estadual de Educacao Tecnologica Paula Souza
R. Bandeirantes, 169 - Bom Retiro - 01124-010
Sao Paulo - SP - Brasil
tel/fax: (11) 3327-3104

Leandro Ramos da Silva
Laboratorio de Pesquisa em Ciencias de Servico (LaPCiS)
Unidade de Pos-Graduacao, Extensao e Pesquisa
Centro Estadual de Educacao Tecnologica Paula Souza
R. Bandeirantes, 169 - Bom Retiro - 01124-010
Sao Paulo - SP - Brasil
tel/fax: (11) 3327-3104