Recuperação de Informações Web para um Ambiente de Aprendizagem Computadorizada: uma Abordagem Multiagentes

Gentil Serra Jr.[1], Roosevelt Oliveira[2], Walker Rabelo[1], Sofiane Labidi[2]

[1]Departamento Acadêmico de Informática – Centro Federal de Educação Tecnológica do Maranhão (CEFET-MA)
Monte Castelo - São Luis - MA – Brazil - CEP 65025-001

[2]Departamento de Engenharia da Eletricidade – Universidade Federal do Maranhão (UFMA)
Campus do Bacanga – São Luís – MA - Brazil
gentil@cefet-ma.br, labidi@ufma.br

Abstract. *In this article, a system of multiple agents for information retrieval in a base of non semantic data is proposed. These software agents have the objective of aiding the teaching-learning process in computer-based learning environment, in what's related to the information retrieval in the Web. For that, ontologies are used to facilitate the communication among the agents in the acting of their tasks and the research of the subject and its context in the Web. The agents' specification and the implementation of some of these agents, exemplified through its use in the Computer-Based Learning Environment Netclass , are described.*

Resumo*. Neste artigo, propõe-se um sistema de múltiplos agentes para recuperação de informações em base de dados não semânticos. Esses agentes de software têm como objetivo auxiliar o processo de ensino-aprendizagem em ambientes de aprendizagem computadorizada, no que diz respeito à recuperação de informações na Web. Para isso, utilizam-se ontologias para facilitar a interoperabilidade entre os agentes no desempenho de suas tarefas e na contextualização do assunto a ser pesquisado na Web. Descreve-se a especificação dos agentes e apresenta-se a implementação de alguns destes, exemplificados através de seu uso no Ambiente de Aprendizagem Computadorizada Netclass.*

1. Introdução

A grande quantidade e diversidade de informações disponíveis na Web configura-se como um repositório relevante para uso em Ambientes de Aprendizagem Computadorizada [Labidi et al. 2004]. Entretanto, em razão de que esse repositório de informações é estruturado para a apresentação apenas a usuários humanos, a maior parte de seu conteúdo torna-se de difícil acesso às máquinas.

Em razão, principalmente, da falta de semântica na Web, a recuperação de informações relevantes que podem ser aproveitadas em ambientes de aprendizagem torna-se uma tarefa difícil. Segundo [Freitas 2002], a solução é: i) dotar as redes de inteligência, fazendo com que as páginas possuam uma semântica mais clara e definida; ou ii) dotar os sistemas de inteligência e autonomia para percorrer e selecionar

informação relevante. As ontologias representam um papel fundamental em ambos os tipos: no primeiro, como elemento de comunicação entre agentes, organização, reuso e disseminação de conhecimento; e no segundo, como parte intrínseca das linguagens propostas para a definição de páginas com semântica.

Considerando que uma ontologia visa desenvolver um conjunto de regras que possibilitem a inferência de forma que a máquina possa, através do acesso a essas regras e a uma coleção de dados e metadados, abstrair um significado semântico das informações disponibilizadas [Uschold 1999] [Rothenfluhh 1996], é pertinente o uso de ontologias em Ambientes de Aprendizagem, no que diz respeito à representação do conhecimento e à recuperação de informações em bases não semânticas.

Utiliza-se o projeto NetClass para exemplificar o uso de ontologias em ambientes de aprendizagem na recuperação de informação em bases de dados não semânticos, como, por exemplo, a Web. O objetivo do NetClass é, através da junção das idéias que fundamentam os Sistemas Tutores Inteligentes aos recursos de uma rede e ao paradigma de aprendizagem cooperativa, definir, projetar e implementar um Ambiente de Ensino Inteligente Cooperativo Computadorizado [Labidi et al. 2003a].

Neste artigo, descreve-se um sistema baseado em agentes que, através do uso de ontologias, disponibiliza informações adequadas para o aprendiz, de acordo com as estratégias de ensino-aprendizagem. Essas informações são obtidas do Modelo do Domínio (Agentes de Domínio) ou Recuperadas da Web (Agentes de Busca). Primeiramente, destaca-se a arquitetura do Ambiente NetClass (seção 2), depois, apresenta-se a especificação dos Agentes de Busca (seção 3), em seguida, descreve-se a implementação desses agentes (seção 4). Finalmente, apresentam-se as considerações finais.

2. Ambiente NetClass

O principal objetivo do Projeto NetClass é a concepção de um ambiente interativo de ensino-aprendizagem cooperativo à distância baseado em uma arquitetura com múltiplos agentes humanos e artificiais. Esse ambiente integra alunos, professores e sistema computacional num espaço que serve para promover o desenvolvimento de atividades cooperativas [Labidi et al. 2003a], favorecendo a aprendizagem do aluno (individual ou em grupo) através de resolução de problemas, tendo a assistência tanto do sistema tutor quanto de professores, de forma individualizada e adaptada às suas necessidades [Labidi et al. 2003b].

No NetClass, os alunos são divididos em grupos distintos, chamados de áreas cooperativas. Eles cooperam e aprendem a partir da interação dentro de seus próprios grupos (interação intragrupo), com os agentes artificiais, com o professor, e com outros grupos (interação intergrupo), através da utilização de recursos multimídia e de tecnologias de redes. Em uma Sessão de Aprendizagem ocorre a interação entre várias áreas cooperativas interligadas através da rede, objetivando o aprendizado por parte dos alunos. Nas etapas da aprendizagem, vários tipos de atividades poderão ser desempenhados pelos alunos e grupos. As atividades de ensino-aprendizagem, no sistema NetClass, são classificadas em seis tipos: i) Preparação de Grupos, ii) Apresentação do Conhecimento, iii) Assimilação do Conhecimento, iv) Aplicação do Conhecimento, v) Avaliação do Grupo, e iv) Avaliação Individual. Cada tipo de atividade tem funções específicas que são desempenhadas com o uso de estratégias pedagógicas apropriadas, escolhidas de acordo com o modelo do aprendiz [Serra Jr.

2001].

O Ambiente NetClass é baseado em uma arquitetura multiagentes composta por agentes inteligentes artificiais (Agentes de Busca Web, Agente Tutor, Agente Estrategista, Agente de Modelagem do Aprendiz e Agentes de domínio) e humanos (alunos/grupos, professor e engenheiro do conhecimento) [Labidi 2000] [Serra Jr 2000]. Para construção dos agentes artificiais utiliza-se o Framework JADE (Java Agent Development framework) [JADE 2005].

A seguir, apresenta-se uma breve descrição de cada um dos agentes artificiais previstos na arquitetura. A sociedade de Agentes de Busca Web, objeto de estudo deste trabalho, será descrita na próxima seção.

2.1. Agente Tutor

O Tutor Artificial tem o papel de apresentar os conhecimentos aos alunos, dirigir a fase de Aplicação do Conhecimento, adotar as melhores estratégias pedagógicas e realizar a avaliação dos alunos/grupos. Ele também tem a responsabilidade de controlar as interações dos grupos de estudantes com o sistema durante o progresso da aprendizagem e executar tarefas ligadas à segurança e permissões. As ações do Tutor dependem do comportamento dos aprendizes. Ele interage com o Agente Estrategista e o Agente de Modelagem do Aprendiz para selecionar as estratégias de ensino mais adequadas. Portanto, o Tutor Artificial determina o que será ensinado, quando e como será feito.

2.2. Agentes de Domínio

O domínio de conhecimento a ser ensinado no NetClass é organizado em uma base de conhecimentos compartilhada pelos Agentes de Domínio. Cada Agente de Domínio mantém um domínio específico, chamado de unidade pedagógica. As unidades pedagógicas são organizadas de acordo com a estrutura pedagógica estabelecida para o domínio a ser ensinado.

Um Agente de Domínio, a partir das ontologias e instâncias, mantém uma unidade pedagógica como sendo uma visão seqüenciada dos conceitos que devem ser apresentados ao aprendiz [Costa 1997]. Cada conceito poderá estar associado a uma instância. Cada instância poderá possuir vários recursos. Recursos são os itens de multimídia (texto, áudio, vídeo) utilizados para apresentar ou reforçar um conhecimento. As instâncias são mantidas na base de conhecimentos.

2.3. Agente de Modelagem do Aprendiz

A modelagem do Aprendiz pode ser vista como o processo de aquisição, representação e manutenção de informações a respeito dos alunos, durante a aprendizagem. A representação é chamada de Modelo do Aprendiz. O Modelo do Aprendiz é utilizado para auxiliar o sistema a personalizar suas ações durante o processo de aprendizagem.

O Agente de Modelagem do Aprendiz é um componente pró-ativo de software que adquire, representa e mantém as informações sobre cada aprendiz [Serra Jr et al. 2000].

2.4. Agente Estrategista

O NetClass possui a possibilidade de múltiplas estratégias pedagógicas, como também introduz o conceito de Agente Estrategista que interage com o Agente de Modelagem do Aprendiz para selecionar a estratégia mais adequada de ensino com base no modelo do aprendiz. Por exemplo, o Agente Estrategista decide quais unidades de conhecimento e em que formato o conteúdo será apresentado aos aprendizes.

3. AGENTES DE BUSCA

O Agente Tutor poderá apresentar informações ao aprendiz com base em duas fontes: Agentes de Domínio e Agentes de Busca. Os Agentes de Domínio possuem uma base de conhecimentos baseada em ontologias. Os Agentes de Busca utilizam as ontologias para recuperar informações da Web. Entretanto, os Agentes de Busca não armazenam recursos Web, mantêm apenas endereços de páginas que podem ser acessados pelo aprendiz.

O conjunto dos Agentes de Busca formam a Sociedade de Busca NetClass. Um Agente de Busca, uma vez instanciado, utiliza mecanismos de busca e interage com outros agentes da Sociedade com a finalidade de realizar busca na Web. As informações recuperadas poderão ser apresentadas ao aprendiz, através do Agente Tutor. Observe na Figura 1 que o aprendiz interage com o Tutor através da Interface do Aprendiz.

Conforme pode ser observado na Figura 1, muitas são as classes envolvidas no processo de ensino-aprendizagem, as principais são: `AgenteTutor`, `AgenteBusca`, `AgenteDominio`, `AgenteModelagemAprendiz`, etc. A `AgenteBusca` interage, principalmente, com a `AgenteTutor` com a finalidade de atender às demandas de busca na Web, feitas no andamento do processo de aprendizagem mediado pelo Tutor Artificial.

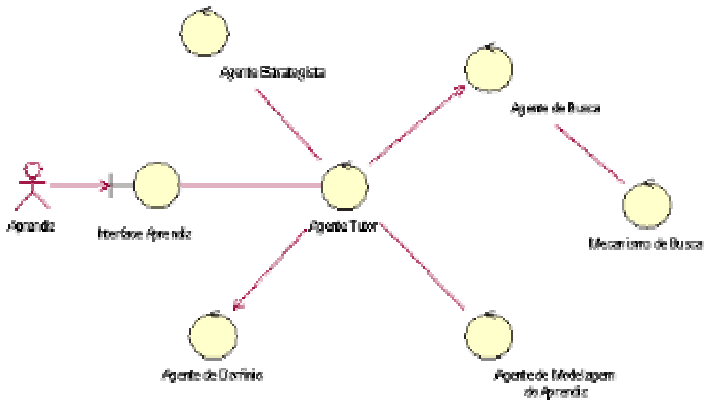

Figura 1. Visão Geral – Principais Classes do NetClass

3.1. Arquitetura da Busca

Os Agentes de Busca se conectam a mecanismos de busca, como, por exemplo, Google, Altavista, Excite e outros, aproveitando os seus índices, uma vez que não é necessário sempre indexar ou percorrer toda a Web para recuperar informações. A arquitetura de busca do NetClass é apresentada na Figura 2.

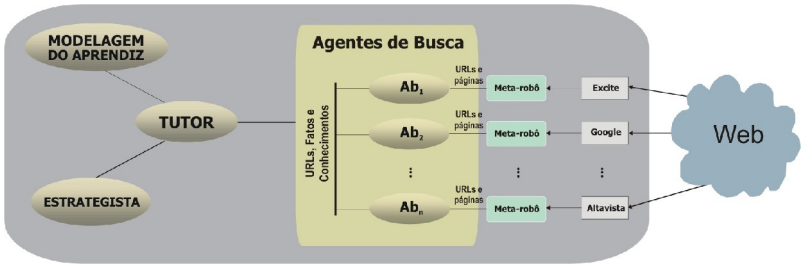

Figura 2. Arquitetura de busca do NetClass.

Um Agente de Busca efetua uma consulta ao mecanismo de busca com palavras-chave. O resultado são endereços de páginas Web (índices), algumas com conteúdo relevante. A partir das ontologias, regras e fatos mantidos pelo agente, este é capaz de discernir sobre os índices que são úteis para o seu domínio específico e os que não o são. Para isso, são utilizados métodos de categorização de recuperação de informação.

Os resultados da busca poderão conter páginas de conteúdo desprezível para um agente, mas relevante para um outro, sendo assim, os Agentes de Busca podem comunicar-se com outros agentes a fim de trocar informações com um agente responsável por um outro domínio específico. Dessa forma, os Agentes de Busca trocam mensagens contendo regras de reconhecimento e fatos (conhecimento dos agentes), além das URLs (dados).

3.2. O Papel do Agente Tutor na Recuperação de Informação

O conhecimento mantido no NetClass pode ser visto a partir do conjunto das sociedades de agentes de software. Um agente, responsável por um domínio específico, é parte do todo e deve ser visto como uma entidade social. Portanto, percebe-se a necessidade de um agente mediador, chamado no NetClass de Agente Tutor, com capacidade de: discernir o que será apresentado ao aprendiz; apresentar informação em um formato compreensível; e notificar ao aprendiz a informação que lhe pode ser útil.

A partir das interações com as sociedades (Domínio e Busca) e demais agentes (Modelagem do Aprendiz e Estrategista), o Tutor adota as decisões do andamento do curso.

É mantida pelo Agente Tutor uma base de dados ontológica com a relação dos nomes de todos os Agentes de Busca e seus respectivos domínios específicos. Através desta, o Tutor tem condições de identificar o agente responsável pela busca de uma determinada classe de páginas (subdomínio). O resultado, que pode ser obtido do Tutor pelo Agente de Busca correspondente ao domínio, será apresentado ao aprendiz como uma relação de endereços que podem ser acessados através da Interface do Aprendiz.

Apresenta-se na Figura 3 um cenário básico das interações entre o Tutor e um Agente de Busca responsável por um subdomínio. Nesse aspecto dinâmico dos objetos, pode-se perceber que o Tutor toma a iniciativa de exibir ao Aprendiz uma relação de *links* relacionados ao assunto que está sendo estudado.

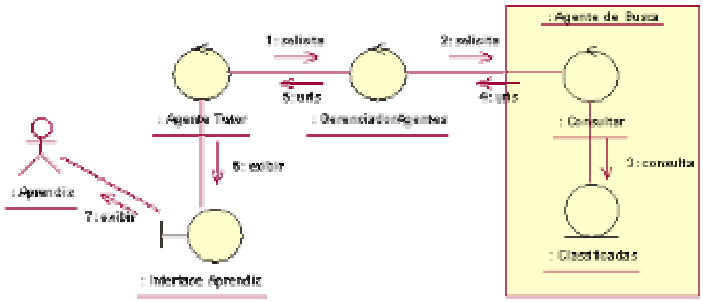

Figura 3. Interações no Cenário Básico AgenteTutor-AgenteBusca

3.3. Tarefas dos Agentes

Com base nas tarefas de agentes descritas em [Freitas 2002, p. 90-95], estabeleceu-se que, durante a busca, cada agente desempenha três tarefas consecutivas no processamento de cada *link* (página) encontrado: Validação, Pré-processamento e Classificação (cf. Figura 4). A tarefa de Pré-processamento depende dos resultados da tarefa de Validação em razão de que páginas inválidas não precisam ser processadas.

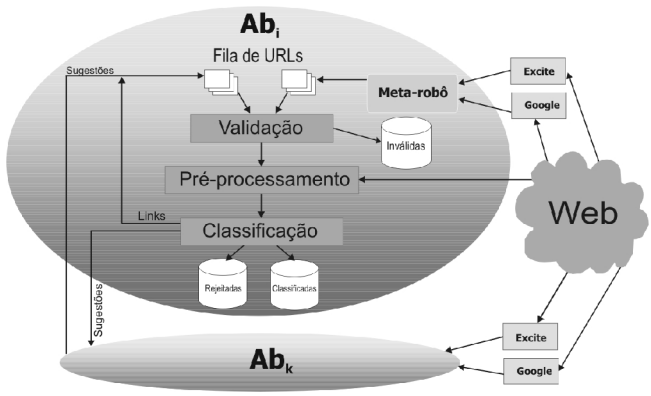

Figura 4. Tarefas de um Agente de Busca

Cada agente manipula três filas de URLs durante sua execução: duas são processadas pelo próprio agente, uma de baixa e outra de alta prioridade, e a terceira contém sugestões de *links* para serem enviadas aos outros agentes.

Ao final do processamento, caso a página não consiga ser reconhecida e classificada, ela estará categorizada como LISTA (Lista Re-avaliável, ou seja, tratada como uma lista de novas URLs), RECOMENDAÇÃO (página reconhecida como pertencente à outra classe de páginas) ou REJEITADA (Mensagem ou Lixo). As próximas subseções apresentam descrições detalhadas das fases de processamento.

Validação

Nesta fase, concretiza-se a recomendação para os mecanismos de busca só recuperarem páginas escritas em formatos que os agentes possam processar. Com efeito, a validação elimina trabalho redundante, verificando se o *link* da página já está presente na base de *links*, seja ela válida, rejeitada ou classificada, indicando se já fora processada. Mesmo inválidas, as páginas ficam armazenadas na base de dados

(endereço, data de acesso), porque o mecanismo de busca, freqüentemente, depara-se com "ponteiros" repetidos, só recuperando-os novamente se a data de última modificação foi alterada.

A maior parte das regras de validação é reusável por todos os agentes. Mas regras específicas podem ser implementadas, referenciando outros campos, tais como data e tamanho, ou mesmo tratando outros protocolos.

Pré-processamento

A fase de pré-processamento tem por meta representar as páginas de diversas maneiras, com dados extraídos delas, aplicando, se necessário, técnicas de recuperação de informação e processamento de linguagem natural. Esses dados (protocolo, formato, data, tamanho etc) são repassados para o motor de inferência para que possam auxiliar a fase posterior.

Classificação

Durante esta fase, o agente classifica as páginas como: LISTA, RECOMENDAÇÃO (página reconhecida como pertencente à outra classe de páginas), REJEITADA ou CLASSIFICADA.

É nesta fase que, a partir das regras, pode-se definir uma página como sendo CLASSIFICADA, ou melhor, reconhecida como membro da classe processada pelo agente. As páginas que contêm conteúdo diferente do esperado para o subdomínio tratado pelo agente poderão ser consideradas: i) desprezíveis (REJEITADAS), caso seu conteúdo seja considerado irrelevante para a Sociedade de Agentes; ii) uma RECOMENDAÇÃO de índice para outro agente; ou iii) uma LISTA de índices para ser processada pelo próprio agente. O reconhecimento de LISTAS deve apresentar boa precisão, caso contrário, uma infinidade de endereços de importância duvidosa, advindos de falsas listas, serão inseridos na fila de processamento de alta prioridade.

Para verificar se a página processada é deste ou de um outro agente (recomendação ou não), palavras-chave da página são comparadas a uma lista de termos, associados aos agentes da Sociedade. Se uma das palavras-chave está contida em um dos termos da lista, testa-se se o termo inteiro está na página e, então, define-se se de fato a página é do agente processador ou não. Existe ainda, para cada agente, uma lista de termos que não devem estar contidos na página. Por exemplo, para o assunto vertebrados não se deseja ter na página termos como "preço", "compre" etc, sob pena da página ser categorizada como rejeitada.

As páginas reconhecidas como sendo do sub-domínio do agente têm seus atributos armazenados em uma base de índices (páginas classificadas), com um nível de relevância determinado pelo agente. Quando consultado, o agente irá responder com essas informações ao Tutor Artificial. As páginas categorizadas como "desprezíveis", são armazenadas na base de páginas rejeitadas para que não sejam analisadas em uma outra oportunidade, evitando-se desperdício de recursos computacionais. As páginas categorizadas como Listas têm as URLs sendo enviadas para a Lista de URLs de Alta Prioridade. As "páginas recomendações" têm seus endereços enviados como sugestões para outros agentes correspondentes.

Para detalhar melhor o processo de classificação, apresenta-se na Figura 5 o cenário básico das interações para o reconhecimento de uma página como membro do domínio (classificada) de um Agente de Busca.

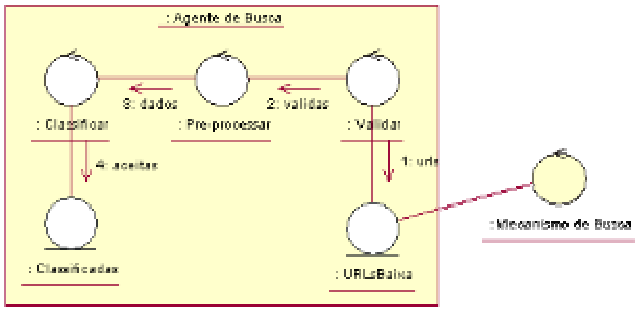

Figura 5. Interações na classificação - Cenário básico

Os passos que compõem o cenário, apresentado na Figura 5, são: i) o objeto da classe `Validar` obtém a relação de Índices (URLs) de Baixa Prioridade; ii) esse objeto envia as URLs válidas para o pré-processamento; iii) o objeto da classe `PreProcessamento` extrai os atributos pertinentes e os envia para o processo de classificação; e iv) as páginas reconhecidas como pertencentes ao domínio do agente são armazenadas como classificadas.

4. IMPLEMENTAÇÃO DOS AGENTES DE BUSCA

Apresenta-se um protótipo de implementação de dois Agentes de Busca com a finalidade de recuperar da Web e manter URLs pertinentes ao domínio de Biologia, especificamente, ao assunto de vertebrados e invertebrados.

Para realizar a implementação, escolheram-se as ferramentas e linguagens mais adequadas para fazê-lo: a linguagem Java [Sun 2005], o motor de inferência Jess [Jess 2005] e o editor de ontologias Protégé [Protegé 2005]. Para a criação dos múltiplos agentes de software, optou-se pela ferramenta de desenvolvimento de sistemas multiagentes JADE [JADE 2005].

O JADE dispõe de uma forma nativa para comunicação entre agentes (compatível com o padrão FIPA-ACL). Considerando que cada agente trata a seqüência de caracteres extraída da mensagem de uma forma própria, pode-se incorporar ontologias à comunicação, estabelecendo-se uma forma padrão para interpretação da informação tratada. As ontologias utilizadas foram escritas no Protégé e convertidas, através do BeanGenerator, para o Modelo CRM (*Content Reference Model*) [Caire 2004], derivado da linguagem ACL, onde temos, basicamente, termos e predicados.

Foram criados dois agentes para fazer a simulação de uma Sociedade de Agentes de Busca. Um agente, chamado de VERTEB, versará sobre o assunto vertebrados no contexto da Biologia e um outro, chamado de INVERTEB, tratará do assunto invertebrados no mesmo contexto.

A persistência dos *links* das páginas recuperadas é feita se utilizado o Firebird [Firebird 2005]. Utiliza-se o *framework* Hibernate [Hibernate 2005] para manter-se transparência dos dados. O agente, quando necessário, cria um objeto para então persisti-lo na base de índices. Como a mobilidade de um agente de busca é desejável para embarcar todos os *links* de páginas processadas, ele seguirá o caminho inverso, convertendo os objetos persistidos na base de índices em POJO (*Plain Old Java Object*), valendo-se do poder do Hibernate para manutenção da sincronia entre a base de dados convencional e os objetos embarcados.

No decorrer da aprendizagem, o Agente Tutor poderá, a qualquer momento, solicitar da Sociedade de Busca uma relação de URLs sobre o assunto que está sendo estudado pelo aprendiz. Por exemplo, para o assunto vertebrados, será acionado o agente VERTEB. O Agente VERTEB informa ao Tutor quais são os resultados alcançados através de pesquisas na Web. Esse resultado é informado na interface do aprendiz como uma relação de *links*, conforme pode ser visto na Figura 6. A interface do aprendiz permite que o aprendiz possa navegar pela relação de *links* exibidos como resultados.

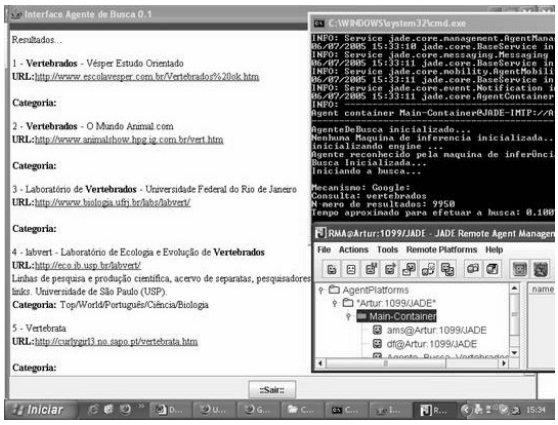

Figura 6. Interface - Protótipo Agente VERTEB

Cada agente possui uma ontologia que contém os conceitos a serem usados apenas por ele. Todavia, boa parte das instâncias relativas ao domínio podem ser úteis e reusadas por outros agentes.

5. Conclusão

Descreveu-se o processo de recuperação de informações no sistema NetClass. Foram especificados dois Agentes de Busca NetClass que serviram como experimento para mostrar que se pode utilizar de forma eficiente agentes de software com o objetivo de recuperar informações na Web a fim de serem utilizadas em ambientes de aprendizagem.

Conforme descrito, o uso de ontologias permite a interoperabilidade entre agentes de software. Por isso, o NetClass utiliza ontologias para representar conhecimentos e recuperar informações da Web. Nesse ambiente, a representação de conhecimentos e recuperação de informações são feitas utilizando-se uma arquitetura multiagentes, onde temos os Agentes de Domínio como a base de conhecimentos e os Agentes de Busca mantendo uma base de índices Web.

Por questão de simplificação, associou-se aos dois agentes protótipos o mesmo mecanismo de busca: o Google, através da importação da API-google fornecida em (Google 2005). Entretanto, poder-se-ia utilizar qualquer outro. Além disso, por enquanto, os agentes somente estão aptos a processar páginas cujos conteúdos estejam codificados em HTML.

Em estudos futuros, pretende-se criar uma Sociedade de Busca com um maior número de agentes com a finalidade de se ter um domínio mais amplo tratado no ambiente de aprendizagem. Nessa ocasião, será possível fazer uma avaliação precisa sobre a eficiência da recuperação de informação realizada pelo sistema multiagentes

estudado.

6. Referências

Caire, G., Cabanillas, D. (2004). "JADE Tutorial: Application-defined content languages and ontologies". TiLab, http://jade.tilab.com/doc/CLOntoSupport.pdf, Abril.

Costa, E. (1997) "Um modelo de ambiente interativo de aprendizagem baseado numa arquitetura multi-agentes". Tese de Doutorado, UFPB, Campina Grande – PB: Dezembro, 1997.

Firebird (2005), http://firebird.sourceforge.net, Janeiro.

Freitas, F. (2002) "Sistemas multiagentes cognitivos para a recuperação, classificação e extração integradas de informação da Web". Tese de Doutorado, UFSC, Florianópolis – SC. http://www.das.ufsc.br/~fred1/fred6.pdf, Fevereiro, 2005.

Google Web APIs (2005), http://www.google.com/apis, Abril.

Hibernate (2005). http://hibernate.bluemars.net, Maio.

JADE - Java Agent DEvelopment framework (2005), http://jade.tilab.com, Março.

Jess, the Rule Engine for the Java Platform (2005), http://herzberg.ca.sandia.gov/Jess, Fevereiro.

Labidi, S., Silva, J., Coutinho, L., Costa, E. (2000) "Agent-Based architecture for cooperative learning environment". Anais do XI Simpósio Brasileiro de Informática na Educação (SBIE'2000). Maceió-AL: 18 a 20 de Novembro, 2000.

Labidi, S., Souza, C., Nascimento, E. (2003a) "NetClass: cooperative learner modeling in a web-based environment". In the 6th Int. Conf. on Computer Based Learning in Science. Proceedings of the 6th Int. Conf. on Computer Based Learning in Science (CBLIS). Nicosia, Cyprus: University of Cyprus, 2003.

Labidi, S., Costa, N., Ferreira, J. (2003b) "Modeling of an authoring tool for an inteligent tutoring system". In the Proceedings of the 6th Int. Conf. on Computer Based Learning in Science (CBLIS). University of Cyprus, Nicosia.

Labidi, S.; Nunes, H. (2004) "Search service to support individual differences within search service to support individual differences within the NetcLass tutoring system". In: International Conference on Computers and Advances Technology in Education (CATE'04). In Proceedings of the International Conference on Computers and Advances Technology in Education. Kauai, Hawai.

Protégé-2000 (Release 3.1) (2005), http://protege.stanford.edu, Março.

Rothenfluhh, T., Gennari, J., Eriksson, H., Puerta, A., Tu, S., & Musen, M. (1996). Reusable ontologies, knowledgeacquisition tools, and performance systems: Protoge-ii solutions to sisyphus-2. International Journal of Human-Computer Studies, 3-4(44):303–332.

Serra Jr, G., Coutinho, L., Labidi, S. e Teixeira, C. (2000) "A learner modeling agent for cooperative learning". Anais (SBIE'2000), Maceió – AL: 18-20 de Novembro, 2000.

Serra Jr, G. (2001) "Agente de modelagem do aprendiz para o sistema MATHNET de

ensino inteligente cooperativo computadorizado". Dissertação de Mestrado, UFMA, São Luís – MA: Setembro.

Sun Microsystems (2005), http://java.sun.com, Março.

Uschold, M., Jasper, R. (1999). "A framework for understanding and classifying ontology applications". In Proceedings of the IJCAI99 Workshop on Ontologies and Problem-Solving Methods(KRR5), Stockholm, Sweden, (August 1999). http://www.cs.man.ac.uk/~horrocks/Teaching/cs646/Papers/uschold99.pdf.