6
Aplicação do Framework ao LMS AulaNet

Este capítulo descreve o trabalho de instanciação do *framework* proposto para o LMS AulaNet, com o objetivo de adaptá-lo ao novo conceito de *learning objects* sugerido por esse trabalho de pesquisa. Mais especificamente, este capítulo inicia com uma apresentação sobre o Ambiente AulaNet e, em seguida, o trabalho de implementação realizado no sistema.

6.1.
Visão Geral do Projeto AulaNet

O Projeto AulaNet [Lucena et al., 1998] vem se realizando desde junho de 1997 no Laboratório de Engenharia de Software (LES) do Departamento de Informática da PUC-Rio, coordenado pelos professores Carlos José Pereira De Lucena e Hugo Fuks. O resultado desse trabalho é um ambiente para ensino e aprendizagem na *Web* denominado AulaNet, que se baseia no trabalho cooperativo gerado pelas interações entres os aprendizes e dos aprendizes com seus instrutores. Neste sentido, o Ambiente AulaNet é considerado um *groupware*, isto é, um ambiente para suporte ao trabalho colaborativo.

O Ambiente AulaNet é considerado fundamentalmente como um LMS devido as suas funcionalidades de administração, criação, manutenção e configuração de cursos e matrículas. O conceito de curso para o Ambiente AulaNet não limita-se apenas a conteúdos instrucionais disponibilizados para os aprendizes, mas a uma série de serviços para apoiar o processo de aprendizagem.

Considerando o objetivo do AulaNet de gerar aprendizado com base no trabalho em grupo, o ambiente vem sendo desenvolvido sob o raciocínio de que para os aprendizes trabalharem em grupo precisam se comunicar e trocar idéias, precisam coordenar suas atividades e, necessariamente, cooperar entre si. É ponderando sobre esse modelo de aprendizagem (comunicação, coordenação e

cooperação) que o Ambiente AulaNet disponibiliza vários serviços para seus aprendizes visando incentivar o trabalho colaborativo.

6.2. Descrição do Ambiente AulaNet

O AulaNet é um sistema desenvolvido em Java para a plataforma Web e, atualmente, encontra-se na versão 2.0. Sendo um sistema *freeware*, pode ser adquirido gratuitamente nos idiomas português, inglês e espanhol (www.eduweb.com.br). Aproximadamente mais de 4.000 cópias gratuitas foram distribuídas pela internet até hoje.

Sua modelagem conceitual foi projetada para possuir basicamente três atores: aprendiz, docente e administrador. Todos os participantes do ambiente, com exceção do administrador, são considerados aprendizes e os atores mencionados podem ter diferentes papéis dependendo do contexto no ambiente. Abaixo, serão detalhados os atores e os papéis que podem assumir:

- O administrador atua na configuração do ambiente, gerenciando as matrículas dos aprendizes, atribuição de permissões no ambiente e a publicação de cursos.
- O docente pode assumir três papéis diferentes: coordenador, docente co-autor e mediador. Como coordenador, o docente possui o papel de criação e configuração dos cursos. Este processo inclui a publicação de conteúdos instrucionais, a escolha e configuração dos serviços a serem disponibilizados para os aprendizes. O docente co-autor é um docente indicado pelo coordenador de um curso para auxiliá-lo na autoria do curso. O mediador é um docente que atua nos cursos criados pelo coordenador com o papel de monitorar a turma e avaliar as interações dos aprendizes.
- O aprendiz, além de ser o papel *default* de todos os participantes do ambiente, pode assumir um papel diferenciado chamado aprendiz co-autor, que caracteriza uma permissão especial dada pelo docente de realizar contribuições de conteúdo para o curso.

O AulaNet oferece ao coordenador uma série de serviços que podem ser selecionados e configurados para que possam ser utilizados na área de trabalho do

curso. Basicamente, os serviços são agrupados com base no modelo de aprendizagem utilizado no AulaNet, isto é, existem serviços de comunicação, serviços de coordenação e de cooperação.

Os mecanismos de comunicação possuem o objetivo de fornecer aos aprendizes a possibilidade de troca e o envio de mensagens e informações. Fazem parte desse grupo os serviços de "Lista de Discussão", "Conferências" (serviço de discussão textual assíncrono – fórum de mensagens), "Contato com Docentes" (envio de mensagens individuais para os docentes), "Debate" (também conhecido como *Chat*), e o serviço de "Mensagem para os Participantes" (serviço de troca instantânea de mensagens).

Os mecanismos de coordenação são compostos por um serviço de "Avisos", "Plano de aulas" (acesso aos conteúdos instrucionais), "Avaliação" (serviço de provas), "Tarefas" e "Relatórios de Participação" para acompanhamento da turma.

Os mecanismos de cooperação compreendem os serviços para a cooperação entre os aprendizes e dos aprendizes com os docentes. Neste grupo estão incluídos os serviços de "Co-autoria de Docente", "Co-autoria de Aprendiz", "Bibliografia", "Webliografia" e uma área de "Download".

6.3. Flexibilizando os Serviços do Ambiente AulaNet

Na seção anterior foram listados os serviços que estão disponíveis no AulaNet para o coordenador durante a criação e configuração de um curso. Atualmente na versão 2.0 do AulaNet, os serviços se encontram fortemente acoplados com o sistema impedindo uma modelagem flexível e baseada em componentes.

A motivação de possibilitar no AulaNet o uso de novos serviços diferentes dos legados, torna-se interessante pela vantagem de permitir uma maior liberdade para o coordenador durante a preparação do ambiente de aprendizado do curso e das turmas. Sob a ótica do administrador, a possibilidade de utilização de novos serviços implementados como *web services* auxilia na tarefa de suprir demandas particulares de alguns coordenadores por funcionalidades específicas para um determinado curso. Um cenário interessante, como exemplo, é o serviço de Lista de Discussão disponível no AulaNet para o coordenador utilizar nos cursos. Este

serviço permite que os aprendizes enviem mensagens para toda a turma através do AulaNet, sendo que as mesmas são armazenadas em ordem cronológica no sistema e são enviadas para o e-mail de cada integrante da turma. Essa ferramenta foi projetada para permitir que os aprendizes se comuniquem, trocando idéias e interagindo. Contudo, considerando a possibilidade de um coordenador estar configurando um curso para um grupo de aprendizes de diferentes nacionalidades, pode ser que o uso dos serviços de comunicação, como a Lista de Discussão, seja impossibilitado devido aos diferentes idiomas falados por cada aprendiz. Neste caso, o administrador poderia realizar uma busca em repositórios de *learning objects* e descobrir a existência de um fornecedor de *learning services* implementados como *web services* que oferece o serviço de Lista de Discussão com a funcionalidade de tradução das mensagens enviadas pelos aprendizes para vários idiomas. Neste caso, um aprendiz enviando o texto da sua mensagem em português, o novo *web service* de Lista de Discussão receberia a mensagem do aprendiz e realizaria a tradução do texto para o idioma de preferência de cada aprendiz da turma. Em seguida, a mensagem seria enviada para os e-mails dos aprendizes com o texto original e com a tradução para o idioma cadastrado no perfil do aprendiz no LMS.

Certamente o cenário descrito acima é possível de ser implementado em sistemas de *e-Learning*, mas é interessante analisar o esforço de implantação. Tomando o estudo de caso o Ambiente AulaNet, este já possui um serviço de Lista de Discussão proprietário e fortemente acoplado ao sistema. Realizar a substituição do serviço proprietário pelo novo *web service* pode não ser a melhor opção simplesmente pelo fato de já existirem outros cursos utilizando o serviço legado de Lista de Discussão. Neste caso, a opção é realizar uma integração do LMS com o novo *web service* e manter a possibilidade do coordenador utilizar um entre os dois serviços. Contudo, necessariamente existirão problemas como incompatibilidade nas interfaces de comunicação e na forma de interação do serviço com a camada de visualização.

É com base nesse cenário e nas características do AulaNet que este trabalho de pesquisa propôs a adaptação do Ambiente AulaNet para o novo conceito de *learning objects*. Com isso, o *framework* descrito no capítulo 5 foi instanciado e aplicado ao AulaNet atribuindo-lhe uma modelagem flexível para facilitar a utilização de *learning services* implementados como *web services* e possibilitar

uma arquitetura baseada em componentes, atribuindo ao sistema o benefício da adaptabilidade e facilidade durante a realização de melhorias e manutenção.

6.4. Processo de Adaptação do Ambiente AulaNet

Este seção descreve as fases do estudo de caso referente ao Ambiente AulaNet, LMS escolhido para a instanciação do *framework* proposto no capítulo anterior, visando o objetivo de validar e embasar as pesquisas relacionadas a este trabalho. Como conseqüência direta do estudo de caso, objetivou-se a adequação do AulaNet ao novo conceito defendido: extensão do conceito de *learning objects* englobando serviços.

Para permitir um estilo de implementação incremental, optou-se por dividir o processo de desenvolvimento em dois estágios. Esta abordagem foi adotada para minimizar o esforço de desenvolvimento devido à possibilidade de realização de testes e aperfeiçoamentos no *framework* entre os estágios. A seguir, encontra-se uma breve descrição de cada estágio:

- Composição estática do AulaNet com um *web service* de um único fornecedor de serviços – nesta etapa, o LMS foi adaptado para possuir a funcionalidade de utilizar um *web service* específico na composição de seus cursos, ou seja, o LMS disponibilizaria para o coordenador os seus serviços locais e o novo *web service* ao qual foi integrado. O termo estático é utilizado devido a uma integração estática entre o LMS e o fornecedor de *web service*.
- Composição dinâmica de vários *web services* com diferentes fornecedores – nesta etapa, o LMS foi adaptado para possuir a funcionalidade de ser integrado a diversos fornecedores de serviço abstraindo as características de implementação dos serviços.

Cada uma das etapas acima gerou modificações na camada de visualização do AulaNet e são detalhadas adiante.

6.5. Modelos de Aprendizagem

Esta seção aborda o modelo de aprendizagem adotado no Ambiente AulaNet, os serviços associados a este modelo e os questionamentos e possibilidades geradas com relação a inserção de novos serviços ao sistema.

O AulaNet foi desenvolvido com base na abordagem de colaboração 3C (comunicação, coordenação e cooperação), onde os serviços disponíveis no ambiente estão subdivididos em serviços de comunicação, serviços de coordenação e serviços de cooperação [WDBC, 2003].

A adaptação do AulaNet ao novo conceito de *learning objects* possibilita de inserção de serviços e permite ao coordenador uma maior flexibilidade no momento de configurar o curso. Contudo, neste momento vale o questionamento que esse cenário também exige que novos serviços agregados ao ambiente sejam classificados com base no modelo de colaboração 3C adotado pelo AulaNet.

Neste sentido, o Ambiente AulaNet não possui na sua camada de visualização nenhuma interface de manutenção dos serviços cadastrados. Com isso, tornou-se necessário incorporar ao desenvolvimento do estudo de caso a capacidade de um ator do sistema realizar a manutenção dos novos serviços eventualmente adicionados ao ambiente.

Essa melhoria no sistema gerou, por conseqüência, outra linha de discussão fundamentada no modelo de aprendizagem adotado pelo AulaNet. Ela é apresentada a seguir, mas não faz parte do objetivo central desse trabalho de pesquisa. Por isso, é abordada de maneira superficial e considerada como um trabalho futuro, sendo mencionado novamente no capítulo 7 de conclusão. A nova linha de discussão é gerada pelo fato de que, considerando a nova flexibilidade do AulaNet, os serviços do modelo 3C seriam estendidos pela integração do ambiente com novos *learning services* implementados como *web services*. Com isso, surge a idéia de possibilitar a flexibilização também com relação ao modelo de aprendizagem, onde existiria a funcionalidade de cadastrar os modelos de aprendizagem e realizar a associação dos serviços que devem ser oferecidos ao coordenador no momento da criação do curso. Este teria a opção de escolher o modelo de aprendizagem que melhor se enquadra com o objetivo do curso.

Devido ao interessante relacionamento entre a flexibilidade do modelo de aprendizagem com a dinâmica de integração de novos serviços, foi incluído no esforço de implementação a verificação de viabilidade técnica dessa implementação no Ambiente AulaNet. Os detalhes dessa implementação são apresentados mais adiante.

6.6. Serviço implementado como *Web Service*

O estudo de caso no Ambiente AulaNet compreende a escolha de um determinado serviço e a sua implementação como um *web service* simulando o cenário desse serviço ser disponibilizado para acesso remoto por um fornecedor de *learning services*.

O serviço escolhido para ser implementado como *web service* é de Conferências, também conhecido como Fórum ou *Newsgroup*. A implementação deste serviço deve possibilitar a criação de conferências textuais assíncronas na forma de discussão orientada para um tópico. A tarefa de criação e configuração das conferências é realizada pelo coordenador do ambiente. Sendo uma discussão orientada a tópico e baseada em troca de mensagens, estas são organizadas hierarquicamente, isto é, estruturadas de acordo com o nível hierárquico da pergunta original. Com isso, é possível responder a uma mensagem e esta resposta será posicionada abaixo da mensagem em questão, formando um fluxo na discussão. Esta organização difere de listas de discussão convencionais onde mensagens de diferentes tópicos são reunidas em apenas um fluxo de discussão. Este relacionamento entre mensagens facilita o acompanhamento das várias linhas de discussão e melhora a visualização do encadeamento das mensagens.

As figuras 15 e 16 apresentam diagramas de casos de uso especificando de maneira resumida as principais funcionalidades disponíveis para alguns atores do AulaNet relativas ao serviço de conferência.

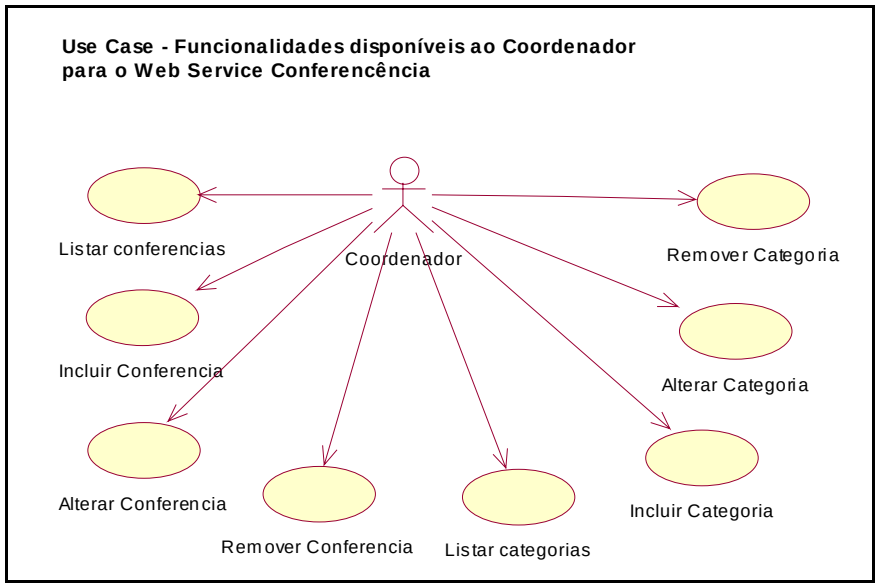

Figura 15 – Diagrama de casos de uso para o serviço de conferência para o coordenador do AulaNet.

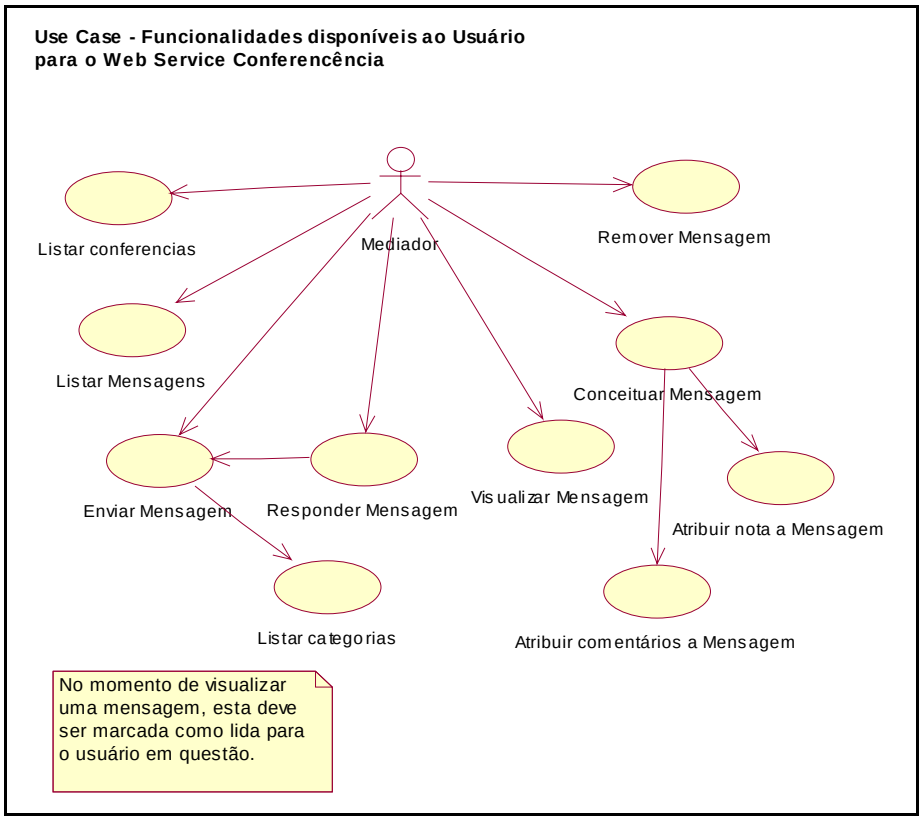

Figura 16 – Diagrama de casos de uso para o serviço de conferência para o mediador do AulaNet.

Considerando a necessidade de realizar o agrupamento dos serviços por domínio, tanto o serviço legado de Conferências no AulaNet quanto o novo serviço desenvolvido em *web service* foram categorizados em um grupo genérico

denominado Fórum. Definido o domínio dos serviços, a especificação da interface de comunicação do LMS com o grupo é gerada a partir das características dos serviços. A interface de acesso para o domínio de serviços Fórum pode ser parcialmente observada na figura 17.

```
public interface IConferenceService
{
    /**
     * Listar conferências
     * @param courseId
     * @param classId
     */
    Conference[]    listConference(long courseId, long classId);

    /**
     * Criar uma conferência
     * @param name
     * @param description
     * @param courseId
     * @param classId
     */
    Conference  createConference(String name, String description,
                                 long courseId, long classId);

    /**
     * Atualizar conferência
     * @param objConference
     */
    Conference  updateConference(Conference objConference);

    /**
     * Remover conferência
     * @param conferenceId
     */
    boolean     removeConference(long conferenceId);

    /**
     * Listar categorias da conferência
     * @param courseId
     */
    ConferenceCategoria[]   listCategoria(long courseId);
```

Figura 17 – Pequeno trecho da interface de acesso genérica para o grupo de serviços Fórum.

A partir da interface de acesso, a camada de visualização do serviço de Conferência do AulaNet foi modificada para somente acessar os métodos especificados nesta interface. Dessa forma, é possível garantir a compatibilidade entre a camada de visualização e os serviços de conferência – legado e *web services*.

6.7. Arquitetura e Implementação

Considerando que toda a comunicação entre os possíveis componentes a serem desenvolvidos é realizada utilizando protocolos-padrão de *web services*, algumas linguagens poderiam ser escolhidas para implementação dos componentes. Contudo, optou-se pela linguagem Java, por ser multi-plataforma e possuir ferramentas de desenvolvimento gratuitas. Entre as plataformas de *web services* mais comuns, optou-se por utilizar no desenvolvimento deste estudo de caso, o Apache Axis e a plataforma Eclipse.

Para a etapa inicial de implementação, o LMS AulaNet utilizou um serviço de conferência desenvolvido como um *web service* por um fornecedor específico. O AulaNet agiu como *service requestor*, enquanto o fornecedor de serviços agiu como o *service provider*. O LMS foi adaptado para possuir a funcionalidade de utilizar um *web service* específico na composição de seus cursos, ou seja, disponibilizar para seus usuários os seus serviços locais e o novo *web service* ao qual foi integrado. Nesta etapa, considerou-se a existência de uma interface pré-definida para o acesso ao *web service*, não necessitando o uso de servidores UDDI no processo. Neste caso, a descrição WSDL do *web service* foi gerada a partir dessa interface pré-estabelecida.

Como o LMS em questão não possuía nenhum tipo de componente integrador para *web services*, foi necessário implantar no mesmo um cliente SOAP para efetivar as chamadas remotas ao serviço. A figura 18 ilustra as características dessa arquitetura.

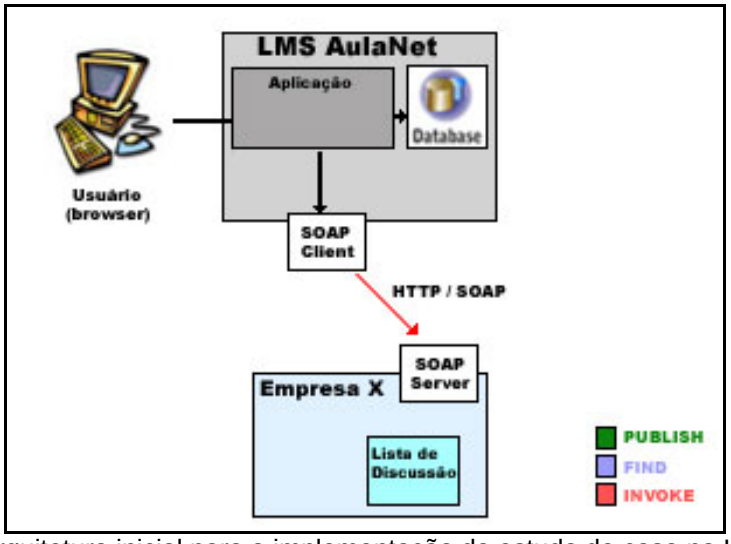

Figura 18 – Arquitetura inicial para a implementação do estudo de caso no LMS AulaNet.

Basicamente, a troca de informações entre o *web service* e o LMS pode ser resumida com base na ilustração da figura 19. No LMS foi necessário a instalação de um cliente SOAP e a criação de um *stub* (*service proxy*) para realizar as chamadas remotas ao serviço, que está implantado num servidor SOAP e possui um arquivo de implantação (DD) para descrevê-lo.

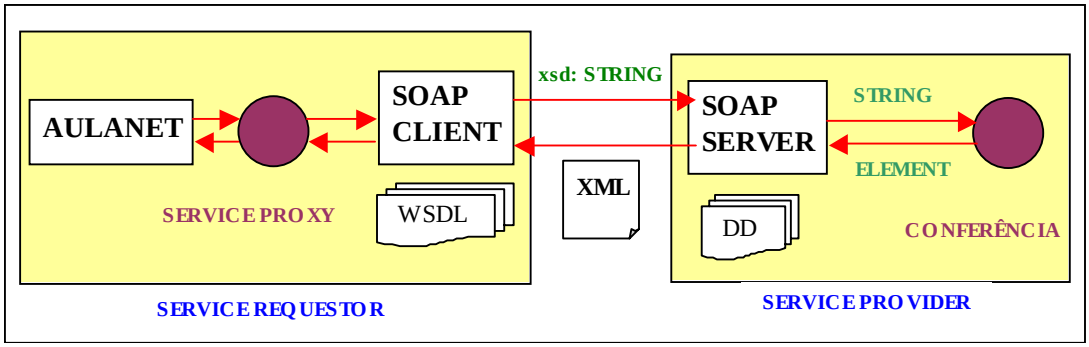

Figura 19 – Visão geral da arquitetura e comunicação implementada na fase inicial do estudo de caso.

A etapa seguinte de implementação do estudo de caso teve início após a efetiva comunicação do LMS com o *web service* implementado. O objetivo na segunda etapa de desenvolvimento foi instanciar o *framework* descrito no capítulo 6 e aplicar uma modelagem capaz de oferecer ao LMS uma flexibilidade com relação aos serviços.

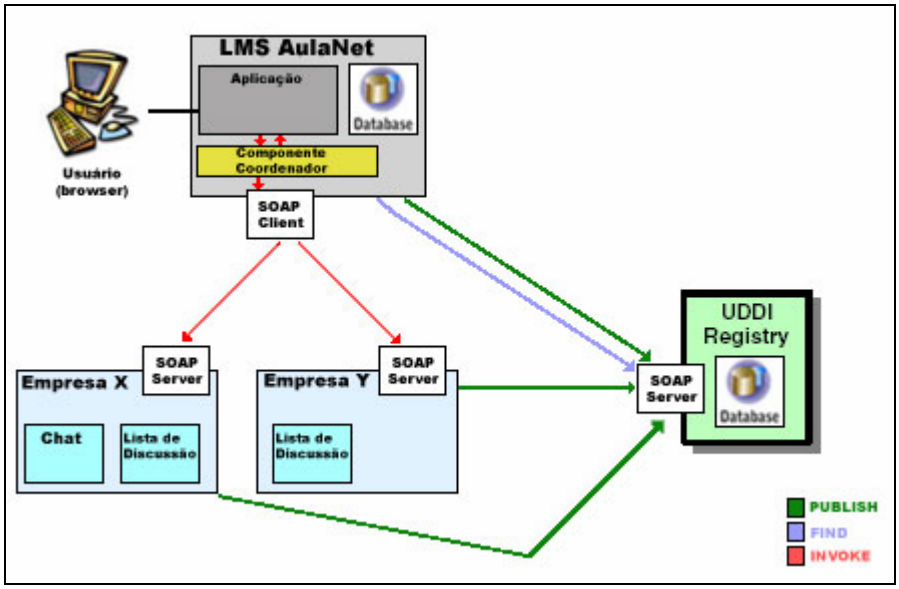

Figura 20 – Arquitetura e comunicação para o processo de instanciação do *framework* no LMS AulaNet.

A figura 20 permite uma visão geral da arquitetura com a aplicação do *framework* no AulaNet, sendo ilustrado como “componente coordenador”. Este realiza a mediação entre o LMS e os serviços disponíveis (legados ou *web services*). Para os *web services*, utiliza-se o cliente SOAP para efetivar a comunicação com diversos fornecedores de *learning services* que tiveram seus serviços integrados no AulaNet. O servidor UDDI representa a possibilidade dos fornecedores publicarem seus *web services* para que sejam encontrados pelo LMS ou alguma outra ferramenta de busca UDDI.

O processo de instanciação iniciou-se pela definição da interface de acesso para o grupo Fórum, como já descrito pela seção 6.6. Essa interface Java permite separar a camada de visualização do AulaNet das diversas implementações de serviços categorizados no grupo Fórum. Contudo, essa modelagem gerou o esforço de modificar a camada de visualização do serviço de conferência para utilizar os cabeçalhos dos métodos definidos na interface Fórum.

Para cada grupo de serviços criado, o *framework* exige a implementação de algumas classes para o gerenciamento das *factories* que instanciam os objetos para cada serviço. O primeiro ponto de implementação é a classe abstrata AbstractFactoryService, que deve ser especializada para o domínio de serviços em questão, neste caso, o Fórum. É nessa classe abstrata que contém a lógica de carregamento sob demanda das *factories* específicas de cada serviço.

Um grupo de serviços pode conter vários *learning services* e cada um deles deve estar associado com uma classe de mediação. Essas classes necessariamente implementam a interface pré-definida do domínio.

A figura 21 apresenta o diagrama de classes para o gerenciamento das *factories* e das classes de mediação do domínio de serviços Fórum. Essa modelagem refere-se ao padrão *Polymorphic factories* [Eckel, 2003] e o diagrama de classe é a especialização da figura 11 mostrada na seção 5.1 do capítulo 5, que discorre sobre as características do *framework*.

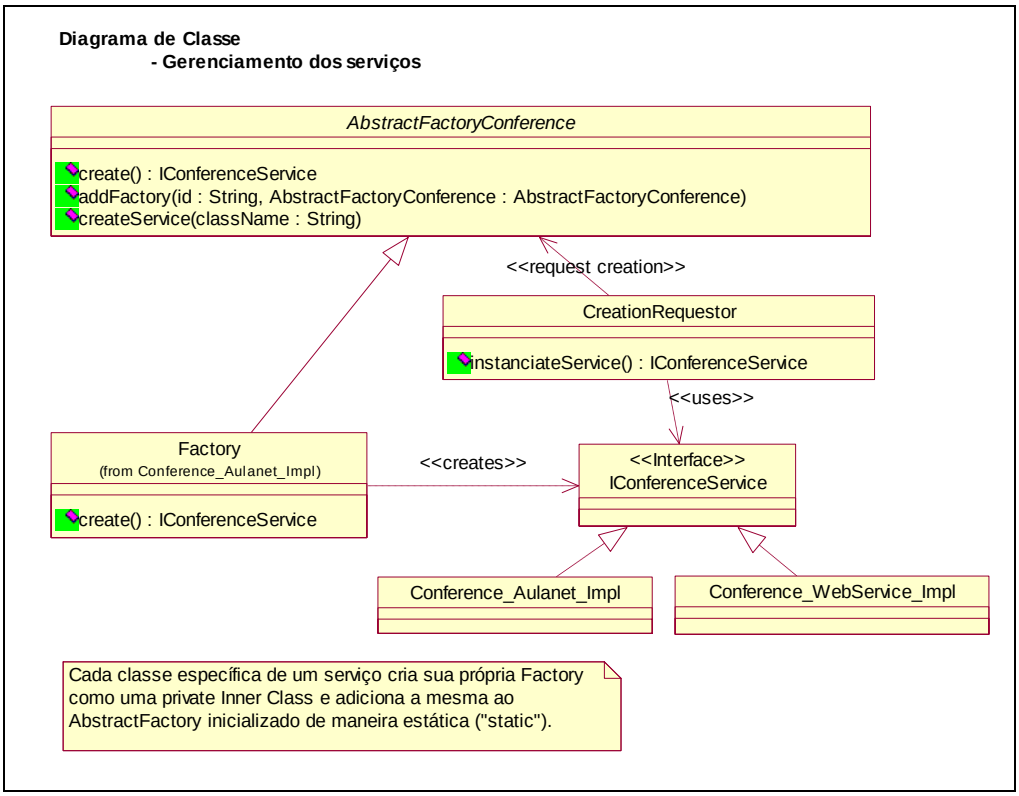

Figura 21 – Diagrama de classe para o gerenciamento das *factories* do grupo Fórum.

Naturalmente, os métodos contendo a lógica de negócios de cada um dos serviços do grupo Fórum não possuem os mesmos cabeçalhos da interface de acesso do grupo. Por isso, como já mencionado, cada implementação de serviço associado ao grupo Fórum tem uma classe para realizar a mediação da comunicação. Essa classe de mediação implementa a interface do grupo, encapsulando, por exemplo, as tarefas de conversão de dados, mapeamentos, compatibilidades, chamadas remotas (para o caso de *web services*) e tratamento de exceções.

Para o caso das chamadas remotas dos serviços implementados como *web services*, é necessária a utilização de *stubs* e estes são mantidos de forma transparente para o LMS nas classes de mediação dos serviços remotos.

A figura 22 apresenta um pequeno trecho da classe de mediação especifica do serviço de Conferência implementado como *web service*. Nesse trecho de código, pode-se observar a *inner class* especificada pelo padrão *Polymorphic factories* que contém o método que dispara a criação da *factory* específica desse serviço.

```
public class Conference_WebService_Impl implements IConferenceService
{
   private static int count = 0;

   /**
    * Construtor
    */
   public Conference_WebService_Impl()
   {
        super();
   }
   /**
    * Inner class para o padrão Polymorphic factories
    *
    * @author Reubem Girardi
    */
   private static class Factory extends AbstractFactoryConference
   {
        protected IConferenceService create()
        {
           count++;
           System.out.println("Conference_WebService_Impl count = " + count);
           return new Conference_WebService_Impl();
        }
   }

   static
   {
      AbstractFactoryConference.addFactory(
       "conference_webservice.Conference_WebService_Impl" , new
          Conference_WebService_Impl.Factory() );
   }
```

Figura 22 – Pequeno trecho da classe de mediação do serviço Conferência implementado como *web service*.

Neste momento, faz-se necessário a apresentação das modificações na camada de visualização do AulaNet para permitir sua integração com o *framework* e a manutenção dos serviços.

O ambiente AulaNet não contempla a possibilidade de gerenciamento dos serviços no sistema, isto é, não possui nenhuma interface de cadastro e alteração dos serviços disponíveis no ambiente. Dado que a implantação do *framework* no AulaNet permite a inclusão e remoção dos serviços, fez-se necessário uma forma de manutenção dos *learning services* cadastrados no AulaNet. Essa interface de cadastro compreende a administração de duas entidades: domínios de serviços e dos serviços propriamente ditos.

Para essa implementação, optou-se por disponibilizar essas funcionalidades na área de administração do Ambiente AulaNet. Dessa forma, o administrador do ambiente poderá realizar o cadastramento de domínios e serviços, sendo que todo serviço deve estar associado a um domínio para utilizar a modelagem do *framework*.

A figura 23 apresenta a listagem de funcionalidades do administrador, incluindo as novas opções de: “Listar Serviços AulaNet”, “Listar *Web Services*” e “Configurar Domínios”.

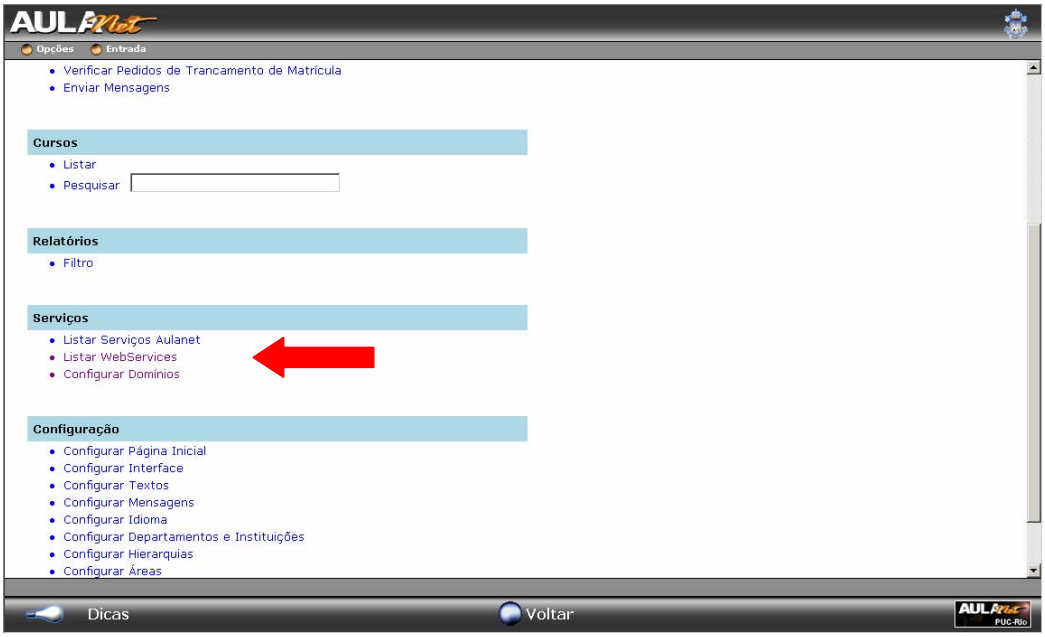

Figura 23 – Configuração de domínio e serviços para o administrador do AulaNet.

Ao acessar a opção de Configurar Domínios, o administrador visualizará uma listagem dos domínios cadastrados. Como dito anteriormente, o domínio é uma categorização dos serviços para permitir uma maior flexibilidade do LMS e a sobrecarga da camada de visualização. A figura 24 mostra a listagem dos domínios e a ocorrência do grupo Fórum.

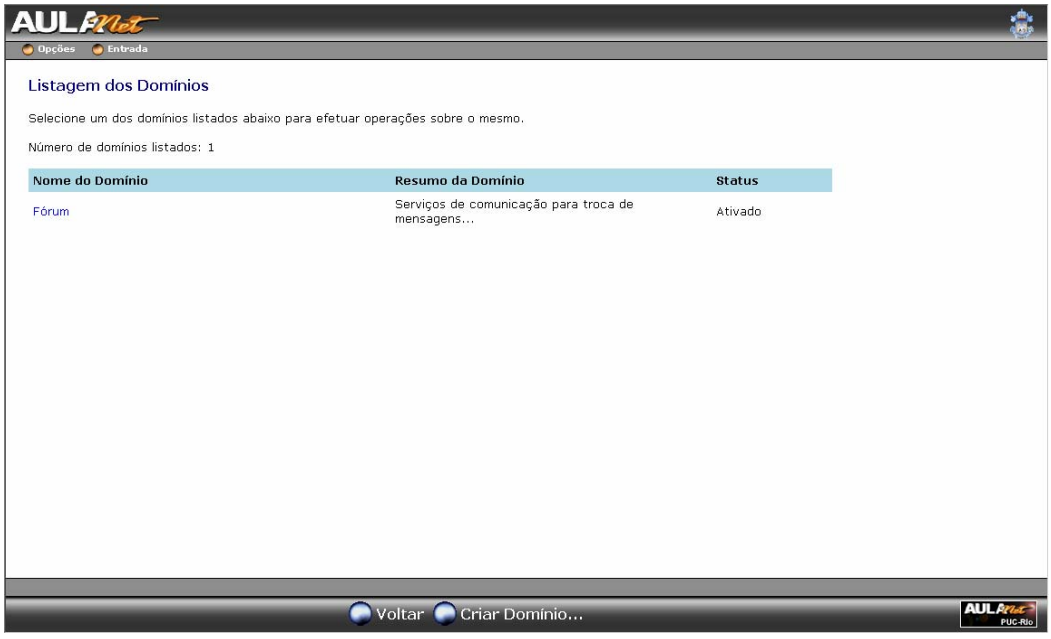

Figura 24 – Listagem dos domínios cadastrados, permitindo sua configuração.

Considerando os domínios de serviços cadastrados, o administrador pode acessar as opções de manutenção dos serviços, legado e remotos, para realizar o cadastramento ou a associação de algum serviço a um determinado domínio. Neste caso, acessando qualquer uma das opções, o administrador encontrará uma listagem dos serviços já cadastrados. As figuras 25 e 26 apresentam, respectivamente, a listagem de *web services* cadastrados no ambiente e os serviços legados do AulaNet.

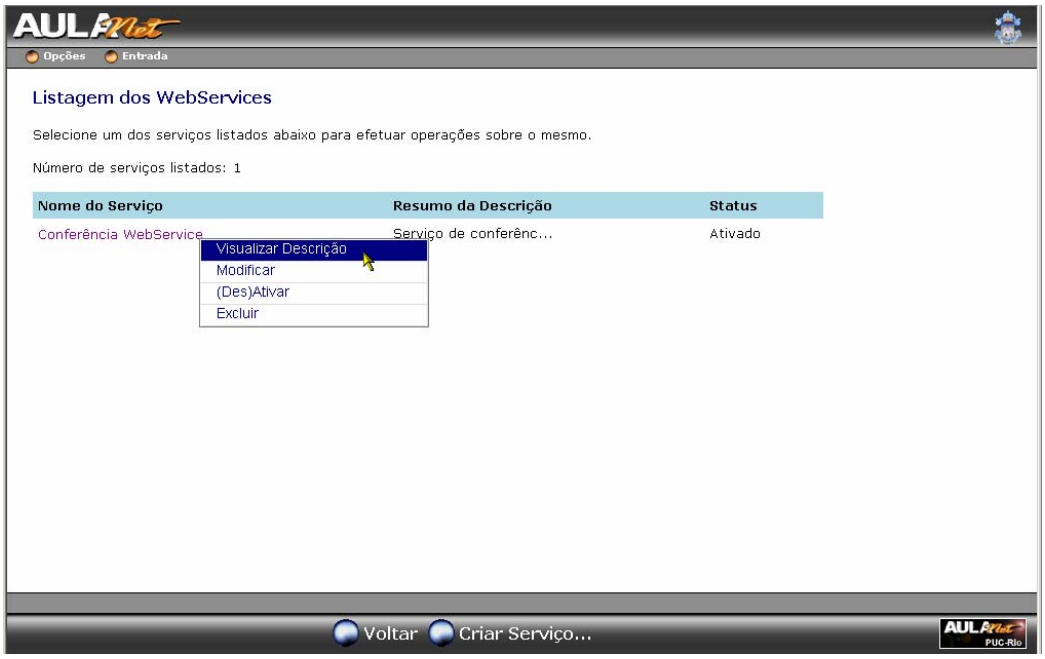

Figura 25 – Listagem dos *web services* cadastrados.

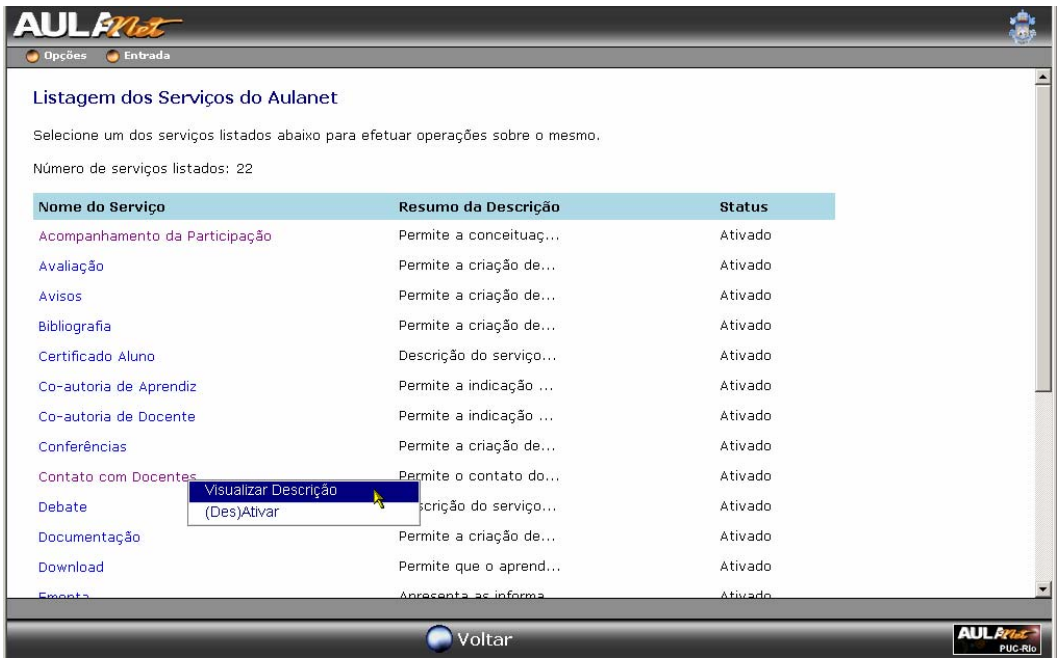

Figura 26 – Listagem dos serviços legados do AulaNet.

Para o caso dos serviços remotos, a opção Modificar permite a visualização do formulário de cadastro e alteração do serviço, ilustrado a seguir pela figura 27.

Figura 27 – Formulário de cadastro dos serviços remotos no AulaNet.

É interessante notar na figura 27 que o formulário de cadastro dos serviços compreende duas importantes informações: domínio e classe. O campo “domínio” representa em qual grupo o serviço é categorizado e o campo Classe contém um identificador para o carregamento dinâmico da classe de mediação durante o gerenciamento das *factories* dos serviços.

Tanto a existência do campo Classe quanto o padrão *Polymorphic factories* foram decisões de modelagem para minimizar o esforço de alterações e o impacto da inclusão de novos serviços no ambiente.

Considerando os domínios e serviços cadastrados, o AulaNet ainda não possui uma forma de disponibilizar para o coordenador o novo *learning service* de conferência implementado como *web service*. Como mencionado na seção 6.2, os serviços legados do AulaNet são organizados com base no modelo de aprendizagem colaborativa (3C) adotado pelo ambiente. Dessa forma, durante a criação do curso, os serviços são apresentados ao coordenador de forma inflexível e imutável. A figura 28 ilustra esse cenário mostrando uma das telas visualizadas

pelo coordenador durante o processo de escolha dos serviços para a criação do curso.

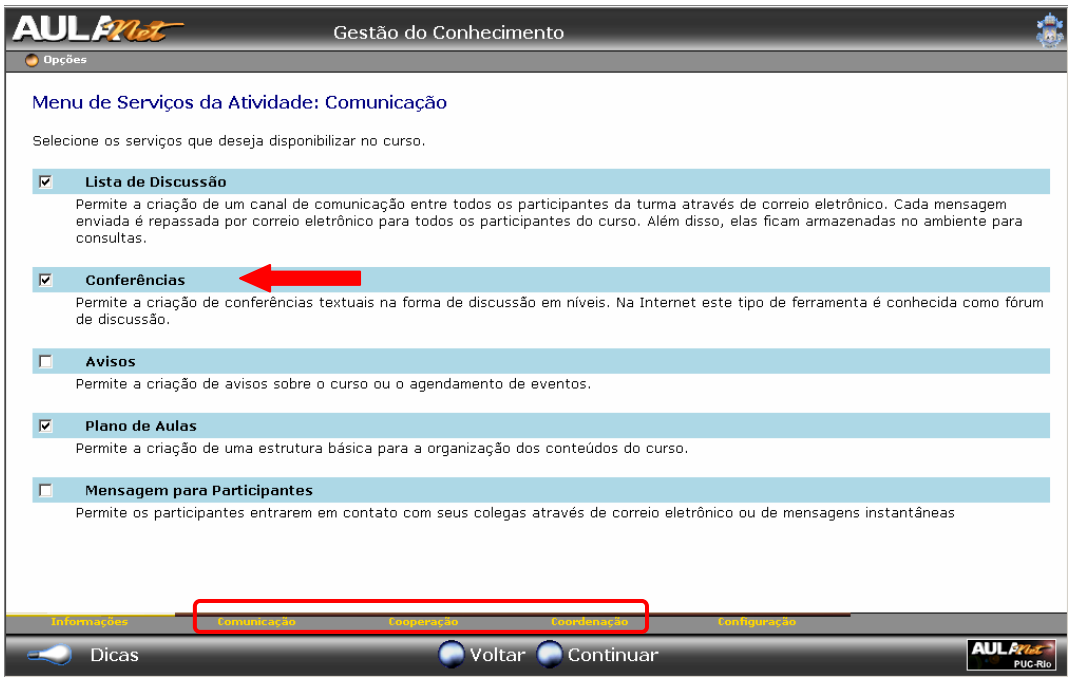

Figura 28 – Mecanismos de comunicação do AulaNet.

Pode-se notar que o modelo de aprendizagem 3C está representado pela barra grifada na figura e que os serviços são disponibilizados ao coordenador segundo esse critério de organização. Ainda na figura 28, são apresentados os mecanismos de comunicação, sendo que um deles é o serviço de Conferências proprietário do AulaNet. Facilmente conclui-se que essa interface de visualização de serviços deve ser flexível com relação a quais *learning services* serão visualizados pelo coordenador. Com isso, foi incluído na implementação do estudo de caso o acréscimo de funcionalidades no AulaNet que permitissem a capacidade de configuração de outros modelos de aprendizagem diferentes do modelo 3C.

A possibilidade do gerenciamento dos modelos de aprendizagem permitiria a atualização do modelo 3C e o cadastramento de outros modelos de aprendizagem, admitindo variadas combinações de serviços. Essa total flexibilidade oferece ao coordenador a oportunidade de criar um curso sob a ótica do modelo de aprendizagem que seja o mais indicado ao perfil dos aprendizes e do conteúdo a ser disponibilizado.

Vale a pena ressaltar que o objetivo desse trabalho de pesquisa não é detalhar os diversos modelos de aprendizagem encontrados na literatura da área. O objetivo principal é possibilitar que o Ambiente AulaNet tenha a capacidade de oferecer aos seus usuários (docentes e aprendizes) um sistema flexível e suficientemente robusto, disponibilizando funcionalidades e configurações que permitam-lhes uma melhor aplicação dos cursos.

A partir da necessidade de flexibilizar o AulaNet com relação aos modelos de aprendizagem, incluiu-se no esforço de implementação a criação de uma interface no ambiente que disponibilizasse essas funcionalidades. Para a tarefa de realizar a manutenção dos modelos de aprendizagem, esse trabalho de dissertação propõe a criação de um ator no ambiente AulaNet – denominado Mentor – especificamente para a realização dessa tarefa. A criação desse novo ator no ambiente e a escolha do termo Mentor foram motivadas pela característica especializada da tarefa de configuração dos modelos.

A criação de um novo ator no ambiente ocasionou em uma área de opções distinta dos outros atores do AulaNet. A figura 29 apresenta essa nova área de acesso para o Mentor.

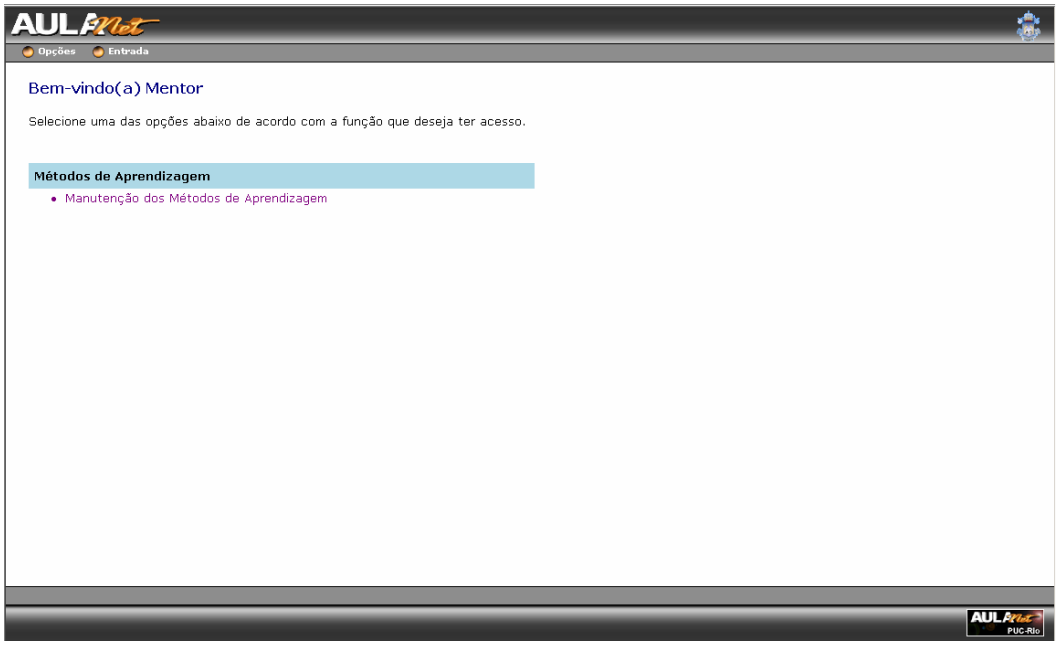

Figura 29 – Interface de acesso do Mentor no Ambiente AulaNet.

O Mentor possui a funcionalidade “Manutenção dos Métodos de Aprendizagem”, que disponibiliza uma listagem com todos os métodos cadastrados no ambiente. Essa modelagem contempla dois conceitos: Métodos de

Aprendizagem e Atividades. Os Métodos são compostos de uma ou mais Atividades, sendo que os serviços encontram-se organizados nas várias Atividades de cada Método. Ao acessar as opções de um determinado Método cadastrado, o Mentor pode configurar as atividades do mesmo. A figura 30 apresenta a listagem disponível para o Mentor dos métodos de aprendizagem.

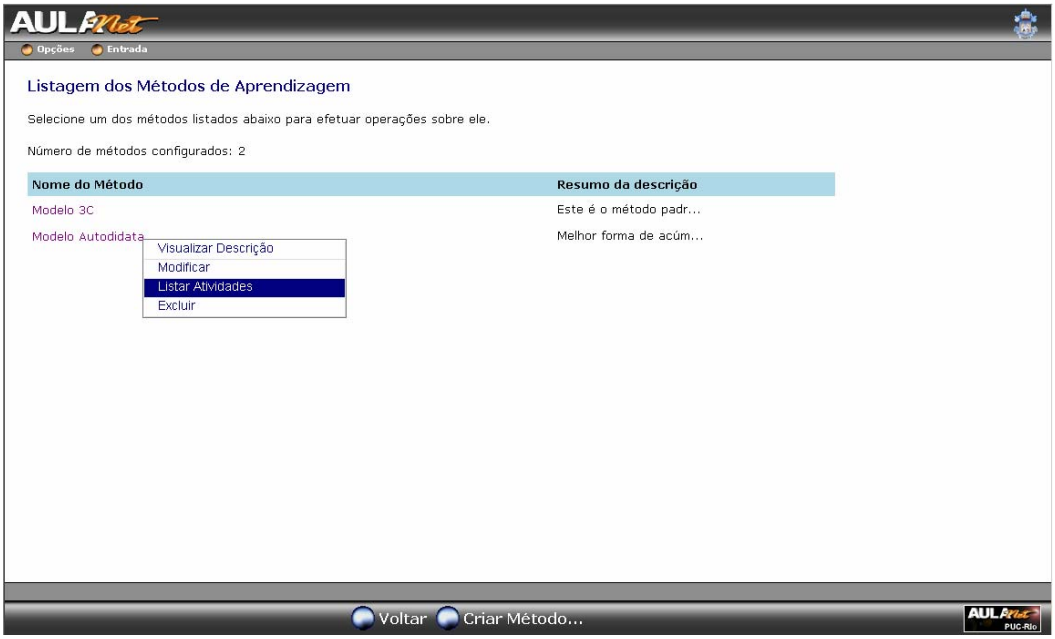

Figura 30 – Listagem dos modelos de aprendizagem.

Obviamente, para garantir a compatibilidade com o estado original do ambiente, o modelo 3C foi configurado com as atividades “comunicação”, “coordenação” e “cooperação”. As atividades possuem duas informações fundamentais: ordem e serviços associados. A ordem representa o critério de ordenação das atividades entre si. Um método pode conter várias atividades e faz-se necessário estabelecer uma ordenação para o momento de disponibilizar as atividades na interface do coordenador durante o processo de criação e configuração do curso. Com relação aos serviços, uma atividade pode conter um ou mais serviços – legados ou *web services*.

As opções disponíveis para o Mentor referindo-se as atividades podem ser visualizadas na figura 31, que lista as atividades do método de aprendizagem 3C original do AulaNet. Ao escolher a opção “(Des)Associar Serviços” entre as opções de uma atividade, o Mentor pode observar a listagem de serviços que já foram associados com a atividade e os serviços ainda disponíveis no AulaNet. É importante ressaltar que a associação de um *learning service* a uma determinada

atividade reflete em alterações na interface de criação de cursos do coordenador. Essas funcionalidades de associação de *learning services* com as atividades podem ser confirmadas pela figura 32, que evidencia a ocorrência do serviço de conferência legado do AulaNet associado com a atividade “Comunicação” do modelo 3C e a possibilidade de substituí-lo pelo novo serviço de conferências implementado como *web service*.

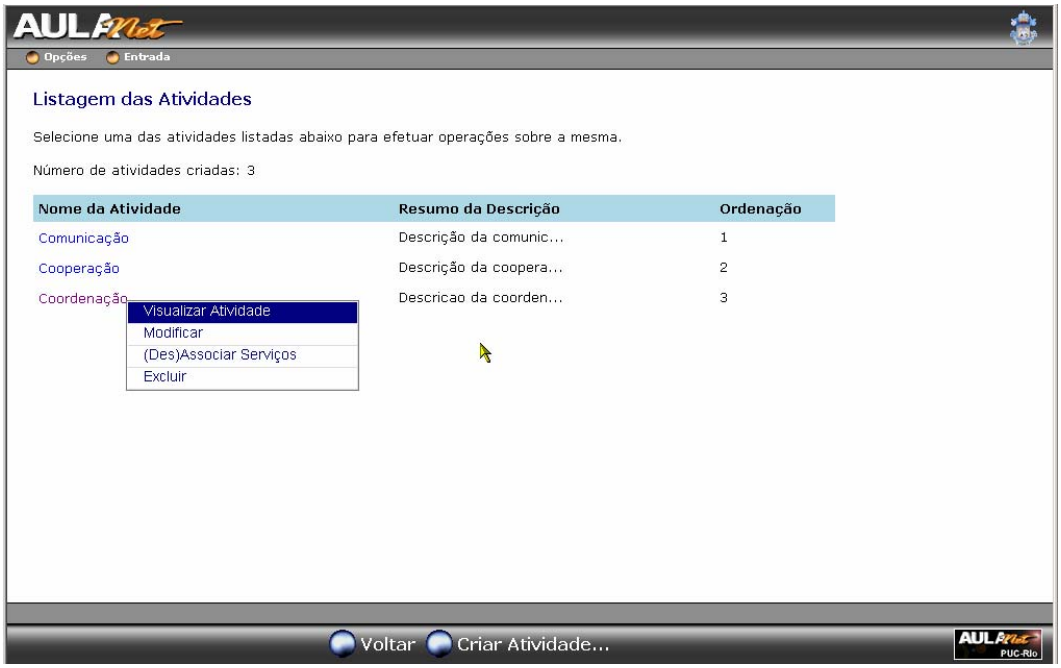

Figura 31 – Listagem das atividades de um método de aprendizagem.

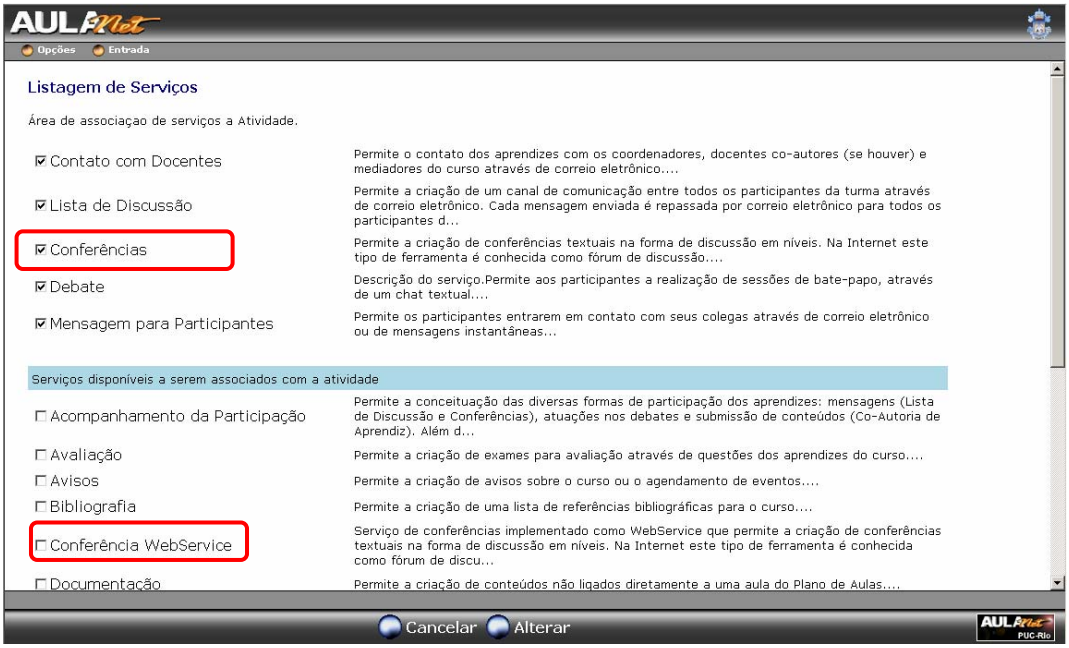

Figura 32 – Interface para a associação de serviços em uma atividade.

Logo que o Mentor realiza a permutação entre os serviços de conferência, novos cursos criados com base no modelo 3C poderão ser configurados para utilizar o novo serviço implementado como *web service*. A figura 33 mostra a interface do coordenador com o novo *learning service* Conferência Web Service. Os destaques na figura 33 evidenciam o novo serviço disponível para o coordenador e a barra de navegação indicando as atividades do modelo 3C.

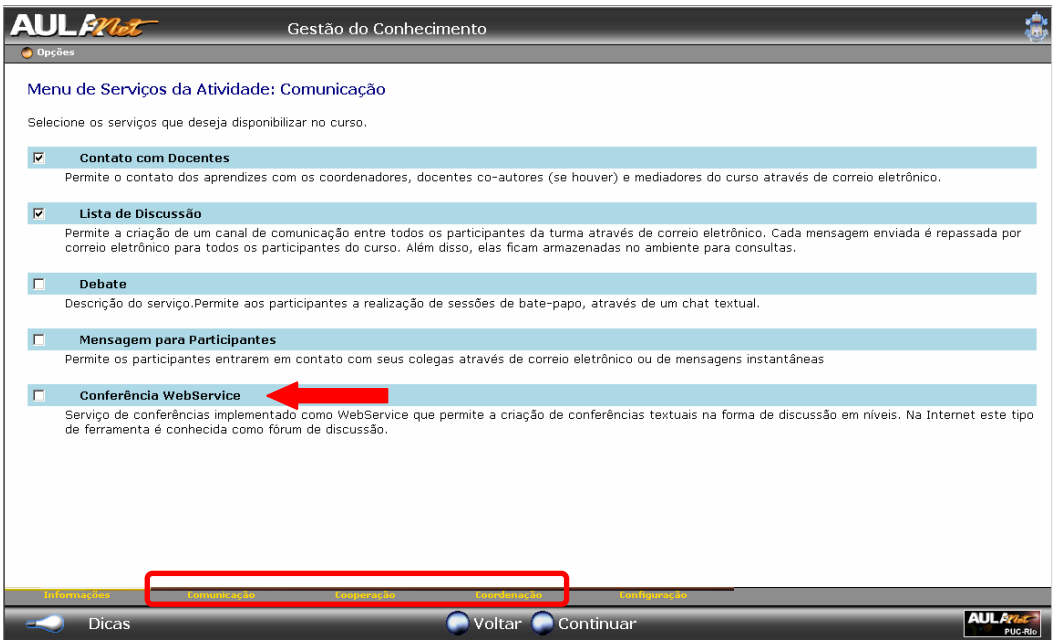

Figura 33 – Interface de criação do curso baseado no modelo 3C com o novo serviço conferência implementado como *web service*.

Mesmo que um novo curso baseado no modelo 3C seja criado e o coordenador habilite o novo serviço de conferência implementado como *web service*, para os aprendizes essa substituição é completamente transparente devido a modelagem de gerenciamento dos serviços e as características da nova implementação da camada de visualização. Como dito anteriormente, a camada de visualização do serviço de conferências foi adaptada para se tornar robusta o suficiente para ser capaz de disponibilizar suas funcionalidades dependendo de qual serviço está sendo utilizado do grupo Fórum. Essas características são discutidas com mais detalhes na seção 6.8.

Para esse estudo de caso no Ambiente AulaNet, criou-se um método de aprendizagem no sistema diferente o método 3C. Com o objetivo de validar a flexibilidade do AulaNet com vários métodos de aprendizagem, realizou-se o

cadastro de um método denominado “Autodidata”, contendo apenas 1 atividade chamada “Estudo Individual” associada a alguns serviços que pudessem auxiliar ao aprendiz a concluir o curso de forma autônoma. Antes de iniciar a criação do curso, o coordenador escolhe o método de aprendizagem do novo curso e prossegue com a configuração do mesmo. Com isso, a criação de um curso baseado nesse novo método de aprendizagem pode ser confirmado através da figura 34, que apresenta a interface de criação de curso no AulaNet disponível para o coordenador. Nessa tela pode-se constatar a barra de navegação grifando a única atividade configurada para o método de aprendizagem “Autodidata”.

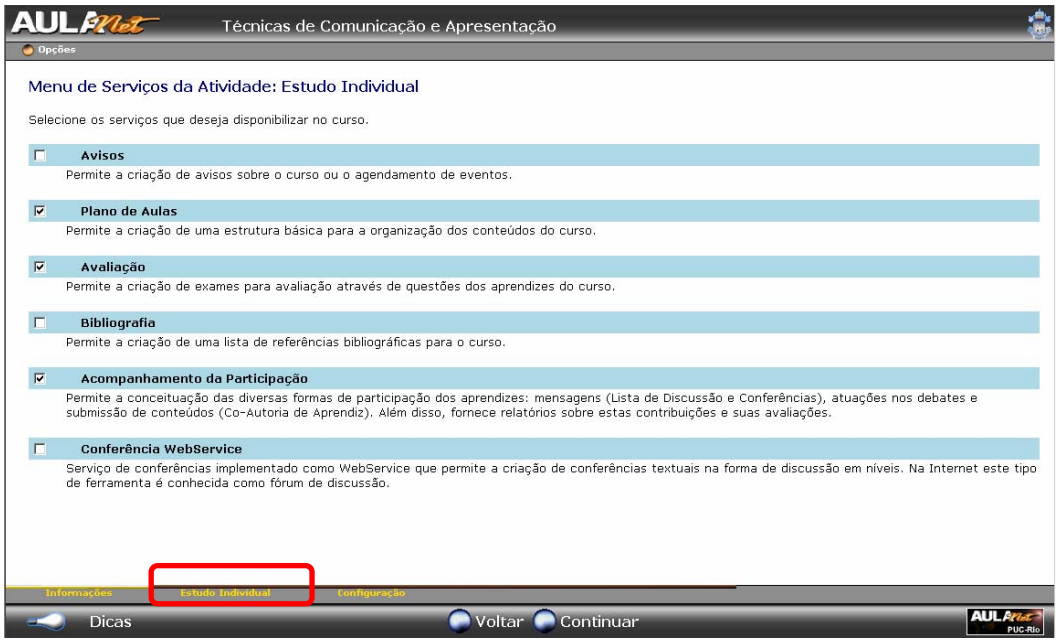

Figura 34 – Interface de criação de curso baseado num modelo diferente do 3C.

Dessa forma, num mesmo ambiente AulaNet, podem existir vários cursos seguindo métodos de aprendizagem diferentes, todos configurados a partir da interface do Mentor.

6.8. Implicações na Camada de Visualização dos Serviços

Essa seção apresenta detalhes das modificações realizadas na camada de visualização do AulaNet visando ampliar os benefícios da flexibilização dos serviços no ambiente. Como já mencionado nas seções anteriores, um dos

objetivos foi manter transparente a permutação de serviços para os usuários do sistema, tanto para o coordenador quanto aprendizes. Esse benefício foi adquirido com a reformulação do código referente a camada de visualização do serviço e com a especificação de uma interface única de comunicação com os vários serviços de um mesmo domínio.

Vale ressaltar que os *learning services* agrupados em um domínio possuem características similares, mas não necessariamente funcionalidades idênticas e integralmente mapeadas. Para o estudo de caso, esse cenário foi considerado com o novo *web service* de conferência e o serviço proprietário do AulaNet. Ambos possuíam as funções básicas de um serviço de fórum convencional, por exemplo, criação e configuração de conferências e envio e remoção de mensagens. Contudo, ambos continham funcionalidades exclusivas, exigindo que a camada de visualização do AulaNet realizasse um esforço maior de gerenciamento das opções a serem disponibilizadas aos usuários. Um exemplo desse cenário pode ser constatado através das figuras 35 e 36, que apresentam a primeira tela de configuração efetiva do serviço de conferências. A tela visualizada nas figuras é a primeira que realmente acessa as funcionalidades do serviço e que depende das características do mesmo para a decisão de quais funcionalidades devem ser oferecidas ao coordenador. A figura 35 mostra a lista de opções do *learning service* conferências proprietário do AulaNet. Um fato relevante é que, originalmente, essa tela de configuração só disponibilizava as três primeiras funcionalidades: “Listar Conferências”, “Definir Categorias” e “Optar por Receber Alertas”. As duas funcionalidades grifadas (“Habilitar Envio de Mensagens E-mail” e “Habilitar Tradução Mensagens E-mail”) foram adicionadas por fazerem parte do novo serviço implementado como *web service*.

As diferenças de funcionalidades entre serviços do mesmo domínio são tratadas pela interface a partir de informações obtidas das respectivas classes de mediação do *framework*. O exemplo pode ser observado na figura 35, onde as novas funcionalidades que são exclusivas do *learning service* implementado como *web service* aparecem desabilitadas, pois o serviço acesso na figura 35 é o proprietário do AulaNet. É importante observar que, para essa implementação, optou-se por desabilitar as opções que não se referem ao *learning service* corrente associado ao cuso em vez de simplesmente não mostrar as tais opções.

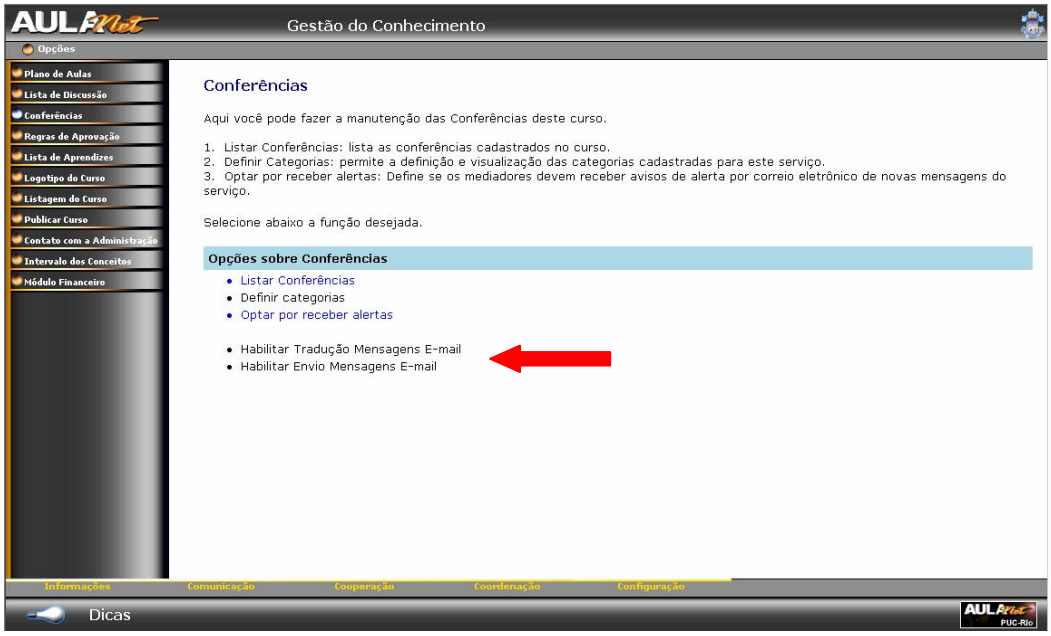

Figura 35 – Configuração do serviço de Conferências utilizando a implementação proprietária do AulaNet.

Por sua vez, a figura 36 apresenta a tela de configuração considerando o uso do *web service*, indicando quais opções se encontram disponíveis. Neste caso, o controle das diferenças ocorre a partir da camada de visualização com base nas informações providas pela interface de comunicação.

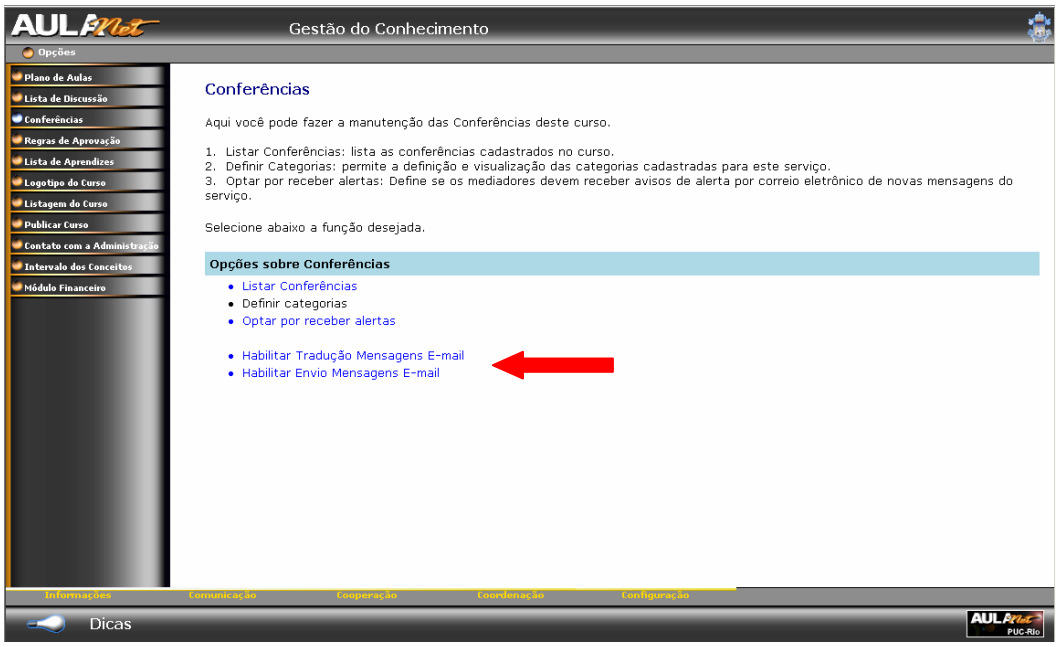

Figura 36 – Configuração do serviço de Conferências utilizando a nova implementação baseada em *web service*.

O processo de especificação da interface para um determinado domínio deve contemplar as principais características dos *learning services* do domínio. Dessa forma, a camada de visualização poderá coordenar a disponibilidade das funções.

A seguir, a figura 37 apresenta mais um trecho da interface Java para a comunicação do AulaNet com os serviços do domínio Fórum. Mais especificamente, o trecho de código evidencia a especificação de métodos que serão utilizados pelo LMS para a execução da lógica de visualização das funcionalidades. No caso, como cada *learning service* do domínio Fórum deve possuir sua classe de mediação que implementa tal interface Java, consequentemente o LMS pode dispor de informações sobre as características de cada serviço. Para o caso do serviço conferência proprietário do AulaNet, a camada de visualização está utilizando os métodos "*hasFunctionTraduzirMensagem*" e "*hasFunctionEnviarMensagemEmail*" para decidir se as novas opções de "Habilitar Envio de Mensagens E-mail" e "Habilitar Tradução Mensagens E-mail", respectivamente, devem ser disponibilizadas para o coordenador.

```
/**
 * Verifica existência da funcionalidade de categorias
 * @return boolean indicando a existência ou não da funcionalidade
 */
boolean hasFunctionCategorias ();

/**
 * Verifica existência da funcionalidade de receber alerta
 * @return boolean indicando a existência ou não da funcionalidade
 */
boolean hasFunctionReceberAlertas ();

/**
 * Verifica existência da funcionalidade de tradução de mensagens
 * @return boolean indicando a existência ou não da funcionalidade
 */
boolean hasFunctionTraduzirMensagem ();

/**
 * Verifica existência da funcionalidade de envio por e-mail das mensagens
 * @return boolean indicando a existência ou não da funcionalidade
 */
boolean hasFunctionEnviarMensagemEmail ();
```

Figura 37 – Trecho da interface Java com os métodos de controle de funcionalidades para os serviços do domínio Fórum.

Para melhor exemplificar a abstração de tarefas e a coordenação das operações referentes aos casos de uso do domínio de serviços Fórum, optou-se por detalhar a funcionalidade de “Enviar Mensagem”. Esse caso de uso corresponde a ação de enviar uma mensagem para uma determinada conferência por um dos atores do AulaNet. A camada de visualização desse caso de uso é apresentada na figura 38, onde se pode notar o formulário para a inserção das informações pertinentes a mensagem.

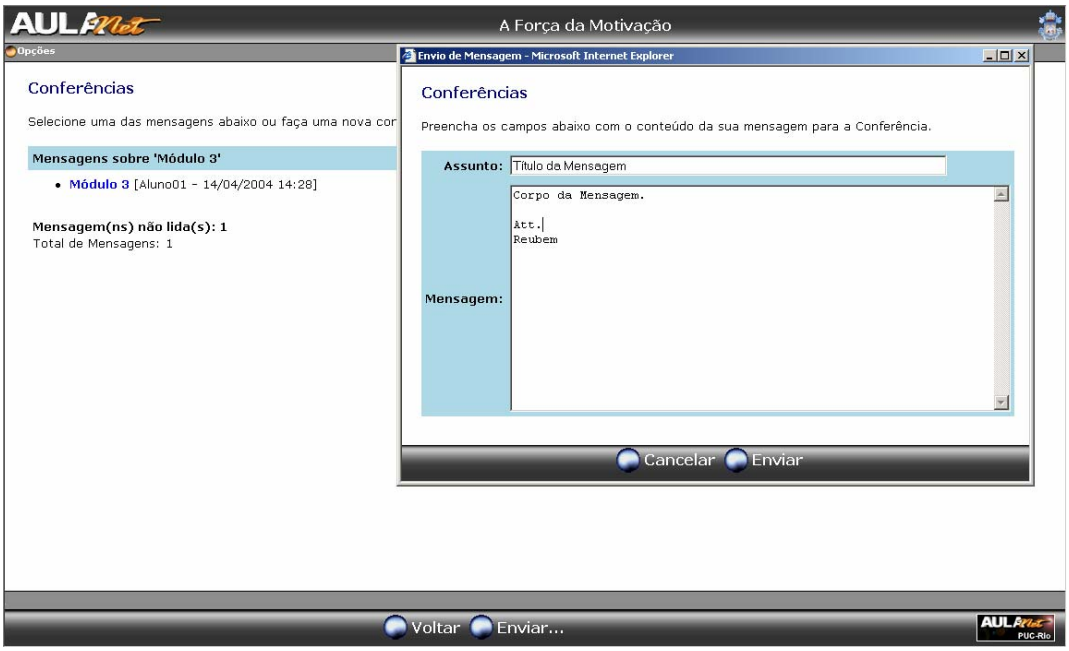

Figura 38 – Interface para o envio de mensagens nas Conferências.

Para utilizar da flexibilidade e outros benefícios adquiridos pela integração com o *framework*, tanto o formulário de envio das mensagens quanto as outras funcionalidades intrínsecas às telas visualizadas na figura 38 também foram adaptadas para utilizar as classes criadas para o domínio de serviço Fórum. Antes dessa adequação, a camada de visualização só refletia a implementação do serviço proprietário do AulaNet e comunicava-se exclusivamente com classes que realizavam o acesso ao banco de dados do sistema. Com a possibilidade da adoção de um serviço de Conferências implementado como *web service*, a classe de mediação do domínio Fórum decidirá qual procedimento deve realizar dependendo do serviço utilizado por determinado curso. A diferenciação, neste exemplo, ocorre entre redirecionar a requisição para as classes proprietárias do AulaNet com acesso ao banco de dados ou efetivar a comunicação com o *web service* através de mensagens SOAP. Vale a pena frisar que para a camada de

visualização do serviço de Conferências, por exemplo, o envio de mensagens, essa diferenciação é totalmente transparente, pois sua comunicação com o *framework* ocorre através de uma API previamente definida.

Com o objetivo de detalhar as mensagens trocadas entre o *web service* de Conferências e o AulaNet, a figura 39 apresenta a mensagem SOAP referente ao envio de uma mensagem pela camada de visualização do serviço de Conferência.

```
<?xml version="1.0" encoding="UTF-8" ?>
<SOAP-ENV:Envelope  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:q0="http://systinet.com/xsd/SchemaTypes/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<SOAP-ENV:Body>
<q0:title>[Titulo] Mensagem de teste</q0:title>
      <q0:message>[Corpo da mensagem] Essa é uma mensagem de teste do serviço
       Conferência implementado como web service.</q0:message>
      <q0:parentMessageId>0</q0:parentMessageId>
      <q0:conferenceId>1</q0:conferenceId>
      <q0:userId>25</q0:userId>
      <q0:nickname>Reubem</q0:nickname>
 </SOAP-ENV:Body>
 </SOAP-ENV:Envelope>
```

Figura 39 – Mensagem SOAP referente ao envio de uma mensagem nas Conferências.

O estudo de caso no AulaNet possibilitou a realização de uma análise do esforço de implementação para a flexibilização dos serviços. O principal ponto verificado são as implicações no código legado do sistema, pois o objetivo foi minimizar as modificações no AulaNet geradas pela aplicação do *framework*. As modificações na camada de visualização foram vantajosas quando comparadas com o esforço de criação de interfaces adicionais para cada novo *learning service*. A possibilidade de reuso da camada de visualização e da permutação dos *learning services* tornou-se estrategicamente benéfico para o ambiente considerando o estado monolítico em que se encontra a maior parte dos sistemas de *e-Learning* no mercado.

A combinação de cenários contemplados pelo AulaNet com base nos novos pontos de flexibilização adquiridos são diversos, permitindo uma maior autonomia do sistema com fornecedores de serviços e com metodologias de aprendizagem.