
# PUC

ISSN 0103-9741

Monografias em Ciência da Computação
n° 32/02

# PRONEX MOBILE 2002

# Frameworks em Tecnologia de Software: Métodos, Ferramentas e Soluções de Domínio Específico

Editores:
Leonardo Magela Cunha
Carlos José Pereira de Lucena

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO
RUA MARQUÊS DE SÃO VICENTE, 225 - CEP 22453-900
RIO DE JANEIRO - BRASIL

PUC RIO - DEPARTAMENTO DE INFORMÁTICA ISSN 0103-9741

Monografias em Ciência da Computação, Nº 32/02

Editor: Carlos J. P. Lucena Dezembro, 2002

# PRONEX MOBILE 2002
# Frameworks em Tecnologia de Software:
# Métodos, Ferramentas e Soluções de Domínio Específico

Editores:

Leonardo Magela Cunha

Carlos José Pereira de Lucena

Trabalho patrocinado pelo Ministério de Ciência e Tecnologia da Presidência da República Federativa do Brasil.

# PRONEX MOBILE 2002

# Frameworks em Tecnologia de Software:

# Métodos, Ferramentas e Soluções de Domínio Específico

Editores:

Leonardo Magela Cunha

Carlos José Pereira de Lucena

{leocunha, lucena}@inf.puc-rio.br

Monografias em Ciência da Computação, Nº 32/02

Editor: Carlos J. P. Lucena Dezembro, 2002

**Abstract:**

The PRONEX 2002 workshop will take place in the Computer Science department at PUC-Rio in December, 17th. Masters students and PhD candidates will present the progress of their research projects. In this paper you will find some short articles about the work developed by the students in cooperation with their advisors and cooperating researchers. The volume includes work-in-progress and also abstracts of concluded theses.

**Keywords:**

Algorithms, Database Systems, Groupware, Multi-agent systems, Quality of Services, Semantic Web, Semiotic Engineering, Software Engineering

**Resumo:**

O seminário PRONEX 2002 acontecerá no dia 17 de dezembro de 2002 e contará com a participação de alunos de mestrado e doutorado do departamento de Informática da PUC-Rio. São apresentados nesta monografia pequenos artigos sobre o trabalho a ser apresentado no seminário e que é desenvolvido pelos respectivos alunos com seus professores orientadores e colaboradores. São apresentados trabalhos de pesquisa concluídos que geraram teses de doutorado e dissertações de mestrado, assim como também pesquisas em andamento.

**Palavras-chave:**

Algoritmos, Bancos de Dados, Engenharia de Software, Engenharia Semiótica Groupware, Qualidade de Serviços, Sistemas Multi-agentes, Web Semântica.

## SUMÁRIO

## ÍNDICE POR AUTORES

| Autor | Página |
|---|---|
| Marco Aurélio Gerosa | 10 |
| Mariano Gomes Pimentel | 14 |
| Mariela Inés Cortés | 34 |
| Matheus Costa Leite | 18 |
| Milene Selbach Silveira | 24 |
| Otavio Rezende da Silva | 49 |
| Raquel Oliveira Prates | 22, 47 |
| Raúl Pierre Rentería | 1 |
| Ricardo Choren | 53 |
| Rubens Nascimento Melo | 3, 5 |
| Ruy Luiz Milidiú | 1, 30 |
| Sergio Scheer | 5 |
| Viviane Torres da Silva | 26 |

# Algoritmos para regressão PLS

Ruy Luiz Milidiú, Raúl Pierre Rentería

Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{milidiu, renteria}@inf.puc-rio.br

## 1. Introdução

O algoritmo para regressão PLS [Wold 1966] tem sido amplamente usado na área de quimiometria para a análise de espectros NIR [Haaland and Thomas 1988]. A simplicidade aliada à robustez do modelo gerado fazem da metodologia PLS uma poderosa ferramenta para a análise fatorial, sendo usada em outras áreas como monitoração de processos, marketing e processamento de imagens [Morineau and Tenenhaus, 1999; Milidiú et al. 1999].

Neste trabalho são apresentadas três variantes do algoritmo para regressão PLS. A primeira é DPLS ou Direct PLS, uma verão não iterativa para o caso de mais de uma variável dependente conhecido também como PLS2. O segundo algoritmo é o PPLS ou Parallel PLS, uma versão paralela do algoritmo convencional para o caso de apenas uma variável dependente conhecido como PLS1. Por último é apresentado LPLS ou Lifted PLS, um algoritmo não linear para a regressão PLS.

## 2. DPLS

Na modelagem clássica PLS, para cada fator desejado o autovetor da matriz $\mathbf{X}^T\mathbf{Y}\mathbf{Y}^T\mathbf{X}$ deve ser calculado, sendo que $\mathbf{X}$ contém as variáveis independentes e $\mathbf{Y}$ as dependentes. Quando $\mathbf{Y}$ possui mais de uma variável (PLS2) é necessário o cálculo do autovetor o uso de um algoritmo iterativo geralmente baseado no método por potência, como por exemplo NIPALS [Wu et al., 1997]. DPLS [Milidiú and Rentería, 2001a,b] ou Direct PLS fornece um método direto não iterativo porém aproximado para o cálculo deste autovetor.

Nove conjuntos de dados foram usados para a avaliação do DPLS. Para cada um foi comparada a curva de PRESS (Predicted Residual Sum of Errors) entre o PLS e o DPLS resultando na escolha de um número de fatores favorável tanto à qualidade de predição quanto à síntese do modelo gerado.

Os resultados obtidos mostram a competitividade do algoritmo DPLS. Para todos os conjuntos, o PRESS mínimo obtido para ambos os métodos é praticamente o mesmo. Além disto o tempo de execução do DPLS foi 40% menor quando comparado com o PLS convencional.

## 3. PPLS

A segunda variante apresentada é PPLS [Milidiú and Rentería, 2000,2001b], uma versão paralela do algoritmo PLS convencional restrita a uma variável dependente (PLS1). Com relação a outras propostas que calculam os fafores PLS via aprendizado Hebbiano (Ham and Kostanic, 1998), PPLS é exato não dependendo de nenhuma aproximação ou critério de convergência.

Para avaliar o PPLS foram usados os primeiros 5340 pontos da série D fornecida durante a competição de Santa Fé já que a preocupação é avaliar o tempo de execução e não a qualidade do modelo gerado. Para a análise de desempenho duas métricas foram usadas: *Speedup* e *Efficiency*. A primeira indica o ganho obtido com cada nó acrescentado e a segunda fornece uma idéia de quanto cada nó esta sendo aproveitado.

Os testes foram executados num cluster de 22 computadores IBM Pentium 2 400MHz com 32MB de RAM cada, ligados por um *switch* de 10MB/s. A biblioteca mpich v1.2.0 foi usada para a codificação do algoritmo. Neste cenário foi observado para o algoritmo um Speedup maior do que 3 para 4 nós. Além disto obtivemos para este caso um *Efficiency* acima de 74% mostrando que a abordagem paralela proposta é eficiente.

## 4. LPLS

LPLS ou Lifted PLS fornece uma versão não linear do algoritmo convencional PLS. A não linearidade é realizada mapeando de forma eficiente as amostras (matriz $\mathbf{X}$) definidas originalmente em $\mathbf{R}^m$ num novo espaço $\mathbf{F}^z$ de alta dimensão ($z >> m$). Um mapeamento possível consiste na geração de novas variáveis correspondentes a todos o monómios de grau $g$ das variáveis originais. Este recurso permite aumentar o poder computacional de algoritmos lineares como o PLS.

Usamos o LPLS em algumas das series empregadas nos experimentos do DPLS. A figura 1 mostra a comparação da curva PRESS entre o PLS e o LPLS para um conjunto de dados

referente à análise de humidade baseada em espectros NIR de amostras de carne. Como podemos observar, não somente o PRESS obtido foi menor mas o número de fatores também.

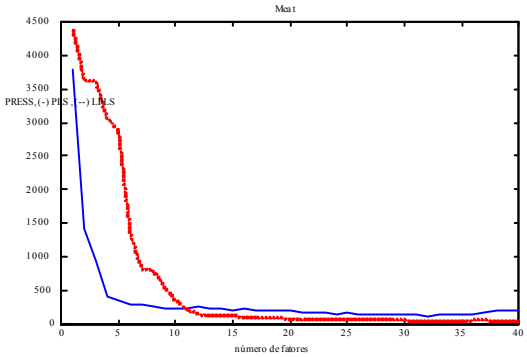



Figura 1 – PRESS para LPLS e PLS

## 5. Conclusão

Foram apresentados 3 variantes do algoritmo para regressão PLS convencional, dentre as principais contribuições podemos citar:

1. eficiência com a aproximação DPLS;
2. aumento do poder preditivo com o LPLS;
3. formulação paralela com o PPLS de fácil implementação e aplicável aos outros algoritmos.

## 4. Referências

Wold. H. 1966: Estimation of principal components and related models by iterative least squares. In: Krishnaiah P.R. (ed) Multivariate analysis II, 391-420.

Haaland D.M., Thomas E.V. 1988: Partial least-squares methods for spectral analysis. 1. relation to other quantitative calibration methods and the extraction of qualitative information. Analytical Chemistry, 60, 11, 1193-1202.

Morineau A., Tenenhaus M. eds. 1999: Les Méthodes PLS, Symposium International PLS'99. Cisia-Ceresta.

Milidiú R.L., Machado R.J., Rentería R.P. 1999: Time series forecasting throught Wavelets and a mixture of experts models. Neurocomputing, 28, 145-156.

Wu W., Massart D.L.,De Jong S. 1997: The kernel PCA algorithms for wide data. part I: theory and algorithms. Chemometrics and Intelligent Laboratory Systems, 36, 165-172.

Milidiú R.L., Rentería. R.P. 2001a: APLS: A fast approximate algorithm for partial least-squares. In V Brazilian Conference on Neural Networks, 661-666, Rio de Janeiro, Brazil.

Milidiú R.L., Rentería. R.P. 2001b: DPLS and PPLS: Two PLS algorithms for large data sets. In 2nd International Symposium on PLS and Related Methods. Capri, Italy.

Milidiú R.L., Rentería R.P. 2000: PPLS: An efficent parallel algorithm for partial least-squares. In 12th Symposium on Computer Architecture and High Performance Computing, São Pedro, SP, Brazil.

Ham F.M., Kostanic I. 1998: A neural network architecture for partial least-squares regression (plsnet) with supervised adaptive modular hebbian learning. Neural, Parallel & Scientific Computations, 6,35-72.

# An Object Model for Collaborative CAD Environments

C.A.M. Barbosa[1], Bruno Feijó[2], Marcelo Dreux[3], Rubens Melo[2], João Bento[4], Sergio Scheer[5]

[1]Institute of Computing (UFG) - barbosa@inf.ufg.br

[2]Dept. of Computing (PUC-Rio) - {bruno, rubens}@inf.puc-rio.br

[3]Dept. of Mechanical Engineering (PUC-Rio) - dreux@mec.puc-rio.br

[4]Technical University of Lisbon - joao@civil.ist.utl.pt

[5]CESEC/TC (UFPR) - scheer@cce.ufpr.br

## 1. Description of the work

A major problem that integrated CAD systems still face, with state-of-the-art technology, is that of mobilising an adequate level of functional integration between the various components to involve [Anumba, 1996], namely at the level of the sharing of data and objects through professionals' teams developing different activities.

Indeed, an important contribution to the integration issue may proceed from the role of systems like database, workflow, solid modelling and others which provided they present a level of adequacy for the purpose at hand. Data models for CAD deal with a more complex structure, due to a number of reasons: the deep hierarchical structure, the multi-representational data aggregates, the correlation across data representations, and the connections across time. Moreover the characteristics of the design process are quite unique, namely the iterative and exploratory nature of design activities and the need for active historical data.

In spite of the importance of CAD software (which is a multibillion dollar industry), research on CAD data models started only in the early 80's and is still not well-understood by the majority of the database community. Research on this area firstly attempted extensions to the relational model [Haskin and Lorie, 1982]; at later stages, object-oriented models started to appear as more promising approach [Katz, 1990]. However, most of the literature still focuses on the issues of version modelling and propagation change. Few contributions present models for collaborative CAD environments. On the one hand, database authors usually investigate object-sharing mechanisms or general aspects of heterogeneous systems, without considering the functional characteristics of the engineering design process and the complex requirements of handling engineering data. On the other hand, CAD researchers mostly concentrate on PDM (Product Data Management) systems and AI (Artificial Intelligence) aspects of collaborative design. The present work presents a pragmatic approach to distributed object modelling for collaborative CAD environments from the perspective of the engineering designer and with a special emphasis on the role of geometry buses for implementation purposes. Version control and consistency are not treated in the present proposal, although the proposed framework encompasses its eventual future integration.

From a designer's viewpoint the kernel of the questions in collaborative CAD systems is the ability to manipulate a so-called virtual prototype. The aim of virtual prototyping is to build a full virtual artefact in such a way that design and manufacturing problems are anticipated and discussed within a cooperative and distributed work environment, which embraces all the departments of a company. Virtual prototyping is much more than building a complete 3D model of an artefact. A virtual prototype works as a spatial and temporal object that can be built and queried by anyone in the enterprise on a heterogeneous computer network. The objects of a virtual prototype have several types of attributes, such as geometric values, design intents, manufacturing specifications, cost data, part numbers and references to documentation. Furthermore, these objects should be defined in the context of workflow management [Georgakopoulos et al., 1995], organizational engineering [Kerzner, 1997] and tractability of requirements [Ramesh et al., 1995].

Figure 1 illustrates a distributed CAD environment, where the virtual prototype is distributed over different networks with different platforms, operating systems, design teams and non-CAD users. The design process requires collaborative work amongst several teams. After the dynamic stage of the design process is completed and as soon as manufacturing drawings are approved, components can be stored in a central database containing the Product Structure file and the STEP file. Essentially, the Product Structure is a collection of pointers to 3D models and 2D drawings. During the design process, data are manipulated over different database systems and networks in a highly dynamic way that goes beyond simple web browsing. A model for

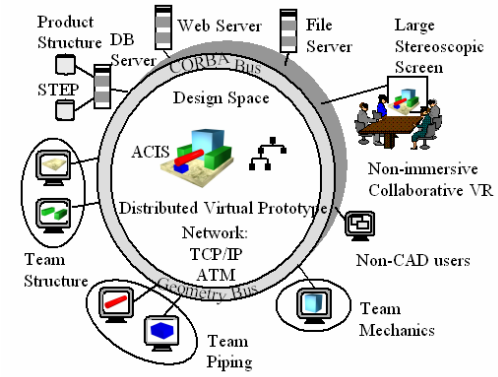



Figure 1 - Distributed CAD environment

distributed CAD objects should allow teams to work cooperatively, accessing and changing information in distributed engineering data systems at run time.

The present work proposes an object model for virtual prototyping based on geometry bus and object-distributed computing with the focus on the design history. Firstly, the object model and the concept of design history are presented. Secondly, a distributed system is proposed as a component architecture, which can be easily customised to work with different middleware for distributed objects (e.g. CORBA (Common Object Request Broker Architecture) or COM (Component Object Model)) and different programming languages at the interface level. This distributed system considers data stored in different data systems, which are distributed on different networks. Thirdly, an integration model for CAD environments is proposed based on the paradigms of workflow management and organisational engineering. Fourthly, the concept of geometry bus is presented as the cornerstone for interoperability and the basic foundation for collaborative work with geometric data. Finally, conclusions and recommendations for further work are presented.

## 2. References

Anumba, C.J.; 1996; Functional Integration in CAD Systems; *Advances in Engineering Software, 25(2/3), pp. 103-109*; UK;

Haskin, R.L., Lorie, R.A.; 1982; On extending the functions of a relational database system; *Proceedings of the ACM SIGMOD Conference, pp. 207-212*; USA;

Georgakopoulos, D., Hornick, M., Sheth, A.; 1995; An overview of workflow management: from process modeling to workflow automation infrastructure; *Distributed and Parallel Databases, 3(2), Kluwer Academic Publ., pp, 119-153*; The Netherlands;

Katz, R.H.; 1990; Toward a unified framework for version modeling in engineering databases; *ACM Computing Surveys, 22(4), pp. 375-408*; USA;

Kerzner, H.; 1997; Project Management: a Systems Approach to Planning, Scheduling, and Controlling, 6th edn.; *John Wiley & Sons*; USA;

Ramesh, B., Powers, T., Stubbs, C.; 1995; Implementing requirements traceability: a case study; *Proceedings of 2nd IEEE Int. Symposium on Requirements Engineering, IEEE Computer Society Press*, pp. 89-95; USA;

# AQEE: uma Máquina de Execução de Consultas Adaptativa

Fausto Véras Maranhão Ayres[1], Fábio André Porto[1], Rubens Nascimento Melo[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{fausto, porto, rubens}@inf.puc-rio.br

## 1. Introdução

O acesso integrado à informações publicadas na *web* permite que sejam oferecidos novos serviços combinando *sites*, a princípio projetados para funcionamento independente. Suponha, por exemplo, três *sites web* de supermercados disponibilizando os preços de seus produtos. Um usuário de uma aplicação de integração destes *sites* poderia submeter uma consulta como esta: *"A partir de uma lista inicial de produtos, informar os seus preços em cada um dos três sites de supermercados"*. Aplicações como essa envolvem a execução distribuída da consulta com acesso integrado aos três *sites*. A consulta formulada a partir do modelo de integração deve ser otimizada e executada pelo sistema integrador que oferece uma visão integrada das fontes de dados. O plano de execução de consultas (PEC) assim produzido, incluirá operações de *BindJoin* [Florescu 1999] que acessam os formulários *web*, modelados aqui como relações virtuais com restrições de acesso permitindo seu tratamento pelo processo tradicional de otimização de consultas baseado em programação dinâmica.

Este processo tradicional gera PECs inadequados para este domínio, devido ao fato de que as fontes web (1) estão sujeitas a grandes variações no tempo de resposta às consultas e (2) sofrem constantes modificações tanto em seu conteúdo quanto em sua forma, tornando tornam extremamente difícil obter informações estatísticas sobre os seus dados. Tal condição particular levou os pesquisadores a buscarem alternativas com variado grau de dinamismo na geração e execução de PECs sobre *sites web*, tais como: *Query Scrambling* [Amsaleg 1996], *Tukwila* [Ives 1999], *Eddies* [Avnur 2000], *Hybrid* [Bouganim 2001].

Neste contexto, desenvolvemos a máquina de execução de consultas *AQEE* (Adaptive Query Execution Engine) baseada no modelo de execução adaptativo. A máquina *AQEE* foi construída a partir do *framework QEEF (Query Execution Engine Framework)* [Ayres 2002] especificado para a construção de máquinas de execução de consulta adaptáveis à diferentes modelos de execução.

O restante deste artigo encontra-se dividido da seguinte forma. A Seção 2 discute o modelo de execução adaptativo. A seção 3 especifica o *framework QEEF*. A Seção 4 apresenta a máquina *AQEE* e um estudo de caso. Finalmente, a Seção 5 apresenta as conclusões finais.

## 2. Estratégia de Execução Adaptativa

Consideramos uma fonte de dados local (L1) contendo a lista inicial de produtos e três *sites* de supermercados (S1, S2 e S3) com informações publicadas na *web*. O esquema relacional exportado pelos seus respectivos *wrappers* são: L1(cod, nome), S1(cod, preco), S2(cod, preco) e S3(cod, preco). Uma aplicação de integração que objetiva recuperar os preços de produtos dos três *sites* seria expressa pela consulta *SQL-like* abaixo:

```
Select  L1.nome, S1.preco, S2.preco, S3.preco
From    L1, S1, S2, S3
Where   L1.cod = S1.cod and L1.cod = S2.cod
        and L1.cod = S3.cod
```

Exemplo 1 – Consulta de integração

O predicado $p$=(*L1.cod=Si.cod)* exemplifica a necessidade do fornecimento de um critério de busca para acesso aos dados publicados pelo *site Si*. Neste caso, cada *site* requer a associação de um código de produto para que se tenha acesso ao preço deste produto. Sendo assim, a fonte L1 deve ser a primeira a ser acessada para obtenção da lista inicial de produtos. Define-se então, uma ordem parcial entre os operadores que comporão o plano de execução da consulta (PEC) do Exemplo 1. Tem-se, para árvores profundas à esquerda, as seguintes possíveis ordens de avaliação dos predicados: $O_1$= *L1→ S1→ S2→ S3,* $O_2$= *L1→ S1→ S3→ S2,* $O_3$= *L1→ S2→ S1→ S3,* $O_4$= *L1→ S2→ S3→ S1 ,* $O_5$= *L1→ S3→ S1→ S2* e $O_6$= *L1→ S3→ S2→ S1*. Outros planos possíveis incluiriam versões para planos *bushy*.

A determinação do melhor plano a ser executado neste cenário é uma tarefa difícil considerando-se as grandes variações no tempo de resposta aos *sites* envolvidos. Estratégias puramente estáticas não são capazes de modelar tais variações. Já estratégias adaptativas podem combinar os diversos planos possíveis durante a execução de uma consulta recorrendo aos operadores que

estejam mais disponíveis a cada instante. Para o Exemplo 1, uma máquina de execução adaptativa executaria o plano equivalente à $O_1 \cup O_2 \cup O_3 \cup O_4 \cup O_5 \cup O_6$.

## 3. O Framework QEEF

*QEEF - Query Execution Engine Framework* [Ayres 2002] foi projetado para construir máquinas de execução de consultas para diferentes modelos de execução. Definimos um modelo de execução de consulta como uma combinação das seguintes características de execução associadas ao processamento dos operadores de um PEC: Sincronismo, Paralelismo, Distribuição, Fluxo de Dados, Fluxo de Controle, Tempo de Resposta e Materialização

Considerando-se que o modelo de execução é definido a partir da seleção das características dentre as acima descritas, o *QEEF* tem como objetivo a instanciação de uma máquina de execução de consulta que suporte esse modelo. A Figura 1 apresenta o diagrama de classes do *framework QEEF*.

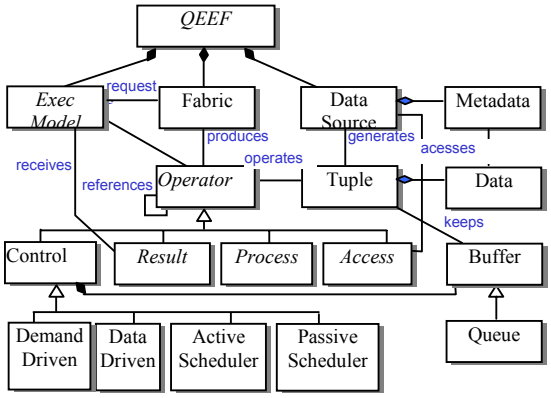



Figura 1 – *Framework QEEF*

A classe *QEEF* é um padrão *facade*[Gamma 1995] responsável pela interface da máquina de execução com a aplicação usuária que submete um Plano de Execução de Consulta (PEC) na forma de uma árvore de consulta profunda à esquerda onde os nós folhas correspondem às fontes de dados e os nós internos correspondem aos operadores algébricos. A classe *Fabric* cria os operadores algébricos e de controle utilizados no PEC. A classe *Operator* implementa, no estilo *pipeline,* uma interface comum a todos os operadores exportando os métodos *open*, *get*, *put* e *close.* Os operadores processam instâncias da classe *Tuple* que representa uma unidade de processamento. As subclasses *Access*, *Process* e *Result* são pontos de flexibilização (*hot spot*) que implementam um dado conjunto de operadores algébricos de acesso, processo e resultado, respectivamente. A classe *Control* representa os operadores de controle que são responsáveis pelo desacoplamento entre os operadores algébricos. A classe *ExecModel* é o principal ponto de flexibilização (*hot spot*) do *QEEF* e precisa ser especializada, via padrão *strategy*[GHJV95], para implementar diferentes modelos de execução. A classe especializada deverá selecionar corretamente os operadores para construir um PEC de acordo com as características presentes no modelo de execução.

## 4. A máquina *AQEE*

A partir da instanciação do *framework QEEF* produzimos a máquina de execução adaptativa *AQEE* (*Adaptive Query Execution Engine*) para execução de consultas tais como a do cenário apresentado no exemplo 1 da Seção 2 que foi usado como estudo de caso. A instanciação dos *hot spots* do *framework* é feita da seguinte forma: a classe *ExecModel* é especializada na classe *AdaptiveModel e* as classes *Access*, *Process* e *Result* são especializadas nas classes *Scan*, *BindJoin* e *Project* respectivamente, correspondendo aos seus operadores algébricos homônimos. As classes de controle *Demand*, *Active* e *Passive* são instanciadas nas classes *Reader*, *Collector* e *Distributer*, respectivamente, as quais incorporam a característica de assincronismo.

## 5. Conclusões

Neste trabalho, apresentamos a máquina de execução de consultas *AQEE* contruída como parte do processo de validação do *framework QEEF*. *AQEE* incorpora o modelo de execução adaptativo. Mostramos através de um estudo de caso, baseado numa consulta de integração de dados a *sites web*, que a adaptatividade está associada a imprevi-sibilidade do tempo de acesso a estes *sites*. Aplicações como essa apresentam desafios na medida em que o tempo de acesso às fontes de dados registram grandes variações e as informações estatísticas são escassas.

As classes do *framework QEEF* e da máquina *AQEE* foram implementadas na plataforma *Java*. A a interface do usuário para o exemplo foi implementada através de *Servlets*,

## 6. Bibliografia

[Amsaleg 1996] "Scrambing Query Plans to Cope with Unexpected Delays", em Proc. PDIS, Miami, USA, 1996.

[Avnur 2000] "Eddies: Continuously Adaptive Query Processing", Proc. SIGMOD, Dallas, Texas, 2000.

[Ayres 2002] "Um Framework para construção de Máquinas de Execução de Consultas Relacionais"., Proc CACIC, Buenos Aires, Argentina, outubro 2002.

[Bouganim 2001] "Processing Queries with Expensive Functions and Large Objects in Distributed Mediator Systems", Proc. ICDE, Heidelberg, Alemanha, Abril 2001.

[Florescu 1999] “Query optimization in the Presence of Limited Access Patterns”, em Proc. SIGMOD, Filadelfia, USA, Junho, 1999.

[Gamma 1995] “Design Patterns”, em Addison-Wesley pub., 1995.

[Graef 1993] “Query Evaluation Techniques for Large Databases”, em ACM Computing Surveys, 25(2), Junho, 1993.

[Ives 1999] “An Adaptive Query Execution System for Data Integration”, em Proc. SIGMOD, Filadelfia, USA, Junho, 1999.

# Development Process Configuration Framework

Leandro Ribeiro Daflon[1], Carlos José Pereira de Lucena[1],Arndt von Staa[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{daflon, lucena, arndt}@inf.puc-rio.br

## 1. Introdução

Atualmente existem diversos modelos de processo de desenvolvimento de software, sendo que cada um deles apresenta vantagens e desvantagens que estão diretamente relacionados com a natureza do projeto a ser desenvolvido. Cada projeto apresenta características intrínsecas, como por exemplo, equipes com pessoas de diferentes *skills*, tempo para o desenvolvimento, recursos financeiros aplicados e o grau de importância para a organização. Cada uma dessas variáveis contribuem diretamente no processo de desenvolvimento do software a ser adotado.

Existem diversos modelos de processos provendo diferentes estratégias de desenvolvimento. Drente os modelos existentes, podemos citar: The Objectory Process [Jacobson, 1992], o Unified Software Development Process [Jacobson, Booch, Rumbaugh, 1999], o Catalysis [D'Souza, 1998], V-Model 97 [Droschel, 1999], o eXtrem Programming [Beck, 1999], o Rational Unified Process (RUP) [Kruchten, 2000], Crystal [Cockburn, 2000] e outros. Entre os processos mencionados acima, existem processos mais formais, nos quais são registrados e documentados detalhadamente cada fase e iteração, e outros processos menos formais, nos quais a documentação é representada basicamente pelo código fonte e pelos casos de testes.

Para elaborar um processo de desenvolvimento a partir de modelos de processos existentes é necessário identificar e caracterizar explicitamente seus componentes (blocos de construção) e os relacionamentos entre os seus diversos componentes [Gnatz, Marschall, Popp, et al]. Dessa forma, será possível configurar um processo de desenvolvimento personalizado que atenda as necessidades específicas de um projeto, combinando aspectos de diversos modelos de processo e utilizando o grau de formalismo adequado.

Este trabalho tem como objetivo, estudar uma maneira de compor diversos componentes de processo de desenvolvimento e padrões, visando a sua reutilização em projetos futuros.

## 2. Motivação

Diversas organizações apresentam dificuldades para produzir software de qualidade no tempo e no orçamento estimado e muitas destas organizações tem mostrado grande preocupação em buscar melhorarias para o seu processo de desenvolvimento, buscando cada vez mais atingir a maturidade de seus processos [SEI, 2000]. Além disso, a indústria do software compete em um mercado altamente dinâmico, no qual, requisitos dos clientes mudam inevitavelmente, novas tecnologias são adotadas durante o projeto, interações entre desenvolvedores e clientes demandam alterações na estratégia do processo de desenvolvimento adotado.

Um outro problema existente no mercado é a rotatividade de pessoas nas equipes de desenvolvimento em um mesmo projeto. Neste caso, informações importantes, muitas vezes são perdidas com a saída de um integrante da equipe, demandando um grande esforço para obter informações que já haviam sido levantadas anteriormente. Dessa forma, é importante armazenar informações em uma base de conhecimento, transformando informações sobre projetos em conhecimento da organização.

É importante também poder avaliar o processo de desenvolvimento definido, identificando as suas falhas, ou seja, os componentes que estejam comprometendo o processo como um todo, possibilitando também identificar pontos críticos e componentes passíveis de aprimoramento para serem aplicados em outros projetos.

## 3. Proposta

Esta dissertação propõe desenvolver um *framework* que permite configurar processos de desenvolvimento de software, integrando os diversos modelos de processos existentes. A integração é baseada na identificação de componentes, padrões de processo e seus relacionamentos.

A configuração do processo de desenvolvimento de um projeto específico se daria na identificação dos componentes e padrões que atendam as características intrínsecas do projeto, como por exemplo, número de pessoas na equipe, o *skill* das

pessoas envolvidas, tempo para o desenvolvimento do projeto, recursos financeiros empregados aplicados, e o grau de importância do projeto para o negócio da organização. Dessa forma, será possível construir um processo de desenvolvimento com o grau de formalismo adequado as varáveis mencionadas acima.

Serão estudadas também, formas de avaliação do processo definido, com o objetivo de distinguir partes do processo (componentes) que falharam no projeto corrente e os componentes que poderão ser melhorados para serem aplicados em um próximo projeto, evitando repetições de erros ocorridos na instância anterior. Dessa forma, é importante armazenar as informações do projeto em uma base de conhecimento, como por exemplo, o processo instanciado, as decisões de projeto, as métricas adotadas, as informações coletadas, os artefatos produzidos e as críticas dos interessados no projeto.

Para configurar um novo processo de desenvolvimento é possível reutilizar partes de um processo definido para um projeto anterior ou o processo como um todo, realizando pequenas alterações – adaptação estática. Será possível também após configurar o processo deixar pontos flexíveis que possibilitem realizar modificações dinamicamente – adaptação dinâmica.

O objetivo é tornar a tarefa de elaboração do processo mais fácil e pragmática, facilitando o trabalho do engenheiro de processo.

## 4. Organização do Trabalho

O estudo será organizado em 4 etapas. Na primeira etapa serão estudados os conceitos de componentes de processo, padrões de processo e como organizá-los e estruturá-los de forma a facilitar a sua integração. Na etapa seguinte, serão estudados formas de avaliação do processo elaborado (métricas). Na terceira etapa será desenvolvido o framework para a configuração do processo e o plugin para o Eclipse. Na quarta e última etapa serão desenvolvidos estudos de caso para validar o framework e os conceitos proposto na dissertação.

## 5. Referências

[Beck, 1999] Beck K. Extreme Programming Explained: Embrace Change. Addison-Wesley, 1999.

[Cockburn, 2000] Alistair Cockburn. Agile Software Development, 2000

[D'Souza, 1998] Desmond Francis D'Souza, Alan Cameron Wills. Objects, Components, and Frameworks with Uml: The Catalysis Approach. Addison Wesley Publishing Company. 1998.

[Dröschel , Wiemers, 1999] Wolfgang Dröschel, Manuela Wiemers. Das V-Modell 97. Oldenbourg. 1999.

[Eclipse, 2001] http://www.eclipse.org, julho 2001

[Gnatz, Marschall, Popp, et al] Towards a Tool Support for a Living Software Development Process. http://citeseer.nj.nec.com/484688.html

[Jacobson, 1992] Ivar Jacobson. Object-Oriented Software Engineering: A Use Case Driven Approach. Addison-Wesley Publishing Company, 1992.

[Jacobson, Booch, Rumbaugh, 1999] Ivar Jacobson, Grady Booch, James Rumbaugh. Unified Software Development Process. Addison Wesley Publishing Company. 1999.

[Kruchten, 2000] Philippe Kruchten. The Rational Unified Process, An Introduction, Second Edition. Addison-Wesley Longman Inc. 2000.

[SEI, 2000] SEI, Process Maturity Profile of the Software Community 1999 Year End Update, SEMA 3.00, Carnegie Mellon University, March 2000

# Engenharia de Groupware: Desenvolvimento de Aplicações Colaborativas

Marco Aurélio Gerosa[1], Alberto Barbosa Raposo[1], Hugo Fuks[1], Carlos José Pereira de Lucena[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{hugo, abraposo, gerosa}@inf.puc-rio.br

## 1. Informações Gerais

A tecnologia gera ambientes que dão suporte às diferentes formas de relacionamento humano e, por conseguinte, revoluciona o modo de se trabalhar na sociedade conectada. A criação de espaços de compartilhamento e troca de informação apoiado por groupware propicia o trabalho colaborativo distribuído e descentralizado.

A Engenharia de Software, que muito avançou no desenvolvimento de aplicações mono-usuário e recentemente começou a considerar o fator humano [DeMarco et al, 1999], falha ao lidar com aspectos de grupo tão necessários em aplicações colaborativas [Grudin, 1994]. O objetivo desta pesquisa é formular uma Engenharia de Groupware, baseada em Engenharia de Software, aprimorada com conceitos originados da área de CSCW e IHC.

Para contextualizar o ciclo de desenvolvimento de groupware, as fases de desenvolvimento de software [Pressman, 1992] são apresentadas na Figura 1, juntamente com os tópicos que estão sendo estudados nesta pesquisa.

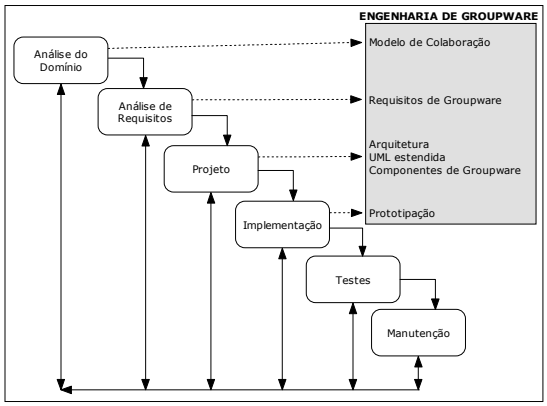



Figura 1. Ciclo de desenvolvimento de software indicando as fases e aspectos abordados nesta pesquisa.

Requisitos de Groupware são levantados na fase de análise de requisitos, onde a atenção está centrada no software. Para instrumentar a fase de projeto, onde o software é concebido numa maneira que satisfaça os requisitos, *toolkits* [Roseman & Greenberg, 1996] e os conceitos de componentes de groupware, arquitetura de componentes e extensões à linguagem UML são necessárias, [Tietze, 2001]. Na fase de implementação, a escolha por prototipação rápida [Schrage, 1996] é útil em aplicações colaborativas, visto que são especialmente susceptíveis a falhas [Grudin, 1994] e demanda avaliação colaborativa durante o desenvolvimento.

Será discutido agora em mais detalhes o modelo de colaboração 3C que embasa a fase de análise do domínio.

## 2. O Modelo de Colaboração 3C

Com o advento da sociedade conectada a maneira de trabalhar mudou. Acostumado ao paradigma de comando e controle que é ensinado, ou melhor, condicionado na sala de aula e largamente difundido no mundo fabril, este trabalhador não se sente habilitado às novas demandas da sociedade da informação. Ele foi preparado para reagir a ordens claras, procedimentos bem definidos e atividades estanques de preferência individuais. Seu entendimento de comunicação é vertical (memorandos que descem e relatórios que sobem) e assim como na sala de aula, a comunicação horizontal, i.e., com o seu colega, além de não ser bem vista, não recebe nenhum suporte tecnológico.

Trabalhadores do conhecimento, por outro lado, conseguem trabalhar em grupo e aprender continuamente novos processos e técnicas para estarem capacitados à realização das suas tarefas. Eles constantemente interagem com os seus colegas de trabalho na busca de informações relevantes à realização das tarefas impostas pela sociedade conectada. Devido à complexidade e interdisciplinaridade destas tarefas, grupos se formam para resolverem os problemas que surgem no dia-a-dia. A organização que era imposta de cima para baixo no paradigma de comando e controle perde eficácia e é substituída por outra menos hierarquizada e mais participativa, onde

predominam a comunicação, a coordenação e a cooperação.

Colaborando, pelo menos potencialmente, pode-se produzir melhores resultados do que individualmente. Em um grupo ocorre a complementação de capacidades, de conhecimentos e de esforços individuais. Colaborando, os membros do grupo têm retorno que permite identificar precocemente inconsistências e falhas em seu raciocínio e, juntos, podem buscar idéias, informações e referências para auxiliar na resolução dos problemas. O grupo também tem mais capacidade de gerar criativamente alternativas, levantando as vantagens e desvantagens de cada uma delas, para selecionar as viáveis e tomar decisões [Turoff and Hiltz, 1982].

Para possibilitar a colaboração, são necessárias informações sobre o que está acontecendo. Estas informações são fornecidas através de elementos de percepção que capturam e condensam as informações coletadas durante a interação entre os participantes. Perceber, neste contexto, é adquirir informação, por meio dos sentidos, do que está acontecendo e do que as outras pessoas estão fazendo. A percepção em si é relativa ao ser humano, enquanto os elementos de percepção estão relacionados ao ambiente.

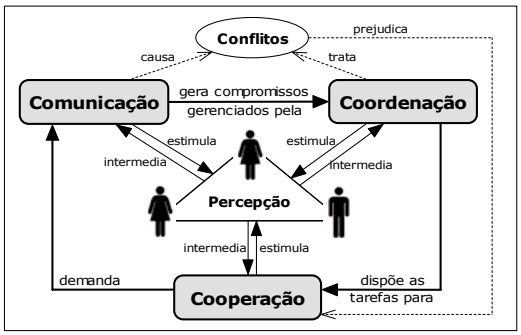



Figura 2. O modelo 3C

Apesar de suas vantagens, colaborar demanda um esforço adicional de coordenação dos seus membros. Sem esta coordenação, boa parte dos esforços da comunicação não será aproveitada na cooperação, i.e., para que o grupo possa operar em conjunto de forma satisfatória, é necessário que os compromissos assumidos nas interações entre os participantes sejam realizados ao trabalharem em conjunto no espaço que compartilham. A coordenação também trata conflitos interpessoais que prejudicam a cooperação.

O diagrama da Figura 2 sumariza os principais conceitos abordados. Este diagrama é um refinamento do modelo 3C apresentado originalmente em [Ellis et al., 1991] e difundido na literatura, como por exemplo em [Borghoff and Schlichter, 2000]. Vale lembrar que apesar da separação destes conceitos para efeito de análise, nem sempre é possível considerá-los monoliticamente, uma vez que são intimamente dependentes e inter-relacionados.

## 3. Referências Bibliográficas

BORGHOFF, U.M., AND SCHLICHTER, J.H. 2000. Computer-Supported Cooperative Work: Introduction to Distributed Applications. Springer, USA.

DeMARCO, T., AND LISTER, T., 1999. Peopleware: Productive Projects and Teams. Dorset House Publishing, USA.

ELLIS, C.A., GIBBS, S.J., AND REIN, G.L. 1991. Groupware - Some Issues and Experiences. Communications of the ACM 34, (1), 38-58.

GRUDIN, J. 1994. Groupware and Social Dynamics: Eight Challenges for Developers. Communications of the ACM 37, (1), 92-105.

PRESSMAN, R. 1992. Software Engineering: A Practitioner's Approach. 3rd ed. McGraw-Hill, USA.

ROSEMAN, M., AND GREENBERG, S. 1996. Building Real-Time Groupware with GroupKit, A Groupware Toolkit. ACM Transactions on Computer-Human Interaction 3, (1), 66-106.

SCHRAGE, M. 1996. Cultures of Prototyping. In: Bringing Design to Software, T. WINOGRAD, ed., ACM Press, USA, 191-205.

TIETZE, D.A. 2001. A Framework for Developing Component-based Co-operative Applications. Ph. D. Dissertation, Computer Science, Technischen Universität Darmstadt, Germany.

TUROFF, M., AND HILTZ, S.R. 1982. Computer Support for Group versus Individual Decisions. IEEE Transactions on Communications 30, (1), 82-91.

# Engenharia de Ontologias no Contexto da Web Semântica

Anarosa Alves Franco Brandão[1], Carlos José Pereira de Lucena[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{anarosa, lucena}@inf.puc-rio.br

## 1. Resumo

A pesquisa em web semântica tem evoluído nos últimos anos, e é muito dependente de ontologias. Ao desenvolvermos aplicações para a web semântica, precisamos dedicar especial atenção às ontologias que iremos utilizar. Algumas linguagens de descrição de ontologias não são adequadas para alimentar este tipo de aplicação e as ferramentas de desenvolvimento, em geral, não são interoperáveis, o que pode impedir a importação de ontologias de uma ferramenta para a outra. Por este motivo, muitos desenvolvedores de aplicações estão optando por desenvolver suas próprias ontologias. Neste trabalho damos uma visão geral das definições para ontologias e web semântica, descrevemos algumas metodologias de desenvolvimento de ontologias e relatamos um estudo de caso sobre o desenvolvimento de uma ontologia para criar conteúdo que vai alimentar uma aplicação para a web semântica, aplicação esta que deve gerar relatórios acadêmicos a partir do conteúdo semântico de um web site.

## 2. Introdução

A popularização do acesso à informação, através da world wide web (www) gerou para seus usuários o problema de excesso de informação, acompanhado da desorientação e consequente falta de informação. As primeiras soluções para o problema de gerenciamento do excesso de informação foram dadas pelas ferramentas de busca (altavista, yahoo, google, etc), baseadas em filtragem da informação através de meta-informações, como palavras-chave, além de heurísticas para categorização. Estas soluções já se mostram ineficientes, dada a quantidade de informação indesejada que o usuário comum recebe ao fazer uma simples busca.

Uma solução para o problema foi proposta por Tim Berners-Lee, o criador da www, ainda no seu documento seminal [Berners-Lee, 1990]. Neste documento ele propunha a utilização da idéia de hipertexto semântico, onde os hiperlinks também poderiam expressar relacionamentos entre documentos. Porém, a linguagem HTML não provia suporte para este tipo de construção. O surgimento de XML (eXtensible Markup Language) mostrou um caminho a ser seguido a fim de tornar possível a materialização da proposta de Berners-Lee de uma "web semântica". A idéia básica de XML, o uso de tags específicas para descrever tipos de dados contidos num documento, registradas em DTDs, serviu de inspiração para a primeira prova de conceito de uma Web Semântica, feita pelo grupo liderado pelo professor Jim Hendler, usando a linguagem de marcação SHOE (Simple HTML Ontology Extensions).

A chave do sucesso desta prova de conceito foi o uso de ontologias, o que possibilitou não só a definição de conceitos num domínio de conhecimento de forma a evitar ambiguidades, como também permitiu a definição de relacionamentos entre estes conceitos e inferências básicas envolvendo-os.

A possibilidade de representação de conteúdo web de forma inteligível e processável por agentes de software abriu caminho para o surgimento de novas tecnologias, como RDF , RDFS, DAML+OIL , e mais recentemente, OWL . Estas tecnologias podem orquestrar uma mudança de paradigma na web, qual seja, da recuperação de informação para o gerenciamento e aquisição do conhecimento nela disponibilizado.

Neste contexto, onde sistemas de software baseados na web são cada vez mais complexos e a web semântica já se mostrou possível, incorporar possibilidades de uso racional da web semântica às aplicações web é uma tendência natural. Assim, direcionarmos esforços na pesquisa relacionada à engenharia de ontologias no contexto da web semântica nos pareceu uma tarefa de grande interesse e motivou este trabalho.

## 3. Ontologias e Web Semântica

O termo Ontologia tem origem na Filosofia, com Aristóteles e está relacionado ao estudo da existência.

Em [Gruber, 1993] temos a definição mais citada na literatura: *uma ontologia é uma especificação explícita de uma conceitualização*, mais tarde re-interpretada por [Borst, 1997]: *ontologia é uma especificação formal de uma conceitualização compartilhada*.

A partir da definição acima podemos deduzir que ontologias são cruciais para sistemas de software

que têm por finalidade a busca ou a combinação e/ou integração de informações provenientes de diversas comunidades. Este é o caso das informações contidas na web, mais precisamente na web semântica, pois ontologias podem representar a semântica dos documentos e permitir que ela seja usada por aplicações web e por agentes de software.

Uma tradução livre para o termo web semântica, dada por [Berners-Lee, 2002] seria: *web semântica é a extensão da web obtida via adição de semântica ao formato atual de representação de dados*.

## 4. Estudo de caso

O problema proposto foi a criação de uma ontologia para grupos e projetos de pesquisa em Ciência da Computação, com a finalidade de criar uma prova de conceito para a Web Semântica anotando as páginas do TecComm e de todos os alunos, professores e pesquisadores ligados a ele. Estas páginas anotadas vão alimentar uma aplicação web baseada em componentes e agentes de software, para gerar relatórios acadêmicos. O TecComm é um laboratório temático, originado a partir do Laboratório de Engenharia de Software (LES) do Departamento de Informática da PUC-Rio, sob coordenação do Prof. Carlos Lucena. Este estudo de caso foi executado em conjunto com Leonardo M. Cunha.

Para o desenvolvimento da ontologia utilizamos uma combinação da metodologia de [Uschold and King, 1995] e o método proposto por um grupo de Stanford [Noy &McGuinness, 2000], cujas etapas podem ser divididas em: busca por ontologias que pudéssemos reutilizar; definição das questões de competência; definição dos conceitos básicos da nossa ontologia; definição de conceitos mais refinados a partir dos conceitos básicos; definição dos relacionamentos entre os conceitos; refinamento da ontologia. O editor de ontologias usado foi o OilEd e o arquivo DAML com a ontologia encontra-se em http://www.teccomm.les.inf.puc-rio.br/daml/onts/v0_6/projeto.daml

## 5. Considerações finais

A experiência que adquirimos durante o desenvolvimento da ontologia para projetos e grupos de pesquisa nos mostrou que a área de engenharia de ontologias tem avançado, mas nos pareceu que o termo "engenharia de ontologias" ainda não está totalmente consolidado na comunidade científica.

Ao direcionarmos esforços para o desenvolvimento de ontologias para serem usadas na web semântica, juntamos um assunto novo a outro novíssimo e dependente do anterior. Esta junção nos trouxe limitações de desenvolvimento naturais quando se usa novas tecnologias, como por exemplo a restrição da linguagem de descrição da ontologia e a escassez de ferramentas para sua edição de verificação.

Concluímos que o uso de ontologias se coloca como um requisito necessário para que a web semântica se torne tão popular quanto a web atual, pois elas proporcionarão a universalização do entendimento do conteúdo disponibilizado na web. Para que isso ocorra, é preciso que os desenvolvedores de ontologias façam o máximo de reuso das ontologias existentes, seja através de composição de ontologias ou de equivalência entre conceitos.

## 6. Referências

[Berners-Lee, 1990] Berners-Lee, T: *Information Management: A Proposal*, CERN, March 1989, May 1990

[Berners-Lee, 2002] Berners-Lee, T: *The World Wide Web - Past Present and Future*, http://www.w3.org/2002/04/Japan/Lecture.html, última visita 20/10/2002

[Borst, 1997] Borst, W.N.: *Construction of Engineering* Ontologies, University of Tweenty, Enschede, NL- Center for Telematica and Information Technology, 1997.

[Gruber, 1993] Gruber, T.R.: *A Translation Approach to* Portable *Ontology Specification*, Knowledge Acquisition 5: 199-220, 1993

[Noy and McGuinness] Noy, N. & McGuinness, D.L.: *Ontology Development 101: A Guide to Create Your First Ontology*, 2000, http://protege.stanford.edu/publications/ontology_development/ontology101.html, último acesso em set/2002.

[Uschold and King, 1995] Uschold, M & King, M.: *Towards a Methodology for Building Ontologies*, Workshop on Basic Ontological Issues in Knowledge Sharing, 1995.

# Ferramentas de Interação Textual

Mariano Gomes Pimentel[1], Hugo Fuks[1], Carlos José Pereira de Lucena[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{mariano, hugo}@les.inf.puc-rio.br

## 1. Introdução

Nos atuais sistemas de Educação a Distância pela Internet, toda a dinâmica educacional é registrada: desde uma longa mensagem enviada por correio eletrônico até uma pequena frase enviada por bate-papo. Numa única turma podem ser geradas centenas de mensagens de e-mail e milhares de mensagens de bate-papo (ver alguns dados em [Fuks *et. al.* 2002]). Esta interação, se analisada adequadamente, pode fornecer informações úteis sobre a dinâmica de ensino e aprendizagem. Esta pesquisa tem por objetivo investigar as ferramentas de interação textual e analisar a interação que ocorre nestas ferramentas - conforme detalhado na próxima seção.

## 2. Objetivos da pesquisa

Os objetivos desta pesquisa encontram-se esquematizados na figura 1.

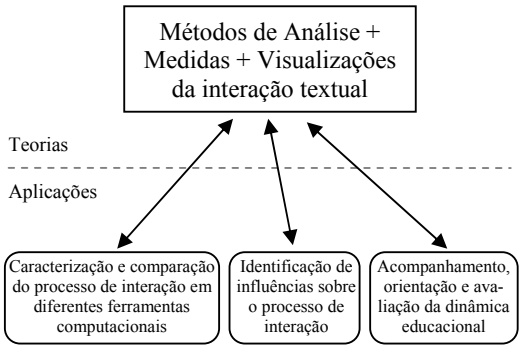



Figura 1 – Objetivos da pesquisa

Esta pesquisa investiga as ferramentas de comunicação textual, como correio eletrônico e bate-papo, objetivando definir métodos de análise, medidas e visualizações da interação entre usuários. Objetiva-se dar continuidade ao trabalho iniciado em [Pimentel 2002], onde – dentre outros métodos, medidas e visualizações – foi investigado o tempo de resposta numa sessão de bate-papo (figura 2), definido um método para analisar os tópicos conversados numa sessão de bate-papo e sua visualização através da representação por "ondas de assunto" (figura 3).

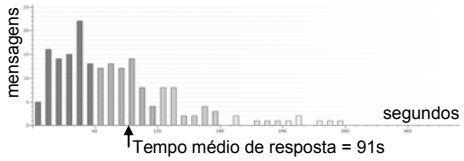



Figura 2 – Análise do tempo médio de resposta numa sessão de bate-papo

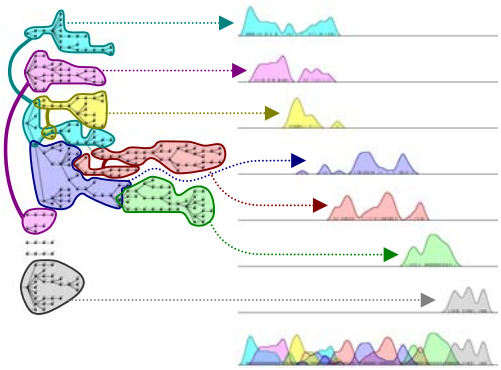


Figura 3 – Análise do encadeamento da conversação e dos tópicos conversados durante de uma sessão de bate-papo

A aplicação desta pesquisa tem como alvo o sistema AulaNet [Fuks 2000], um ambiente de Educação a Distância baseado na Web, para o qual objetiva-se: caracterizar o processo de interação que ocorre em suas ferramentas de interação textual (seção 2.1), aperfeiçoar e desenvolver novas ferramentas (2.2), e desenvolver mecanismos computacionais para apoiar o acompanhamento, orientação e avaliação da dinâmica educacional (2.3).

### 2.1. Caracterização do processo de interação

O ambiente AulaNet possui diferentes ferramentas para interação textual: "Lista de Discussão", "Conferências" (no estilo de fórum), "Debate" (no estilo de bate-papo), "Mensagem para Participante" (semelhante a um *messenger*), dentre outras. É de interesse para esta pesquisa caracterizar os diferentes processos de interação que ocorrem nestas ferramentas, investigando e

caracterizando medidas como o tamanho médio de mensagens, o tempo médio de resposta, etc.

## 2.2. Influências das ferramentas sobre a interação

Em função de certas especificações de cada ferramenta, são oferecidas algumas potencialidades e impostas certas limitações para a interação, o que influencia na qualidade e quantidade de texto produzido pelos usuários, no tempo de envio de mensagens, na compreensão da conversação, dentre diversos outros fatores. Com esta pesquisa objetiva-se fornecer alguns mecanismos para evidenciar estas influências e assim possibilitar investigar e desenvolver ferramentas que sejam mais adequadas para uma determinada atividade.

## 2.3. Acompanhamento da dinâmica educacional

A análise do registro das interações nas diferentes ferramentas de comunicação tem o potencial para apoiar o acompanhamento, a orientação e a avaliação dos aprendizes e da dinâmica educacional [Pimentel e Sampaio 2002]. A análise destes registros, contudo, ainda é pouco apoiado tecnologicamente [Otsuka 2002]. Com esta pesquisa objetiva-se dar maior suporte computacional para a realização desta atividade.

## 3. Conclusões

A pesquisa em andamento, apresentada neste artigo, tem por objetivo teórico o desenvolvimento de métodos para análise da interação que se realiza nas ferramentas de comunicação textual, bem como a definição de algumas medidas e visualizações desta interação. Como objetivo prático, esta pesquisa propõe a análise e caracterização das interações que ocorrem nas diferentes ferramentas do AulaNet, investigar e desenvolver novas ferramentas de interação para este ambiente, e construir mecanismos computacionais para apoiar a investigação da dinâmica educacional.

## 4. Referências

PIMENTEL, Mariano Gomes. *HiperDiálogo: uma ferramenta de bate-papo para diminuir a perda de co-texto*. Dissertação de Mestrado em Informática, NCE/IM/UFRJ, março 2002.

PIMENTEL, Mariano Gomes, SAMPAIO, Fábio Ferrentini. Comunicografia: uma metodologia para análise de processos de interação(...). *Revista Brasileira de Informática na Educação (SBC)*, n. 1, v. 10, abril 2002.

FUKS, Hugo. Aprendizagem e trabalho cooperativo no ambiente AulaNet. *Revista Brasileira de Informática na Educação (SBC)*, n. 1, v. 6, abril 2000.

FUKS, Hugo, GEROSA, Marco Aurélio, LUCENA, Carlos José Pereira de. Usando a categorização e estruturação de mensagens textuais em cursos pelo ambiente AulaNet. *Revista Brasileira de Informática na Educação (SBC)*, n. 1, v. 10, abril 2002.

OTSUKA, Joice Lee. *Um modelo baseado em agentes de interface para o suporte à avaliação formativa em ambientes de educação a distância*. Exame de qualificação, UNICAMP, novembro 2002.

# Formação de Grupos em Turmas de um Curso no Ambiente AulaNet Utilizando Agentes de Software

Leonardo Magela Cunha[1], Hugo Fuks[1], Carlos José Pereira de Lucena[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{leocunha, fuks, lucena}@inf.puc-rio.br

## 1. Descrição do Trabalho

Atualmente a complexidade do trabalho e a disseminação das tecnologias da informação e comunicação valorizam e potencializam o trabalho em grupo. O apoio computacional fornecido para o trabalho em grupo, denominado groupware, baseia-se na pesquisa de Computer Supported Cooperative Work. O suporte aos trabalhadores deve ser fornecido tanto para a criação dos grupos de trabalho assim como para a sua dissolução, passando pelo apoio ao trabalho em grupo propriamente dito.

Em Engenharia de Software a utilização de sistemas multi-agentes possibilita um nível de abstração mais adequado para o tratamento de problemas complexos e distribuídos. Um exemplo destes problemas é o caracterizado pelos ambientes de trabalho e aprendizagem em grupo na Web. Este trabalho apresenta o estudo e a forma como foi implementado um sistema multi-agentes para o auxílio à formação de grupos no ambiente AulaNet.

A proliferação de ambientes computacionais heterogêneos, o acesso a grandes quantidades de informação distribuídas pelas redes e a complexidade do mundo real está transformando rapidamente como as pessoas aprendem. A tecnologia de agentes se apresenta como uma estratégia promissora para ser aplicada aos desafios dos ambientes educacionais modernos que estão cada vez mais influenciados por tecnologias como Internet e Inteligência Artificial.

Com o surgimento da Inteligência Artificial Distribuída [Weiss, 1999] vivenciou-se e vivencia-se um momento de fortalecimento dos conceitos de agentes de software e também se percebe um redirecionamento destes conceitos para a Engenharia de Software [Jennings and Wooldridge, 2000]. As plataformas computacionais e os ambientes de informação modernos são distribuídos, grandes, abertos e heterogêneos. Essas características se aplicam diretamente aos ambientes de instrução baseada na Web. Agentes de software podem influenciar diferentes campos em sistemas educacionais. Eles fornecem novos paradigmas educacionais, suportam teorias, e podem auxiliar tanto aprendizes quanto professores na tarefa de aprendizagem auxiliada por computador [Aroyo and Kommers, 1999].

Este trabalho apresenta como foi implementado um sistema multi-agentes para a formação de grupos de aprendizes em turmas de um curso no ambiente AulaNet [Cunha, 2002]. Apresentam-se conceitos sobre agentes de software e então se explica com base nestes conceitos porque utilizar agentes de software para formação de grupos. É também apresentado como foi feita a implementação da modelagem dos aprendizes e do SMA para formação de grupos.

Conforme destacado por Dertouzos [2001], tanto o trabalho em grupo como a educação são importantes na nova era em que a computação está voltada para os humanos (human-centered computing). Este trabalho se insere nestes dois campos fornecendo uma perspectiva tecnológica para a formação de grupos de trabalho. Esta perspectiva é representada pela tecnologia de CSCW incrementada com a utilização de agentes de software. A utilização dos grupos de trabalho dentro do ambiente AulaNet fornece uma visão realista das necessidades dos aprendizes e professores e colabora para uma pesquisa contextualizada.

Além disso, a preocupação em utilizar padrões IMS [2002] e a abordagem groupware que é adotada no sistema possibilita uma fácil correspondência entre o mundo educacional e o mercado de trabalho. A utilização do AulaNet como uma ferramenta para o suporte ao trabalho já vem sendo pesquisada e é promissora. A correspondência de um curso com um projeto, de uma turma com uma equipe, de um grupo de aprendizagem com um grupo de trabalho é uma visão estimulante neste processo de pesquisa.

Pela literatura pesquisada, acredita-se que a utilização de agentes pode favorecer vários aspectos psico-pedagógicos em educação baseada na Web. Ao fornecer apoio para a formação e o trabalho em grupo, os agentes de software estão também apoiando a aprendizagem por projetos e a aprendizagem colaborativa. Quando os grupos

formados apresentam um alto nível de heterogeneidade, a atitude e prática interdisciplinares também podem ser beneficiadas. Da mesma forma que estes aspectos são influenciados é necessário ressaltar que habilidades profissionais podem ser desenvolvidas e influenciadas através do trabalho em grupo como as capacidades de se automonitorar, de ouvir, de apresentar idéias e de persuadir entre outras.

## 2. Referências

[Weiss, 1999] WEISS, G. Multiagent Systems – A Modern Approach to Distributed Artificial Intelligence. Cambridge, MA: MIT Press, 1999.

[Jennings and Wooldridge, 2000] JENNINGS, N. and WOOLDRIDGE, M. Agent-Oriented Software Engineering. In: BRADSHAW. J. (ed.), Handbook of Agent Technology, AAAI/MIT Press, 2000.

[Aroyo and Kommers, 1999] AROYO, L. and KOMMERS, P. Preface - Intelligent Agents for Educational Computer-Aided Systems. Journal of Interactive Learning Research, v. 10, n. 3/4, p. 235-242, 1999.

[Dertouzos, 2001] DERTOUZOS, M. L. Unfinished revolution: human-centered computers and what they can do for us. New York: HarperCollins Publishers, Inc., 2001.

[IMS 2002] Instructional Management Systems Global Consortium, Inc. [online] [citado em 03 de novembro de 2002] Disponível na World Wide Web em <URL: http://www.imsproject.org>.

[Cunha, 2002] CUNHA, L. M. Formação de Grupos de Trabalho Utilizando Agentes de Software. Dissertação de mestrado. Rio de Janeiro: PUC-Rio, 2002.

# Introduzindo Object Circuits

Matheus Costa Leite[1], Carlos José Pereira de Lucena[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{matheus, lucena}@inf.puc-rio.br

## 1. Resumo

Neste artigo, introduzimos o conceito de Object Circuits, uma técnica de programação que aborda a programação orientada a objetos tradicional através da metáfora de circuitos elétricos. Nossa motivação é o estudo da utilidade de Object Circuits sob o domínio específico de modelagem e simulação.

## 2. Funcionamento Geral

Object Circuits é um paradigma para construção de software que combina a expressividade da programação orientada a objetos com a semântica bem estabelecida de circuitos elétricos tradicionais. Uma lista incompleta de características importantes é dada abaixo:

- Modelo intrinsecamente concorrente: Tipicamente, um paradigma onde programas são representados graficamente e não textualmente tem uma vantagem no design de concorrência. A razão é que os elementos que compõem o programa são visualmente distribuídos em duas ou mais dimensões ao invés de uma, resultando numa clara separação de blocos independentes. Na abordagem adotada por Object Circuits, por exemplo, cada nó computacional opera independentemente dos demais.
- Suporte à evolução dinâmica: Um dos principais objetivos do desenvolvimento de software orientado a componentes é o suporte à evolução dinâmica, ou seja, a habilidade de alterar um sistema em execução colocando ou retirando partes. Paradigmas convencionais têm notória dificuldade em tratar este problema [Koksal 1999]; o mesmo não ocorre com circuitos elétricos. Não fosse este o caso, seria necessário desligar a chave geral de energia de uma casa antes de trocar uma única lâmpada queimada.
- Fraco acoplamento entre partes: Diferentemente do paradigma OO convencional, Object Circuits utiliza conceitos de programação orientada a componentes para tornar essencialmente nulo o acoplamento entre as partes que compõem o sistema [Szyperski 1997]. No paradigma OO, existe uma interface de entrada (métodos públicos de um objeto), mas não a de saída (métodos invocados por um objeto), o que gera o acoplamento. Já em Object Circuits, tanto a interface de entrada quanto a de saída são bem conhecidas e documentadas.
- Uso natural de ambientes visuais: Dada a natureza essencialmente gráfica de circuitos, a construção de ambientes visuais torna-se decorrente.

## 3. Simulação e Object Circuits

De uma perspectiva histórica, orientação a objetos e simulação tem muito em comum, pois alguns conceitos-chave do primeiro – incluindo classes e objetos, herança e *dynamic binding* – foram introduzidos por uma linguagem de simulação dos anos 60, Simula I [Birtwistle 1979]. De fato, os resultados obtidos no domínio de simulação causaram Simula I ser reescrita mais tarde como Simula 67: uma linguagem completa e genérica que influenciou muitas linguagens OO modernas.

Por outro lado, a teoria de circuitos também está intimamente ligada a simulação, no sentido de que tanto uma quanto outra lidam com partes comunicantes fortemente concorrentes em um ambiente dinâmico. Logo, a união de orientação a objetos e circuitos é perfeita: o primeiro, como uma poderosa e expressiva ferramenta de modelagem; o segundo, como uma plataforma onde sistemas concorrentes têm uma descrição natural e elegante.

## 4. Referências

[Szyperski 1997] C. Szyperski. **Component Software: Beyond Object-Oriented Programming**. Addison-Wesley, 1st edition, December 1997

[Birtwistle 1979] G.M. Birtwistle. **SIMULA Begin**. Van Nostrand Reinhold, June 1979.

[Koksal 1999] P. Koksal, I. Cingil, A. Dogac. **A Component-based Workflow System with Dynamic Modifications**. In Proceedings of the Next Generation Information Technologies and Systems (NGITS'99), Israel, 1999.

# Mediated Chat 2.0
# Uma Instanciação do Framework Canais de Comunicação

Juliana Lucas de Rezende[1], Hugo Fuks[1], Carlos José Pereira de Lucena[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)
R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{jlucas, hugo}@inf.puc-rio.br

## 1. Informações Gerais

O desenvolvimento de uma ferramenta de bate-papo cujo objetivo é a facilitação de debates provendo técnicas de comunicação para ajudar o mediador na condução da conversação entre os aprendizes de um curso é o principal objetivo deste trabalho. Essa ferramenta foi batizada de Mediated Chat 2.0 (MC2) e será disponibilizada no ambiente AulaNet [Lucena et al., 1999], substituindo a atual ferramenta que é o Mediated Chat 1.0 (MC1).

O curso TIAE – Tecnologias da Informação Aplicadas à Educação [Fuks, Gerosa & Lucena, 2002] tem como objetivo que seus alunos aprendam a trabalhar com o grupo as tecnologias da informação, tornando-se educadores baseados na Web. O curso é oferecido desde 1998 como uma disciplina do Departamento de Informática da PUC-Rio e é ministrado totalmente via Internet pelo ambiente AulaNet [Fuks & Lucena, 2002].

O serviço **Debate** viabiliza a troca de mensagens em tempo real entre os participantes através de um *chat* textual que é o MC1 (Figura 1). No TIAE, os temas são divididos em aulas e o Debate é utilizado para discuti-los semanalmente.

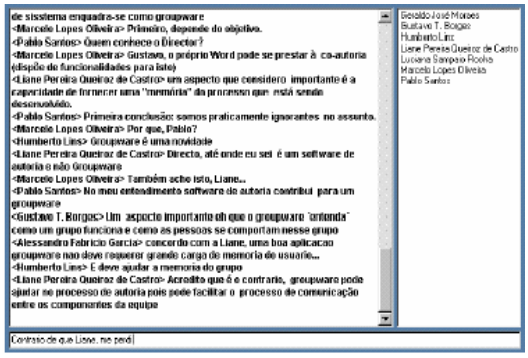



Figura 1 – Interface do MC1

As grandes dificuldades dos mediadores do debate são: fazer com que todos participem, manter o foco da conversa evitando que a discussão caminhe para tópicos irrelevantes, e manter o ritmo do debate.

Num chat é muito fácil "se perder": um monte de gente teclando, um monte de idéias diferentes, assuntos paralelos e não relacionados. Então como manter o foco? No MC1 a única forma de coordenação é o protocolo social. O moderador deve estar extremamente atento para perceber quando a discussão começa a tomar um rumo inadequado ou a se dispersar. Quando isto ocorrer, ele deve lembrar a todos do FOCO. Se não der certo, deve chamar atenção individualmente dos participantes, ou até "GRITAR" de vez em quando. Se o moderador não tiver pulso firme, o debate vira um bate-papo improdutivo ou se discute a escalação da seleção brasileira.

Percebeu-se que durante o curso os aprendizes evoluiam e conseguiam se organizar melhor durante o debate. Pelo fato desta evolução nem sempre ser satisfatória, surgiu o pensamento: "Seria possível oferecer um suporte tecnológico que ajudasse neste processo de evolução do aprendiz?"

Assim, desenvolveu-se uma ferramenta de bate-papo chamada de MC2 (Figura 2) que possui o suporte tecnológico pretendido. O MC2 é uma instanciação do framework Canais de Comunicação [Ferraz & Fuks, 2000], que foi desenvolvido no Laboratório de Engenharia de Software da PUC-Rio em 1999, da mesma forma que o MC1, que é a ferramenta de bate-papo que atualmente fornece o serviço de Debate no AulaNet, já é uma instanciação do framework citado.

O MC2 é uma evolução do MC1, pois além de possuir as funcionalidades já existentes no MC1, ele adicionou técnicas de comunicação para alcançar o principal objetivo do projeto.

## 2. Estudo e Aplicação

Uma analogia entre as técnicas de trabalho em grupo [Minicucci, 1992] e as políticas de comunicação foi realizada para tentar oferecer o suporte tecnológico usando conceitos da informática.

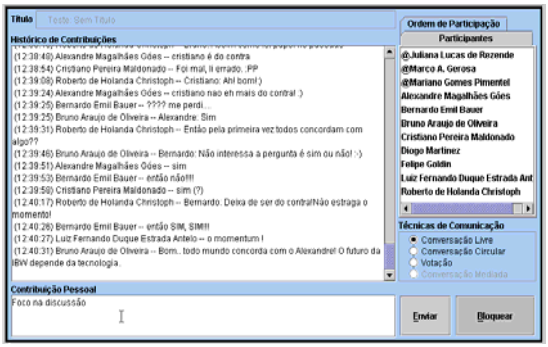


Figura 2 – Interface do MC2

Uma das características mais importantes de um sistema de comunicação é o conjunto de políticas de comunicação disponibilizadas por ele. De maneira geral, as políticas de comunicação são responsáveis por coordenar a interação dos atores envolvidos.

A escolha das políticas iniciou-se com a percepção da semelhança entre as políticas e as técnicas de escalonamento [Tanenbaum & Woodhull, 1997] utilizadas em Sistemas Operacionais. Um participante seria tratado como se fosse um processo em uma técnica de escalonamento. Quando um ou mais participantes requisitam a palavra no debate, é preciso decidir a ordem em que os pedidos serão deferidos. Os algoritmos de escalonamento estudados foram: o FIFO, o FILO, o Round Robin, o por Prioridades e o Aleatório.

Após o estudo dos algoritmos de escalonamento, verificou-se que estes não propiciavam coordenação de forma satisfatória, pois os objetivos são muito distintos. Os algoritmos de escalonamento têm como objetivo a utilização do canal com o menor desperdício possível, e dando oportunidade a todos os processos. Já no caso da mediação da comunicação, o objetivo é dar suporte tecnológico ao mediador fornecendo ferramentas que facilitem a coordenação de um debate.

A seguir serão descritas as técnicas utilizadas no MC2. Essas técnicas foram escolhidas pois com elas era possível aplicar a dinâmica desejada em TIAE, que foi onde a nova ferramenta de bate-papo foi utilizada.

**Conversação Livre**: o aprendiz está livre para enviar suas contribuições, sem ter que seguir uma ordem pré-estabelecida e nem se preocupar com o número de mensagens enviadas.

**Conversação Circular:** o aprendiz deverá enviar sua contribuição no momento em que estiver habilitado, o que acontece de forma circular; o número de rodadas é definido pelo mediador.

**Votação:** cada aprendiz poderá manifestar a sua posição com apenas uma contribuição; e só estarão habilitados novamente quando todos tiverem votado.

As técnicas apresentadas não são técnicas rígidas que devem ser seguidas à risca. A iniciativa de adaptá-las às características do grupo e às exigências da situação vale muito.

## 3. Referências

[Ferraz & Fuks, 2000] Ferraz, F.G. & Fuks, H. (2000), "*Framework Canais de Comunicação*", Julho 2000.

[Fuks, Gerosa & Lucena, 2002] Fuks, H., Gerosa, M.A. & Lucena, C.J.P. (2002), "*The Development and Application of Distance Learning on the Internet*", The Journal of Open and Distance Learning, Vol. 17, N. 1, ISSN 0268-0513, Fevereiro 2002, pp. 23-38.

[Fuks & Lucena, 2002] Fuks, H. & Lucena, C.J.P. (2002), "*Tecnologias de Informação Aplicadas à Educação (TIAE): Manual do Aprendiz*", Maio 2002.

[Lucena et al., 1999] Lucena, C.J.P., Fuks, H., Milidiú, R., Laufer, C., Blois, M., Choren,R., Torres, V., Daflon, L. "*AulaNet: Helping Teachers to Do Their Homework*". In: Multimedia Computer Techniques in Engineering Education Workshop. Graz, Austria: Technische Universitat Graz, 1999.

[Minicucci, 1992] Minicucci, A. "*Técnicas do Trabalho de Grupo*". 2. ed. São Paulo : Atlas, 1992.

[Tanenbaum & Woodhull, 1997] Tanenbaum, A. & Woodhull, A. "*Operating Systems: Design and Implementation*". Prentice-Hall, 1997.

# MetaCom-G*: Especificação da Comunicação entre Membros de um Grupo

Clarissa Mª de Almeida Barbosa[1], Clarisse Sieckenius de Souza[1], Raquel Oliveira Prates[2]

[1]SERG, Departamento de Informática – PUC-RIO

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

[2]Departamento de Informática e Ciência da Computação, IME/UERJ

R. São Francisco Xavier, 524, 6º andar, Bloco B, Tijuca– 20550-013– Rio de Janeiro – RJ – Brasil

{cbarbosa, clarisse}@inf.puc-rio.br, raquel@ime.uerj.br

## 1. Visão Geral

A complexidade inerente à atividade de design e os desafios próprios do projeto de interfaces multi-usuário motivam o desenvolvimento de modelos, métodos, técnicas e ferramentas que apóiem o processo de design de interfaces multi-usuário. O MetaCom-G [Prates,1998] é um destes modelos. Fundamentando-se na Engenharia Semiótica [de Souza,1993], Prates, através do MetaCom-G, propõe que se ofereça ao designer um ambiente que lhe permita descrever de que forma um grupo se organiza para desempenhar a sua atividade apoiado por computador e que lhe forneça indicadores qualitativos sobre sua descrição que o auxiliem a tomar suas decisões de projeto e construção da interface. Entretanto, como os meios dos quais o designer dispõe para descrever a comunicação entre os membros de um grupo permitem-lhe apenas descrevê-la em um alto nível de abstração, o apoio que o ambiente proposto no MetaCom-G pode oferecer ao designer é ainda restrito. Para ampliar este apoio, estendemos o MetaCom-G, definindo, assim, o MetaCom-G*. Neste modelo, permitimos ao designer descrever de forma mais precisa o processo de comunicação de um grupo. Como a nossa proposta de extensão baseia-se nas teorias dos Atos de Fala [Austin,1962; Searle,1969,1979] e Análise de Discurso [Brown&Yule,1983], as informações fornecidas pelo ambiente de apoio ao processo de design de interfaces multi-usuário proposto no MetaCom-G* levam o designer a refletir sobre o impacto da comunicação sobre os membros do grupo e, assim, auxiliam-no a tomar suas futuras decisões de design relativas ao processo de comunicação do grupo.

## 2. Contribuições

Todos os modelos fundamentados na Engenharia Semiótica atuam no nível do **conhecimento** que o designer possui sobre a situação a ser resolvida. À medida que ele interage com os diversos produtos gerados com o apoio das ferramentas construídas com base nestes modelos, o designer reflete sobre o contexto corrente, passando a conhecer e compreender um pouco mais o problema a ser tratado. A partir destas reflexões e do novo conhecimento obtido, ele toma suas futuras decisões de projeto [Schön,1983; Schön&Bennett,1996]. Acredita-se que quanto maior for o conhecimento do designer sobre o problema, maior será a qualidade do produto resultante. Os modelos da Engenharia Semiótica, então, visam capacitar o designer a projetar uma aplicação com boa qualidade de IHC.

Observe que os modelos fundamentados na Engenharia Semiótica não atuam diretamente sobre o produto resultante do processo de design. Dessa forma, não cabe avaliarmos o produto, mas sim o conhecimento que o designer pode vir a adquirir sobre a situação a ser resolvida caso tenha à sua disposição um ambiente de apoio ao processo de design fundamentado no modelo correspondente.

Seguindo esta linha de raciocínio, no caso particular do MetaCom-G*, temos que este modelo necessariamente leva o designer a refletir sobre certos aspectos do processo de comunicação do grupo que não são abordados no modelo original. Assim, o MetaCom-G* potencializa o conhecimento que o designer pode adquirir a respeito da comunicação entre membros de um grupo. Podemos, portanto, afirmar que, dentro da proposta da Engenharia Semiótica, o MetaCom-G* oferece contribuições importantes para o apoio ao processo de design de interfaces multi-usuário.

## 4. Referências

AUSTIN, J.L. **How to do things with words**. Cambridge, MA: Harvard University Press, 1962.

BROWN, G.; YULE. G. **Discourse analysis**. Cambridge: Cambridge University Press, 1983.

de SOUZA, C.S. The Semiotic Engineering of user interface languages. **International Journal of Man-Machine Studies**, v.39, n.5, p.753-773, November 1993.

PRATES, R.O. **A Engenharia Semiótica de Linguagens de Interfaces Multi-Usuário**. Rio de Janeiro, 1998. Tese de Doutorado - Departamento de Informática, PUC-Rio.

SCHÖN, D.A. **The reflective practitioner**: how professionals think in action. New York, NY: Basic Books, 1983.

SCHÖN, D.A.; BENNETT, J. Reflective conversation with materials. In: Winograd, T.(Ed.) **Bringing design to software**. New York, NY: ACM Press. 1996.

SEARLE, J. **Speech acts**: an essay in the philosophy of language. Cambridge: Cambridge University Press, 1969.

SEARLE, J. **Expression and meaning**: studies in the theory of speech acts. Cambridge: Cambridge University Press, 1979.

# Método para Construção de Sistemas de Ajuda *Online*

Milene Selbach Silveira[1,2], Clarisse Sieckenius de Souza[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

[2]Faculdade de Informática – Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)

Av.Ipiranga, 6681 – 90619-900 – Porto Alegre – RS – Brasil

{milene, clarisse}@inf.puc-rio.br

## 1. Introdução

A construção do sistema de ajuda *online* de uma aplicação é um dos passos mais críticos no design de IHC (Interação Humano-Computador). Este trabalho é baseado na Engenharia Semiótica [de Souza 1993], que vê a interface como uma mensagem do designer para os usuários, representando a solução do designer para o que ele acredita ser o problema do usuário. Na Engenharia Semiótica, o sistema de ajuda *online* é um componente essencial da aplicação. É através dele que os designers podem explicitamente "falar" aos usuários, mostrando como a aplicação foi construída, como pode ser usada e para que propósito.

Neste resumo será apresentado um método para o design de sistemas de ajuda *online* à luz da Engenharia Semiótica, e baseado em comunicabilidade e na técnica retórica de *layering*.

## 2. O Sistema de Ajuda *Online* à Luz da Engenharia Semiótica

É essencial que os usuários entendam a mensagem do designer para que consigam melhor compreender e tirar vantagens da aplicação a ser utilizada. Uma forma de tornar esta mensagem explícita é através do sistema de ajuda *online* da aplicação, onde os usuários devem ser capazes de expressar mais precisamente suas dúvidas e necessidades, e os designers devem ser capazes de antecipar tais dúvidas e necessidades e organizar sua mensagem de acordo com elas.

Baseado no método de avaliação de comunicabilidade [Prates, de Souza e Barbosa 2000] e em *layering* em documentação minimalista [Carroll 1998; Farkas 1998], o usuário pode escolher uma de um conjunto de expressões disponíveis para indicar sua dúvida durante a interação [Silveira, Barbosa e de Souza 2001]. Com isso, ele obterá um conteúdo de ajuda contextualizado pela expressão escolhida. Este conteúdo será apresentado de forma minimalista e poderá ser aprofundado, através do uso de novas expressões sobre o conteúdo apresentado (*layering*). O exemplo a seguir ilustra esta abordagem de ajuda. Este é um exemplo fictício, baseado em conteúdos de ajuda encontrados no MS Word® como resposta a uma requisição de ajuda sobre 'controle de alterações' (figura 1).

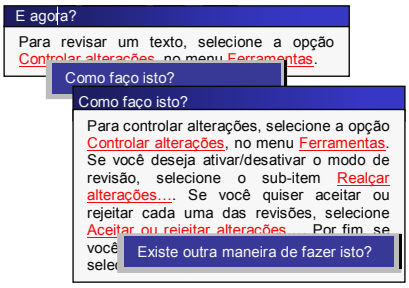



Figura 1 - Exemplo de respostas de ajuda.

A proposta de ajuda em que este trabalho se baseia, inclui, também, um modulo de ajuda geral, onde é possível encontrar informações sobre o domínio e sobre a aplicação como um todo, bem como cenários de uso desta aplicação.

## 3. Uso de um Método para Construção de Sistemas de Ajuda *Online*

A fim de ilustrar como esta visão comunicativa da ajuda pode ser desenvolvida, será apresentado um método de construção de sistemas de ajuda *online* à luz da Engenharia Semiótica.

Os passos a serem seguidos neste método são: construção dos modelos de design de IHC, considerando-se questões específicas relacionadas à ajuda; geração de um rascunho do conteúdo da ajuda; refinamento do conteúdo e especificação dos pontos de recorrência; construção do módulo de ajuda geral; conexão dos pontos de acesso à ajuda aos elementos de interface; testes preliminares; análise de uso durante a avaliação de comunicabilidade e refinamento ou re-design do conteúdo da ajuda. Estes passos são ilustrados na figura a seguir (Figura 2).

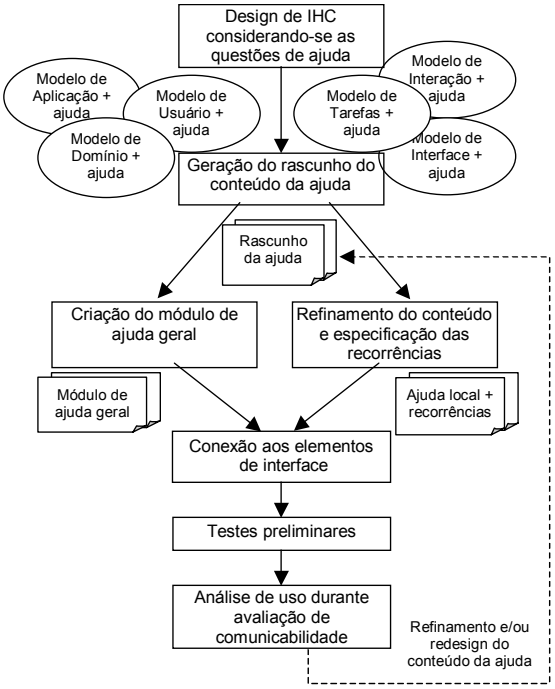



Figura 2 - O método proposto.

## 4. Considerações Finais

A Engenharia Semiótica considera o sistema de ajuda *online* como uma parte essencial da aplicação. É através da ajuda que o designer pode se comunicar diretamente com os usuários, mostrando as motivações por trás de seu design e como os usuários podem melhor utilizar a aplicação por ele desenvolvida.

Os usuários, por sua vez, podem expressar suas dúvidas mais diretamente, usando uma das expressões de ajuda local disponíveis durante a interação. A resposta a esta expressão será construída a partir do ponto de vista e do *rationale* do designer capturados durante o processo de design da aplicação. Além disto, os usuários podem aprofundar o conteúdo de ajuda através de pontos de recorrência disponíveis em cada resposta de ajuda, em uma indefinida cadeia de associações, a partir de suas necessidades locais. Este processo é associado a alguns conceitos fundamentais na teoria semiótica: semiose e abdução.

O método apresentado facilita o design do sistema de ajuda sob esta perspectiva. E o custo de desenvolvimento do mesmo é reduzido se for seguido um design de IHC baseado em modelos de design.

Para Engenharia Semiótica, em particular, o design baseado em modelos é muito importante, porque ele torna possível manter a consistência entre os produtos de design construídos a cada fase [Barbosa et al. 2002; Silveira, de Souza e Barbosa, 2002]. Isto permite ao designer criar e passar aos usuários uma mensagem coesa, a fim de aumentar suas chances de entender o sentido desta mensagem.

Além destes esforços técnicos e teóricos, é necessário um esforço, também, em mudar a visão dos usuários sobre os sistemas de ajuda. Como regra, usuários utilizam a ajuda somente como último recurso [Ceaparu et al. 2002]. Isto pode vir tanto de experiências frustradas no uso da ajuda no passado, quanto mesmo de não se ter um entendimento sobre qual a utilidade de um sistema de ajuda. Com isso, mesmo o sistema de ajuda mais cuidadosamente projetado não aumentará a experiência dos usuários com a aplicação. Como esta proposta abre um canal de comunicação direto dos designers para os usuários, acreditamos ser este um primeiro passo na introdução desta nova cultura.

## 5. Referências

Barbosa, S.D.J., de Souza, C.S.; de Paula, M.G.; Silveira, M.S. (2002) Modelo de Interação como Ponte entre o Modelo de Tarefas e a Especificação da Interface. *Anais do Simpósio Brasileiro de Fatores Humanos em Sistemas Computacionais*.

Carroll, J.C. (ed.) (1998) *Minimalism Beyond the Nurnberg Funnel*. Cambridge, The MIT Press, Cambridge.

Ceaparu, I., Lazar, J., Bessiere, K., Robinson, J., Shneiderman, B. (2002) Determining Causes and Severity of End-User Frustration. Technical Report, HCIL-2002-11, CS-TR-4371, UMIACS-TR-2002-51 [ftp://ftp.cs.umd.edu/pub/hcil/Reports-Abstracts-Bibliography/2002-11html/2002-11.pdf, visited September 2002]

de Souza, C.S. (1993) The Semiotic Engineering of User Interface Languages. *International Journal of Man-Machine Studies*, 39, 753-773.

Farkas, D.K. (1998) Layering as a Safety Net for Minimalist Documentation. Carroll, J.C. (ed.) *Minimalism Beyond the Nurnberg Funnel*. Cambridge, The MIT Press, Cambridge.

Prates, R.O., de Souza, C.S., Barbosa, S.D.J. (2000) A Method for Evaluating the Communicability of User Interfaces. *ACM Interactions*, 31–38. Jan-Feb 2000.

Silveira, M.S.; Barbosa, S.D.J.; de Souza, C.S. (2001) Augmenting the Affordance of Online Help Content. *Proceedings of IHM-HCI 2001*, Lille, Springer-Verlag.

Silveira, M.S.; de Souza, C.S.; Barbosa, S.D.J. (2002) Design de Sistemas de Ajuda Online baseado em Modelos. *Anais do Simpósio Brasileiro de Fatores Humanos em Sistemas Computacionais*

# Modelagem de Sistemas Multi-Agentes por Refinamentos e Validações Sucessivas

Viviane Torres da Silva[1], Carlos José Pereira de Lucena[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{viviane, lucena}@inf.puc-rio.br

## 1. Motivação

O paradigma de computação através de agentes de software trouxe um novo enfoque para a área de engenharia de software. Agentes são entidades autônomas e adaptativas que desempenham papéis ao se relacionarem com os outros agentes do sistema a fim de atingir seus objetivos. Este novo paradigma introduziu a necessidade de desenvolver novas metodologias [DeLoach 1999][Elamari and Lalondi 1999][Kinny and Georgeff 1997][Lind 2000][Wooldridge et al. 2000][Yu and Schmid 1999] para modelagem de software e introduziu a necessidade de criação de novas ferramentas que auxiliem o processo de modelagem e de implementação de sistema multi-agente (MAS).

No estudo que realizamos sobre metodologias para modelagem de MAS e propostas para a descrição de MAS verificamos que estas não tratam de maneira apropriada a fase de identificação dos agentes. É nesta fase que os agentes são identificados com base nas informações descritas na definição do problema.

Tendo em vista a problemática, propomos o método de refinamento que pretende abordar a fase de análise dos agentes do sistema. O método se baseia no refinamento das metas do sistema e na associação destas metas à entidades. O método aborda o sistema primeiramente como uma entidade única que possui metas e se comunica com entidades externas ao sistema para atingir estas metas. Em seguida, através de uma seqüência de passos de decomposição e especialização, isto é, de refinamento da entidade sistema em sub-entidades e das metas do sistema em sub-metas, identificamos os agentes que farão parte da solução do sistema.

## 2. Método de refinamento

O método é composto de 4 passos. A cada passo da decomposição identificam-se as sub-metas das metas identificadas no passo anterior e relacionam-se estas sub-metas às sub-entidades das entidades também anteriormente identificadas. Por último, associa-se a cada sub-entidade papéis e relacionamentos com outras sub-entidades através dos quais elas irão atingir as suas metas. É através do processo de refinamento das entidades e das metas do sistema que os agentes são identificados juntamente com os papéis que desempenham e os relacionamentos existentes entre eles.

Os agentes do sistema são justamente as entidades que aparecem no final do processo de refinamento. Desta forma, a descrição dos agentes fica facilitada dado que as suas metas, os papéis que desempenham e os relacionamentos dos quais participam já foram definidos.

Com o objetivo de ajudar o projetista no processo de decomposição das metas e das entidades do sistema, o método define uma série de regras de refinamento que devem ser aplicadas a cada passo da decomposição. O objetivo das regras é garantir que o que foi definido em um determinado nível de decomposição seja mantido em todos os níveis subseqüentes. Desta maneira pretende-se garantir que durante o processo de refinamento todos elementos modelados em um determinado nível apareçam especializados nos níveis subseqüentes. As regras pretendem garantir também que as características de cada elemento sejam mantidas.

### 2.1. Passo 1

Neste passo, o sistema é considerado uma entidade única (unidade) que possui uma meta ou um conjunto de metas e que desempenha papéis e se relaciona com entidades externas a fim de cumprir suas metas. Então, durante este passo a meta ou o conjunto de metas da entidade deve ser identificado juntamente com seus papéis e o relacionamento com as entidades externas.

### 2.2 Passo 2

O objetivo deste segundo passo é decompor e especializar a meta ou o conjunto de metas identificado no passo anterior. Para cada entidade identificada no passo anterior, deve-se decompor e especializar as suas metas. Em seguida deve-se associar estas sub-metas a sub-entidades que são decomposições e especializações das entidades identificadas no passo anterior. Cada sub-entidade deverá ser associada a pelo menos uma sub-meta.

### 2.3 Passo 3

Este passo deve ser aplicado para cada entidade definida no passo anterior. É necessário identificar

o relacionamento que a entidade tem com outras entidades do sistema e com entidades externas que a possibilitam atingir suas metas definidas no passo anterior. Por último, os papéis desempenhados por cada entidade precisam ser identificados.

## 2.4 Passo 4

Este passo define a condição de parada da especialização e decomposição das metas e das entidades do sistema. Enquanto a decomposição e a especialização das metas e das entidades não tiver chegado ao fim, os passos 2 e 3 devem ser executados consecutivamente permitindo a geração de novas sub-metas e sub-entidades. A decomposição e especialização das metas e entidades devem cessar quando as seguintes condições forem identificadas: (i) o conjunto de metas definido para cada entidade do sistema pode ser cumprido por um agente (ii) o conjunto de metas das entidades modeladas tiver abordado o nível de detalhamento descrito no problema.

## 3. Regras de refinamento

As regras de refinamento têm o objetivo de auxiliar o processo de decomposição e especialização dos modelos gerados a partir do método proposto. As regras foram definidas a fim de garantir que as propriedades descritas no modelo de mais alto nível de abstração estejam presentes durante todo o processo de refinamento até o modelo de mais baixo nível de abstração onde as classes dos agentes estão identificadas. Além disso, as regras pretendem demonstrar como as propriedades podem aparecer nos outros níveis do modelo.

São ao todo 3 tipos de regras. (i) regras para as metas e entidades – dada uma meta relacionada a uma entidade, esta pode ser decomposta em sub-metas que devem ser associadas às sub-entidades da entidade com a qual a meta se relaciona; (ii) regras para os papéis – papéis identificados nas entidades serão desempenhados pelas suas sub-entidades, isto é, dado um papel desempenhado por uma entidade, pelo menos uma sub-entidade irá desempenha-lo. No refinamento, novos papéis também podem ser criados; (iii) regras para os relacionamentos: os relacionamentos definidos pelas entidades em um determinado passo devem ser cumpridos pelas suas sub-entidades no passo seguinte. Além de garantirmos que o relacionamento seja mantido, devemos garantir também a semântica do relacionamento.

## 4. Referências

DeLoach, S. A.: Multiagent Systems Engineering: a Methodology and Language for Designing Agent Systems. Proceedings of Agent Oriented Information Systems (AOIS99), Seattle Washington, May (1999).

Elammari, M. and Lalonde, W.: An agent-oriented methodology: High-level and intermediate models. In G. Wagner and E. Yu, editors. Proc. of the 1st Int. Workshop. on Agent-Oriented Information Systems (1999).

Kinny, D. and Georgeff, M.: Modelling and design of multi-agent systems. In: Müller, J. P., Wooldridge, M., and Jennings, N. R. (Eds.), Intelligent Agents III (LNAI Volume 1193), Springer-Verlag, (1997) pp. 1-20.

Lind, J. MASSIVE: SoftwareEngineering for Multiagent Systems. PhD thesis, University of the Saarland (2000).

Wooldridge, M., Jennings, N. R., and Kinny, D.: The Gaia methodology for agent-oriented analysis and design. Journal of Autonomous Agents and Multi-Agent Systems, 3, (2000) pp. 285-312.

Yu, L. and Schmid B.: A conceptual framework for agent-oriented and role-based workflow modeling. In G. Wagner and E. Yu, editors, Proc. of the 1st Int. Workshop. on Agent-Oriented Information Systems (1999).

# Modelagem Semântica para Aplicações Web usando Meta-Modelos

Fernanda Lima[1], Daniel Schwabe [1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{ferlima, schwabe}@inf.puc-rio.br

## 1. Introdução

Métodos de projeto para aplicações Web utilizam modelos de Entidade-Relacionamento ou de Orientação a Objetos para expressar as informações de um domínio específico. Entretanto, domínios de aplicações Web podem não ser representados de forma precisa apenas contando com o uso destes paradigmas, uma vez que os dados existentes na Web não podem ser considerados estruturados, como estas notações presumem.

O método OOHDM-estendido vem sendo desenvolvido no grupo de Web Engineering do Departamento de Informática da PUC-Rio com o objetivo de modelar aplicações Web utilizando conceitos da Web Semântica.

De acordo com Tim Bernes-Lee [Berners-Lee 2000], a Web Semântica é composta de uma arquitetura onde tanto XML quanto RDF [W3C 2000b] desempenham papéis de destaque. A idéia da Web Semântica é alcançar "uma teia de dados que possam ser processados direta ou indiretamente por máquinas".

## 2. Método OODHM-estendido

O método OOHDM-estendido define meta-modelos com base nos modelos conceitual e navegacional do método OOHDM [Schwabe and Rossi 1998], propondo extensões que facilitam o mapeamento dos modelos para uma implementação capaz de realizar consultas tanto a dados e metadados integradamente.

### 2.1. Projeto Conceitual

O meta-modelo conceitual OOHDM descreve um domínio específico através de classes e relacionamentos utilizando a UML [OMG1999] com dois detalhes adicionais: atributos podem ser multi-tipados e possuem multiplicidade.

Através de regras, o meta-modelo conceitual pode ser mapeado para um esquema serializável XML utilizando uma linguagem própria do método.

Esta linguagem é proposta para permitir o mapeamento das primitivas do método para a arquitetura em camadas da Web Semântica. A linguagem faz uso de marcações já definidas em outras linguagens propondo extensões (novas marcações) somente onde é necessário. As principais linguagens utilizadas são: OWL [W3C 2002], DAML+OIL [DAML+OIL 2001] , RDF-Schema [W3C 2000] e XML-Schema [W3C 2001].

### 2.2. Projeto Navegacional

O meta-modelo navegacional faz uso de primitivas de modelagem de aplicações Web: objetos navegacionais (visões de objetos conceituais) e contextos navegacionais (conjuntos de objetos navegacionais de acordo com regras determinadas pelo projetista da aplicação).

Assim como no meta-modelo conceitual, definimos regras de mapeamento que permitem transpor o modelo navegacional para um esquema serializável.

Para extrair as visões que compõem o modelo navegacional é necessário utilizar uma linguagem de consulta com os seguintes pre-requisitos: capacidade de extrair conjuntos (para representar os contextos); capacidade de consultar tanto esquemas quanto instâncias (para representar contextos e grupos de contextos, dentre outros usos); capacidade de inferência (para extrair informação que não foi previamente modelada); modo declarativo (para permitir que a consulta seja descrita em cartões de especificação); suporte aos tipos de dados definidos na linguagem XML Schema (para oferecer melhor suporte que apenas tipos literais como em DTDs).

Escolhemos a linguagem de consulta RQL [Karvounarakis et al. 2002], pois até o momento da publicação deste trabalho, é a única linguagem capaz de combinar consultas a esquemas e instâncias de forma integrada.

## 3. Especificação de "Frameworks" de Aplicação

Um exemplo de uso desta proposta é na definição de "Frameworks" de Aplicação [Schwabe et al. 2001a, 2001b], onde são fornecidas definições 'parametrizadas' de contextos. Outros exemplos de

consultas a esquemas e instâncias de aplicações são citados em [Lima and Schwabe 2002].

Consideremos, por exemplo, um catálogo de Comércio Eletrônico. Um "framework" de catálogos pode ser descrito contendo diversas classes abstratas e pontos de flexibilização a serem especializados para cada catálogo específico. A classe abstrata Produto tem que ser especializada em cada definição de catálogo, onde toda a hierarquia de tipos de produtos pode ser definida. Não é sabido, a priori, quantos níveis de TiposDeProdutos ("Product Types") serão criados em cada catálogo.

Este "framework" pode ser visto como um meta-modelo para catálogos. A cada instanciação do "framework", podemos obter um modelo para um catálogo específico e posteriormente, uma instância com os produtos. Por exemplo, é possível definir restrições para todas as subclasses de Produto, como "defina um contexto com todos as subclasses de Produto que possuam um atributo chamado 'cor' e cujas instâncias tenham cor= azul".

A abordagem OOHDM-estendida pode ser usada para modelar todos os níveis deste "framework". As consultas podem recuperar todas as subclasses de TiposDeProdutos desta hierarquia, e também as instâncias, de modo integrado.

## 4. Referências

[Berners-Lee 2000] Berners-Lee, T.: "Weaving the Web: The original design and ultimate destiny of the World Wide Web", New York, NY, HarperCollins, 2000.

[DAML+OIL 2001] "Reference description of the DAML+OIL (March 2001) ontology markup language", Disponível em <http://www.daml.org/2001/03/reference.html>.

[Esmeraldo 1999] Esmeraldo, L.: "Frameworks para Projetos de Aplicação Hipermídia", Dissertação de Mestrado, Depto de Informática, PUC-Rio, 1999.

[Karvounarakis et al. 2002] Karvounarakis, G., Alexaki, S., Christophides, V., Plexousakis, D. and Scholl, M.: "RQL: A Declarative Query Language for RDF", In Proceedings of the 11th International World Wide Web Conference (WWW2002), Honolulu, Hawaii, USA, May, 2002. Disponível em http://139.91.183.30:9090/RDF/RQL/index.html>

[Lima and Schwabe 2002] Lima, F. and Schwabe, D.: "Aspectos de Modelagem Semântica para Aplicações Web", SBMIDIA 2002, Anais do VIII Simpósio Brasileiro de Sistemas Multimídia e Hipermídia, 2002.

[OMG1999] OMG: "Unified Modeling Language Specification version 1.3 (UML 1.3)", June 1999.

[Schwabe and Rossi 1998] Schwabe, D. and Rossi, G.: "An object-oriented approach to Web-based application design" Theory and Practice of Object Systems (TAPOS), October 1998, 207-225.

[Schwabe et al. 2001a] Schwabe, D., Rossi, G., Esmeraldo, L. and Lyardet, F.: "Engineering Web Applications for reuse" IEEE Multimedia 8(1) – Special Issue on Web Engineering, Jan-Mar 2001, 20-31.

[Schwabe et al. 2001b] Schwabe, D., Rossi, G., Esmeraldo, L. and Lyardet, F.: "Web Design Frameworks: An Approach to Improve Reuse in Web Applications", Lecture Notes in Computer Science (Hot Topics) 2016 - In Proceedings of the 2nd International Workshop on Web Engineering, 9th International World Wide Web Conference, Springer Verlag, 2001, 335-352.

[W3C 1999] W3C: "Resource Description Framework (RDF) Model and Syntax Specification", W3C Recommendation 22 February 1999. Disponível em <http://www.w3.org/TR/1999 /REC-rdf-syntax-19990222/>.

[W3C 2000] W3C: "Resource Description Framework (RDF) Schema Specification 1.0", W3C Candidate Recommendation 27 March 2000. Disponível em <http://www.w3.org/TR/2000/CR-rdf-schema-20000327/>.

[W3C 2001] W3C: "XML Schema Part 2: Datatypes", W3C Recommendation 02 May 2001. Disponível em <http://www.w3.org/TR/xmlschema-2/>.

[W3C 2002] W3C: "OWL Web Ontology Language 1.0 Reference", W3C Working Draft 29 July 2002. Disponível em <http://www.w3.org/TR/owl-ref/>.

# Otimização de Transporte em Redes de Dutos e Busca com Custos de Acesso Variados

Artur Alves Pessoa[1], Ruy Luiz Milidiú[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{artur, milidiu}@inf.puc-rio.br

## 1. Introdução

Neste trabalho, consideramos dois problemas de otimização combinatória bastante distintos: o problema de transporte em redes de dutos e o problema de construção de árvores para busca com custos de acesso variados.

O primeiro problema é uma abstração do transporte de produtos de petróleo através de redes de oleodutos. O estudo deste problema é motivado pelo projeto RSB*, que visa desenvolver ferramentas para o planejamento das operações da Transpetro através de uma parceria entre o Laboratório LEARN da PUC-Rio e o CENPES/Petrobras.

O segundo problema considera a busca por um elemento em um vetor ordenado e armazenado em meios físicos heterogêneos. Neste caso, o custo de cada acesso pode variar em função da posição do elemento no vetor. O estudo deste problema é motivado pelo projeto AMYRI, que é produto de uma cooperação entre pesquisadores íbero-americanos na área de Recuperação de Informações.

## 2. Transporte em Redes de Dutos

Um duto é um tipo abstrato de dados que respeita a regra FIFO (*first in first out*) e cujo número de elementos deve se manter fixo. Chamamos os elementos de um duto de bateladas. Como conseqüência disto, sempre que uma batelada é inserida (*push*) no final de um duto, a batelada do início deve ser removida (*pop*). Desta forma, as operações *push* e *pop* são simultâneas, resultando numa operação híbrida que denominamos *push-pop*.

Uma rede de dutos é definida por um grafo orientado $G=(N,A)$ onde cada arco $a \in A$ tem um duto associado com $v(a)$ posições, cada posição contendo uma batelada. No PTD, também é dado um conjunto $L$ de bateladas. Cada batelada $b$ tem um nó de destino $d(b) \in N$ e uma posição inicial. Esta posição inicial pode ser um nó ou uma posição de um duto da rede. Para cada arco $a = (i,j)$, uma operação PP consiste em inserir no duto uma batelada contida no nó $i$ e guardar no nó $j$ a batelada removida deste duto. Assumimos que os nós da rede têm capacidade ilimitada. Também é dado um subconjunto $F \subset L$ de bateladas *proteláveis*, que não precisam chegar aos seus destinos. Com isso, o objetivo do problema é encontrar uma seqüência de operações que transporte todas as bateladas não-proteláveis aos seus respectivos nós de destino.

Poucos trabalhos encontrados na literatura apresentam resultados teóricos sobre o problema de transporte em redes de dutos. Em [Hane and Ratliff 1995], os autores consideram redes de dutos cujos grafos associados são caminhos simples.

Neste trabalho, apresentamos novos resultados em relação ao PTD [Milidiú, Pessoa and Laber 2000] [Milidiú, Pessoa and Laber 2002a] [Milidiú, Pessoa and Laber 2000b]. Demonstramos que encontrar uma solução viável para o PTD é um problema NP-completo, mesmo que o grafo $G$ seja acíclico. Por isso, consideramos o PTD síncrono (PTDS), uma variação do PTD onde nenhuma batelada protelável está inicialmente contida em um duto. Assumindo que cada batelada $b$ tem um peso $w(b)$ associado, o custo de uma operação de inserção e remoção em um duto é modelado como uma combinação linear dos pesos das bateladas envolvidas. Neste caso, o PTDS de custo mínimo (PTDSC) consiste em encontrar uma solução de custo mínimo para o PTDS, onde o custo de uma solução é dado pela soma dos custos de suas operações. Para este problema, apresentamos o algoritmo BPA (*Batch-to-Pipe Assignment*), um algoritmo polinomial que obtém uma solução viável para o PTDSC. Caso o grafo $G$ seja acíclico, a solução dada pelo algoritmo BPA é ótima. Além disso, o algoritmo BPA é estendido para o caso em que bateladas com mesma origem, destino e peso podem ser agrupadas em ordens. Neste caso, o tempo de execução do algoritmo permanece polinomial em relação a uma entrada representada de forma mais compacta.

Também consideramos o PTDS de *makespan* mínimo (PTDSM), que consiste em obter uma solução para o PTDS que minimiza tempo necessário para terminar a última operação PP.

* Financiado pela FINEP/CTPetro

Neste caso, assumimos que cada duto para pode executar uma operação PP a cada unidade de tempo. Demonstramos que o PTDSM não admite nenhum algoritmo polinomial $\eta^{1-\varepsilon}$-aproximado para nenhum $\varepsilon > 0$, a menos que P = NP, onde $\eta$ é o tamanho da instância. Este resultado ainda vale se o grafo $G$ é acíclico e planar. Além disso, mostramos que o algoritmo BPA fornece uma solução $|A|$-aproximada para o caso em que o grafo $G$ é acíclico.

## 3. Busca com Custos de Acesso Variados

Seja $A = [a_1, ..., a_n]$ um vetor ordenado onde $c(a_i)$ é o custo de acessar o seu $i$-ésimo elemento. O objetivo do problema é encontrar uma estratégia de busca de custo mínimo para o vetor $A$. Se todos os custos de acesso são iguais, então a melhor estratégia de busca possível é a busca binária padrão, que permite encontrar qualquer elemento após $O(\log n)$ acessos. Por outro lado, se os custos de acesso são variados, então a busca binária não é necessariamente uma estratégia ótima.

Consideramos duas formas de calcular o custo de uma estratégia de busca: pelo custo médio de uma busca e pelo custo da busca mais cara. Chamamos os respectivos problemas de problema da busca média (PBM) e de problema da pior busca (PPB). No caso do PBM, assumimos que todos os elementos de $A$ têm a mesma probabilidade de busca. Em ambos os casos, o melhor algoritmo exato conhecido executa em tempo $O(n^3)$ e utiliza um espaço de memória $O(n^2)$ [Knight 1988] [Navarro, Barbosa, Baeza-Yates, Cunto and Ziviani 2000].

Neste trabalho, apresentamos algoritmos aproximados para o PBM e para o PPB [Laber, Milidiú and Pessoa 1999] [Laber, Milidiú and Pessoa 2002b] [Laber, Milidiú and Pessoa 2001] [Laber, Milidiú and Pessoa 2002a]. Assumimos sem perder generalidade que a soma dos custos de acesso é 1. No caso do PBM, apresentamos o algoritmo da razão. Este algoritmo sempre calcula uma solução de custo menor ou igual a $4\ln(n+1)/n$ em tempo $O(n^2)$, e utilizando um espaço $O(n)$. Além disso, introduzimos uma nova abordagem para desenvolver algoritmos aproximados para problemas de busca com custos de acesso variados. Com base nesta abordagem, desenvolvemos o algoritmo de escala de custos para busca média (ECBM), para o PBM, e o de escala de custos para a pior busca (ECPB), para o PPB. Ambos constroem soluções $(2+ \boldsymbol{\varepsilon} + o(1))$-aproximadas, para qualquer $\boldsymbol{\varepsilon} > 0$, em tempo $O(n)$ e utilizando um espaço $O(n)$.

## 4. Referências

[Hane and Ratliff 1995] Sequencing inputs to multi-commodity pipelines. *Annals of Operations Research*, 57, 1995. Mathematics of Industrial Systems I.

[Knight 1988] Search in an ordered array having variable probe cost. *SIAM Journal on Computing*, 17(6):1203-1214, December 1988.

[Laber, Milidiú and Pessoa 1999] Strategies for searching with different access costs. In *Seventh European Symposium on Algorithms*, LNCS, pages 236-247, Prague, Czech Republic, July 1999.

[Laber, Milidiú and Pessoa 2001] On binary searching with non-uniform costs. In *Proceedings of SODA'2001*, pages 855-864, Washington, DC, USA, January 2001.

[Laber, Milidiú and Pessoa 2002a] On binary searching with non-uniform costs. *Siam Journal on Computing*, 31(4):1022-1047, 2002.

[Laber, Milidiú and Pessoa 2002b] Strategies for searching with different access costs. *Theoretical Computer Science* (to appear), 2002.

[Milidiú, Pessoa and Laber 2000] Transporting petroleum products in pipelines (abstract). In *ISMP 2000 - 17th International Symposium on Mathematical Programming*, pages 134-135, Atlanta, Georgia, USA, August 2000.

[Milidiú, Pessoa and Laber 2002a] Pipeline transportation of petroleum products with no due dates. In *Proceedings of the LATIN'2002*, pages 248-262, Cancún, Mexico, april 2002.

[Milidiú, Pessoa and Laber 2000b] Complexity of makespan minimization for pipeline transportation of petroleum products. In *Proceedings of the APPROX'2002*, pages 243-255, Roma, Italy, 2002.

[Navarro, Barbosa, Baeza-Yates, Cunto and Ziviani 2000] Binary searching with non-uniform costs and its application to text retrieval. *Algorithmica*, 27(2):145-169, 2000.

# *Software Devices*: Uma nova visão sobre Componentes de Software

Gustavo Robichez de Carvalho[1], Carlos José Pereira de Lucena[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{guga, lucena}@inf.puc-rio.br

## 1. Introdução

A existência ou evolução de contratos de especificação de cenários de reutilização de componentes tende a facilitar e promover o processo de especificação e desenvolvimento de aplicações baseadas em componentes [Fiadeiro e Lopes].

Facilitar o processo de geração e evolução destas aplicações será determinante para o sucesso destas abordagens e da plena utilização da Engenharia de Software Baseada em Componentes.

Resumidamente, um *Software Device* é uma associação de componentes de software, seus contratos de cenários de reutilização e mecanismos de coordenação de componentes. Através desta associação prevê-se o desenvolvimento de um processo de geração de aplicações que tem como objetivo promover e facilitar a reutilização e a evolução destes componentes de software.

Este estudo tem como objetivo analisar a aplicabilidade e as restrições desta abordagem.

## 2. Motivação

A principal motivação deste trabalho é promover a reutilização de componentes de software e facilitar a evolução de aplicações que os utilizem. Entende-se por reutilização, o processo de criação de sistemas a partir de unidades de software existentes ao invés de criá-las do zero [Sametinger].

Componentes podem ser definidos como unidades de composição com interfaces contratualmente especificadas e com dependências explícitas de contexto. Um componente de software pode ser disponibilizado de forma independente e é passível de composição por terceiras partes [Szyperski].

Uma outra definição é que componentes de software são unidades reutilizáveis de uma solução, que precisam ser integradas a um todo para que desempenhem um papel na solução [Sametinger].

O pleno uso de componentes de software pode trazer como conseqüência uma melhora na qualidade das soluções geradas, um aumento da produtividade da equipe e do processo de desenvolvimento utilizado e um aumento da confiança e segurança de artefatos, pois estes são devidamente testados e validados [Sametinger].

Exemplos de componentes variam desde funções, classes, pacotes, subsistemas, componentes CORBA, processos distribuídos, web-services até agentes de software.

Para que seja possível otimizar o processo de reutilização, coordenação, ou composição de componentes é necessário que uma especificação, ou contrato, contendo todas as informações necessárias para esta finalidade, esteja bem definido sintática e semanticamente.

Esta documentação deve explicitar o tipo de componente, sua natureza, suas funcionalidades, cenários de reutilização e restrições de uso. Esta documentação deve ser explicitamente especificada em contratos de reutilização.

Estes contratos de reutilização são especificações contendo propriedades em que um usuário, um cliente, ou um objeto podem confiar, pois qualquer implementação deste componente deve satisfazê-las [Fiadeiro e Lopes].

Existem diversas linguagens de representação de processos de coordenação de componentes [Ciancarini, Gelernter and Carriero, Papadopoulos e Arbab]. Estas linguagens geram um conjunto de artefatos ou contratos de componentes, que poderão ser utilizados neste processo de coordenação.

Além disto, estes contratos podem especificar pontos de acessos e protocolos de interação de componentes, além de padrões de reutilização de componentes ou outras especificações necessárias para a coordenação de componentes.

É com o intuito de promover e disseminar a reutilização de componentes a partir de sua especificação através de contratos que está sendo estudada a abordagem baseada em *Software Devices*. Abaixo estará descrito um resumo para o trabalho de mestrado que está sendo desenvolvido.

## 3. Proposta

Esta dissertação de mestrado visa estudar, desenvolver e aplicar o conceito de *Software Devices* ao desenvolvimento de aplicações. Esta proposta é a associação de componentes de software e seus contratos a mecanismos capazes de coordenar a execução destes componentes.

Com esta abordagem é esperada a separação[Malone e Crowstone] entre:

- *Computational Concerns :* Qual a funcionalidade oferecida pelo componente;
- *Compositional Concerns:* Como explicitar formas e restrições de composição, e
- *Coordinational Concerns:* Como interagir com componentes para atingir determinado objetivo.

O conceito de componentes de software será utilizado como uma abstração de soluções, que compreende diferentes unidades de software com suas especificações explícitas em contratos e bem definidas, dentre elas possíveis formas de interação com demais componentes.

## 4. Organização do Estudo

O trabalho a ser desenvolvido durante a dissertação de mestrado compreenderá o levantamento de conjecturas sobre o que vem a ser um Software Device.

A partir desta conjectura, um conjunto de hipóteses compreendendo a abordagem de *Software Devices* será desenvolvido e avaliado para que seja possível compreender quais são as etapas, dificuldades e requisitos de um processo de utilização, composição e coordenação de componentes de software.

Visando colaborar, auxiliar e amadurecer este processo de compreensão será levantado um conjunto de evidências através de estudos experimentais.

São previstas a elaboração de no mínimo três experimentos com a utilização de pelo menos quatro diferentes tecnologias de componentes de software. Dentre as tecnologias em análise se destacam: agentes de software, web-services, componentes orientados a objetos, CORBA e bibliotecas de funções.

Após elaborar, desenvolver e avaliar estes experimentos será possível consolidar os enfoques e conceituar de melhor forma a proposta de Software Devices.

## 5 Trabalhos Futuros

Como trabalho futuro se prevê três etapas necessárias para a plena utilização desta abordagem que estarão descritas abaixo.

Na primeira etapa será formalmente definido o conceito de *Software Devices* com o objetivo de

poder reutilizar esta definição sem ambigüidade. Além disto, ao se utilizar fundamentos e técnicas formais na definição desta abordagem será possível efetuar validações e verificações em especificações geradas a partir dela.

Com relação à segunda etapa, além do processo para a geração de *Software Devices*, serão estudadas e avaliadas formas e linguagens para a representação de contratos de componentes de software. Estas linguagens, em conjunto com o modelo de coordenação de componentes, será a base para este processo de geração de aplicações.

O processo de geração de *Software Devices* consiste no desenvolvimento incremental de um mapeamento entre a linguagem de definição de cenários de reutilização e o framework para a coordenação de componentes.

Após este mapeamento estar amadurecido será possível disponibilizar um gerador de Software Devices. Este receberá como entrada uma instância de um contrato, especificado na linguagem utilizada pelo mapeamento, e terá como saída o código fonte a ser compilado e executado, sendo que este código comporá a chamada máquina de execução de processos.

O objetivo da terceira etapa é estudar e evoluir mecanismos de coordenação de componentes que estarão associados aos *Software Devices* gerados. Entende-se por mecanismos propostas de frameworks para a coordenação de componentes.

## 6. Referências

[Ciancarini] P. Ciancarini. Coordination models and languages as software integrators. ACM Computing Surveys, 28(2), 1996.

[Fiadeiro e Lopes] L. Fiadeiro and A. Lopes. Semantics of architectural connectors. In Proceedings of TAPSOFT'97, 505-519p. Springer-Verlag, 1997.

[Gelernter e Carriero] D. Gelernter e N. Carriero. Coordination languages and their significance. Communications of the ACM, 35(3):97-107, 1992.

[Malone e Crowstone] T. Malone e K. Crowstone. The interdisciplinary study of coordination. ACM Computing Surveys, 26(1):87-119, 1994.

[Papadopoulos e Arbab]G.A. Papadopoulos e F. Arbab. Coordination models and languages. Advances in Computers, 46: The Engineering of Large Systems, 1998.

[Sametinger]J. Sametinger. Software Engineering with Reusable Components.Springer-Verlag, Berlim, 1997.

[Szyperski] C.Szyperski. Component Software: Beyond Object-Oriented Programming. Addison-Wesley, New York, 1999.

# Suporte a Evolução de FrameWorks

Mariela Inés Cortés[1], Carlos José Pereira de Lucena[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{mariela, lucena}@inf.puc-rio.br

## 1. Introdução

Os altos custos de desenvolvimento motivam o reuso e evolução de software existente. Frameworks OO constituem uma das mais promissoras técnicas para a reutilização de código e design [1].

Um framework é um design abstrato que define uma arquitetura reusável para um determinado domínio de problemas. Esta arquitetura é definida em termos de colaborações entre classes e pontos de variação através dos quais o framework pode ser estendido para a criação de diversas aplicações.

Frameworks maduros são usualmente o resultado de muitas iterações e muito trabalho envolvendo mudanças estruturais. O problema mais complexo em relação a evolução e desenvolvimento iterativo de frameworks é o impacto que mudanças no design podem causar sobre o resto do sistema, e.g., possíveis incompatibilidades com instanciações previas. O problema, é referenciado como divergência arquitetural [2], e ocorre quando o framework não suporta uma customização e os usuários precisam violar sua estrutura.

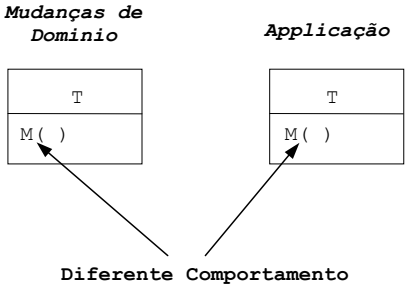



Figure 1. Divergência arquitetural

A Figura 1 ilustra o problema da divergência arquitetural, onde mudanças de dominio representam os novos requisitos: no exemplo, a semântica do método M() difere da semântica previamente estabelecida pelo framework. Como resultado, a aplicação tende a divergir da arquitetura original do framework.

Outros problemas em relação a evolução de frameworks [2] são sumarizados a seguir:

- Complexidade estrutural: a evolução do framework pode fazer a estrutura difícil de manipular e entender.
- Mudanças de dominio: assim como o framework evolui e novas instâncias do framework são criadas, novas abstrações podem ser derivadas como parte do framework.
- Novos insights de design: o design do framework pode precisar de melhorias para dar suporte às características não consideradas ou esquecidas na fase inicial do projeto.

## 2. Solução Proposta

Neste trabalho de tese é proposta a utilização de duas abordagens complementares: refactorings [3, 4] e regras de evolução [5], para assistir na manutenção e evolução de frameworks.

Refactorings são mudanças na estrutura interna do software para melhorar o preservando a semântica original. Exemplos de refactorings são mover o comportamento comum à superclasse, renomear métodos, etc.

Regras de evolução são refactorings especiais que não só preservam, mas também estendem a semântica do framework com novas abstrações. Estas regras podem ser usadas para evitar o problema de divergência arquitetural através da restruturação dos pontos de variação do framework.

### 2.1. Regras de Evolução

As regras de evolução automatizam a incorporação dos metapatterns propostos por Pree [6]. O termo metapattern é usado para um conjunto de design patterns que descrevem como constuir frameworks de maneira independente a um dominio específico.

Metapatterns definem as formas possíveis de implementar pontos de variação como combinação de metodos templates e hooks [7, 8]. Um método template fornece um esqueleto de comportamento, enquanto que um método hook é chamado pelo método template e pode ser sobrescrito para fornecer diferentes comportamentos. Actualmente existem quatro regras de evolução:

- Add Hook Pattern Rule é usada para incorporar um método hook no design.
- Add Unification Pattern Rule é usada para incorporar o metapattern unification no design. O metapattern ocorre quando um objeto da classe template referencia objetos da classe hook (classe template e hook unificadas na mesma classe).
- Add Separation Pattern Rule é usada para incorporar o metapattern separation no design. O metapattern ocorre quando um objeto da classe template referencia objetos da classe hook (classe template diferente da classe hook).
- Add Recursive Pattern Rule é usada para incorporar o metapattern recursion no design. O metapattern ocorre quando o objeto da classe template referencia objetos da classe hook (classe template descendente da classe hook).

## 2.2. Formalização e Verificação

O objetivo desta etapa é calcular a semântica do projeto e verificar que as transformações que implementam os processos evolutivos (refactorings e regras de evolução) são válidas, garantindo que a semântica do projeto não é alterada.

## 3. Ferramenta

O presente trabalho propõe o desenvolvimento de uma ferramenta para dar suporte computacional à abordagem proposta.

O protótipo da ferramenta de suporte a regras de evolução está sendo implementado como extensão da ferramenta JRefactory [9]. Trata-se de uma ferramenta "open source", 100 % java, que suporta a aplicação de refactorings através de uma interface baseada em diagramas UML para a visualização das classes java envolvidas.

Outra funcionalidade considerada é a atualização dos cookbook recipes uma vez que novos pontos de flexibilização podem ser gerados como resultado dos processos evolutivos e precisam ser descritos.

## 4. Contribuições

As principais contribuições desta pesquisa são:

- Identificar o conjunto de transformações necessárias para dar suporte a evolução de frameworks orientadas a objetos.
- Definir em forma detalhada os processos evolutivos envolvidos.
- Especificação formal da semântica do framework e dos processos evolutivos.
- Geração de uma ferramenta semi-automática de suporte a refactorings e regras de evolução.

## 5. Referências

1. Fayad, M.E. Application Frameworks. In Building Application Frameworks, John Wiley, New York, NY, 1999.
2. Codenie W., Hondt K., Steyaert P., and Vercammen A., From Custom Applications to Domain-Specific Frameworks, Communications of the ACM, 40(10), 71-77, 1997.
3. Opdyke W., Refactoring Object-Oriented Frameworks, Ph.D. Dissertation, Computer Science Department, University of Illinois, Urbana-Champaign, 1992.
4. Fowler M., Refactoring: Improving the design of existing code, Addison-Wesley, 1999.
5. Fontoura M., Crespo S., Lucena C., Alencar P., Cowan D., Using Viewpoints to Derive Object-Oriented Frameworks, Journal of Systems and Software, Elsevier Science, 54(3), 239-257, 2000.
6. Pree W., Design Patterns for Object-Oriented Software Development, Addison-Wesley, 1995.
7. Gamma E., Helm R., Johnson R., and Vlissides J., Design patterns – Microarchitectures for reusable object-oriented software. Reading, Massachusetts, Addison-Wesley, 1994.
8. Pree W., OO versus conventional construction of user interface prototyping tools. Ph thesis, University of Linz, 1991.
9. JRefactory. http://jrefactory.sourceforce.net/

# Um Framework para provisão de QoS em Redes Móveis Sem Fio

Luciana dos Santos Lima[1], Luiz Fernando Gomes Soares[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{lslima, lfgs}@inf.puc-rio.br

## 1. Introdução

Nas últimas décadas, tem-se observado um crescente interesse nas tecnologias relacionadas a ambientes de comunicação móvel sem fio. Em grande parte, esse interesse vem acompanhando o crescimento do mercado de telecomunicações, mais especificamente dos sistemas de telefonia móvel celular. Os usuários vêm absorvendo rapidamente essas tecnologias, originando novas necessidades, como a utilização de serviços de dados multimídia, que exige garantias de qualidade.

Este trabalho propõe uma arquitetura para provisão de qualidade de serviço (QoS) fim-a-fim em redes móveis sem fio. Para alcançar esse objetivo, são propostas modificações aos *frameworks para provisão de QoS em ambientes genéricos de processamento e comunicação* [Gomes 1999], visando atender às necessidades trazidas pelas redes móveis, gerando uma arquitetura adaptável para fornecer QoS em ambientes que ofereçam suporte à mobilidade. Uma instanciação dos *frameworks para provisão de QoS em redes móveis sem fio* é descrita através de um cenário de mobilidade, no qual é simulada uma rede infra-estruturada de serviços integrados funcionando sobre o IP Móvel, com o auxilio da ferramenta MobiCS [Rocha & Endler 2000], de modo a validar a proposta.

## 2. O Modelo Genérico de Provisão de QoS

Ao longo dos anos, tem-se testemunhado um aumento crescente na demanda por aplicações distribuídas e multimídia. Essas aplicações são extremamente sensíveis ao retardo e necessitam de algum mecanismo adicional para garantir o seu funcionamento de modo que os usuários fiquem satisfeitos com a qualidade dos serviços oferecidos. Esse cenário tem sido beneficiado por um crescente aumento nas taxas de transmissão, tornando possível se falar mais amplamente em qualidade de serviço. Tradicionalmente, as redes foram projetadas para oferecer apenas o serviço de melhor esforço (*best-effort*), que trata os pacotes de informação individualmente, esforçando-se por entregá-los ao seu destino mas sem oferecer nenhuma garantia.

Qualidade de serviço é fundamental para diversos tipos de aplicações. A definição de *QoS* tem um aspecto subjetivo, podendo variar de aplicação para aplicação de acordo com as características desejadas para o serviço oferecido pela rede. Porém, de alguma forma, esta subjetividade inicial deve ser traduzida em parâmetros que serão negociados entre as aplicações e a rede.

Muitas propostas têm sido apresentadas com a finalidade de modelar os mecanismos envolvidos na provisão da QoS. Com esse mesmo objetivo, alguns trabalhos foram desenvolvidos no Laboratório TeleMídia [Gomes 1999; Gomes et al. 2001] definindo uma arquitetura para a provisão de qualidade de serviço em ambientes genéricos de processamento e comunicação. Segundo Gomes [1999], o modelo genérico de operação de um sistema que ofereça suporte à provisão de QoS pode ser dividido nas seguintes fases: (i) iniciação do sistema, (ii) requisição de serviços, (iii) estabelecimento de contratos de serviço e (iv) a manutenção desses contratos.

## 3. Avaliando o Modelo Genérico para as Redes Móveis Sem Fio

Como foi apresentado em [Lima 2002] os *frameworks genéricos* não contemplam questões específicas referentes às redes móveis sem fio, como (i) a influência da mobilidade dos usuários, que envolve questões como a predição da mobilidade [Santos 2000], relacionando os padrões de deslocamento ao comportamento do usuário [Chan & Seneviratne 1999] e, como conseqüência desse deslocamento, (ii) o estabelecimento de reservas antecipadas [Wolf & Steinmetz 1997].

Algumas características inerentes aos ambientes móveis foram consideradas de modo que os *Frameworks para Provisão de QoS em Redes Móveis Sem Fio* pudessem ser especificados. Como resultado, este trabalho propõe a especialização dos *frameworks* genéricos [Gomes 1999] para as redes móveis sem fio, adicionando o módulo de gerenciamento de mobilidade e aplicando, conceitos complementares, como reservas antecipadas, intervalos de QoS e políticas de controle. Para que esses objetivos fossem alcançados, duas etapas distintas foram executadas, a saber:

- A introdução do *framework para gerenciamento de mobilidade* ao conjunto de *frameworks* que compõem o modelo genérico, com a finalidade de manter os históricos de deslocamento dos usuários e realizar a predição da mobilidade, especificando os caminhos através dos quais as reservas antecipadas de recursos serão estabelecidas; e
- O preenchimento de pontos de flexibilização definidos no modelo genérico, cobrindo questões referentes à mobilidade do usuário, tais como:
  - A pré-alocação de recursos, de modo que as reservas possam ser efetuadas não somente através do ponto de localização atual do nó móvel, mas também de todos os caminhos que possam ser visitados durante os deslocamentos do nó;
  - Os intervalos de tolerância de QoS; e
  - As políticas de controle, que irão definir as permissões de acesso dos usuários em diferentes domínios administrativos.

## 4. Conclusões

O desenvolvimento dos *Frameworks para Provisão de QoS em Redes Móveis Sem Fio* permitiu um estudo aprofundado dos ambientes que oferecem suporte à mobilidade, possibilitando o fornecimento de serviços com qualidade, levando em consideração as peculiaridades desses ambientes, como as mudanças dinâmicas da topologia da rede, o deslocamento dos nós móveis entre áreas de cobertura vizinhas, o desvanecimento das conexões e as flutuações na disponibilidade de recursos, entre outros.

As principais contribuições obtidas com a conclusão deste trabalho foram a introdução dos conceitos de reservas de recursos antecipadas e intervalos de QoS, tanto para redes móveis quanto para redes fixas, e a definição de um *framework para gerenciamento de mobilidade*, permitindo a especificação do grafo de prováveis provedores para os quais o nó móvel pode migrar em função do tempo. Esse grafo* contém todos os caminhos através dos quais as reservas serão efetuadas, quer sejam reservas imediatas ou reservas antecipadas. Os trabalhos que se propõem a oferecer QoS aos serviços disponibilizados em redes móveis não incorporam o mecanismo de gerenciamento de mobilidade, fundamental para o estabelecimento de reservas antecipadas. Este trabalho acopla o módulo de gerenciamento de mobilidade à arquitetura de provisão de QoS proposta por Gomes [1999], promovendo uma cooperação entre os mecanismos de predição de deslocamento e de reserva de recursos.

* Quando se trata de redes *ad hoc*, esse grafo tem a sua representação reduzida, pois a definição de todas as possíveis topologias dessas redes é um problema NP-completo. São utilizadas heurísticas para se obter um grafo com um menor detalhamento, definindo rotas alternativas e/ou redundantes.

## 5. Referências

CHAN, J.; SENEVIRATNE, A. *A Practical User Mobility Prediction Algorithm for Supporting Adaptive QoS in Wireless Networks*. In: IEEE International Conference on Networks, 1999.

GOMES, A.T.A. *Um Framework para Provisão de QoS em Ambientes Genéricos de Processamento e Comunicação*. Dissertação de Mestrado. Departamento de Informática. PUC-Rio, Maio de 1999.

GOMES, A.T.A.; COLCHER, S.; SOARES, L.F.G. *Modeling QoS Provision on Adaptable Communication Environments*. In: IEEE International Conference on Communications, Finlândia, Junho de 2001.

LIMA, L. dos Santos. *Um Framework para Provisão de QoS em Redes Móveis Sem Fio.* Dissertação de Mestrado. Departamento de Informática. PUC-Rio, Agosto de 2002. SANTOS, V. de Lima. *Qualidade de Serviço em Redes Móveis Sem Fio*. Dissertação de Mestrado. Departamento de Ciência da Computação, UFMG, Março de 2000.

WOLF, L.C.; STEINMETZ, R. *Concepts for Resource Reservation in Advance*. In: Journal of Multimedia Tools and Applications. Vol. 04, No. 3, Maio de 1997.

ROCHA, R.C. A. da; ENDLER, M. *Um Simulador de Protocolos Distribuídos para Computação Móvel* (2º. Workshop de Comunicação sem Fio-WCSF), pp. 33-48, DCC/ICEx/UFMG, Belo Horizonte, Maio de 2000.

# Um Framework para Provisão de QoS em Sistemas Operacionais

Marcelo Ferreira Moreno[1], Luiz Fernando Gomes Soares[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{moreno, lfgs}@inf.puc-rio.br

## 1. Introdução

A progressiva demanda por aplicações multimídia distribuídas, caracterizadas por fortes exigências sobre os recursos computacionais, torna evidente a necessidade de provisão de qualidade de serviço (QoS) em cada um dos subsistemas envolvidos, como redes de comunicação e sistemas operacionais. Ao mesmo tempo, tais subsistemas devem ser flexíveis para que possam oferecer novos serviços a aplicações futuras, ou seja, devem ser adaptáveis em tempo de execução.

Especificamente, em sistemas operacionais multimídia, é necessário garantir ao fluxo de dados, alvo principal da provisão de QoS, que suas necessidades de utilização dos recursos de comunicação sejam respeitadas ao longo do caminho entre aplicação e rede. Em várias etapas desse caminho, porém, é observada uma dependência entre o fluxo de dados e a capacidade de processamento de toda a pilha de protocolos, além da aplicação. Portanto, o processamento do fluxo de instruções pela CPU deve ter garantias análogas àquelas dadas ao fluxo de dados, de forma que a interferência de um fluxo sobre o outro seja contabilizada para a gerência e controle da QoS como um todo.

Sistemas operacionais de uso geral, contudo, provêm pouco ou nenhum suporte a QoS e à adaptabilidade dos serviços, impulsionando muitas pesquisas em tais áreas. Observando-se algumas dessas tecnologias, nota-se que os mecanismos de provisão possuem várias semelhanças funcionais. Dessa forma, este trabalho propõe uma arquitetura adaptável para a provisão de QoS nos subsistemas de rede e de escalonamento de processos de sistemas operacionais, independente de implementação, através da descrição de frameworks genéricos.

## 2. Descrição da Arquitetura

A arquitetura QoSoS, como foi denominada, foi definida a partir da especialização dos frameworks genéricos descritos em [Gomes 1999], acrescidos de algumas novas estruturas aqui introduzidas. A Figura 1 mostra como os tipos de hot-spots descritos podem ser completados para a construção de uma arquitetura de provisão de QoS em sistemas operacionais. Os frameworks genéricos definem as estruturas para a provisão de QoS, as quais são comuns aos vários subsistemas que participam do fornecimento do serviço fim-a-fim. A primeira etapa de especialização é feita para que sejam incluídas funcionalidades específicas de sistemas operacionais, como os mecanismos pertinentes aos subsistemas de escalonamento de processos e de comunicação em rede. A etapa seguinte de particularização define aspectos relacionados à provisão do serviço, como o conjunto de políticas de QoS que cada um dos subsistemas disponibilizará a seus usuários.

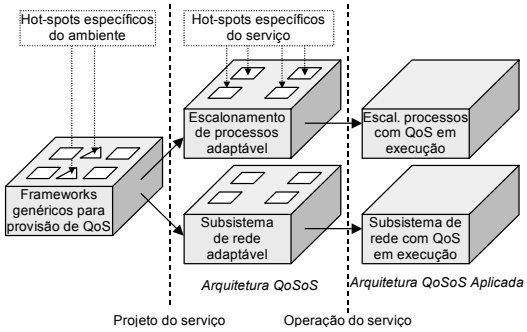



Figura 1 - Tipos de hot-spots de uma arquitetura modelada pelos frameworks genéricos

Os frameworks foram subdivididos em quatro conjuntos, de acordo com a funcionalidade que representam na arquitetura. São eles: Parametrização de Serviços; Compartilhamento de Recursos; Orquestração de Recursos; Adaptação de Serviços. Tais frameworks encontram-se documentados de forma completa em [Moreno 2002].

O framework para parametrização de serviços modela uma estrutura responsável por definir um esquema de parâmetros de caracterização de serviços, de forma genérica, independente dos possíveis serviços a serem oferecidos pelo sistema operacional. Os frameworks para compartilhamento de recursos se baseiam no conceito de recurso virtual para modelar os mecanismos de escalonamento e alocação de recursos. Recursos virtuais são parcelas de utilização de um ou mais recursos reais distribuídas entre os fluxos submetidos pelos usuários.

Na área de atuação dos sistemas operacionais, vários são os recursos que devem ter seus mecanismos de escalonamento e de alocação gerenciados de forma integrada, de modo a viabilizar a orquestração dos recursos no ambiente como um todo. Na arquitetura QoSoS, a modelagem da orquestração de recursos é apresentada pela especialização de dois frameworks distintos: o framework para negociação de QoS e o framework para sintonização de QoS.

O framework para negociação de QoS (Figura 2) modela os mecanismos de negociação e mapeamento que operam durante as fases de solicitação e estabelecimento de serviços, além dos mecanismos de admissão que atuam somente na fase de estabelecimento. Já o framework para sintonização de QoS modela os mecanismos de sintonização e monitoração que atuam na fase de manutenção do serviço.

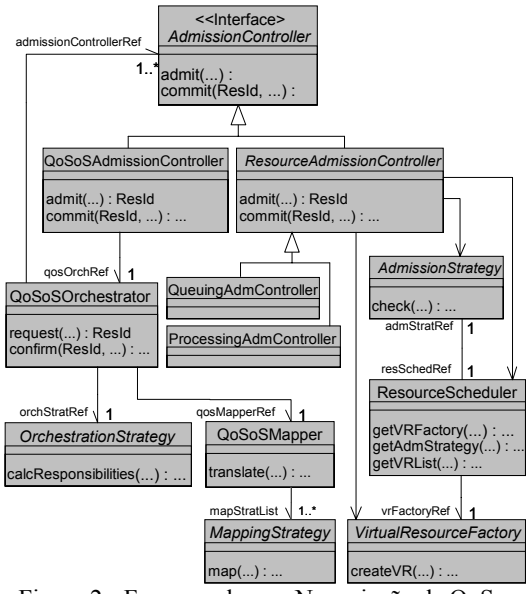


Figura 2 - Framework para Negociação de QoS

A implementação de "meta-mecanismos" que automatizem a adaptação do sistema a novos serviços ou a novas políticas de provisão de QoS, e que observem questões como manutenção de consistência e restrições de reconfiguração relacionadas a segurança, é altamente desejável. O framework para adaptação de serviços (Figura 3) foi elaborado neste trabalho para preencher parte dessa lacuna deixada pelos frameworks genéricos para provisão de QoS, em uma abordagem específica para sistemas operacionais.

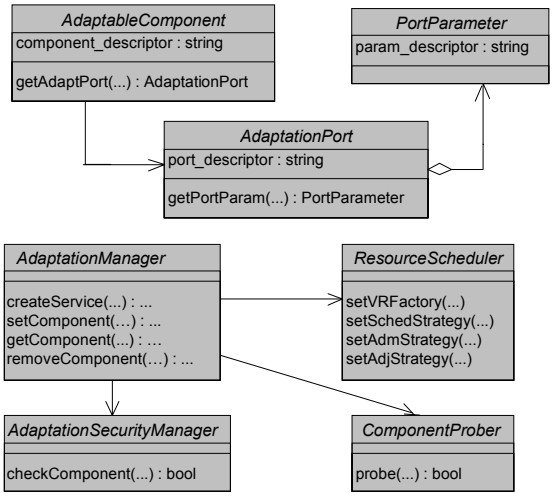


Figura 3 - Framework para Adaptação de Serviços

## 3. Conclusões

O emprego de frameworks como ferramenta de modelagem facilitou a identificação dos pontos de flexibilização (hot-spots) que permitem a instanciação da arquitetura para a implementação de vários cenários reais. Os exemplos de aplicação da arquitetura foram apresentados em [Moreno 2002] considerando a implementação de um subsistema de escalonamento de processos hierarquizado regido por um meta-algoritmo capaz de emular o comportamento de diversos algoritmos de escalonamento. Por fim, foi implementado um cenário real de uso da arquitetura, no qual o sistema operacional Linux sofreu pequenas alterações para oferecer uma infra-estrutura adaptável de suporte a QoS sobre os buffers de comunicação de estações finais e roteadores.

## 4. Referências

GOMES, Antônio T. Um framework para provisão de QoS em ambientes genéricos de processamento e comunicação. Dissertação de Mestrado, Pontifícia Universidade Católica do Rio de Janeiro, 1999.

MORENO, Marcelo F. Um framework para provisão de QoS em sistemas operacionais. Dissertação de Mestrado, Pontifícia Universidade Católica do Rio de Janeiro, 2002.

# Um Processo Unificado para Projeto de Ontologias e Bases de Conhecimento

Daniel Abadi Orlean[1], Carlos José Pereira de Lucena[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{orlean, lucena}@inf.puc-rio.br

## 1. O Crescimento da Web

Concebida com base em uma série de princípios de projeto como simplicidade de protocolos e interoperabilidade, a Web atingiu níveis espetaculares tanto com relação a quantidade de usuários quanto a disponibilidade de documentos. No entanto, há um preço elevado a ser pago por esse sucesso. A crescente disponibilidade de recursos e fontes de informação vem dificultando a busca, o acesso, a apresentação e a manutenção das informações requeridas pelos usuários [Ding et al., 2002]. Neste contexto, as tarefas mais complexas de acesso, extração, interpretação e manutenção são deixadas para o ser humano [Madche, 2002], uma vez que a interoperabilidade de dados e informações ainda é majoritariamente restrita ao nível sintático, em detrimento do semântico. Entenda-se aqui como semântica a capacidade de se processar e interpretar algum tipo de informação computacionalmente [Uschold, 2001], e não apenas através da mente humana.

## 2. Web Semântica

A segunda geração da Web, uma visão de Tim Berners-Lee [Berners-Lee et al.,2001] que já está se tornando realidade, vem com o objetivo de solucionar este problema. A Web Semântica, como é chamada, tem como característica principal dotar os documentos ora disponibilizados na Web com informações que possam ser processadas semanticamente, através do oferecimento de serviços automáticos [Ding et al., 2002], como agentes de software, máquinas de busca inteligentes e *web services,* por exemplo. Não só a apresentação do conteúdo, mas também a automação, integração e reuso dos dados devem ser possibilitados através de diversas aplicações [W3C 2001].

## 3. Ontologias

No entanto, tornar os documentos disponibilizados na Web semanticamente processáveis não é uma tarefa imediata. Infelizmente (ou felizmente!), os computadores não são capazes de desenvolver, por livre e espontânea vontade uma linguagem consensual de comunicação. É nesta etapa que entram em cena as ontologias. Conhecida no ramo da filosofia como uma "teoria sobre a natureza da existência", as ontologias têm sido encaradas de maneira consideravelmente diferente no universo computacional. Segundo [Gruber, 1993], as ontologias representam um entendimento comum e compartilhado sobre um domínio específico, e têm como o objetivo principal permitir a comunicação entre pessoas e aplicações.

Ontologias consistem tipicamente em definições de conceitos, suas relações e axiomas [Staab, Schnurr, Studer, Sure, 2000]. As relações entre os conceitos da ontologia podem ser tanto taxonômicas (*"um professor é um [tipo de] funcionário"*) como não taxonômicas (*"um professor orienta um aluno*"), o que permite um mapeamento bastante próximo da realidade do domínio em questão, e, conseqüentemente, uma linguagem mais adequada para a comunicação e para seu entendimento.

## 4. Metodologias para o Desenvolvimento de Ontologias

Como a própria definição adotada denota, uma ontologia deve ser capaz de capturar o conhecimento de um domínio de uma forma genérica e de promover um entendimento compartilhado, o que tipicamente exige o comprometimento de grupos de pessoas [Sure et al., 2002] e o uso de metodologias, processos e/ou métodos bem definidos. O projeto e desenvolvimento de ontologias vêm deixando de ser uma arte para se transformar em uma ciência.

Diversas propostas nesse sentido já foram apresentadas ([Uschold and King, 1995], [Uschold and Gruninger, 1996], [Guarino and Welty, 2000], [Staab, Schnurr, Studer, Sure, 2000], [Fernandez, Gomez-Perez, Juristo, 1997]), apoiadas por estudos de caso acadêmicos e industriais. No entanto, é importante notar que nenhuma dessas metodologias – que em muitos casos resumem-se apenas a heurísticas extraídas da experiência de seus autores ou a orientações sobre como alcançar bons resultados – atende por completo os requisitos potenciais do projeto de uma ontologia.

Por ter como um de seus objetivos permitir o processamento semântico de informações, uma ontologia deve ter um grau de formalismo que permita com que esta seja computada. Deve, além disso, representar a teoria compartilhada e consensual sobre um determinado domínio,

permitindo a comunicação entre pessoas e aplicações, levando em consideração questões antagônicas como complexidade *(toda a informação necessária – hoje ou em um futuro próximo - deve estar disponível)* versus facilidade de implementação, uso e manutenção *(nenhuma informação desnecessária deve estar disponível)*. Desta forma, uma ontologia pode apresentar diferentes graus de abstração e detalhamento que permitam o cumprimento dos diferentes objetivos para os quais está sendo desenvolvida, seja permitir a comunicação entre agentes de software ou mesmo a geração de um portal de conhecimento dotado com informações semânticas sobre seu conteúdo. Por fim, deve levar em conta as evoluções às quais uma ontologia, por ser a representação de um domínio do mundo real, está propensa a sofrer com o passar tempo, exigindo sempre seu refinamento e manutenção.

## 5. Um Processo Unificado para Projeto de Ontologias e Bases de Conhecimento

No entanto, o que se observa nas metodologias e processos existentes é que geralmente é dado um foco maior em uma dessas características. Enquanto algumas possuem um processo bem especificado de levantamento de requisitos e análise do domínio, outras focam em questões referentes ao formalismo da ontologia, em sua manutenção ou na alimentação de sua base de conhecimento.

A base de conhecimento é, diferentemente da ontologia, representada pelo conhecimento específico do domínio e não a sua estrutura. Ou seja, a base de conhecimento é formada pela população de instâncias que podem ser classificadas pelos conceitos da ontologia, bem como a ocorrência de relações entre tais instâncias. Alguns autores consideram que esta separação não é tão clara assim, e consideram outras questões para diferenciação entre o que faz parte da ontologia e o que compõe a base de conhecimento como em [Maedche et al., 2000].

O que se propõe neste trabalho é a unificação de diversos métodos oriundos de metodologias distintas em um processo abrangente, que permita o projeto e desenvolvimento de ontologias e bases de conhecimento. Entende-se como um processo um conjunto de atividades e resultados associados a essas atividades com o objetivo de garantir a geração de um produto final [Sommerville, 1998], seja este produto um software, uma ontologia ou uma ontologia associada a sua base de conhecimento.

Resumidamente, este processo deverá ser composto pelas seguintes atividades:

1. Identificação do problema, Levantamento de Requisitos e Levantamento de Fontes de Informação
2. Identificação de Conceitos, Relações Taxonômicas e Não-taxonômicas
3. Formalização da Ontologia – Definição de Axiomas
4. Avaliação e Refinamento
5. Modelagem da Visão de Inferências
6. Modelagem da Visão de Navegação
7. Implementação da ontologia
8. Implementação da Aplicação Hipermídia
9. Alimentação e Manutenção

## 6. Bases de conhecimento: Consumo humano e automático

Todas essas fases, com exceção de (6) e (8), já são abordadas ou identificadas de alguma forma pelas metodologias estudadas. O processo unificado deverá integrar e aprimorar essas abordagens, agregando novas técnicas e resultados sempre que se mostrar necessário. Há, por exemplo, uma necessidade premente de se desenvolver ontologias que dêem suporte tanto ao processamento automático de informações, focando nos agentes de software e aplicações, quanto para o consumo-humano, através da navegação dos usuários na base de conhecimento (visão de navegação).

As estratégias implementadas pelas abordagens estudadas para navegação na ontologia e na base de conhecimento consideram apenas a estrutura taxonômica da ontologia, permitindo que os usuários naveguem em profundidade pelos conceitos e instâncias. Este tipo de navegação traz uma série de desvantagens para o usuário, pois é orientada pela estrutura da ontologia, e não pelas tarefas para as quais ela foi projetada. Não se tem notícia, portanto, de nenhuma metodologia ou processo de engenharia de ontologias que aborde o consumo humano da ontologia e da base de conhecimento de forma satisfatória.

Existem, no entanto, diversos métodos para projeto e desenvolvimento de aplicações hipermídia [Orlean et al, 2001] abordando o assunto sob perspectivas distintas.

Neste caso em especial, o foco principal deve ser dado na modelagem navegacional, considerando-se os elementos que compõem a navegação como visões dos conceitos da ontologia e suas relações. Esta noção é inspirada no método OOHDM [Schwabe and Rossi, 1998], o qual terá diversos elementos incorporados neste processo unificado para permitir, também, a geração de aplicações hipermídia a partir das ontologias desenvolvidas.

Estas aplicações deverão ter como estrutura navegacional uma visão da estrutura conceitual da ontologia orientada a resolução das tarefas. Essas tarefas representam que tipos de informação o usuário humano será capaz de extrair através da navegação entre os conceitos da ontologia e suas instâncias na base de conhecimento e que tenham sido projetadas para o usuário através das chamadas questões de competência – ou seja, o que a ontologia deve ser capaz de responder.

## 7. Estudos de Caso

Todo processo de desenvolvimento precisa ser testado e avaliado por uma quantidade considerável e representativa de estudos de caso para ter sua eficácia e qualidade comprovadas. Estes deverão ser escolhidos de forma a oferecer subsídios para a identificação de pontos fortes e fracos em cada atividade e nos resultados produzidos, induzindo a uma melhora contínua do processo. A ontologia e a base de conhecimento do Grupo de Pesquisa do TecComm, ligado ao Laboratório de Engenharia de Software, está sendo projetada e já está permitindo várias adaptações neste processo.

## 8. O Framework Portalware

Para dar suporte a essa metodologia foi desenvolvido um Framework que facilita a geração de portais semânticos de conhecimento projetados segundo o processo descrito. Este Framework contempla principalmente as fases 7, 8 e 9 do processo unificado, permitindo, através da definição de uma ontologia, a geração de toda a infraestrutura da base de conhecimento e de serviços de alimentação da base, além dos mecanismos de navegação e consulta a informações [Orlean et al., 2002].

## 9. Referências

[Berners-Lee et al.,2001] T. Berners-Lee, J. Hendler, O. Lassila: The Semantic Web, Scientific American, Maio de 2001

[Gruber, 1993] T. R. Gruber: A Translation Approach to Portable Ontology Specifications, Knowledge Acquisition 5, 1993

[Ding et al., 2002] Y. Ding, D. Fensel, M. Klein, B. Omelayenko: The Semantic Web: Yet Another Hip?, Data and Knowledge Engineering, 2002

[Schwabe and Rossi, 1998] D. Schwabe, G. Rossi: An Object Oriented Approach to Web-Based Application Design, Theory and Practice of Object Systems 4(4), 1998

[Orlean et al, 2001], D. Orlean, F. Ferreira, C. Lucena : EveryWare: Dealing with E-Commerce Pervasiveness, International Conference on Internet Computing, 2001

[W3C, 2001] W3C: Semantic Web Activity Statement, 2001

[Sure et al., 2002] Y. Sure, M. Erdmann, J. Angele, S. Staab. R. Studer, D. Wenke: OntoEdit: Collaborative Ontology Development for the Semantic Web, 2002

[Uschold and King, 1995] M. Uschold, M. King: "Towards a Methodology for Building Ontologies, IJCAI, 1995

[Uschold and Gruninger, 1996] M. Uschold, M. Gruninger: Ontologies: Principles, methods and applications, Knowledge sharing and Review, 1996

[Guarino and Welty, 2000] N. Guarino and C. Welty: Indentity, unity, and individuality: towards a formal toolkit for ontological analysis, ECAI, 2000

[Staab, Schnurr, Studer, Sure, 2000] S. Staab, H. Schnurr, R. Studer, Y. Sure: Knowledge Processes and Ontologies, 2000

[Fernandez, Gomez-Perez, Juristo, 1997] M. Fernandez, A. Gomez-Perez, N. Juristo: Methontology: From Ontological Art Towards Ontological Engineering, AAAI-Spring Symposium on Ontological Engieering, 1997

[Uschold, 2001] M. Uschold: Where are the Semantics in the Semantic Web?, Autonomous Agents Conference, 2001

[Maedche et al., 2000] A. Maedche, S. Staab, N. Stojanovic, R. Studer, Y. Sure: SEAL – A Framework for Developing Semantic Web Portals, 2000

[Orlean et al., 2002] D. Orlean, C. Rocha, F. Ferreira, C. Lucena – A Framework for Ontologies and Knowledge Bases Generation

[Sommerville, 1998] Ian Sommerville: Software Engineering, Addison-Wesley

# Uma Abordagem Baseada em *Frameworks* para a Construção e Gerenciamento de Teste de Software em Sistemas Multi-Agentes

Aluízio Haendchen Filho[1], Arndt von Staa[1], Carlos José Pereira de Lucena[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{aluizio, arndt, lucena}@inf.puc-rio.br

## 1. Problemática e objetivos

Questões relacionadas ao desenvolvimento de Sistemas Multi-Agentes estão entre os mais importantes desafios que dirigirão as pesquisas na área de Engenharia de Software nos próximos anos [Hannoun 2000]. Após o surgimento da *Internet* e com o advento de tecnologias baseadas na *Web*, tais como *e-commerce*, o domínio de aplicação de sistemas multi-agentes está se expandindo e atualmente é utilizado em larga escala em aplicações *Web* e em várias outras plataformas de aplicações de negócios e de sistemas de informação.

Uma importante questão que surge neste contexto diz respeito aos aspectos de testabilidade e confiabi-lidade destes sistemas. Para fins de especificação e teste, a complexidade inerente de um agente e suas características implicam em lidar com um nível de detalhamento bastante mais refinado e complexo do que aquele necessário para manipular um objeto. Estas particularidades sugerem a aplicação de funções e técnicas de teste diferentes daquelas utilizadas em testes procedurais e teste de objetos.

O objetivo deste trabalho de pesquisa é descrever uma arquitetura composta por *frameworks* para facilitar a construção e gerenciamento de software para testar agentes. Esta arquitetura permite a um testador adotar de forma prática e segura a verificação sistemática da confiabilidade de sistemas multi-agentes, considerando suas caracte-rísticas e os padrões adotados.

## 2. Abordagem proposta

Nós entendemos um sistema multi-agente como sendo uma 'sociedade artificial' ou organização, caracterizada por uma coleção de papéis que são desempenhados por agentes [Wooldridge 1999, Yu 2000]. Esta coleção de papéis define a estrutura e o comportamento de um grupo de agentes trabalhando para atingir seus objetivos individuais e os objetivos da organização. Um papel define uma função prototípica de um agente, determinando um padrão para o comportamento esperado.

Para compor o contexto e o ambiente de Engenharia de Software, nossa abordagem considera a construção de dois modelos: um modelo de especificação, denominado *Specification Model* e um modelo de teste, denominado *Test Model*.

A Figura 1 mostra a arquitetura genérica do sistema, os relacionamentos e o fluxo de informações existentes entre os principais modelos que compõem o ambiente de Engenharia de Software.

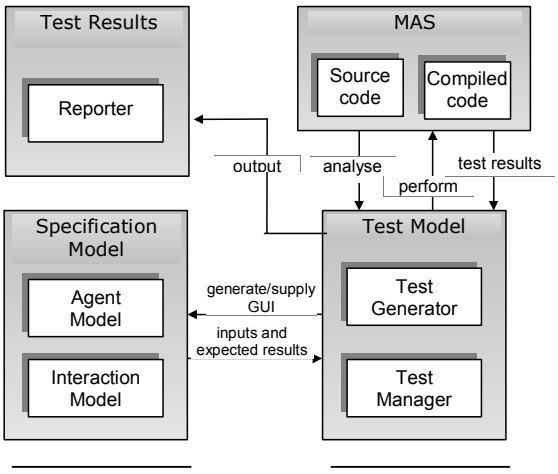



Figura 1 – Arquitetura genérica do sistema

Dada uma organização de agentes (MAS), um conjunto de classes implementadas em *Test Model* analisa o código fonte dos agentes e mapeia o conteúdo para o modelo abstrato de especificação. Um conjunto de interfaces gráficas é gerado de forma automática, e através destas interfaces o testador define os parâmetros (*inputs* e resultados esperados) para os casos de teste. As interfaces facilitam a construção dos casos de teste, minimizando o trabalho de codificação do testador e facilitando a montagem de cenários para teste de unidade e integração.

Os resultados dos casos e suites de teste executados são apresentados sob a forma de laudo (*Reporter*). Ao analisar o laudo, o testador diagnostica e corrige as falhas encontradas. Este

processo iterativo é um importante fator para aprimorar o software produzido.

## 2.1. Specification Model

*Specification Model* é um modelo abstrato que visa capturar as especificações de um agente e de uma organização de agentes. É composto por dois sub-modelos: *Agent Model* e *Interaction Model*. As especificações definidas nestes modelos formam um conjunto completo de informações, que possibilita obter os dados necessários para especificar um sistema multi-agente.

*Agent Model* define as especificações individuais de um agente. Estas especificações são mapeadas para um modelo baseado em papéis, tomando por base modelos definidos em [Hannoun 2000, Yu 1999]. Atributos tais como *goals*, planos, atividades, recursos e relacionamentos são especificados em um processo interativo e iterativo entre o testador e o sistema.

*Interaction Model* [Wooldridge 2000] identifica as dependências e relacionamentos entre os vários agentes em um sistema multi-agente. Este modelo consiste de um conjunto de definições de protocolos, um para cada tipo de interação entre agentes. O padrão de interação é formalmente definido através de uma seqüência de passos a serem executados. Nós utilizamos *workflows* [Baral 2000] e técnicas baseadas em estado para modelar a seqüência de passos, onde as especificações de cada transição de estado são mapeadas para regras ECA.

## 2.2. Test Model

*Test Model* define como automatizar a geração e a execução de casos de teste. O princípio geral é testar se cada transição de especificação atinge seu estado destino, aplicando a entrada e checando se a saída gerada corresponde aos resultados esperados.

O conjunto de casos de teste é formado por testes de unidade, baseados nas especificações definidas no *Agent Model* e testes de integração, baseados nas especificações estabelecidas no *Interaction Model*. O teste de unidade enfatiza a verificação da organização interna dos módulos que constituem o código de um agente. O teste de integração enfatiza o teste de protocolos, a seqüência de execução e a coordenação entre os agentes. A construção dos casos de teste é suportada por um *framework* que contém duas principais superclasses: *Test Generator* e *Test Manager*.

*Test Generator* é composto por um conjunto de classes geradoras que tem por objetivo auxiliar o testador a construir os casos de teste através de técnicas parcial ou integralmente automatizadas. Estas classes geradoras capturam os atributos e relacionamentos dos agentes analisando o código fonte, e disponibilizam interfaces gráficas através das quais o testador pode definir as especificações e *scripts* para os casos de teste.

*Test Manager* tem por função armazenar, gerenciar e prover interfaces para executar os processos de teste. Interfaces auxiliares são geradas automatica-mente sob a forma de árvore, e os casos e suites de teste são armazenados nos nodos da hierarquia. O propósito é auxiliar o testador a selecionar e executar casos e suites de teste, bem como facilitar a adaptação de casos de teste existentes para novos testes ou testes de regressão.

## 3. Conclusão

Este estudo é parte de um trabalho de pesquisa em engenharia de software baseada em agentes que vem sendo desenvolvido pelo grupo de pesquisa SoC + Agentes, na PUC-Rio. O trabalho visa minimizar custos, tempo e esforço associado com o desenvolvimento de técnicas para testar sistemas multi-agentes. Os resultados de nossa pesquisa trazem uma contribuição para preencher uma lacuna existente nas abordagens especialmente projetadas para a construção e gerenciamento de teste em sistemas multi-agentes.

## 4. Referências

[Wooldridge 1999] Wooldridge, M., Jennings N. and Kinny D. The Gaia Methodology for Agent-Oriented Analysis and Design.

[Yu 2000] Yu L, Schmid B.F. A Conceptual Framework for Agent Oriented and Role Based Workflow Modeling.

[Hannoun 2000] Hannoun M., Boisser O. MOISE: An Organizational Model for Multi-Agent Systems.

[Baral 1999] Baral C., Lobo J. Formalizing Workflows as Collections of Condition-Action Rules.

# Uma Abordagem Orientada a Aspectos para o Desenvolvimento de Sistemas Multi-Agentes

Alessandro Fabricio Garcia[1], Carlos José Pereira de Lucena[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)
R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{afgarcia, lucena}@inf.puc-rio.br

## 1. Problema e Objetivos

A tecnologia de agentes tem sido proposta como uma abordagem complementar a tecnologia de objetos para construção de sistemas distribuídos complexos. Entretanto, o desenvolvimento de sistemas multi-agentes não é uma tarefa trivial. Agentes e objetos possuem muitas similaridades uma vez que ambos encapsulam estado e serviços [Garcia 2002a]. Entretanto, agentes de software têm um estado mental – composto de componentes como crenças, objetivos, capacidades e planos – e um conjunto de propriedades comportamentais, tais como interação, adaptação, autonomia, colaboração, aprendizagem e mobilidade (Figura 1).

Em sistemas orientados a objetos, estas propriedades naturalmente espalham-se e replicam-se através dos componentes de uma aplicação multi-agente. Ademais, cada propriedade atua sobre o estado e serviços de um agente, não é ortogonal em relação as outras propriedades, e introduz complexidade adicional ao sistema de software. Consequentemente, tais propriedades dificultam a satisfação dos requisitos de facilidade de compreensão, manutenção e reutilização de sistemas multi-agentes.

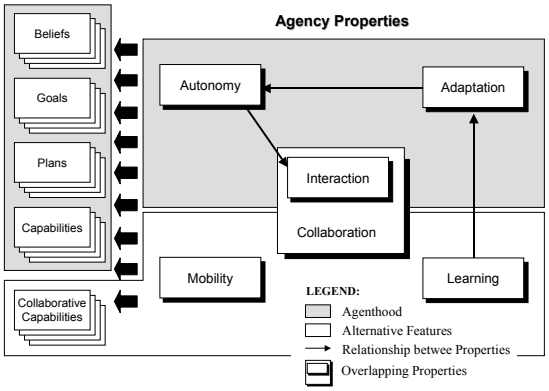
Idealmente, desenvolvedores de sistemas baseados em agentes deveriam aplicar métodos e técnicas especiais de estruturação e formas disciplinadas de identificar e associar as diferentes propriedades dos agentes durante análise, projeto e implementação. De forma geral, são poucas as pesquisas na literatura que propõem tal abordagem sistemática para tratamento das propriedades de agência ao longo do ciclo de vida de software. Nexte contexto, o principal objetivo desta proposta de tese é propor uma abordagem sistemática baseada em aspectos para a análise, projeto e implementação de sistemas multi-agentes.

## 2. Solução Proposta

Engenharia de software orientada a aspectos tem emergido como uma área promissora com o princípio de modularizar *concerns*[*] que normalmente espalham-se e replicam-se pelos vários componentes do sistema ao longo do ciclo de desenvolvimento de software. Nós acreditamos que o princípio de separação de *concerns* é um dos melhores meios de lidar com a complexidade existente na construção de aplicações multi-agentes. Propostas orientadas a agentes existentes somente proporcionam suporte à definição dos serviços e componentes de estado dos agentes, mas não orientam em como lidar e estruturar as propriedades comportamentais dos agentes durante as diferentes fases de desenvolvimento.

**Figura 1** – Uma definição para *agenthood*.

Nossa proposta é baseada na pressuposição de que a estruturação das propriedades dos agentes são a




fonte da complexidade de sistemas multi-agentes e devem ser organizadas desde as primeiras fases de desenvolvimento. Idealmente, as propriedades dos agentes devem ser modularizadas como aspectos de tal forma a lidar com a complexidade de agentes de software, permitindo a produção de aplicações que sejam mais fáceis de manter, entender e reutilizar. Entretanto, embora a identificação, estruturação e composição das propriedades de agência sejam tarefas desejáveis, não são triviais. Neste sentido, nossa abordagem será composta de:

[*] Palavra da língua inglesa que pode ser entendida como conceito, interesse, propriedade, preocupação, etc...

(1) *Fase De Análise*: Extensão de uma metodologia existente para análise orientada a agentes que identifica as propriedades dos agentes de uma aplicação, bem como um conjunto de diretivas que ajuda desenvolvedores na estruturação do sistema multi-agentes;

(2) *Fase de Projeto - Estado e Propriedades*: um conjunto de diretivas, possivelmente baseadas em padrões de software orientados a aspectos, que guiam engenheiros de software a estruturar os serviços, componentes de estado e propriedades dos agentes;

(3) *Fase de Projeto - Composição*: um conjunto de diretivas que auxilia desenvolvedores na composição dos serviços, componentes, propriedades de agentes;

(4) *Fase de Implementação*: um conjunto de diretivas que orienta a implementação do projeto dos agentes utilizando linguagens que proveêm suporte a programação orientada a aspectos.

## 3. O Grupo SoC+Agents

Este trabalho está sendo desenvolvido no contexto do grupo SoC+Agents - Separation of Concerns and Multi-Agent Systems (URL: http://www.teccomm.les.inf.puc-rio.br/socagents/), um dos subgrupos do grupo TecComm/PUC-Rio. Este subgrupo é composto atualmente de cerca de 6 alunos de doutorado e 6 alunos de mestrado, sob a coordenação do Prof. Carlos Lucena. Além da proposta descrita aqui, os componentes estão envolvidos no desenvolvimento de um *framework conceitual* para sistemas multi-agentes, estudo de metodologias para desenvolvimento de software orientado a agentes, implementação de estudos comparativos, bem como experimentos e estudos de casos. Ademais, o grupo TecComm desenvolve uma série de sistemas que fazem uso da tecnologia de agentes, tais como Portalware, Ideal, VBroker, VGroups, CommercePipe, e VMarket. Este cenário contribui decisivamente para o desenvolvimento e validação da proposta de tese apresentada neste documento.

## 4. Resultados Preliminares

A conclusão desde trabalho de tese está prevista para março de 2004. Os artigos apresentados nas referências contém resultados preliminares sobre nossa abordagem proposta.

## 5. Referências

[Garcia 2001a] A. Garcia et al. “An Aspect-Based Approach for Developing Multi-Agent Object-Oriented Systems”. XXI Brazilian Symp. on Software Engineering, Rio de Janeiro, Brazil, October 2001, pp. 177-192.

[Garcia 2001b] A. Garcia et al. “Promoting Advanced Separation of Concerns in Intra-Agent and Inter-Agent Software Engineering”. Workshop on Advanced Separation of Concerns (ASoC) at OOPSLA'2001, Tampa Bay, USA, October 2001

[Garcia 2002a] A. Garcia, C. Lucena. “Software Engineering for Large-Scale Multi-Agent Systems – SELMAS 2002”. (Post-Workshop Report) ACM Software Engineering Notes, September 2002.

[Garcia 2002b] A. Garcia et al. “Engineering Multi-Agent Systems with Aspects and Patterns”. Journal of the SBC, November 2002. (To Appear)

[Garcia 2003a] A. Garcia, C. Lucena, D. Cowan. “Agents in Object-Oriented Software Engineering”. Software: Practice & Experience, Elsevier, 2003. (To Appear)

[Garcia 2003b] A. Garcia et al. "Software Engineering for Large-Scale Multi-Agent Systems". Springer-Verlag, LNCS, January 2003. (To be published)

[Silva 2001] O. Silva, A. Garcia, C. Lucena, “A Unified Software Architecture for System-Level and Agent-Level Dependability in Multi-Agent Object-Oriented Systems”, 7th ECOOP Workshop on Mobile Objects Systems, Budapest, June 2001.

[Silva 2003] O. Silva, A. Garcia, C. Lucena. “The Reflective Blackboard Architectural Pattern”. In: "Software Engineering for Large-Scale Multi-Agent Systems". Springer, LNCS, January 2003. (To Appear)

# Uma Análise Semiótica de Ambientes Virtuais de Discussão

Elton José da Silva[1,2], Clarisse Sieckenius de Souza[1]
Raquel Oliveira Prates[1,3], Ana Maria Nicolaci-da-Costa[4]

[1]Departamento de Informática (PUC-RIO)
[2]Departamento de Computação (UFOP-MG)
[3]Departamento de Computação (UERJ)
[4]Departamento de Psicologia (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil
*{elton, clarisse}@inf.puc-rio.br, raquel@les.inf.puc-rio.br, anicol@psi.puc-rio.br*

## 1. Introdução

Ambientes multiusuário têm sido amplamente utilizados por diversos tipos de grupos com os mais diferentes propósitos. Neste trabalho, estamos interessados no apoio à comunicação de grupos heterogêneos, de pequeno a médio porte, reunidos em torno de um tema de discussão comum, onde a interação entre os membros ocorre de forma alternada nos ambientes presencial e virtual. Ao invés de partir direto para o uso uma aplicação disponível na Internet ou para o desenvolvimento de uma nova aplicação, investigamos, primeiramente, quais eram as expectativas que um grupo de potenciais usuários desse tipo de tecnologia tinha sobre a mesma. De posse dessas descobertas, fizemos uma análise de alguns ambientes tecnológicos disponíveis gratuitamente na Internet. Essa análise nos permitiu identificar alguns acertos e desacertos da tecnologia, com base nas expectativas dos usuários. Esses resultados nos dão alguns subsídios interessantes para informar o projetista de ambientes multiusuário que apóiem o tipo de grupo em questão.

## 2. Demandas do grupo

Para se descobrir quais eram as expectativas de um determinado grupo em relação a um ambiente de discussão online, foi utilizada uma metodologia de análise qualitativa do discurso denominada MEDS (Método de Explicitação do Discurso Subjacente), proposta por Nicolaci-da-Costa [1989, 1994]. Foi enviado um questionário de perguntas abertas aos participantes, cuja grande maioria não tinha qualquer

experiência prévia no uso de ambientes de discussão virtuais, mas que estava motivada com a possibilidade de utilização futura de um ambiente dessa natureza para apoio às discussões das quais participavam de forma presencial. O MEDS é particularmente apropriado porque, sendo originário da área de psicologia clínica, é capaz de revelar medos, desejos, motivações, conflitos e expectativas escondidas nas falas dos participantes[*]. Os resultados obtidos dessa análise revelaram um conjunto de demandas bastante genuínas em relação ao que esses futuros usuários gostariam que estivesse presente em um ambiente virtual de discussão. Essas demandas, com os seus respectivos tópicos associados, são apresentadas resumidamente na Tabela 1 (para uma discussão mais ampla, ver [da Silva et al., 2002]).

## 3. Análise semiótica

Foram selecionadas algumas aplicações[#] públicas na Internet, tipicamente usadas por grupos de discussão online, e realizada uma análise semiótica de porções ilustrativas e relevantes desses softwares usando como fio condutor os tópicos apresentados na Tabela 1. Essa análise está fundamentada na Engenharia Semiótica ([de Souza, 1993], [de Souza et al. 2001]), que muda o foco da avaliação de *como os usuários entendem e usam o software,* tradicional das abordagens cognitivas, para *como o software foi expresso pelo designer*. A análise consistiu em uma inspeção de cada software, a partir de três pontos de observação, que representam a ordem natural em

[*] Para um outro exemplo de utilização do MEDS na fase de elicitação de requisitos, ver [Barbosa et al., 2002]

[#] As *URLs* dos ambientes analisados são: www.*yahoogroups*.com, www.*smartgroups*.com, www.*nossogrupo*.com.br, http://support.discusware.com, www.*cscwplace*.com e http://factotem.com/kneeboard.

que a maioria dos usuários costumam explorar as aplicações computacionais.

(a) uma leitura transversal inicial da aplicação, captando os significados mais salientes dos signos estáticos

(b) uma visita exploratória da aplicação, captando os significados mais salientes dos signos dinâmicos

(c) uma leitura da documentação online oferecida, como os Termos de Serviço, Política de Privacidade e o Sistema de Ajuda associado aos signos estáticos e dinâmicos analisados em (a) e (b).

| **Demanda** | **Tópicos associados** |
|---|---|
| Apresentação do conteúdo das discussões | ▪ Filtragem da informação<br>▪ Sumários e Resumos<br>▪ Estruturação e Visualização |
| Realismo na Interação | ▪ Acesso a informações de contexto<br>▪ Expressão de gestos e emoções |
| Privacidade | ▪ Entre os membros:<br>– conversas reservadas<br>– espaços individuais e coletivos<br>▪ Entre os membros e os visitantes ou membros recém-chegados:<br>– conteúdos discutidos e material compartilhado pelo grupo. |
| Liderança | ▪ Estímulo à participação dos membros<br>▪ Tomada de decisão<br>▪ Convergência das discussões |
| Participação na gestão | ▪ Decisões sobre a permissão de acesso, por visitantes, às discussões do grupo<br>▪ Decisões a respeito de membros que não contribuem nas discussões<br>▪ Decisões sobre a configuração de informações e aparência do grupo |

Tabela 1 – Demandas do grupo em relação a um ambiente virtual de discussão

## 4. Considerações finais

A partir da análise descrita na seção 3, foi identificada uma lacuna entre as expectativas dos usuários e o valor pragmático do que está tecnologicamente disponível no conjunto de ambientes avaliados. Esses resultados têm servido de base para a elaboração de recomendações para projetistas de aplicações multiusuário com ênfase no suporte a grupos de discussão.

## 5. Referências

Barbosa, C. M. O., Sieckenius, C. S., Nicolaci-da-Costa, A. M., and Prates, R. O. P. [2002], *Using the Underlying Discourse Unveiling Method to Understand Organizations of Social Volunteers*, IHC'2002, Fortaleza, Brazil.

da Silva, E. J., de Souza, C. S., and Nicolaci-da-Costa, A. M. [2002], *Visões e Idealizações de Usuários Potenciais a respeito de Aplicações de Groupware,* Rio de Janeiro: PUC-Rio/ Departamento de Informática, Relatório Técnico – MCC XX/2002 (a ser publicado)

de Souza, C. S. [1993], *The Semiotic Engineering of User Interface Languages*, International Journal of Man-Machine Studies, v. 39, pp. 753-773.

de Souza, C. S., Barbosa, S. D. J., and Prates, R. O. [2001], *A Semiotic Engineering Approach to User Interface Design*. Knowledge Based Systems. Amsterdam, v. 14, n. 8, pp. 461-465.

Nicolaci-da-Costa, A. M. [1989], *Questões metodológicas sobre a análise de discurso*, Psicologia: Reflexão e Crítica, 4(1/2), pp. 103-108

Nicolaci-da-Costa, A. M. [1994], *A análise de discurso em questão*, Psicologia: Teoria e Pesquisa, V. 10, N. 2, pp. 317-331.

# Uma Arquitetura de Software para Sistemas Multi-Agentes Baseada em Espaços de Tuplas Reflexivos

Otavio Rezende da Silva[1], Carlos José Pereira de Lucena[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{otavio,lucena}@inf.puc-rio.br

## 1. Motivação

A tecnologia de software está passando por uma transição de arquiteturas monolíticas, construídas a partir de um projeto único e coeso para arquiteturas compostas por agentes e sistemas multi-agentes semi-autônomos e heterogêneos. Estas arquiteturas são marcadas pela existência de novas propriedades do nível do sistema (ou inter-agentes) [Garcia 2001, Silva 2001e] que requerem estratégias (ou políticas) que controlem a lógica (isto é seus agentes) e os dados da aplicação.

Dentre os problemas inerentes a tal transição, nenhum é mais sério que a dificuldade de incorporar e compor múltiplas estratégias de controle, o que leva à necessidade de uma abordagem mais sofisticada desde a fase arquitetural de um sistema multi-agente. As funcionalidades básicas de agentes de software já são bastante complicadas, de forma que suas estratégias de controle necessitam ser projetadas e implementadas separadas do seu comportamento básico.

Na verdade, os níveis de satisfação dos requisitos de qualidade (por exemplo, capacidade de reutilização e manutenção) de um sistema multi-agente são fortemente dependentes de sua arquitetura. Desta forma, se uma arquitetura que oferece suporte ao tratamento de múltiplas estratégias de controle for escolhida desde o início do desenvolvimento de um sistema multi-agente, torna-se mais simples atingir os atributos de qualidades necessários no restante do desenvolvimento do sistema.

Com o objetivo de resolver este problema é proposto neste trabalho, o padrão arquitetural Reflective Blackboard [Silva 2002a, Silva 2003]. De forma complementar, é apresentado o framework T-Rex [Silva 2001a, Silva 2001b, Silva 2001d], que é uma implementação do padrão que também provê suporte a diferentes propriedades inter-agentes.

## 2. O Padrão Arquitetural Reflective Blackboard

No contexto de sistemas multi-agentes, o padrão arquitetural Blackboard tem sido largamente utilizado como uma metáfora útil para o tratamento da comunicação e coordenação de organizações de agentes heterogêneos. Este padrão arquitetural possui três componentes principais: múltiplos *agentes* independentes, cada um implementando uma parte específica da lógica da aplicação, interagem entre si usando o componente denominado *blackboard*; o *blackboard* por sua vez é uma estrutura de dados que é usada como um mecanismo geral de comunicação e coordenação para os diferentes agentes, e é gerenciado por um componente de *controle*. No entanto, o padrão não especifica de forma explícita como o componente de controle deve lidar com as distintas estratégias de controle necessárias para gerenciar o *blackboard*, nem como separar estas estratégias de controle dos agentes e dados que compõem uma aplicação, o que pode levar a arquiteturas de software multi-agente que são mais difíceis de manter, compreender e reutilizar.

Neste trabalho, é proposto o padrão arquitetural Reflective Blackboard [Silva 2002, Silva 2003] que é construído a partir da composição de dois outros padrões arquiteturais bem conhecidos: o próprio padrão Blackboard e o padrão Reflection. Como resultado da composição proposta, os componentes do padrão Reflection são utilizados para refinar a estrutura geral definida pelo padrão Blackboard e com o objetivo de promover uma melhor separação de responsabilidades[*]. A separação de responsabilidades é atingida em arquiteturas reflexivas (baseadas no padrão Reflection) através da separação do sistema em dois níveis: o nível base e o meta-nível.

O padrão arquitetural Reflective Blackboard segue esta organização: enquanto a lógica e os dados da aplicação multi-agente são encapsulados no nível base, o componente de controle é situado no meta-nível. Desta forma, as estratégias de controle do

[*] Do inglês *separation of concerns*

sistema podem ser tratadas completamente separadas de sua lógica e dados. Por sua vez, os agentes da aplicação são projetados de forma independente de suas estratégias de controle. Esta abordagem é importante no que se refere a aplicações multi-agentes de grande porte uma vez que sua lógica já é bastante complicada devido à complexidade inerente a sistemas multi-agentes de grande porte.

O padrão arquitetural Reflective Blackboard pode ser utilizado para minimizar a complexidade causada pela presença de numerosas propriedades inter-agentes [Garcia 2001, Silva 2001e] em sistemas multi-agentes. O padrão provê suporte à implementação de estratégias de controle para os diferentes agentes de um sistema e para as propriedades do nível do sistema.

## 3. O Framework T-Rex

Além da definição do padrão arquitetural, também faz parte do objetivo deste trabalho a implementação de uma infra-estrutura para o desenvolvimento de sistemas multi-agentes baseados na arquitetura proposta pelo Reflective Blackboard. Foi então desenvolvido o framework T-Rex (Espaços de **T**uplas **Re**fle**x**ivos) [Silva 2001a, Silva 2001b, Silva 2001d] que além de ser uma implementação do padrão proposto, oferece suporte a diferentes propriedades do nível do sistema, como mobilidade, comunicação, persistência e coordenação.

T-Rex é baseado em uma variante do padrão Reflective Blackboard denominada Espaços de Tuplas Reflexivos. Esta variante utiliza espaços de tuplas à la Linda como infra-estrutura para o blackboard do nível base. Além disso, eles são também utilizados para armazenar as diferentes estratégias de controle existentes no meta-nível.

As propriedades do nível do sistema em T-Rex (mobilidade, comunicação e persistência) são também implementadas sempre através da infra-estrutura de seus espaços de tuplas. Desta forma, as estratégias de controle podem ser facilmente associadas a estas propriedades, através da reflexão, simplificando sua coordenação e composição. Além disso, a utilização de uma tecnologia única pode ajudar na manutenção e suporte de sistemas multi-agentes que se baseiam na infra-estrutura fornecida por T-Rex.

Além da implementação da infra-estrutura de T-Rex foi desenvolvida também, com o objetivo de validar sua utilização em sistemas com múltiplas propriedades do nível do sistema e múltiplas estratégias de controle, uma aplicação de Marketplace [Silva 2001c]. Nesta aplicação, T-Rex foi utilizado para prover a comunicação entre os agentes, a mobilidade e persistência destes. Além disso, diferentes estratégias de controle foram utilizadas para compor a propriedade de comunicação com a mobilidade e persistência dos agentes. Estratégias de controle também foram utilizadas para implementar atividades de controle do próprio Marketplace.

## 4. Referências

[Garcia 2001] A. Garcia, C. Chavez, O. Silva, V. Silva, C. Lucena. “Promoting Advanced Separation of Concerns in Intra-Agent and Inter-Agent Software Engineering”. ASoC at OOPSLA'2001, Tampa Bay, EUA, 2001

[Silva 2001a] O. Silva, A. Garcia, C. Lucena, “A Unified Software Architecture for System-Level and Agent-Level Dependability in Multi-Agent Object-Oriented Systems”, 7th ECOOP MOSW, Budapeste, Hungria, 2001

[Silva 2001b] O. Silva, A. Garcia, C. Lucena, T-Rex: A Reflective Tuple Space Environment for Dependable Mobile Agent Systems. III Workshop de Comunicação Sem Fio e Computação Móvel WCSF at IEEE MWCN, Recife, Brasil, 2001

[Silva 2001c] O. Silva, D. Orlean, F. Ferreira and C. Lucena. “A Shared-Memory Agent-Based Framework for Business-to-Business Applications”. IRMA'2001, Toronto, Canadá, 2001

[Silva 2001d] O. Silva, C. Lucena “Um Ambiente Para Aplicações de Agentes Móveis Baseado em Arquiteturas Reflexivas de Espaços de Tuplas” WTES do SBES, Rio de Janeiro, Brasil, Outubro 2001

[Silva 2001e] V. Silva, O. Silva, A. Garcia, C. Chavez, C. Lucena Separation of Concerns for Multi-agent Software Engineering. Poster Session at OOPSLA'2001, Tampa Bay, Florida, EUA, Outubro, 2001.

[Silva 2002] O. Silva, A. Garcia, C. Lucena, “The Reflective Blackboard Architectural Pattern for Developing Large Scale Multi-Agent Systems” SELMAS 2002 at ICSE 2002, Orlando, EUA, 2002.

[Silva 2003] O. Silva, A. Garcia, C. Lucena. “The Reflective Blackboard Pattern: Architecting Large-Scale Multi-Agent Systems.” In:. Software Engineering for Large-Scale Multi-Agent Systems. A. Garcia, et al (Ed.) Springer-Verlag, 2003 (a ser publicado)

# Uma arquitetura para aplicações baseadas em serviços utilizando a Web Semântica

Francisco Eduardo dos Reis Ferreira[1], Carlos José Pereira de Lucena[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{chico, lucena}@inf.puc-rio.br

## 1. Introdução

A Web caminha hoje em dia para um novo paradigma de serviços e funcionalidades. Nos últimos anos, uma série de websites disponibilizaram para os usuários da Internet serviços de reservas, compras, consultas e notícias, dentre outros.

No entanto, a maioria dessas aplicações é baseada em bancos de dados locais centralizados que contém informações sobre uma entidade apenas. A integração de sistemas diferentes visando agrupar e compilar a informação para potenciais clientes é um caminho natural para as aplicações na Web, principalmente pela explosão de fontes de dados associadas aos inúmeros sites espalhados pela rede.

Neste contexto de evolução das aplicações atualmente presentes na Web, Berners-lee apresenta em [Beners-Lee 2001] uma visão sobre o futuro dos padrões e técnicas que hoje governam a WWW, a Web Semântica.

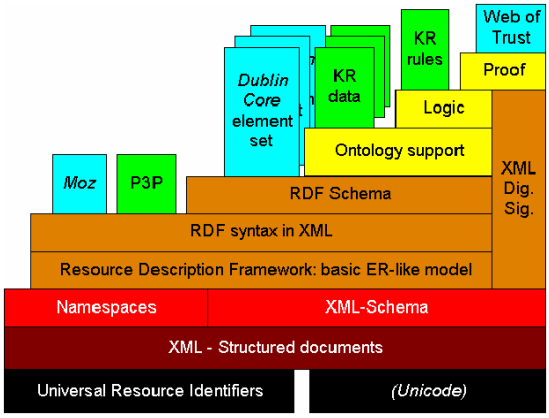



**Figura 1 - A arquitetura da Web Semântica por Berners-Lee**

A objetivo deste trabalho é realizar um estudo sobre a implantação real de aplicações em uma web semântica, levando em consideração os principais aspectos levantados por Berners-lee em sua visão sobre o futuro da WWW. Serão analisados mecanismos de representação de conhecimento, construção de ontologias, processamento distribuído e infra-estrutura de desenvolvimento e implantação. O objetivo fundamental deste trabalho é verificar a viabilidade do uso de arquiteturas orientadas a serviços na construção deste tipo de aplicação, verificando quais os principais problemas e dificuldades para a criação de aplicações reais e escaláveis na Web Semântica.

## 2. Aplicações baseadas em serviços

Web Services [Kirtland 2001], [Kreger 2001] podem ser considerados um padrão para implementação de arquiteturas orientadas a serviços.

Nesta arquitetura, os componentes são vistos como serviços que realizam uma tarefa específica. Uma aplicação é simplesmente uma composição de serviços, que unidos, atendem a uma necessidade de um determinado usuário. Uma aplicação baseada em web services possui três tipos básicos de entidades: Service Provider, Service Broker e Service Requester

Um determinado serviço é fornecido por um Service Provider. Este publica suas funcionalidades em algum sistema de busca por serviços, que recebe o nome de Service Broker. Um Service Requester é uma entidade que irá utilizar alguma funcionalidade de um Web Service para realizar alguma tarefa mais completa. Para tal, ela entra em contato com algum Broker de Serviços através de algum padrão para descoberta (i.e. UDDI). Ao receber a resposta, o Service Requester terá as interfaces padrão para acesso aos serviços que atendem às suas necessidades. Estas interfaces fazem parte da descrição do serviço, que normalmente é feita em uma linguagem específica para isso, a WSDL (Web service Description Language). A partir deste ponto, os serviços fornecidos pelos Service Providers poderão ser acessados utilizando um protocolo de acesso a objetos remotos (i.e. SOAP [SOAP] ou IIOP [IIOP]).

## 3. Trabalhos Relacionados

Diversos mecanismos e propostas foram criados para permitir o desenvolvimento de aplicações baseadas em serviços. É possível citar [Fensel 2002], [Poneekanti 2002], [TAP] e [Heflin 2001]

## 4. Everyware : Uma plataforma para desenvolvimento de aplicações baseadas em serviços

O objetivo do trabalho é criar uma abordagem pragmática para o desenvolvimento de aplicações na Web Semântica que sejam implementadas utilizando uma infra-estrutura de software e de servidores escalável, confiável e de fácil evolução e manutenção, assim como as principais aplicações presentes hoje em dia na Web.

Nesta arquitetura, uma aplicação na Web Semântica pode ser vista como uma composição de serviços disparada diretamente ou indiretamente (através do uso de agentes de software) por um usuário do sistema. Esta composição de serviços pode ser guiada por uma ontologia de domínio que explicita como os serviços estão relacionados e como devem ser compostos.

Outro ponto importante é a formação de ontologias através da composição de serviços. Muitas vezes uma aplicação ou agente de software na web semântica não possui a priori uma ontologia completa do domínio sobre o qual está atuando. Neste caso, serão necessários outros serviços capazes de completar estas ontologias (com novas instâncias na taxonomia ou novas regras de inferência) de forma a permitir o processamento completo da informação.

Finalmente, a arquitetura deverá oferecer mecanismos de composição de serviços e de ontologias de forma simples e direta, sem necessidade de criação de código por parte do desenvolvedor das aplicações.

## 5. Referências

[Beners-Lee 2001] Berners-Lee, T., Hendler, J. and Lassila, O. **The Semantic Web**. Scientific American, Maio 2001.

[Fensel 02] Fensel, D. and Bussler, C. **The Web Services Modeling Framework – WSMF Extended Abstract.**

[Heflin 2001] Heflin, J. and Hendler, J. **Searching the Web with SHOE.**

[Kirtland 2001] Kirtland, M. **Web Services Essentials.** Microsoft Developer Network. January, 2001.

[Kreger 2001] Kreger, H. **Web Services Conceptual Architecture.** IBM Software Group White Paper, May 2001.

[Palmer 2001] Palmer, S. **The Semantic Web : An Introduction.** [online] http://infomesh.net/2001/swintro

[Poneekanti 2002] Poneekanti, S. ; Fox, A. **SWORD: A developer Toolkit for Web Service Composition**. Proceedings of the Eleventh International World Wide Web Conference. Hawaii, 2002.

[TAP] **Tap Project.** [online] http://tap.stanford.edu

# Uma Linguagem de Modelagem para Sistemas Baseados em Agentes

Ricardo Choren[1], Carlos José Pereira de Lucena[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{choren, lucena}@inf.puc-rio.br

## 1. Introdução e Motivação

A tecnologia de agentes tem recebido uma grande atenção nos últimos anos e, como conseqüência, a indústria está começando a se interessar em adotar esta tecnologia para o desenvolvimento de produtos. A área de sistemas multi-agentes (SMA) tem por objetivo oferecer princípios para a construção de sistemas complexos envolvendo múltiplos agentes.

Para que o desenvolvimento de sistemas baseados em agentes de software consiga se estabelecer, é preciso criar metodologias que auxiliem todas as fases do ciclo de vida do desenvolvimento deste tipo de sistemas. Hoje, já existe um notável esforço na pesquisa por arquiteturas, métodos de desenvolvimento, linguagens de modelagem, linguagens de desenvolvimento, modelos de reutilização e etc. específicos para sistemas multi-agentes. E a necessidade do emprego de técnicas de engenharia de software, uma engenharia de software para SMA, será cada vez mais importante a medida que as arquiteturas de desenvolvimento se tornem distribuídas, como p ex., com a Web semântica.

O objetivo deste trabalho é apresentar uma linguagem de modelagem que suporte o desenvolvimento de aplicações com a tecnologia de agentes. O enfoque seguido neste trabalho consiste em oferecer uma linguagem que ofereça notação para aspectos específicos de sistemas multi-agentes, como, distribuição, adaptação, aprendizado, modelo de conhecimento e autonomia.

O resultado deste trabalho, a Linguagem de Modelagem para Sistemas Multi-agentes (LM-SMA), apresenta modelos com elementos de notação próprios para agentes, identificando e especificando as características de agência do sistema.

A motivação deste trabalho é oferecer um mecanismo de engenharia de software - uma linguagem de modelagem - que seja própria para a abstração de agentes. Esta linguagem serve para caracterizar (especificar) o sistema utilizando uma semântica de agentes.

Isto não significa que este trabalho simplesmente busca tornar obsoleto o conceito de objetos. Também é uma motivação deste trabalho a identificação dos pontos onde a estrutura do sistema de agentes pode ser razoavelmente expressa como objetos. Estes pontos devem incluir componentes da linguagem de comunicação de agentes (incluindo estrutura de mensagens e elementos de vocabulário), protocolos de comunicação (mostrando a estrutura da interação que ocorre entre agentes) e representação do conhecimento existente no sistema (incluindo a definição de uma ontologia de termos).

## 2. Abordagem

Este trabalho propõe uma técnica para o suporte do desenvolvimento de sistemas AO. Uma solução AO é composta por agentes que agem em prol do alcance de seus objetivos. Assim, o primeiro passo para o desenvolvimento da linguagem de modelagem proposta neste trabalho é a divisão dos objetivos do sistema. Depois da divisão dos objetivos, busca-se explicitar os cenários (exemplos de atuação dos agentes no contexto de um objetivo a ser alcançado) para a descrição, de alto nível, dos agentes, suas funcionalidades e suas interações.

A partir da descrição das atuações, é possível se descrever os planos de ação dos agentes, com as descrições das atividades que estes devem executar, e as interações, com as descrições das mensagens que devem ser trocadas entre os agentes.

A linguagem de modelagem proposta, então, busca instrumentalizar o desenvolvedor com diagramas que permitam a modelagem dos objetivos do sistema, dos tipos de agentes que formam a solução e de sua atuação, com os planos e as interações que ocorrem. A linguagem proposta se preocupa em oferecer itens de notação para as características específicas de agência, como adaptação, aprendizado e autonomia, pois é importante que estas fiquem documentadas.

Além de descrever os agentes que fazem parte da solução da aplicação, linguagem de modelagem proposta neste trabalho assume a existência de

elementos não-agentes na modelagem de um problema baseado em agentes. Elementos não-agentes são itens que existem no sistema, são manipulados pelos agentes, mas não são da mesma natureza que os agentes. Mesmo não sendo agentes, é importante permitir que o desenvolvedor possa fazer a modelagem destes itens.

# Using an Agent-Based Framework and Separation of Concerns for the Generation of Document Classification Tools

João Alfredo Pinto de Magalhães[1], Carlos José Pereira de Lucena[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

{magalha, lucena}@inf.puc-rio.br

## 1. Introduction

The problem we address in this work is document classification - that is, given a set of documents and a set of classes, to decide which documents best match each class. When a tool for classifying documents is generated, many concerns need to be taken into consideration. These concerns can be divided into two main groups:

· The first group deals with concerns related to general functionalities, such as document processing (extraction of important parameters used during classification), document location (finding the places where the documents are stored), detection of duplicates (to avoid the useless re-classification of already classified documents) and report generation (in which way the results will be used), just to mention a few;

· The second group deals with concerns related to the document classification algorithm that is used.

Due to its intrinsic characteristics, the second group of concerns strongly suggests that it should be handled by someone with accurate skills in algorithm engineering, whereas the first group does not require such a specialization. The first group of concerns can be "frozen" in a framework [Pree 1995] to allow for algorithm research. Such a framework, in which one can plug and use different algorithms in an easy fashion, plays the role of a laboratory where one can analyze the overall performance of different algorithms, processing different types of documents (for instance, documents about economy, politics, documents written in HTML, PDF or PS). After testing different algorithms, the best one can be chosen in order to create a complete application that uses the functionalities of the first group.

Following this strategy, we have developed Avestruz, a multi-agent framework that generates tools to classify documents. Since their work is automatic, the generated tools can be highly autonomous: the only inputs required are the places where the documents are stored. The existing hot spots [Pree 1995] are the type of documents supported, the reports generated, the degree of memory and pro-activity of the agent, the document selection strategy and the classification algorithm. Clearly, the last hot spot is directly related to the second group of concerns (which we call classification concerns), whereas the others comprise the first group (which we call concern platform).

The main goal of the framework architecture is to entirely detach the concerns related to the classification algorithm from those related to the platform by using the concept of separation of concerns [Tarr et al. 1999]. Once the concerns platform is defined, a specialist from the documents domain can handle the concerns related to the classification algorithm. The great advantage is the ability to test different algorithms: a specialist in algorithms has a powerful tool for making tests.

## 2. Methodology

The decision of adopting a design approach based on multi-agents for the development of the framework is meant to facilitate the understanding of the problem at hand. A classifier agent can be seen as an entity that is constantly searching for documents, while selecting the ones that must be processed. It processes them according to a classification algorithm and reports the results.

In order to prevent that concerns related to a specific agent conflict with concerns related to the multi-agent system [Zambonelli et al. 2001] [Silva, Garcia, Lucena 2001] we have adopted a two-layered approach. The software agents that are controlled by the second layer are placed in the first layer, which is the management layer. Figure 1 illustrates this separation

The strategy used to obtain the desired independence between platform and classification concerns was to concentrate the concerns platform on both the management layer and the software agent layer while the classification concerns are located only in the agent layer. We do this in a way that turning the software agent into a user of external services hides the software agent concept. Once the platform is instantiated (generated), the framework instantiator is free from the software agents concerns and free to concentrate its attention on the classification algorithm. It does

not need to know that the framework was built using software agents - this capability is completely encapsulated. All the frozen spots are located in the management layer, and they deal with concerns such as agent cooperation and coordination. Thus, in order to benefit from the proposed two-group approach, it is necessary to instantiate the framework following two steps: the first step deals with the concerns platform, where all the corresponding hot spots are completed; the second step is dedicated exclusively to the classification hot-spot.

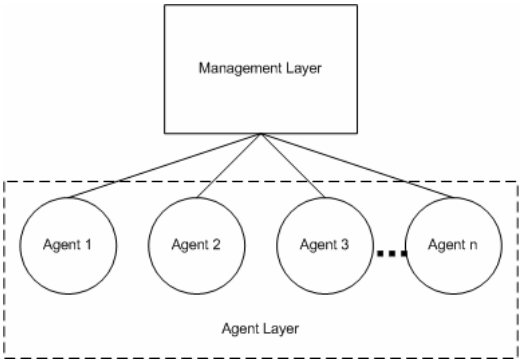



Figure 2: General view of the layers

The main purpose for making the tools generated from the framework highly autonomous is to minimize the need for interference from the user during the classification process. Also important is the fact that the instances of the framework are not limited to self-contained applications: it is possible to generate independent components that can be connected to other knowledge management applications.

## 3. Ongoing Work

Avestruz was developed using the object-oriented paradigm and Java as the programming language. There already are four substantially different instances derived from the framework: two applications and two components. One of the components classifies documents in the SHOE format [Helfin, Hendler, and Luke 1999], a proposal for marking up documents for the Semantic Web [Buranarach 2001], and it is currently being used to evaluate how the abundance of meta-information in documents affects the classification process. We are currently working on improvements to the framework and the development of new instances in order to validate our two-step development approach.

## 4. References

Buranarach. M. 2001. The Foundation for Semantic Interoperability on the World Wide Web. Ph.D. diss. Science Computer Department, University of Pittsburgh.

Helfin, J. D.; Hendler, J.; Luke, S. 1999. SHOE: A Knowledge Representation Language for Internet Applications, Technical Report, TR-99-71, Science Computer Department, University of Maryland.

Pree, W. eds. 1995. *Design Patterns for Object-Oriented Software Development*. Addison Wesley.

Silva, O.; Garcia, A.; Lucena, C. J. 2001. A Unified Software Architecture for System-Level Dependability in Multi-Agent Object-Oriented Systems. 7th ECOOP Workshop on Mobile Objetcts Systems.

Tarr, P.; Ossher, H.; Harrison, W.; and Sutton, S.M. 1999. N Degrees of Separation: Multi-Dimensional Separation of Concerns. In Proceedings of the International Conference on Software Engineering (ICSE'99).

Zambonelli, F.; Jennings, N.; Omicini, A.; Wooldridge, M. 2001. Agent-Oriented Software Engineering for Internet Applications. In Coordination of Internet Agents.

# Utilização de Ontologia no Segmento B2C

Francisco José Z. Guimarães[1], Carlos José Pereira de Lucena[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

chicao@inf.puc-rio.br

## 1. Motivação

A principal dificuldade dentro do segmento B2C está em aumentar a utilidade da WWW para o comércio eletrônico através da melhoria de sua interface. Algumas das vantagens proporcionadas pelo meio eletrônico são dificultadas devido aos seguintes problemas:

- Apesar de a WWW permitir ao comprador ter acesso a uma grande quantidade de informação, obter a informação do fornecedor certo que venda o produto desejado a um preço razoável, pode ser uma tarefa muito custosa.
- Dificuldade de integração das informações colhidas de diversos fornecedores na WWW devido à falta de padronização. Dessa forma se torna difícil a comparação automática de produtos entre várias lojas on-line, exigindo que o consumidor extraia e compare as informações ele mesmo.

Uma das formas de melhorar a interface é através do uso de agentes inteligentes de busca de informação, isto é, agentes de compra, que auxiliam os compradores a encontrar produtos de seu interesse. Para que isso ocorra se esbarra em uma dificuldade inerente à própria WWW: A mistura da linguagem natural, *gifs* e informação de *layout* de HTML são uma das maiores barreiras para a automatização do comércio eletrônico, pois a semântica da informação é somente compreensível por seres humanos. [Fensel 2001]

Espera-se conseguir agentes de compra mais eficientes quando associados ao uso de ontologias, e lojas virtuais que tenham anotações especiais que sigam uma ontologia.

## 2. Situação Atual

Agentes de compra são agentes inteligentes que ajudam o consumidor na escolha e comparação entre produtos ou serviços oferecidos por lojas on-line. Alguns exemplos de agentes desse tipo são: Miner, BizRate e mySimon.

A arquitetura típica de um agente de compra é dada na figura 1. Como cada loja on-line descreve as informações de seus produtos de uma forma especifica, grande parte do esforço para o desenvolvimento desses agentes de compra está no desenvolvimento de conversores específicos para cada loja. Esses convertem as descrições dos produtos de cada loja para um padrão de descrição que o mediador do agente pode entender.

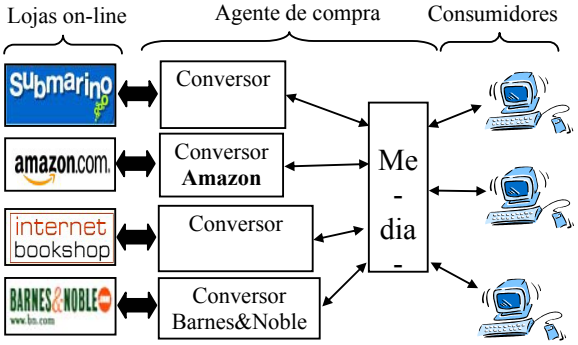



Figura 1 – Arquitetura de um agente de compra.

As principais dificuldades no desenvolvimento dos conversores são as seguintes:

- Ele deve extrair a informação do produto que está em HTML (linguagem natural).
- Cada loja on-line deve ter um conversor feito sob medida.
- Qualquer mudança no *layout* da loja exige uma mudança no conversor.
- Ele deve converte a descrição do produto feita pelo site, para uma descrição padrão entendida pelo mediador.

## 3. Ontologias

Uma das melhores definições do que vem a ser uma ontologia é dada por T. R. Gruber em seu artigo "*A Translation Approach to Portable Ontology Specification*" [Perez 1999] :

"Uma ontologia é uma especificação formal explicita de uma conceitualização compartilhada."

É importante explicitarmos alguns dos termos utilizados nessa definição. O termo "conceitualização" refere-se a um modelo abstrato de algum fenômeno que identifique conceitos relevantes desse fenômeno. O termo "explicita" significa que os tipos de conceitos usados e as limitações do uso desses conceitos devem ser

definidas de forma explícita. O termo "formal" refere-se ao fato que a ontologia deve ser passível de ser processada por uma máquina. E finalmente o termo "compartilhada" reflete a noção de que a ontologia captura um conhecimento consensual aceito por um grupo de pessoas .

Uma ontologia é tipicamente formada pelos seguintes componentes:

- Um conjunto de conceitos e a taxinomia entre esses conceitos.
- Um conjunto de relacionamentos entre esses conceitos.
- Um conjunto de funções. Uma função é uma caso especial de relacionamento em que um conjunto de elementos tem uma relação única com um outro elemento.
- Um conjunto de axiomas, ou seja regras que são sempre verdade.
- Um conjunto de instâncias que na realidade fazem parte da base de conhecimentos.

As principais tecnologias que estão por trás da utilização de ontologias na ciência da computação são: RDF [Swick and Lassila 1999], RDF Schema [Brickley and Guha 2002], DAML+OIL [Connolly et al. 2001], SHOE [Luke and Heflin 2000]. As três primeiras tecnologias estão sendo usadas na implementação do estudo de caso.

## 4. Ontologia em B2C

Com o uso de ontologias praticamente não é mais necessário o uso de conversores entre os formatos das descrições dos produtos das lojas, uma vez que as lojas seguem uma mesma ontologia. Caso duas lojas sigam ontologias diferentes para a descrição de seus produtos, basta existir uma ontologia de mapeamento entre as duas ontologias. A figura 2 mostra a nova arquitetura do agente de compra com o uso de ontologias.

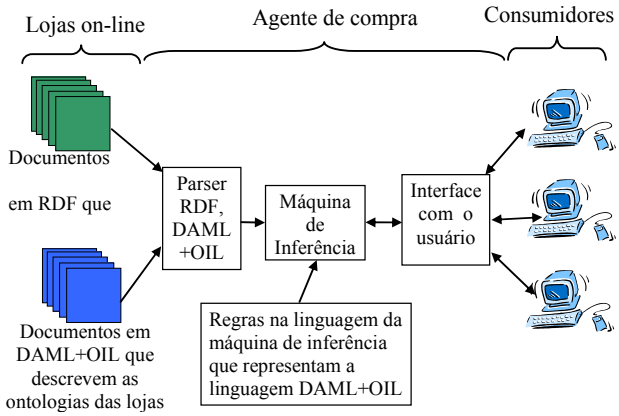



Figura 2 – Agente de compra com ontologias.

## 5. Referências

[Brickley and Guha 2002] - Dan Brickley, R.V. Guha - RDF Vocabulary Description Language 1.0: RDF Schema - W3C Working Draft 30 April 2002

[Connolly et al. 2001] - Dan Connolly, et al. - DAML+OIL (March 2001) Reference Description – W3C Note 18 December 2001

[Fensel 2001] - Dieter Fensel - Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce - August 15, 2001

[Lee; Hendler and Lassila 2001] - Tim Berners-Lee, James Hendler, Ora Lassila - The Semantic Web - Scientific American - May 2001

[Luke and Heflin 2000] - Sean Luke, Jeff Heflin - The SHOE Specification - April 28, 2000

[Perez 1999] - A. Gómez-Pérez. Ontological Engineering: A state of the Art. Expert Update. British Computer Society - Autumn. Vol. 2. Nº 3. 1999

[Swick and Lassila 1999] - Ralph R. Swick, Ora Lassila - Resource Description Framework (RDF) Model and Syntax Specification - W3C Recommendation 22 February 1999

# Web Life - Uma arquitetura para a implementação de sistemas multi-agentes para a Web

Marcelo Blois Ribeiro[1,2], Carlos José Pereira de Lucena[1]

[1]Departamento de Informática – Pontifícia Universidade Católica – Rio de Janeiro (PUC-RIO)

R. Marquês de São Vicente, 225, Gávea – 22453-900 – Rio de Janeiro – RJ – Brasil

[2] Faculdade de Informática – Pontifícia Universidade Católica – Rio Grande do Sul (PUC-RS)

Av. Ipiranga, 6681, Prédio 16 - 90619-900 - Porto Alegre - RS - Brasil

{blois, lucena}@inf.puc-rio.br

## 1. Descrição do Trabalho

O desenvolvimento de sistemas multi-agentes é uma área em franca expansão devido a sua adaptação a resolução de problemas de natureza distribuída e, em especial, a possibilidade de sua aplicação na construção de sistemas para a Internet. O tema, historicamente restrito ao ambiente de pesquisa de Inteligência Artificial, passou a influenciar outras áreas da computação, especialmente a área de Engenharia de Software.

Começaram a surgir os primeiros estudos sobre a estruturação de métodos que apoiassem o desenvolvimento de sistemas multi-agentes. A identificação de novos requisitos para a construção destes sistemas, motivou o surgimento de ferramentas conceituais e computacionais que incorporam agentes de software como entidades fundamentais para a resolução de problemas complexos com distribuição de processamento. Em especial, a necessidade da definição de uma plataforma de prototipação e implementação de sistemas multi-agentes tornou-se evidente.

Algumas iniciativas foram tomadas por parte de instituições de ensino e por consórcios de empresas para a criação de uma infra-estrutura padrão multi-agentes, que servisse de base para a criação de sistemas multi-agentes interoperáveis. Uma das primeiras plataformas criadas com este objetivo foi o FIPA agent platform [FIPA, 2000]. Com o uso de uma forma padrão de comunicação e a presença de entidades básicas de gerenciamento das funções da plataforma, a FIPA criou um modelo capaz de mapear as principais funções envolvidas na interação entre os agentes de uma sociedade. Embora não tenha atingido plenamente o seu objetivo de se tornar uma plataforma padrão, o modelo da FIPA serviu como base para a implementação de diversas plataformas multi-agentes.

Além da infra-estrutura de comunicação e administração da plataforma de agentes, outros requisitos podem ser identificados como fundamentais a criação de sistemas multi-agentes. Este é o caso das formas de representação do conhecimento que permitem a troca estruturada de conhecimento entre agentes de software. Esta capacidade auxilia a adaptação dos agentes a diferentes situações, assim como o aprendizado de novas funções e objetivos.

Para que uma plataforma de implementação de sistemas multi-agentes seja completa, é preciso o suporte ao desenvolvimento da parte interna dos agentes, assim como a criação da infra-estrutura de atuação dos agentes em sua organização (parte externa). Algumas plataformas foram propostas para a implementação de sistemas multi-agentes, mas poucas oferecem suporte total à sua confecção. Dentre os trabalhos que mais se destacam com relação ao suporte das partes interna e externa de um sistema multi-agente, encontram-se o Zeus [Zeus, 2002] e o Jack [Jack, 2001].

Algumas plataformas pretendem dar suporte a criação de SMA genéricos, enquanto outras concentram esforços em determinados tipos de aplicações. Um dos tipos de aplicação para a qual faltam estudos aprofundados é o desenvolvimento de sistemas multi-agentes para a Web. Embora ainda não existam plataformas voltadas para a Web, a idéia de uso da Web como um ambiente onde agentes se relacionam foi proposta claramente em [Berners-Lee, Hendler and Lassila 2001]. Neste trabalho, será ilustrada a criação de uma nova Web que estenda a atual, incorporando descrições semânticas sobre os seus conteúdos. Os exemplos apresentados no decorrer do trabalho mostram como os requisitos da nova Web afetam a criação e interação de agentes de software.

O objetivo deste trabalho é a apresentação de uma arquitetura que permita o desenvolvimento de sistemas multi-agentes para a Web, incorporando os requisitos necessários à criação da Web Semântica. A arquitetura Web Life utiliza o modelo conceitual FIPA 2000 como camada de base para a criação de sua infra-estrutura de

interação entre agentes. Para o desenvolvimento dos agentes de software o Web Life utiliza uma abordagem orientada a componentes, visando auxiliar o desenvolvedor a implementar os aspectos de agência organizados por unidades semanticamente acopladas.

A implementação da arquitetura tem como diretriz o reuso de mecanismos existentes para funções especializadas dos seus componentes, como é o caso do mecanismo de tomada de decisões. Embora não seja obrigatório o uso de máquinas de inferência no modelo conceitual do Web Life, a arquitetura traz uma integração nativa com o JESS - Java Expert System Shell [Jess, 2002] para o uso de inferências como técnica de tomada de decisões. O uso de padrões e mecanismos existentes facilita a adoção da arquitetura e aproveita todo um conjunto de ferramentas criadas para agregar valor a estes padrões, acelerando o desenvolvimento e expansão da plataforma. Em especial, a utilização da infra-estrutura da Web atual, como o uso de protocolo HTTP na troca de recursos, é característica fundamental para a elaboração do Web Life.

## 2. Referências

Berners-Lee, T.; Hendler, J. & Lassila, O. The Semantic Web. Scientific American, v.1, n.29. 2001.

FIPA. FIPA Abstract Architecture Specification. 2000. Disponível em <http://www.fipa.org/repository/fipa2000.html>. Acesso em 27 nov 2002.

JACK. What is JACK? Desenvolvido por: The Agent Oriented Software Group. 2001-2002. Disponível em <http://www.agent-software.com/shared/products/index.html>. Acesso em 27 nov 2002.

JESS. Jess, the Expert System Shell for the Java Platform. 2002. Disponível em <http://herzberg.ca.sandia.gov/jess/>. Acesso em 27 nov 2002.

ZEUS. Desenvolvido por British Telecommunications plc, 1997-2002. Disponível em <http://more.btexact.com/projects/agents/zeus/>. Acesso em: 27 nov 2002.