OCtoPUS: Um Modelo para Classificação de Perfil de Usuários usando Trilhas em Sistemas Ubíquos

Douglas M. da Silva[1], Jorge Barbosa[1], Renata Vieira[2]

[1]Programa Interdisciplinar de Pós-Graduação em Computação Aplicada (PIPCA)
Universidade do Vale do Rio dos Sinos (UNISINOS) – São Leopoldo – RS – Brasil

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

michaelsen@gmail.com, jbarbosa@unisinos.br, renata.vieira@pucrs.br

Abstract. *This work presents the OCtoPUS model which uses the user behavior information, characterized by trails in a ubiquitous environment, to identify the user profile. The model uses semantic web technologies to represent and infer knowledge about the user, specially the classification inference. A prototype was developed to evaluate the OCtoPUS model in the learning field. The prototype classifies the user in terms of abilities and competencies based on which trails the user have enacted during the course.*

Resumo. *Neste trabalho é apresentado o modelo OCtoPUS, que utiliza informações sobre o comportamento do usuário, caracterizado por trilhas em um ambiente ubíquo, para identificar o seu perfil. As tecnologias da web semântica são usadas para representar e inferir – através do raciocínio de classificação – novos conhecimentos a respeito do usuário. Para testar e validar o modelo OCtoPUS foi desenvolvido um protótipo na área educacional, que classifica os alunos de acordo com suas habilidades e competências baseado em suas trilhas percorridas durante o curso.*

1. Introdução

A web semântica pretende organizar o conhecimento disponível de forma a esses dados serem úteis também para sistemas computacionais, possibilitando assim um melhor uso dos recursos disponíveis na web. Para atingir esse objetivo são utilizadas técnicas de representação do conhecimento, da lógica e de inteligência artificial [Antoniou 2004]. Na representação do conhecimento ontologias são usadas para representar domínios e para definir vocabulários comuns que facilitam a equiparação de conceitos. Por ser o nível mais alto de camadas sucessivas de metadados a ontologia consegue agregar estrutura, metadados e lógica.

Neste trabalho[1] ontologias com embasamento em lógica são utilizadas para a representação e gerenciamento do conhecimento. Já os conceitos da computação ubíqua são utilizados para a definição do histórico de interação do usuário. Dessa forma, os conhecimentos representados são: (i) o perfil do usuário, mais precisamente estereótipos

[1] Este trabalho recebeu apoio do Conselho Nacional de Desenvolvimento Científico e Tecnológico – CNPq – Brasil (Edital 15/2007 – Universal).

de usuários. Esse tipo de modelagem de usuários segue os conceitos de modelo de conhecimento do usuário em subgrupos [Chin 1989]; (ii) as ações que o usuário realiza, ou seja, seu histórico de interação com o sistema ubíquo. O histórico de interação do usuário, chamado de trilha (do inglês *trail*) [Levene 2002], é uma coleção de locais visitados, com informações de contexto associadas, e que segue uma ordem de visitação. As trilhas representam a interação do usuário com o sistema ubíquo, assim como a navegação do usuário na web representa a interação em sistemas de hipermídia adaptativa [Palazzo 2002]. Um registro de trilhas fornece a navegação do usuário em uma sessão, seja ela virtual ou física, e o conjunto de trilhas do usuário fornece um modelo de suas ações [Levene 2002]. Por isso o histórico de trilhas foi identificado como a informação que potencialmente oferece a melhor caracterização do perfil, em termos de comportamento, do usuário em um sistema ubíquo.

Por fim, na web semântica são os agentes que recebem tarefas e, baseados nas preferências do usuário, procuram por informações, comunicam-se com outros agentes e tomam decisões para dar respostas personalizadas ao usuário. Dessa forma o conceito de agentes pessoais da web semântica é utilizado neste trabalho como o software que irá utilizar as ontologias e a lógica para raciocinar sobre o perfil do usuário. O texto está organizado como segue: a seção 2 apresenta o modelo OCtoPUS; a seção 3 apresenta os trabalhos relacionados; a seção 4 apresenta a implementação do protótipo e discute um cenário de utilização; na seção 5 é feita a avaliação do modelo; na seção 6 são feitas as considerações finais.

2. Modelo OCtoPUS

O modelo OCtoPUS possibilita o reconhecimento do perfil do usuário através do raciocínio de classificação a partir de informações sobre a interação do usuário com o ambiente ubíquo. O software responsável por esse raciocínio é um agente pessoal, que foi modelado para consultar informações sobre as trilhas realizadas pelo usuário e inferir o seu perfil, definido na ontologia de domínio da aplicação. Dessa forma o modelo OCtoPUS define as ontologias – desenvolvidas em OWL[2] – e as capacidades do agente de forma genérica para que possa ser especializado de acordo com o sistema no qual ele será utilizado.

2.1. Arquitetura

A Figura 1 mostra a arquitetura do modelo OCtoPUS, onde a separação entre o modelo e o sistema ubíquo fica evidente através da interface entre ambos definida pelos protocolos de comunicação, exibidos na Figura 3. O módulo “Sistema de Trilhas” é o responsável por fornecer o histórico de trilhas do usuário. Já o módulo “Sistema de Perfis” será o consumidor do perfil classificado pelo modelo. A comunicação do sistema com o modelo OCtoPUS é detalhada seção 2.4. Uma ontologia é definida para os conceitos de trilhas, conforme explicado na seção 2.2 de forma a permitir extensão e reuso. Semelhantemente a ontologia de perfil faz parte do modelo para explicitar os conceitos relativos à modelagem de usuário como estereótipos. Essa ontologia está detalhada na seção 2.3.

[2] Web Ontology Language: disponível em: http://www.w3.org/TR/owl-features/

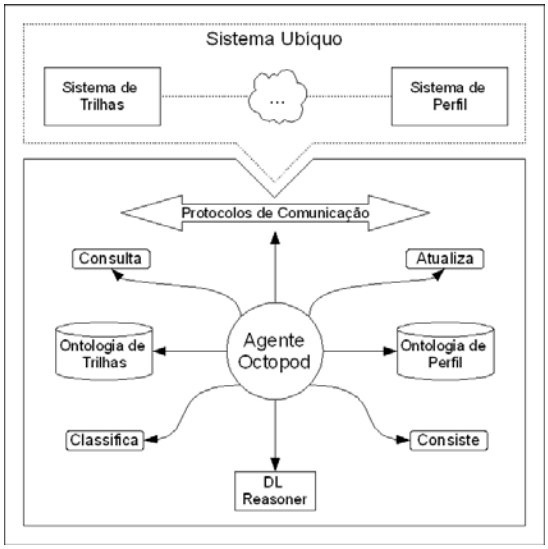

Figura 1 . Arquitetura do Modelo OCtoPUS.

O agente Octopod é o responsável por manipular as informações presentes nas ontologias. Para isso ele possui as capacidades de "consultar" e "atualizar" as ontologias. Para realizar o raciocínio de classificação necessário o agente deve ter acesso a um *reasoner* de lógicas de descrição. Tendo esse acesso ele será capaz de "classificar" e "consistir" o perfil do usuário.

2.2. Ontologia de Trilha

O sistema de trilhas ao qual o modelo terá acesso deve armazenar o histórico de interação do usuário. Esse histórico de interação armazena os objetos aos quais o usuário interagiu. Tais objetos podem ser tanto reais como virtuais. Por exemplo, a interação de um usuário em um museu ou em uma feira é representada por uma trilha de objetos reais, como obras de arte ou pôsteres. Já a interação do usuário através do acesso a materiais disponíveis *online* é formada por uma trilha de objetos virtuais. A ontologia de trilha prevê essas duas possibilidades em um ambiente ubíquo. Ainda, uma trilha pode ser espontânea ou planejada, ou seja, pode ser criada a partir da interação aleatória do usuário ou pode ter sido criada previamente para determinar a seqüência a ser seguida. Para fazer a ligação desses conceitos de trilha com o Sistema de Trilhas foi desenvolvida a ontologia apresentada na Figura 2.

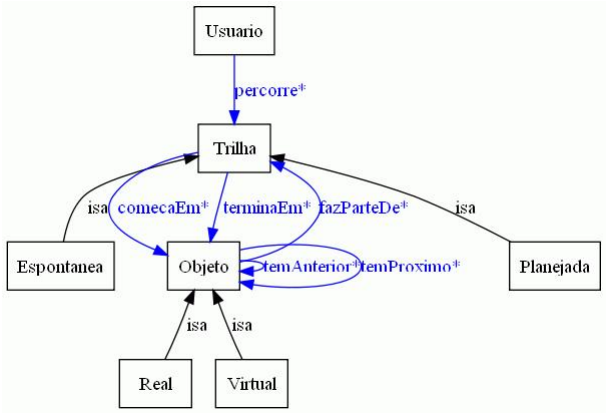

Figura 2 . Ontologia de Trilha.

Uma trilha começa e termina em um objeto. Os objetos fazem parte de uma trilha e estão ligados entre si pelas propriedades **temProximo** e **temAnterior** para fornecer a representação na ontologia da navegação feita na trilha. Durante o percurso de uma trilha o usuário deve ter realizado uma atividade, pois a trilha como um todo representa uma atividade planejada. É essa atividade que ajudará a modelar os usuários do domínio da aplicação. Dessa forma, as trilhas serão usadas como as características para definir os perfis em termos de estereótipos na ontologia de perfil.

2.3. Ontologia de Perfil

O que está sendo sugerido no modelo, através da ontologia de perfil, é que através de um conjunto de atividades definidas pelas trilhas de um determinado usuário, seja possível inferir a quais perfis ele pertence. A ontologia de perfil é definida por uma ou mais trilhas realizadas. Ao estender as ontologias para um domínio de aplicação deve-se especificar cada estereótipo identificado como um conceito complexo em lógica de descrição. A Figura 5 apresenta um exemplo de uma especificação desse tipo.

2.4. Agente Octopod

O agente Octopod foi modelado utilizando a metodologia Prometheus [Padgham 2003], que consiste em três fases: a fase de especificação do sistema foca na identificação das funcionalidades básicas do sistema com suas entradas (percepções), saídas (ações) e quaisquer bases de dados compartilhados; a fase de arquitetura usa as saídas da fase anterior para determinar que agentes o sistema irá conter e como eles irão interagir; por fim, a fase de detalhamento cuida da parte interna de cada agente e como eles irão resolver suas tarefas dentro do sistema. Na Figura 3 são apresentados os protocolos de comunicação definidos para fazer a ligação do agente com os sistemas do ambiente ubíquo. Eles especificam a ordem de interação que deve ser estabelecida com o agente. Além disso, as ontologias do domínio devem ser usadas para validar o vocabulário comum utilizado entre o agente e os sistemas.

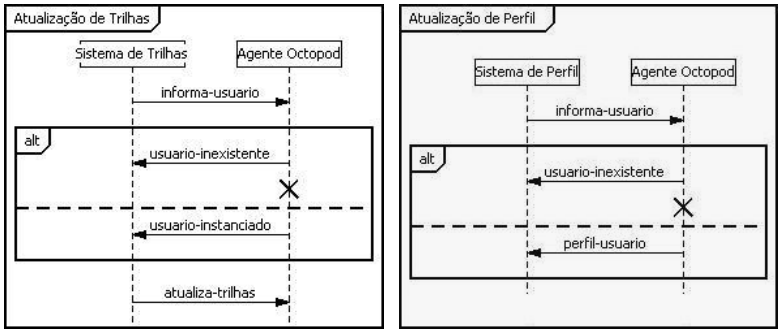

Figura 3 . Protocolos de atualização dos sistemas de Trilhas e Perfil.

A estrutura interna do agente Octopod, apresentada na Figura 4 é definida por módulos, que são: capacidades, eventos internos e estruturas de dados. Cada capacidade deve ser descrita com informações sobre a interface externa da capacidade. Ou seja, quais eventos são entradas e quais eventos são produzidos pela capacidade (e que servem de entrada para outras capacidades). A metodologia Prometheus permite na modelagem do agente definir as capacidades em níveis de complexidade que vão sendo especializados até chegarem ao nível dos planos do agente.

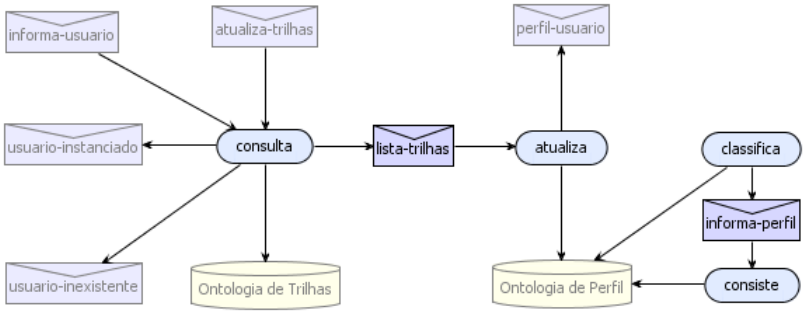

Figura 4 . Diagrama do agente Octopod: os envelopes representam os eventos de entrada e saída em cada capacidade do agente.

As capacidades do agente Octopod foram definidas de forma abrangente com o intuito de fornecer generalidade. Os eventos de entrada e saída foram extraídos dos protocolos de comunicação apresentados na Figura 3. São eles que ativam as capacidades do agente e que são retornados ao ambiente. Também estão representados no diagrama os eventos trocados entre as capacidades, como "lista-trilhas" e "informa-perfil". Os protocolos de comunicação definem a interação do Octopod com outros agentes possíveis. Essa definição fica a critério da aplicação e são discutidas na fase de prototipação e avaliação, apresentadas na seção 4.

3. Trabalhos Relacionados

Os trabalhos relacionados possuem ao menos uma das características utilizadas pelo modelo OCtoPUS, pois o foco do modelo está na utilização das tecnologias da web semântica e também no uso de trilhas. Dessa forma a Tabela 1 mostra os projetos avaliados e suas características relativas à como as informações são representadas, qual técnica de inferência utilizada e suas aplicações.

Tabela 1 - Comparação entre trabalhos

Modelo	Representação	Inferência	Aplicação
FLAME2008	Ontologias Frame-Logic	Raciocínio de subclassificação e instâncias	Serviços Web / Dispositivos móveis / Grandes Eventos
Grafos Conceituais	Grafos Conceituais e conjuntos *fuzzy*	Teoria de Atribuição de Massa	Sistemas de Recomendação
PAPI/LIP	Textual	Similaridade de Textos	Educação
PeLeP	Textual	Fator de Certeza	Educação Ubíqua
OCtoPUS	OWL	DL	Sistemas Ubíquos

O projeto FLAME2008 [Weißenberg 2004] não trata diretamente do perfil de usuários, mas utiliza as tecnologias semânticas para a representação da situação do usuário com o fim de inferir os serviços de forma personalizada. É um trabalho que utiliza ontologias e raciocínio de forma semelhante ao modelo OCtoPUS. Já o trabalho de Baldwin (2000), que utiliza grafos conceituais, relaciona-se mais com o modelo proposto nesse trabalho, pois foi idealizado de forma genérica. Através do *log* de interação do usuário ele constrói e aprimora protótipos de usuários. A diferença entre os

trabalhos é que no OCtoPUS os estereótipos são definidos em lógica de descrição e não através de clusterização, permitindo uma melhor definição dos perfis de acordo com o domínio. Os IMS-LIP (2001) e PAPI (2002) são as principais especificações de perfis que incluem informações detalhadas para personalização: histórico do aprendiz, atividades atuais e objetivos. Contudo, eles provem somente um *framework* sintático para armazenar tais informações, fornecendo uma estrutura para descrições textuais. Dessa forma, o processamento textual do perfil é limitado, pois essas seções dos perfis são projetadas para serem lidas por pessoas ao invés de dispositivos computacionais. Sem qualquer formalismo ou padrão para expressar as informações dos aprendizes qualquer tentativa dos dispositivos entenderem essa informação será baseada em técnicas de similaridade de textos não estruturados. O uso das tecnologias da web semântica é uma forma de definir tais vocabulários de forma que a informação seja processada tanto por humanos quanto por máquinas.

Por fim, o modelo PeLeP [Levis 2008] é o que mais se aproxima do modelo proposto, pois foi idealizado para estar conectado a um sistema ubíquo e utiliza o histórico de interação dos aprendizes para aperfeiçoar o perfil. Porém, por estar baseado no PAPI/LIP também dificulta o processamento das informações de perfil por dispositivos computacionais. Esse diferencial o modelo OCtoPUS oferece ao usar as ontologias baseadas em lógica de descrição para a representação das informações do usuário.

4. Protótipo

O protótipo desenvolvido armazena as atividades dos alunos da ComGRefe[3] durante o curso de forma a sugerir suas habilidades e competências adquiridas. As atividades dos alunos são as trilhas percorridas através dos objetos de aprendizagem cadastrados pelos professores. As habilidades desenvolvidas são definidas a partir das trilhas cadastradas. E as competências são uma especialização do conceito de perfil da ontologia e são definidas a partir das habilidades existentes, ou seja, uma competência é definida por uma ou mais habilidades.

Inicialmente foram cadastradas na ontologia as trilhas identificadas no cenário. Em seguida foram especializadas as ontologias para o domínio do protótipo. O conceito Perfil foi especializado nos conceitos Competência e Habilidade. Por sua vez, o conceito Habilidade foi especializado de acordo com as duas disciplinas do cenário: Sistemas Operacionais e Programação. Dando seqüência ao processo de especialização, os conceitos SisOp e Programação foram derivados em cada uma das habilidades que os alunos devem adquirir no decorrer das disciplinas. Essas habilidades foram definidas de acordo com as trilhas cadastradas. Por exemplo, a Habilidade em Processos é definida pelo conceito *percorre has trilha_silberschatz_cap4*. Já a Habilidade em Programação Orientada a Objetos é definida pelos conceitos *percorre has trilha_java* e/ou *percorre has trilha_C++*.

[3] Graduação de referência em Engenharia da Computação, disponível no site: http://www.unisinos.br/graduacao/bacharelado/eng_comp/

Tendo definidas as habilidades, o próximo passo foi a definição das competências que os alunos deveriam adquirir nessas duas disciplinas. Na Figura 5 são apresentadas as três especializações do conceito Competências. Por exemplo, o conceito Programação Avançada foi definido em lógica de descrição como: *possui some Programação and possui some (h_concorrencia or h_escalonamento or h_processos)*. Isso quer dizer que foi identificado o perfil de aluno em Programação Avançada como sendo aquele que possui alguma habilidade em Programação e alguma habilidade em concorrência, escalonamento ou processos.

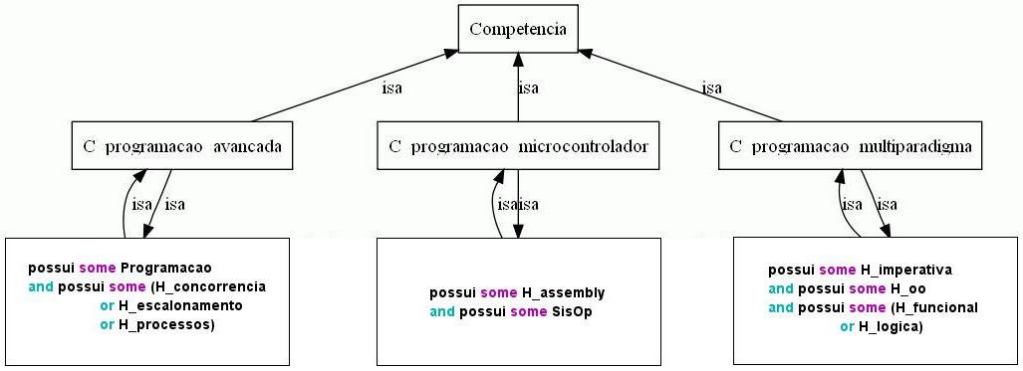

Figura 5 . Especialização das Competências.

Após o processo de especialização das ontologias foram criadas as instâncias dos alunos e informadas quais trilhas eles percorreram, simulando o papel do Sistema de Trilhas do modelo OCtoPUS. Para fazer essa simulação foram definidas as trilhas percorridas de forma a testar se cada aluno seria corretamente classificado em cada uma das competências. O que se tem de fato na ontologia são as trilhas percorridas pelos alunos. As outras informações, como as habilidades e competências, são inferidas pelo agente Octopod. Para representar o Professor e os Alunos foram criados dois agentes. Esses agentes fazem alusão aos Sistemas de Trilha e de Perfil, pois são eles que vão interagir com o agente Octopod. Para fins de avaliação do protótipo esses dois agentes conseguem interagir facilmente com o agente Octopod.

O agente Professor envia uma mensagem ao agente Octopod solicitando a ativação do plano *consulta_habilidades(Aluno)*.

```
+!consulta_habilidades(Aluno): aluno(Aluno)[o(t)]
    <-  +aluno(Aluno);
        jasdl.ia.get_types(Aluno, t, true, Types);
        createAndShowGUI(I);
        showList(Types,"Habilidades e Competencias: ").

+!consulta_habilidades(Aluno): not aluno(Aluno)[o(t)]
    <-  .print("Aluno não existe").
```

Figura 6 . Consulta das habilidades do Agente Octopod.

Se o aluno informado pelo agente Professor existe na ontologia, então o plano é executado. Essa verificação ocorre no contexto do plano onde o predicado *aluno(Aluno) [o(t)]* está anotado conforme define o JASDL (Jason Agent-Speak Description Logics) [Klapiscak 2008] com a ontologia de trilhas, definida pela letra *t*. Se *Aluno* for unificado com a instância na ontologia, então o plano é executado. Se o aluno não for uma instância da ontologia, então o plano com o contexto *not aluno(Aluno) [o(t)]* resulta

verdadeiro e a mensagem de aluno inexistente é exibida. Durante o processo de classificação o agente atualiza automaticamente a relação *aluno possui habilidade* na ontologia. Ao perceber essa adição de crença o agente ativa o plano mostrado na Figura 7.

```
+atualiza_habilidade(Habilidade): habilidade(Habilidade)[o(t)]
    <-  ?aluno(Aluno);
        +possui(Aluno,Habilidade)[o(t)];
        jasdl.ia.get_types(Aluno, t, true, Types);
        updateList(Types,"Habilidades e Competencias: ").
```

Figura 7 . Atualização de habilidades do Agente Octopod.

A cada nova atualização de habilidade o agente verifica se a habilidade informada pelo ambiente está presente na ontologia e retorna sua instância. No desenvolvimento do protótipo foram utilizadas tecnologias da web semântica e agentes inteligentes. Para a criação das ontologias foi usada a ferramenta Protege [Horridge 2004] que possui suporte a linguagem OWL. Como mecanismo de raciocínio, já embutido no JASDL, foi utilizado o Pellet[4] que oferece raciocínio para a lógica de descrição da OWL DL, baseado no algoritmo *trableaux*. Além disso, a versão 4.0 do Protege traz o Pellet como seu mecanismo de raciocínio padrão.

5. Avaliação

De forma a avaliar o modelo proposto nesse trabalho são usadas as premissas apresentadas por Kobsa (2007), como "serviços" que sistemas de modelagem de usuários devem prover. Para representar as características comuns aos usuários em subgrupos, ou perfis, foi especializada a ontologia de perfil. Dessa forma as habilidades e competências foram definidas de acordo com características que agrupam alunos. Outra premissa é classificar o usuário em um ou mais perfis. Para avaliar esse quesito foi executado o protótipo desenvolvido. Como resultado para a consulta feita pelo agente *Professor* para o *aluno1* foi obtida a classificação apresentada na Figura 8.

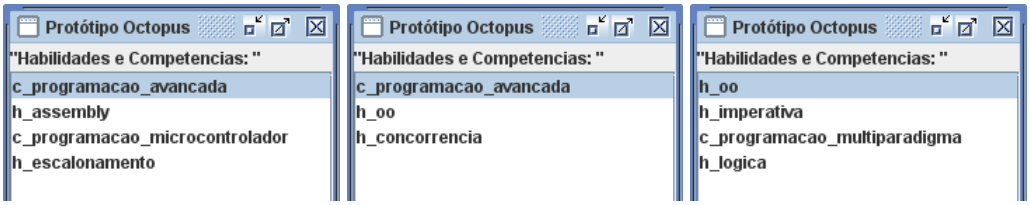

Figura 8 . Classificação dos alunos 1, 2 e 3.

Conforme dito na seção anterior, foi afirmado que o *aluno1* percorreu as trilhas *assembly* e *silberschatz_cap6*. Com base nisso o agente Octopod classificou o aluno com as habilidades *h_assembly* e *h_escalonamento*. Em seguida o agente atualizou a ontologia com essas habilidades e reclassificou o aluno, retornando que ele também possui as competências de Programação Avançada e Programação de Microcontroladores. O aluno foi classificado em dois perfis de competências, pois na descrição do conceito de Programação Avançada qualquer usuário que possuir algum tipo de habilidade em programação e também habilidade em concorrência, escalonamento ou processos é um Programador Avançado. E também em Programação

[4] Código aberto em Java disponível em: http://pellet.owldl.com/

de Microcontroladores, pois a sua habilidade em *assembly* e alguma habilidade em Sistemas Operacionais o capacitam a ser classificado nessa competência. Pelo mesmo motivo que o *aluno1* foi classificado em Programação Avançada o *aluno2* também o foi. Porém não foi classificado em Programação de Microcontroladores por não possuir habilidade em *assembly*, ou seja, a trilha *assembly* não foi percorrida. O *aluno3* por ter percorrido as trilhas de C++, Pascal e Prolog foi classificado como possuidor de habilidades em programação Orientada a Objetos, Imperativa e Lógica. Após atualizar a ontologia com essas habilidades o agente Octopod classificou o *aluno3* como pertencente à competência em Programação Multiparadigma.

Dessa forma o objetivo de classificar o usuário em um ou mais perfis foi atendido pelo protótipo. A premissa seguinte apresentada por Kobsa (2007) é a de representar o comportamento do usuário, em particular a interação com o sistema. Esse objetivo é atendido no protótipo através da Ontologia de Trilhas. Essa ontologia possui os conceitos de trilhas como uma seqüência de Objetos reais ou virtuais. Uma trilha pode ser planejada ou espontânea. As trilhas do protótipo são trilhas planejadas e são informadas diretamente na ontologia. O modelo provê o conceito de trilhas espontâneas, onde o usuário interage aleatoriamente com o sistema. Porém, testar essa representação demandaria uma integração com um sistema ubíquo real ficando fora do escopo do protótipo, que é um cenário simulado.

As duas últimas premissas: raciocinar sobre o usuário com base na história de sua interação com o sistema e manter a consistência do perfil estão implícitos no processo de classificação, pois o agente Octopod, através do JASDL, teve que raciocinar baseado nas trilhas e nas habilidades do usuário para determinar suas competências. Além disso, caso alguma inconsistência seja identificada no processo de atualização das habilidades o JASDL não realiza a atualização.

6. Considerações Finais

A computação ubíqua abre novas possibilidades para antigas questões da computação. Na área de personalização e modelagem do usuário, por exemplo, a forma como o usuário interage com um sistema ubíquo é mais dinâmica e envolve questões de localização física. Logo, novos modelos para identificar o perfil do usuário devem surgir como uma conseqüência natural.

O modelo OCtoPUS identificou como principal característica de interação em um ambiente ubíquo as trilhas do usuário. Assim, vislumbrando aplicações práticas, o modelo pretende fornecer uma abstração capaz de identificar o usuário baseado em suas ações. Embora o protótipo tenha explorado apenas as trilhas virtuais de um usuário, as bases para o desenvolvimento de trilhas reais estão presentes no modelo. O usuário pode ser classificado baseado em locais em que visitou, contemplando assim o aspecto ubíquo. Passar de um cenário virtual para um real depende apenas de desenvolver o protótipo usando tecnologias de localização, que são mais difíceis de encontrar em pleno funcionamento para integração.

A grande vantagem do modelo OCtoPUS está na descrição semântica dos conceitos envolvidos. Essa abordagem facilita a integração entre o usuário e os dispositivos computacionais e permite que o conteúdo representado seja útil pra ambos, correspondendo à premissa fundamental da Web Semântica.

Referências

Antoniou, G. and van Harmelen, F. (2004) “A semantic web primer”, The MIT Press, 2004.

Baldwin, J., Martin, T., Tzanavari, A. (2000) “User modeling using conceptual graphs for intelligent agents”, In B. Ganter & G.W. Mineau (Eds.), Conceptual structures: logical, linguistic, and computational issues: 8th international conference on conceptual structures, Springer, 2000.

Brusilovsky, P., Kobsa, A., Nejdl, W. eds. (2007) “The Adaptive Web: Methods and Strategies of Web Personalization”, Berlin, Heidelberg, New York: Springer-Verlag.

Chin, D. (1989). “KNOME: Modeling what the user knows in UC”, In Kobsa, A., & Wahlster, W. (Eds.), User models in dialog systems, pp. 74–107. Springer Verlag, Berlin, 1989.

Horridge, M., Knublauch, A., Rector, R., Stevens, C. (2004) “A Practical Guide To Building OWL Ontologies Using the Protégé-OWL Plugin and CO-ODE Tools”, Technical Report, Ed. 1.0, The University Of Manchester.

IEEE LTSC (2002). “PAPI Learner, Draft 8 Specification”, Disponível em: [http://edutool.com/papi/]. Acesso em: setembro de 2008.

IMS Global Learning Consortium (2001). “IMS Learner Information Packaging Information Model Specification’, Final Specification Version 1.0, Disponivel em: [http://www.imsglobal.org/profiles/lipinfo01.html]. Acesso em: setembro de 2008.

Klapiscak, T., Bordini, R. (2008) “JASDL: A Practical Programming Approach Combining Agent and SemanticWeb Technologies”, In: AAMAS Workshop: Declarative Agent Languages and Technologies (DALT), 2008, Estoril. Baldoni, Endriss, Omicini and Torroni (Eds) Declarative Agent Language Technologies VI - Sixth International Workshop, DALT 2008 - Selected and Revised Papers - LNAI. Berlin: Springer, 2008.

Levene, M., Peterson, D. (2002) “Trail Record and Ampliative Learning”, Research Report BBKCS-02-01, School of Computer Science and Information Systems. University of London, 2002.

Levis, D., Barbosa, J., Pinto, S., Barbosa, D. (2008) “Aperfeiçoamento Automático do Perfil do Aprendiz em Ambientes de Educação Ubíqua”, Revista Brasileira de Informática na Educação, v. 16, p. 29-41, 2008.

Padgham, L., Winikoff, M. (2003) “Prometheus: A Methodology for Developing Intelligent Agents”, LNCS, vol. 2585, pp. 174--185. Springer 2003.

Palazzo, L. (2002) “Sistemas de Hipermídia Adaptativa”, XXI Jornada de Atualização em Informática. XXII Congresso da SBC. Florianópolis, julho de 2002.

Weißenberg, N., Voisard, A., Gartmann, R. (2004) “Using Ontologies in Personalized Mobile Applications”, In Pro-ceedings Intl. ACM GIS Conference, ACM Press, pp. 2-11.