

Uma Solução Interoperável para Apoiar a Decisão Colaborativa em Ambiente Suportado por Ontologias

Daniel F. Pires¹, Cesar A. C. Teixeira², Evandro E. S. Ruiz³

¹LINCE — Faculdades COC – Ribeirão Preto(SP) – Brasil

²LINCE – DC – UFSCAR – São Carlos(SP) – Brasil

³DFM – FFCLRP – USP – Ribeirão Preto(SP) – Brasil

dfpires@coc.com.br, cesar@dc.ufscar.br, evandro@usp.br

Abstract. *This paper presents an interoperable solution to support Web applications development that supports decision making in a distributed computer environment based on service-oriented architecture. The solution establishes a model to design a knowledge base oriented ontology, and computer mechanisms to create, share and search knowledge bases. The paper still presents a case study developed as a system prototype to support clinical diagnosis decision, and expose some considerations about its integration to a Web environment that supports medical educational collaborative activities with multimedia communication tools.*

Resumo. *Este trabalho apresenta uma solução interoperável para apoiar o desenvolvimento de aplicações Web que suportam a tomada de decisão colaborativa em ambiente computacional distribuído e orientado a serviço. A solução estabelece um modelo para a criação de uma ontologia orientada a bases de conhecimento, e de mecanismos de construção, compartilhamento e consulta de bases de conhecimento. São apresentados também um estudo de caso para apoiar a decisão diagnóstica colaborativa na área médica, e considerações sobre a integração deste a um ambiente Web que suporta atividades em educação médica com o uso de ferramentas de comunicação multimídia.*

1. Introdução

Sistemas computacionais fornecem informações a um sistema de apoio a decisão (SAD) para que uma base de conhecimento possa ser criada, representada, e inferências realizadas. A partir desta base de conhecimento, o usuário do SAD pode então ser apoiado durante uma tomada de decisão. Caso os sistemas computacionais contêm informações criadas apenas por usuários locais, o SAD apoiará seus usuários com o aprendizado realizado em experiências locais. Por outro lado, uma situação interessante é quando os sistemas podem inserir dados em um SAD com informações criadas por diferentes usuários e provenientes de diferentes sistemas. Neste cenário, um usuário poderia ser apoiado durante uma tomada de decisão por vários outros profissionais localizados remotamente, como um especialista de referência na área. Sendo assim, cria-se um ambiente colaborativo e de computação distribuída em que profissionais podem auxiliar e serem auxiliados durante uma tomada de decisão a partir do compartilhamento de bases de conhecimento.

Um ambiente computacional formado por aplicações distribuídas na Web que precisam compartilhar informações para tomada de decisão pode se tornar heterogêneo,

pois os serviços são normalmente criados e utilizados por diferentes protocolos de comunicação. Como consequência, o ambiente pode se tornar fortemente acoplado e não promover a interoperabilidade de computação distribuída. A arquitetura orientada a serviços SOA (*Service Oriented Architecture*) e as especificações de Serviços Web, uma arquitetura projetada para suportar interoperabilidade entre aplicações na Internet, mostram-se interessantes de serem aplicadas ao ambiente distribuído descrito anteriormente. Assim, um SAD pode ser visto como um potencial serviço Web.

Normalmente, as aplicações na Web que participam de um ambiente computacional, como o exposto anteriormente, não seguem um conjunto padrão de tipos semânticos necessários para a criação de uma estrutura que suporte o compartilhamento de bases de conhecimento, de modo que possam ser utilizadas por profissionais no apoio à tomada de decisão, a exemplos de [Bilykh et al. 2006, Kazemzadeh 2005, Achour et al. 2001], criando assim problemas de interoperabilidade semântica. Deste modo, torna-se fundamental a pesquisa e a criação de mecanismos computacionais que promovam a interoperabilidade semântica durante a troca de informações para apoio à tomada de decisão em ambiente Web.

Este artigo tem como objetivo apresentar uma solução interoperável para apoiar o desenvolvimento de aplicações Web que suportam a tomada de decisão em ambiente computacional distribuído. Esta solução é baseada em um novo modelo denominado *DSModel* para a criação de ontologia orientada a bases de conhecimento, e de novos mecanismos de construção, compartilhamento e consulta de bases de conhecimento. Estes mecanismos são baseados em duas especificações: a primeira denominada *DSModelWs* e utilizada na construção de um conjunto de serviços Web, e a segunda denominada *DSModel API*, utilizada na construção de programas para geração de documentos semânticos RDF. São apresentados também um estudo de caso desenvolvido a partir da solução proposta para apoiar a tomada de decisão diagnóstica na área médica, e uma proposta de integração deste a um ambiente Web, o projeto TIDIA-Ae¹, que apóia, com ferramentas de comunicação multimídia, atividades colaborativas em educação médica.

Este artigo está organizado da seguinte maneira: a seção 2 discute padrões e tecnologias relacionadas a este trabalho; a seção 3 apresenta a solução proposta; a seção 4 apresenta os resultados do estudo de caso desenvolvido para exemplificar o uso da solução proposta; a seção 5 relata cenários de atividades educacionais quando da integração do estudo de caso ao ambiente do projeto TIDIA-Ae; a seção 6 tece as considerações finais, indica possíveis pesquisas na linha do trabalho, e apresenta trabalhos relacionados.

2. Padrões e Tecnologias Relacionadas

A arquitetura orientada a serviços (SOA) é baseada na definição da interface dos serviços, o que torna as aplicações fracamente acopladas. SOA contém basicamente 3 regras [Graham et al. 2005]: um provedor de serviço, um requisitante de serviço, e um registrador ou divulgador de serviços. Ainda, inclui 3 operações: publicação, busca, e ligação. O provedor de serviço é responsável por criar um serviço e publicar a descrição do mesmo em um registrador de serviços, e ainda por receber conexões dos requisitantes de serviços. Estes, por sua vez, são responsáveis por procurar um serviço através de sua definição e se conectar às máquinas que hospedam o serviço encontrado. Finalmente, o

¹ Projeto TIDIA-Ae - <http://tidia-ae.incubadora.fapesp.br>

divulgador de serviço é responsável por registrar e divulgar as descrições dos serviços publicados pelos provedores e que serão procurados pelos requisitantes.

As especificações SOAP [Mitra 2006], um protocolo de comunicação para computação distribuída, WSDL [Chinnici et al. 2006], uma linguagem de descrição de serviços, e UDDI ², uma especificação para registrar e divulgar serviços, mostram-se interessantes de serem aplicadas ao ambiente distribuído proposto neste trabalho para promover a interoperabilidade de computação distribuída, visto que são padrões fracamente acoplados e independentes de plataforma computacional [Booth et al. 2006], são soluções gratuitas, e utilizam XML como formato de seus documentos associados.

Liderada pela W3C, a Web Semântica tem como objetivo fornecer um conjunto de padrões e especificações que pode ser utilizado para se adicionar informação semântica aos dados na Web através de ontologias. Segundo Berners-Lee e co-autores em [Berners-Lee et al. 2001], somente assim as informações na Web poderão ser automaticamente processadas por computadores, e não somente através de intervenção humana. São alguns destes padrões a *RDF* [Miller et al. 2006] e a *OWL* [Miller and Hendler 2006]. *RDF* é um arcabouço que oferece uma linguagem XML para descrever recursos Web e ontologias, enquanto que *OWL* oferece um vocabulário adicional para a formalização semântica de ontologias e portanto possui maior poder de interpretabilidade em relação a *RDF*.

3. A Solução Proposta

A solução interoperável apresentada nesta seção pode ser utilizada por aplicações Web que precisam oferecer aos seus usuários suporte à tomada de decisão em ambiente computacional distribuído e orientado a ontologias e a serviços. Essa solução é orientada a ontologias e a serviços para garantir às aplicações Web, respectivamente, compatibilidade semântica e as características da computação distribuída, durante as atividades computacionais de criação, compartilhamento e consulta de base de conhecimento no contexto de apoio à tomada de decisão. Para isso, a solução determina que as aplicações Web precisam conter: (1) uma ontologia criada a partir de um conjunto de tipos semânticos definidos por um modelo denominado *DSModel* (*Decision Support Model*), (2) um conjunto de serviços Web definidos conforme a especificação *DSModelWs* (*Decision Support Model Web Services*); e (3) uma API para auxiliar as aplicações a gerarem documentos semânticos *RDF* conforme a especificação *DSModel API*.

3.1. Modelo *DSModel*

O modelo de tipos semânticos *DSModel*, apresentado na Figura 1, é proposto para facilitar a construção de uma ontologia de domínio que apóie semanticamente a criação de uma estrutura que suporte o compartilhamento de bases de conhecimento para serem utilizadas por profissionais no apoio à tomada de decisão.

Com o objetivo de explicitar e justificar a criação e o relacionamento dos tipos semânticos apresentados na Figura 1, pode-se supor um conjunto de casos que treinam um SAD e que, posteriormente, outros casos poderão ser classificados com o conhecimento do SAD. Neste exemplo, os casos dizem respeito a decisão diagnóstica relacionada a dor

² <http://www.uddi.org/>

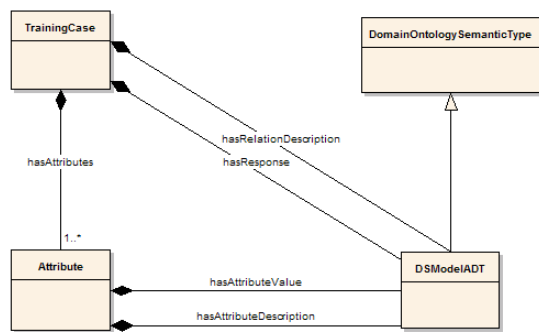


Figura 1. Modelo DSModel

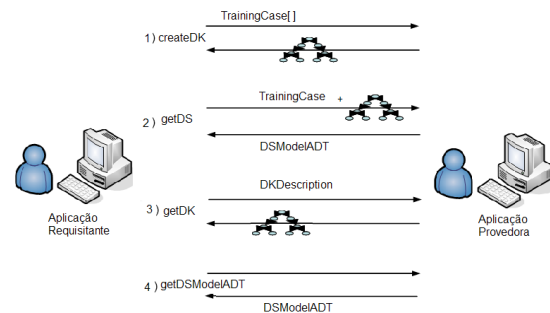


Figura 2. Especificação DSModelWs

abdominal aguda. São possíveis sintomas, juntamente com seus possíveis valores: (intensidade: moderada, severa, desconforto), e (vômito: bile, feculento, claro). As possíveis respostas diagnósticas são colecistite ou gastrite.

O primeiro conjunto de termos do exemplo apresentado anteriormente é representado por: ‘dor abdominal aguda’, ‘intensidade’, ‘moderada’, ‘severa’, ‘desconforto’, ‘vômito’, ‘bile’, ‘feculento’, ‘claro’, ‘colecistite’, e ‘gastrite’. Este conjunto de termos pode ser classificado, de acordo com o modelo apresentado na Figura 1, através do tipo semântico DSModelADT (*Decision Support Model Abstract Data Type*). Este tipo semântico deve ser estendido de um tipo semântico de mais alto nível da ontologia de domínio que representa o universo sendo aplicado, ilustrado aqui pelo tipo *DomainOntologySemanticType*. Não foram definidas propriedades para estes tipos semânticos, visto que ambos são definidos exclusivamente em função da necessidade de cada aplicação. Um exemplo destas propriedades é apresentado e melhor detalhado quando da explicação do estudo de caso, na seção 4.

O segundo conjunto de termos do exemplo apresentado anteriormente é representado pelos pares: (‘intensidade’, ‘moderada’); (‘intensidade’, ‘severa’); (‘intensidade’, ‘desconforto’); (‘vômito’, ‘bile’); (‘vômito’, ‘feculento’); (‘vômito’, ‘claro’). Este conjunto de termos pode ser classificado, de acordo com o modelo apresentado na Figura 1, através do tipo semântico *Attribute*. Este tipo semântico deve possuir duas propriedades: *hasAttributeDescription*, que representa semanticamente a descrição do atributo, e *hasAttributeValue*, que representa semanticamente o valor do atributo.

O terceiro e último conjunto de termos do exemplo, apresentado para ilustrar a criação do modelo, é representado pelos exemplos de trios: (‘dor abdominal aguda’, [(‘intensidade’, ‘severa’); (‘vômito’, ‘bile’)], ‘gastrite’); (‘dor abdominal aguda’, [(‘intensidade’, ‘desconforto’); (‘vômito’, ‘claro’)], ‘gastrite’). Este conjunto de termos pode ser classificado, de acordo com o modelo apresentado na Figura 1, através do tipo semântico *TrainingCase*. Este tipo semântico deve possuir três propriedades: *hasRelationDescription*, que representa semanticamente a descrição da relação sendo discutida, *hasAttributes*, que representa semanticamente os atributos do caso de treinamento, e finalmente *hasResponse*, que representa semanticamente a resposta do caso de treinamento. As representações semânticas dos termos que compõem um caso de treinamento irão depender diretamente das propriedades contidas no tipo semântico *DSModelADT*.

Procurando permitir e facilitar o uso da ontologia criada a partir do modelo ap-

resentado anteriormente por parte das aplicações computacionais que desejam trocar informações para apoiar a tomada de decisão em ambiente computacional distribuído, este trabalho projetou duas especificações nomeadas DSModelWs e DSModel API, utilizadas na construção de um conjunto de serviços Web e na construção de programas para geração de documentos semânticos RDF, respectivamente.

3.2. DSModelWs

A especificação DSModelWs é proposta para ser utilizada na construção de um conjunto de serviços Web que apóie as atividades de criação, de compartilhamento e de consulta por uma base de conhecimento. Esta especificação permite também que este conjunto de serviços possa ser utilizado por profissionais no apoio à tomada de decisão em ambiente computacional distribuído e orientado a serviços. O conjunto é formado por 4 serviços detalhados a seguir, e ilustrados na Figura 2:

1. *createDK (create Database Knowledge)*: Este serviço cria uma base de conhecimento a partir de um conjunto de casos de treinamento representados por um vetor de objetos da classe *TrainingCase*. Na Figura 2, a base de conhecimento é representada através de uma árvore de decisão;
2. *getDS (get Decision Support)*: Este serviço obtém uma resposta representada como um objeto do tipo semântico *DSModelADT*. Os parâmetros de entrada são a base de conhecimento representada por um objeto da técnica de aprendizado de máquina, e um caso duvidoso representado por uma classe *TrainingCase*;
3. *getDK (get Database Knowledge)*: Este serviço recupera bases de conhecimento de outros participantes. Os parâmetros de entrada são informações representadas por um objeto da classe *DKDescription*, explicada a seguir, que descreve a base de conhecimento sendo compartilhada;
4. *getDSModelInfo (get DSModel Info)*: Este serviço recupera informações do tipo semântico *DSModelADT*. Em razão deste serviço ser muito específico e dependente das informações contidas no tipo semântico herdado de uma ontologia de domínio sendo aplicada, não definimos parâmetros de entrada para o mesmo.

3.3. DSModel API

A especificação DSModel API é proposta para ser utilizada na construção de um conjunto de classes que auxiliam os programas na geração de documentos semânticos RDF de acordo com a ontologia projetada e a partir do modelo DSModel. Isto significa que as informações contidas nos documentos criados por estas classes estarão associadas semanticamente a algum tipo semântico da ontologia. Ainda, os documentos RDF são utilizados na chamada dos serviços da DSModelWs para a serialização dos objetos.

A DSModel API, ilustrada pela Figura 3, sugere que as classes sejam construídas utilizando-se da JENA RDF API, que fornece um conjunto de classes e interfaces sob licença de software livre, para a leitura e escrita de documentos RDF de maneira simples. Especificamente, a DSModel API permite que aplicações que precisam trocar informações de apoio à tomada de decisão possam gerar e manipular documentos em formato RDF. A API é composta pelos seguintes componentes de software:

- *DSModelADT*, *Attribute* e *TrainingCase*: contém variáveis e métodos necessários para manipulação destas na forma de um recurso e de um modelo RDF, conforme a especificação da JENA;

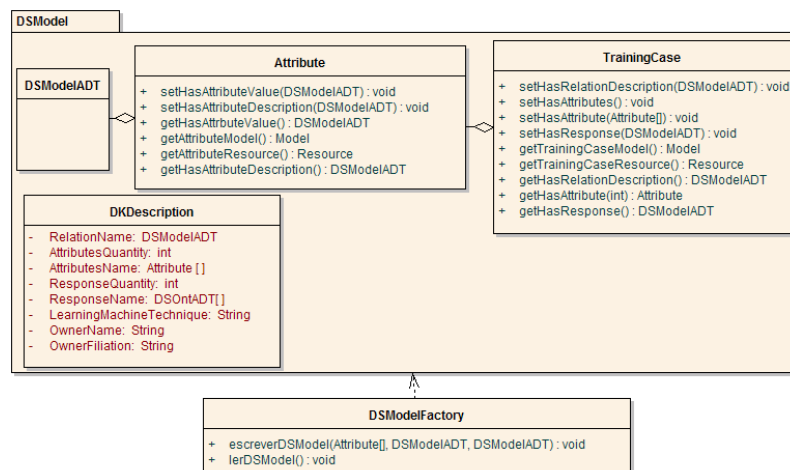


Figura 3. Especificação DSModel API

- *DSModel Factory*: contém a implementação exemplo dos métodos *escreverDSModel()* e *lerDSModel()*, de modo a ilustrar a escrita e leitura, respectivamente, de um documento RDF da aplicação;
- *DKDescription*: contém informações para se disponibilizar e procurar por uma base de conhecimento.

4. Estudo de Caso

Como forma de verificar a viabilidade da solução proposta, um estudo de caso para apoio à tomada de decisão diagnóstica entre profissionais da área médica foi desenvolvido - também relatado em mais detalhes em [Pires 2007].

4.1. Utilizando o DSModel

O resultado da utilização do modelo *DSModel* foi a construção da ontologia *DDSOnt* (*Diagnosis Decision Support Ontology*), ilustrada pela Figura 4. O tipo semântico *DomainOntologySemanticType* da *DSModel* é representado neste estudo de caso por dois tipos semânticos, *UMLS Entity* e *UMLS Event*. Estes são os tipos semânticos de topo na hierarquia da ontologia médica *UMLS SN* (*Unified Medical Language System - Semantic Network*) [System 2006], ontologia de domínio utilizada no estudo de caso para representar e classificar semanticamente os termos clínicos que compõe um caso clínico de treinamento. O tipo semântico *DSModelADT* do modelo *DSModel* é representado neste estudo de caso pelo tipo semântico *DDSOntADT* (*Diagnosis Decision Support Ontology - Abstract Data Type*). Este tipo semântico representa qualquer termo clínico na área médica, como um conceito qualitativo, um sinal, uma doença, uma substância do corpo, um resultado laboratorial ou teste, funções do organismo, uma descoberta, funções patológicas, funções fisiológicas, e outras mais. Por representar qualquer termo clínico, este tipo semântico está relacionado com os tipos semânticos *UMLS Entity* e *UMLS Event* através da relação ‘é_um’, como mostra a Figura 4.

A Figura 4 mostra ainda que o tipo semântico *DDSOntADT* contém um conjunto de 8 propriedades. Estas propriedades são necessárias para identificar unicamente um termo clínico no sistema *UMLS MetaThesaurus*. Este sistema unifica termos de mais de

100 sistemas de terminologia clínica, relacionando-os conceitualmente. Identificar unicamente um termo clínico é importante pois termos diferentes de sistemas de terminologia diferentes podem representar o mesmo conceito, ou ainda termos iguais de sistemas de terminologia diferentes podem representar conceitos diferentes. Assim, através dos conceitos do sistema UMLS *MetaThesaurus*, uma aplicação que faz uso de um sistema de terminologia ‘A’ pode receber e identificar o conceito de um termo recebido proveniente de outra aplicação que se utiliza de um sistema de terminologia ‘B’.

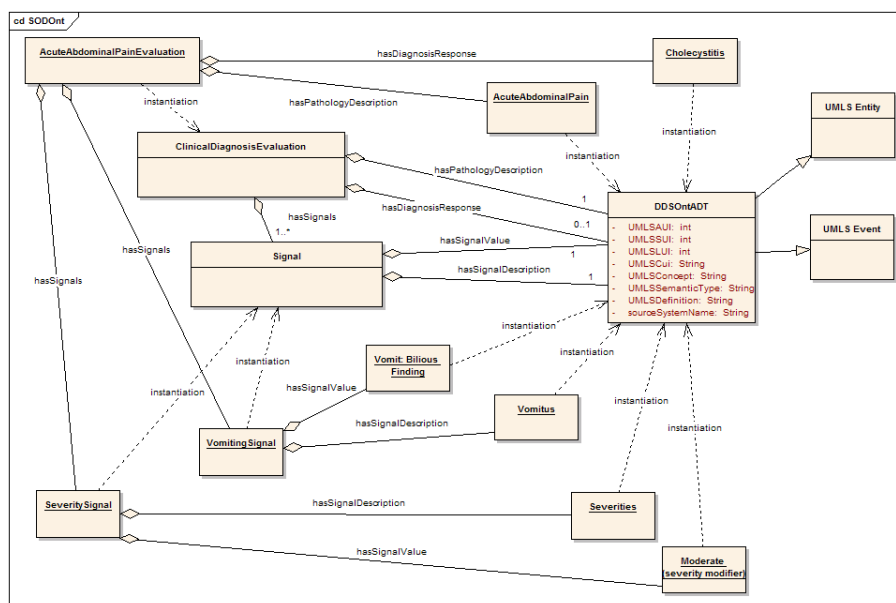


Figura 4. Ontologia DDSOnt

Continuando a explicação da criação da ontologia DDSOnt com o modelo DSModel, o tipo semântico *Attribute* do modelo é representado no estudo de caso pelo tipo semântico *Signal*. Apenas por ajustes semânticos, as propriedades *hasAttributeDescription* e *hasAttributeValue* foram renomeados para *hasSignalDescription* e *hasSignalValue*. Finalmente, o tipo semântico *TrainingCase* é representado no estudo de caso pelo tipo semântico *ClinicalDiagnosisEvaluation*. O nome das propriedades foram alteradas apenas por ajustes semânticos.

4.2. Utilizando a DSModelWs

Um conjunto de serviços Web foi criado e projetado conforme a especificação DSModelWs da solução proposta e apresentado na seção anterior. Primeiramente, os nomes dos serviços Web foram alterados de acordo com o domínio do estudo de caso. O serviço *createDK* foi renomeado para *createCDK* (*Create Clinical Database Knowledge*), o serviço *getDS* foi renomeado para *getDDS* (*Get Diagnosis Decision Support*), o serviço *getDK* foi renomeado para *getCDK* (*Get Clinical Database Knowledge*), e finalmente o serviço *getDSModelInfo* foi renomeado para *getDDSOntADTInfo* (*Get Diagnosis Decision Support Ontology ADT Information*). A técnica de aprendizado de máquina utilizada para a implementação dos serviços *createDK* e *getDS* foi a árvore de decisão, especificamente o algoritmo J48 da API WEKA [Witten and Frank 2000].

Em relação a implementação do serviço `getDSModelInfo`, que obtém informações relativas às 8 propriedades do tipo semântico `DDSOnt`, foi utilizado a UMLSKS API, que possibilita que programas computacionais possam fazer acesso a base de dados da UMLSKS (*Unified Medical Language System Knowledge Server*). Os parâmetros de entrada definidos para o serviço Web `getDSModelInfo` são um termo clínico e o seu respectivo sistema de terminologia de origem. Os serviços foram implementados utilizando a tecnologia AXIS, que é um servidor da Apache que implementa a especificação SOAP.

De modo a publicar e descobrir os serviços Web implementados, este estudo de caso fez uso de um servidor UDDI privado, através da instalação do servidor `jUDDI`, também da Apache, que implementa a especificação UDDI. A criação dos programas para disponibilização e consulta por serviços foi realizada com a API `uddi4j`.

4.3. Utilizando a DSModel API

Os programas da API disponíveis na documentação da proposta, como apresentado na seção anterior, foram utilizados como modelo inicial para a criação da `DDSOnt` API. Como programas originais estão diretamente relacionados ao modelo `DSModel`, pequenas alterações foram realizadas, como a necessidade de criação das variáveis e dos métodos da classe `DDSOntADT`, e a alteração dos nomes originais das classes, variáveis e métodos para nomes semanticamente relacionados ao estudo de caso.

5. Integrando o Estudo de Caso ao Ambiente do Projeto TIDIA-Ae

O projeto *TIDIA-Ae* tem como objetivo criar um conjunto de esforços para fomentar pesquisa e desenvolvimento de tecnologias computacionais que suportem o gerenciamento de atividades de ensino-aprendizagem na Web, e ao mesmo tempo exijam uma rede de computadores de baixa latência e grande largura de banda. A partir do ambiente do projeto *TIDIA-Ae*, pode-se instanciar ferramentas de comunicação como comunicador instantâneo (CI), *chat*, *whiteboard*, etc.

O estudo de caso apresentado na seção anterior foi adicionado ao ambiente do projeto *TIDIA-Ae* de modo que este ambiente Web, suportado também por ontologias e serviços Web, possa apoiar a troca de informações diagnósticas objetivando oferecer alternativas de atividades didáticas no ensino médico e ainda de refinar bases de conhecimento que auxiliam na decisão diagnóstica. Com essas adições, os seguintes cenários são apresentados, considerando a existência de salas remotas com alunos e professores de medicina participando de uma reunião em diferentes locais:

1. Configuração do contexto onde será realizada a reunião: definição dos participantes, da patologia discutida, das ferramentas de comunicação que serão utilizadas. Pode-se definir ainda se as informações criadas serão compartilhadas apenas para o grupo envolvido, ou para todos os participantes do ambiente;
2. Análise entre alunos e professores dos diagnósticos que serão considerados para o estudo da patologia. Por exemplo, para a patologia de dor abdominal aguda serão considerados os diagnósticos de gastrite e colecistite, como na figura 5;
3. Discussão entre alunos e professores dos sintomas que serão considerados para a patologia, bem como seus valores possíveis, como ilustrado na Figura 5;
4. Discussão entre alunos e professores dos casos clínicos que compõem o conjunto de treinamento para criação de uma base de conhecimento;

5. Durante a definição dos diagnósticos, dos sintomas e dos casos clínicos, pode acontecer de eventualmente alguns professores definirem valores alternativos com objetivos de se criar bases de conhecimento diferentes para comparações futuras de diferentes respostas diagnósticas;
6. Efetivamente realizar consultas por respostas diagnósticas e comparar o resultado com outros alunos que eventualmente utilizaram outras bases de conhecimento, como na Figura 5.

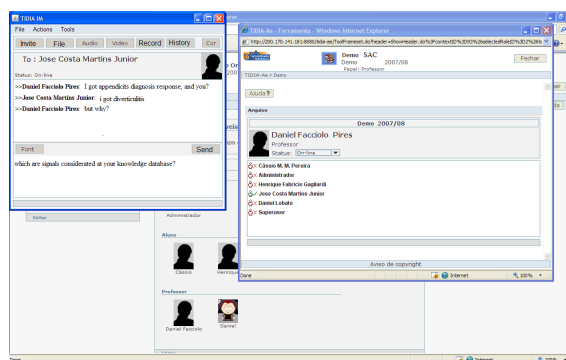


Figura 5. Discussão com ferramentas de comunicação multimídia

Objetivando mapear as situações descritas anteriormente para a arquitetura SOA, o módulo de software dos professores e monitores são os provedores de serviço, enquanto que o módulo dos estudantes representam a regra de requisitantes de serviços. Os módulos de software são relatados em mais detalhes em [Pires et al. 2008].

6. Conclusões

Este trabalho apresentou uma nova solução interoperável, baseada em ontologias, serviços Web e documentos semânticos, para apoiar o desenvolvimento de aplicações Web que suportam a tomada de decisão em ambiente computacional distribuído e orientado a serviço. A solução foi utilizada para apoiar a decisão diagnóstica médica colaborativa explorada no ambiente TIDIA-Ae, e suportar atividades de ensino médico. Através da exploração de tecnologias computacionais e do estudo de como integrá-las, este trabalho apresenta uma contribuição para a informática médica com uma solução que promove o compartilhamento de informações para apoiar a decisão diagnóstica colaborativa. Este trabalho é uma evolução dos trabalhos apresentados em [Pires 2007] e [Pires et al. 2008] pois apresenta um novo modelo *DSModel* e dois novos mecanismos de construção, compartilhamento e consulta de bases de conhecimento, *DSModelWs* e *DSModel API*.

Com o objetivo de comparar nossa proposta com trabalhos relacionados, duas iniciativas relevantes foram encontradas na literatura. A primeira delas foi proposta por Bilykh e co-autores[Bilykh et al. 2006]. Eles apresentam EGADSS, um sistema que pode ser suprido com informações de aplicações heterogêneas. A solução é baseada nas tecnologias HL7. Assim, os tipos semânticos da ontologia RIM são utilizados para a representação semântica das informações, enquanto que a CDA é utilizada para estruturar e formalizar os documentos clínicos. A segunda iniciativa, descrita por Achour e co-autores[Achour et al. 2001], expõe uma ferramenta de aquisição de conhecimento que

produz bases de conhecimento capazes de serem trocadas e reutilizadas. O sistema UMLS é utilizado para promover a compatibilidade terminológica e ontológica.

Alguns aspectos do nosso trabalho podem ser enfatizados quando comparados com os trabalhos citados. Um deles são as facilidades de exploração de um ambiente colaborativo para a decisão diagnóstica, como as facilidades de interação entre professores, médicos e estudantes objetivando aperfeiçoar o ensino através da informação diagnóstica. Estas facilidades podem oferecer atividades alternativas de aprendizado médico e podem ser utilizadas para refinar bases de conhecimento que auxiliam a decisão diagnóstica. Enquanto que o foco das outras iniciativas é a representação de regras de decisão médica presentes em bases de conhecimento utilizando linguagens como MLM[Hripcsak G et al. 1994] e Arden Syntax[Jenders et al. 2003], este trabalho se concentra na troca de dados clínicos e como utilizá-los para a criação de bases de conhecimento. Talvez a principal diferença de nosso trabalho e a principal contribuição original seja a definição de tipos semânticos junto à ontologias para promover a compatibilidade semântica durante a decisão diagnóstica colaborativa.

7. Agradecimentos

Agradecemos às Faculdades COC, ao CNPq e à FAPESP pelo suporte financeiro, este último através do processo 05/60653-1 do Projeto TIDIA-AE.

Referências

- Achour, S., Dojat, M., Rieux, C., Bierling, P., and Lepage, E. (2001). A umls-based knowledge acquisition tool for rule-based clinical decision support system development. *Journal of American Medical Informatics Association*, 8(4):351–360.
- Berners-Lee, T., Hendler, J., and Lassila, O. (2001). The semantic web. *Scientific American*, 5:28–37.
- Bilykh, I., Jahnke, J., McCallum, G., and Price, M. (2006). Using the clinical document architecture as open data exchange format for interfacing emrs with clinical decision support systems. In *Proceedings of the 19th IEEE International Symposium on Computer-Based Medical Systems*, volume 1, pages 855–860, Salt Lake City, USA. IEEE.
- Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C., and Orchard, D. (2006). Web service architecture w3c working group, <http://www.w3.org/tr/ws-arch/>. Última visita:2007.
- Chinnici, R., Gudgin, M., Moreau, J., Schlimmer, J., and Weerawarana, S. (2006). Web services description language (wsdl) version 2.0 part 1: Core language, <http://www.w3.org/tr/2004/wsdl20-20040803/>. Última visita:2007.
- Graham, S., Davis, D., and Simeonov, S. e. a. (2005). *Building Web Services with Java: Making Sense of XML, SOAP, WSDL, and UDDI*. Sams Publishing.
- Hripcsak G, P., Ludemann, T., Pruor, O., and Wigertz, P. (1994). Rationale for the arden syntax. *Computers in Biomedical Research*, 27(4):291–324.
- Jenders, R., Corman, R., and Dasgupta, B. (2003). Making the standard more standard: a data and query model for knowledge representation in the arden syntax. In *Proceedings of the AMIA Annual Symposium*, volume 1, pages 323–330, Washington, USA. AMIA.
- Kazemzadeh, R.S. and Sartipi, K. (2005). Interoperability of data and knowledge in distributed health care systems. In *Proceedings of the 13th IEEE International Workshop on Software Technology and Engineering Practice*, volume 1, pages 230–240, Budapest, Hungary. IEEE.
- Miller, E. and Hendler, J. (2006). Web ontology language (owl), <http://www.w3.org/2004/owl/>. Última visita:2007.
- Miller, E., Swick, R., and Brickley, D. (2006). Resource description framework (rdf), <http://www.w3.org/rdf/>. Última visita:2007.
- Mitra, N. (2006). Soap version 1.2 part 0: Primer, <http://www.w3.org/tr/2003/rec-soap12-part0-20030624/>. Última visita:2007.
- Pires, D. (2007). Uma solução interoperável, baseada na umls, para apoiar a decisão diagnóstica colaborativa na web. tese de doutorado. dfm – fclrp – usp.
- Pires, D., Teixeira, C., and Ruiz, E. (2008). A umls interoperable solution to support collaborative diagnosis decision making over the internet. In *Proceedings of the 2008 ACM Symposium on Applied Computing*, volume 2, pages 1400–1404, Fortaleza, CE, Brasil.
- System, U. M. L. (2006). Unified medical language system - national library of medicine, <http://www.nlm.nih.gov/research/umls/>. Última visita:2007.
- Witten, I. and Frank, E. (2000). *Data Mining: Practical machine learning tools with Java implementations*. Morgan Kaufmann.