5
Framework para coordenação e mediação de Web Services para ambientes de aprendizado à distância

O capítulo anterior apresentou uma discussão sobre a inclusão dos chamados *learning services* no processo de padronização de *learning objects* visando obter uma maior interoperabilidade e reusabilidade por parte dos serviços. Observaram-se as vantagens de propor a padronização de funcionalidades básicas dos domínios de serviços e permitir que fornecedores estendam o padrão incluindo funcionalidades proprietárias incentivando a criatividade e inovação nos serviços. Contudo, este trabalho não possui o foco na criação de um padrão proprietário para domínios de serviços. Com isso, optou-se por analisar o modelo atual de interação entre sistemas de *e-Learning* e fornecedores de serviços, que transfere para aquele a tarefa de adaptação no momento de permutar entre serviços de mesmo domínio.

Dessa forma, com objetivo de embasar a proposta teórica desse trabalho de pesquisa, propõe-se a criação de um *framework* para inserir os *learning services* no conceito de *learning objects* e facilitar a adequação de sistemas de *e-Learning* ao novo conceito.

Este *framework* possui os principais objetivos:

- Facilitar a coordenação de *web services* no LMS;
- Flexibilizar a interface de comunicação entre um LMS e os variados domínios de *learning services* co-relacionados;
- Criar uma camada entre o LMS e os *web services* permitindo a permutação de *learning services* de mesmo domínio sem a necessidade de alterações na visualização do serviço por parte do LMS;
- Encapsular a mediação entre os *web services* e o LMS com relação à semântica das informações trocadas;
- Abstrair tarefas ao LMS, principalmente, com relação às chamadas remotas aos *web services*;
- Facilitar integração com sistemas de *e-Learning* já desenvolvidos;

- Reduzir o esforço de flexibilizar o LMS em relação à integração com *web services*.

O objetivo principal do *framework* é auxiliar a adaptação de um LMS para permitir a integração com *learning objects* baseados em serviços e desenvolvidos como *web services*. Neste caso, a finalidade é atribuir ao LMS a possibilidade de possuir um comportamento dinâmico relativo à sua composição e funcionalidades disponíveis. Esta característica dinâmica é justificada com base no fato de que o sistema possuirá a capacidade de usar diferentes serviços, atribuindo-lhe novas funcionalidades, que não fazem parte da sua modelagem e implementação original.

O principal cenário idealizado é permitir que durante a configuração do ambiente de aprendizado do participante no LMS, o sistema possa utilizar um serviço remoto que não seja legado do próprio ambiente.

Neste sentido, a maior dificuldade no esforço de adaptar o LMS está relacionada com a interface de acesso ao serviço remoto, que a princípio, é totalmente desconhecida pelo LMS. Uma solução é a especificação de um padrão de comunicação e integração entre o LMS e o *web service* em questão. Contudo, como questionado anteriormente, a adoção de um padrão nesta comunicação gera limitações indesejadas nas implementações dos *web services*. Esse é um dos objetivos intrínsecos ao *framework* proposto, criar uma camada de mediação entre o LMS e os *web services*.

Essa camada de mediação possibilita ao LMS abstrair as especifidades das interfaces de acesso dos diferentes serviços e permite a integração com um novo serviço através do mapeamento dos seus métodos com as interfaces pré-definidas para os domínios de serviços configurados e utilizados pelo LMS.

A seguir, encontra-se a figura 9 que ilustra a arquitetura da utilização do *framework*. Neste caso, o *framework* instanciado está representado como o componente coordenador destacado na figura, criando uma interface entre os *web services* e o LMS.

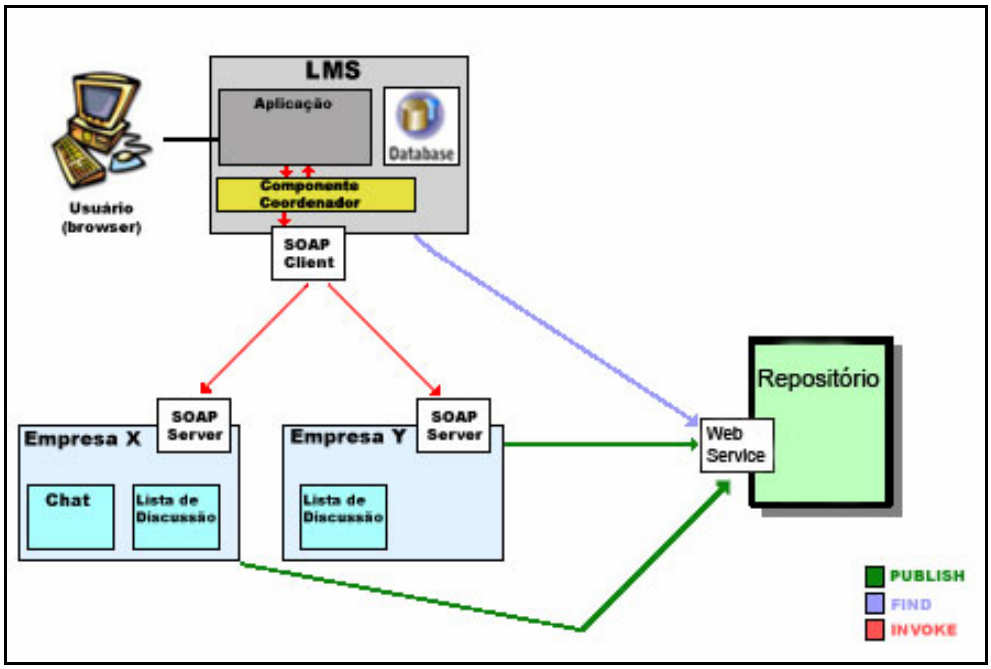

Figura 9 – Visão geral da arquitetura do *framework* para a coordenação e mediação de *web services* para sistemas de *e-Learning*.

O *framework* possui as seguintes funcionalidades:

- Oferecer ao LMS uma interface pré-definida para serviços de um mesmo domínio de aplicação, evitando modificações na visualização do serviço;
- Encapsular informações relativas aos *web services* tornando transparente ao LMS as chamadas remotas e conversão de dados;
- Possuir modelagem de fácil extensão para a inclusão de novos *web services*;
- Permitir a permutação de *web services* de um mesmo domínio nos cursos do LMS;
- Oferecer interface para buscas em servidores UDDI.

5.1. Domínios de Serviços

Com o objetivo de viabilizar a abstração de tarefas para as chamadas remotas e permitir a flexibilidade de serviços minimizando o impacto na camada de visualização, é proposto o conceito de domínio de serviços e a sua implantação no LMS. Essa forma de agrupamento e categorização dos serviços permite que a

comunicação do LMS com os serviços ocorra de forma transparente para o sistema e seja realizada por intermédio do *framework*. Nesse sentido, a especificação dos domínios de serviços encontra-se diretamente relacionada com a modelagem de uma interface de comunicação (API) para cada domínio. É com base nessa API pré-definida que o LMS realizará toda a comunicação com os serviços de um determinado domínio. Dessa forma, para o LMS torna-se indiferente se o *learning service* acessado é um serviço legado do LMS ou um novo *web service* integrado ao ambiente.

A interface de comunicação sugerida para cada domínio de serviço deve contemplar todas as funcionalidades dos *learning services* agrupados no domínio e as informações relevantes que permitam a camada de visualização do LMS realizar o controle da disponibilização de tais funcionalidades para os usuários.

Para o controle da camada de visualização do LMS, são sugeridos na API métodos de verificação de disponibilidade apenas para as funcionalidades específicas dos variados serviços, pois as funcionalidades comuns a todos os serviços do domínio não necessitam de controle pela camada de visualização.

5.2. Modelagem

A seguir, são apresentadas informações básicas de modelagem do *framework* proposto, incluindo diagrama de casos de uso, diagrama de classes e outros.

O *framework* foi idealizado para sistemas de *e-Learning* que não possuem suporte a serviços remotos para a configuração do ambiente de aprendizado do participante. Com isso, o LMS será integrado ao *framework* provendo-lhe uma interface de comunicação e permitindo que seja transparente ao sistema qual serviço está sendo invocado para os variados domínios de serviços configurados.

Os pontos de flexibilização se referem ao LMS, aos *web services* dos fornecedores de serviços e a interface de comunicação utilizada nas chamadas remotas. Dessa forma, o domínio de aplicação do *framework* é o de sistemas *e-Learning* e permite a integração desses com *learning services* de variados domínios de serviços através de interfaces de comunicação não padronizadas.

Abaixo, a figura 10 apresenta um diagrama resumido de casos de uso para explicitar as principais funcionalidades básicas do *framework*.

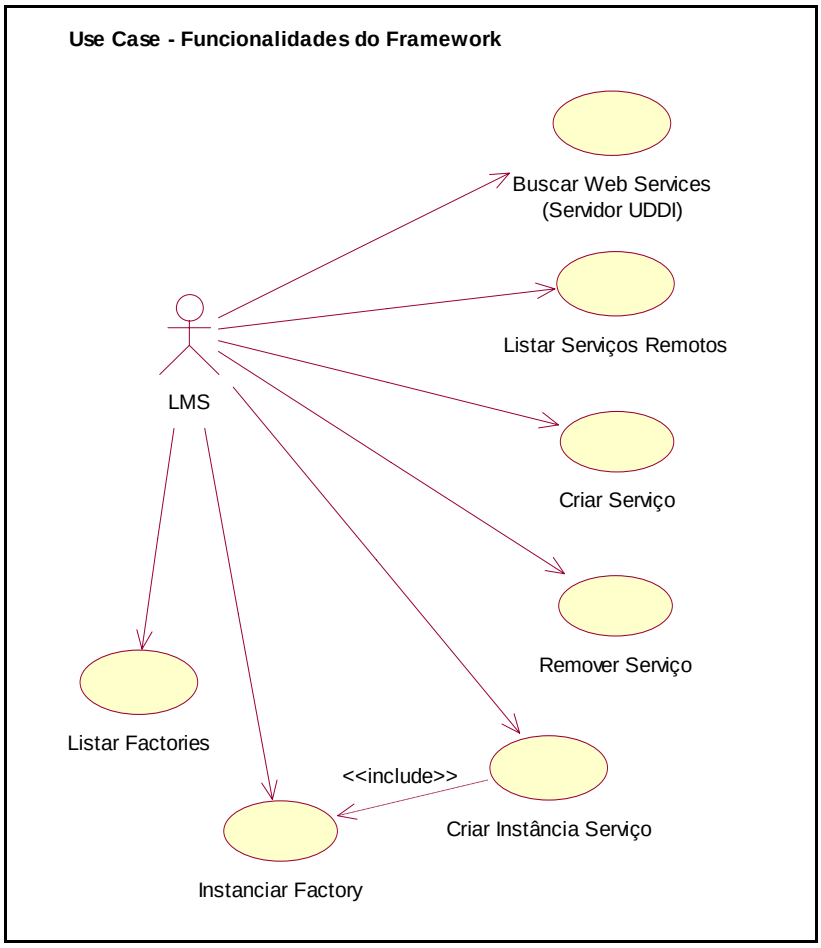

Figura 10 – Diagrama com algumas funcionalidades do *framework*.

O *framework* atua na área de administração do LMS, provendo funcionalidades que permitem a configuração de serviços remotos, e nos locais de acesso aos serviços, onde será executada a lógica de decisão para as chamadas aos serviços. O caso de uso “Listar Serviços Remotos” permite a visualização dos *web services* já configurados e integrados no LMS. A modelagem de persistência para um serviço remoto contempla as informações básicas que permitem a sua invocação pelo próprio LMS, resumidamente, a informação da classe que encapsula a implementação do serviço.

Os casos de uso “Listar Factories” e “Instanciar Factory” representam as funcionalidades de permitir que o LMS obtenha a listagem das *factories* disponíveis e realize a instanciação de uma determinada *factory* para realizar a criação de objetos que permitem o acesso aos serviços de um mesmo domínio. Neste caso, cada domínio de serviço possui uma *factory* para o gerenciamento da

instanciação de objetos. Os casos de uso “Criar Serviço” e “Remover Serviço” representam a tarefa do LMS de realizar o cadastramento e configuração dos serviços remotos.

A seguir, a figura 11 apresenta um diagrama de classe com a modelagem para o gerenciamento dos serviços e suas *factories*.

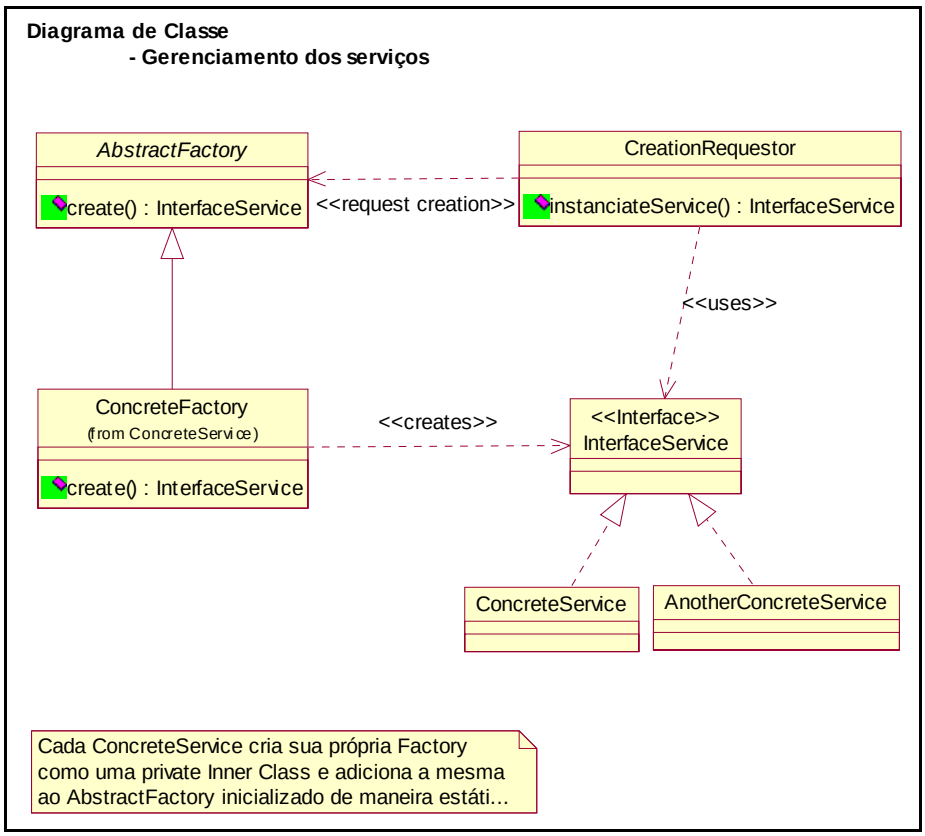

Figura 11 – Diagrama de classe do *framework* – Gerenciamento dos serviços.

A modelagem adotada para o gerenciamento dos serviços é o padrão *Polymorphic factories* [Eckel, 2003]. Esta abordagem difere do padrão *factory* tradicional com relação ao local onde o método de criação da *factory* se encontra. Neste caso, o método de criação está localizado em cada uma das várias sub-classes que implementam a interface da entidade modelada. É interessante notar que com esse padrão, os serviços específicos serão dinamicamente carregados sob demanda. Uma das grandes vantagens dessa modelagem é a fácil extensão do código com mínimas modificações no código original.

Na figura 12, encontra-se um diagrama de seqüência com as principais interações entre as classes de gerenciamento das *factories*. Neste diagrama, pode-

se notar o carregamento dinâmico da classe *ConcreteService*, que encapsula os detalhes de uma implementação específica de um determinado serviço.

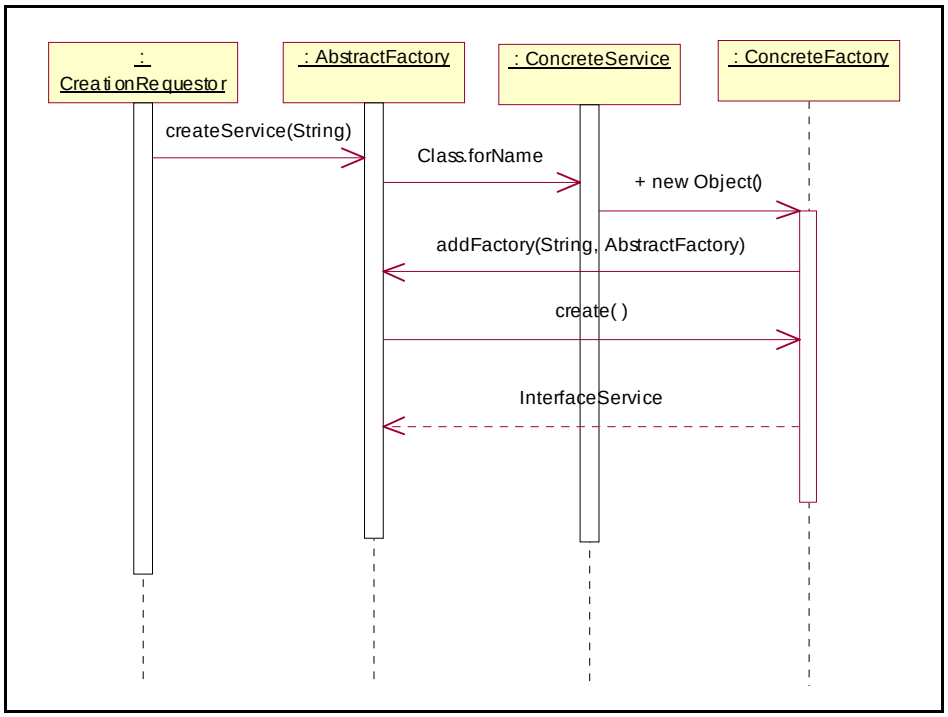

Figura 12 – Diagrama de seqüência do *framework* para a criação de uma factory de Serviços.

Essa modelagem necessita que o LMS possibilite o agrupamento dos *learning services* com base em seus domínios de aplicação, permitindo que o acesso a cada grupo de serviços seja realizado através de uma interface pré-definida, garantindo a abstração de qual serviço está sendo utilizado. Esta modelagem incentiva a componentização do sistema e a sua flexibilidade em permutar entre diferentes implementações de um serviço dentro de um domínio específico. Realizado o agrupamento dos serviços e a especificação das interfaces de comunicação, basta a criação das classes “ConcreteService” para encapsular a implementação particular de cada serviço.

Caso seja necessária qualquer mediação durante a comunicação entre o LMS e um determinado serviço, esta tarefa será transparente para o LMS e ficará contida especificamente na classe do serviço.

Uma vantagem obtida com as características dessa modelagem é permitir que a interface de visualização do serviço no LMS seja sobrecarregada para diversas implementações do mesmo domínio de serviços. Ao mesmo tempo, a interface de visualização pode ser considerada como genérica quando analisada

sob o ponto de vista das funcionalidades básicas dos serviços de um mesmo domínio. Com relação às características proprietárias de cada fornecedor, a modelagem proposta permite que a camada de visualização possa extrair informações da interface pré-definida do domínio do serviço para decidir se deve disponibilizar tal função para o usuário.

Uma conseqüente vantagem desse cenário é a possibilidade de um LMS utilizar um determinado serviço que contenha várias funcionalidades ainda não implementadas pela camada de visualização e realizar a adaptação gradualmente, sem interferir na compatibilidade com serviços de outros fornecedores.

Dependendo das características da implementação dos serviços incorporados ao LMS, pode ser necessário que o *framework* realize uma mediação mais complexa para garantir a abstração para o LMS. Neste caso, para o gerenciamento da persistência de dados do *framework*, este incorpora uma API específica para sistemas de gerenciamento de banco de dados relacionais. Na figura 13, pode-se visualizar a existência das classes *ConnectionManager* e *ConnectionPool* que gerenciam a criação e instanciação dos *pools* de conexão e das conexões propriamente ditas. As outras classes são auxiliares com relação à manipulação dos dados a serem armazenados e a flexibilidade do SGBD.

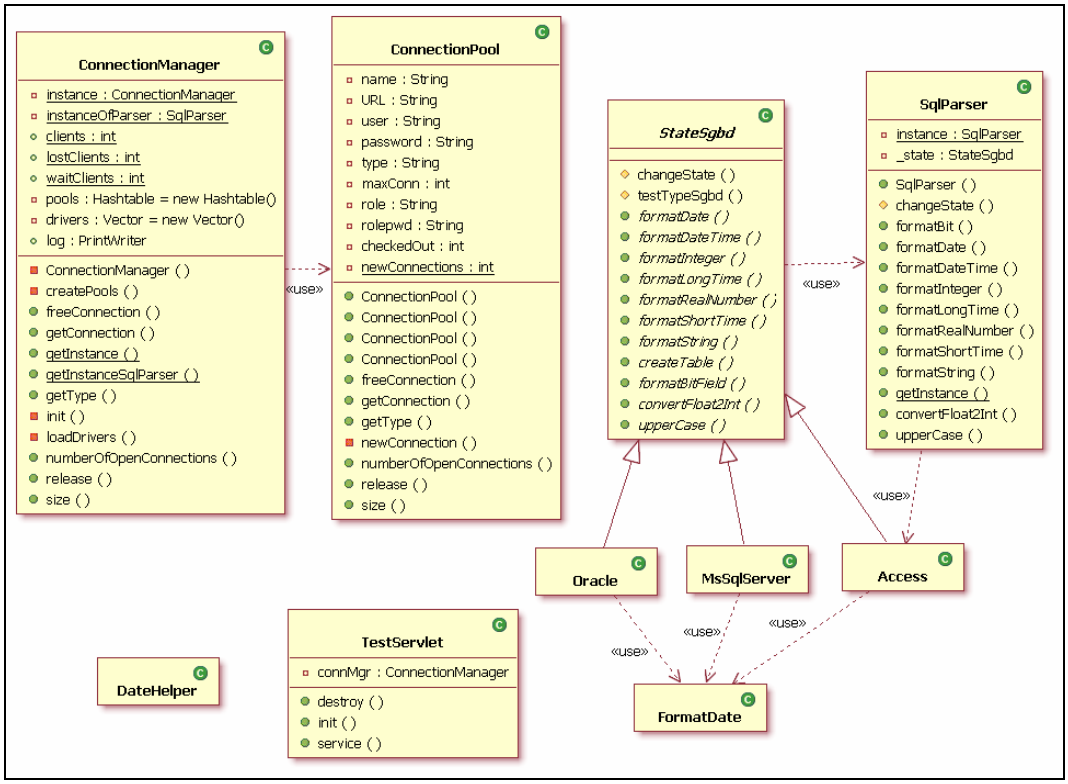

Figura 13 – Diagrama de classe do *framework* com relação a API de persistência dos dados.

Para facilitar a busca de novos *web services* no LMS, o *framework* possui uma API para facilitar a integração com diferentes componentes de busca em servidores UDDI. Neste caso, o *framework* está integrado com a API da IBM. A busca UDDI ocorre no método "*findServiceImplementations*()" declarado na interface *UDDIRegistryInterface*, que retorna em um vetor contendo as *URLs* (pontos de acesso) de todos os serviços para os respectivos parâmetros passados para o método - "*service provider*" e "*service type*" (nome originado do atributo *targetNamespace* na interface wsdl). Foi utilizado o padrão *Factory* devido a possibilidade de existirem diferentes APIs de comunicação com servidores UDDI. Como dito anteriomente, foi implementada a classe *IBM_UddiAccessor*, que implementa a interface *UDDIRegistryInterface* para o acesso a servidores UDDI da IBM. A modelagem das classes responsáveis pela busca em servidores UDDI pode ser visualizada na figura 14.

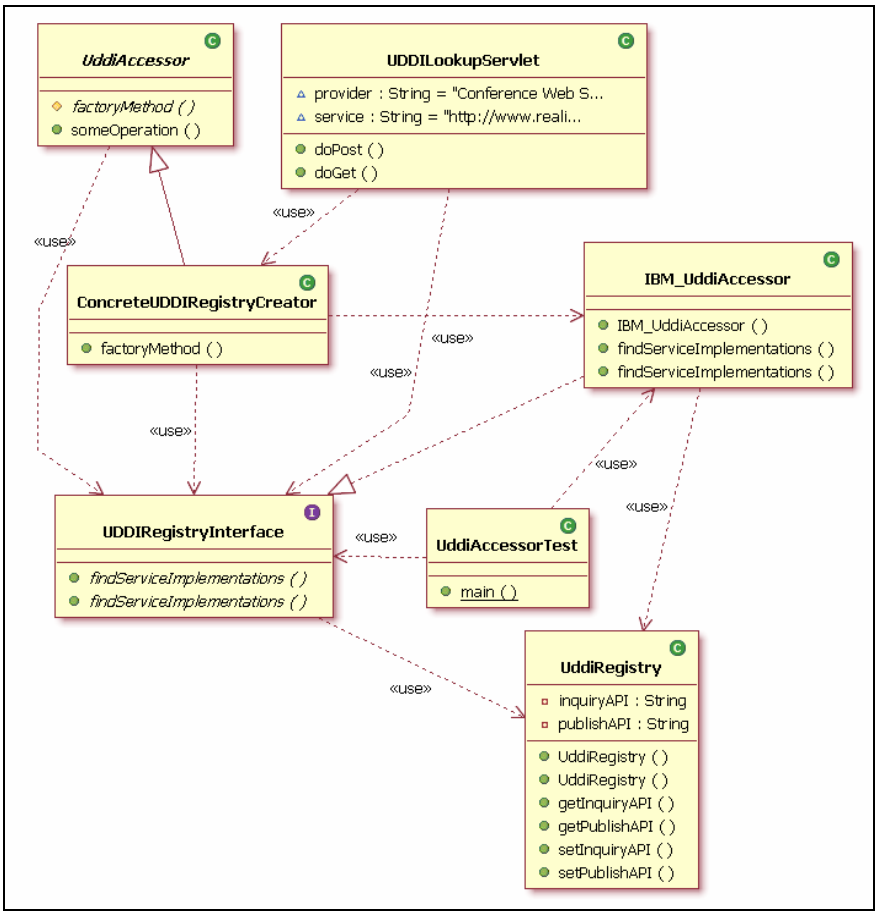

Figura 14 – Diagrama de classe do *framework* para a API de comunicação com servidores UDDI.

5.3. Modificações no LMS

A adequação do LMS aos learning services através do *framework* pode exigir, ainda assim, um relativo esforço. Mesmo o *framework* abstraindo várias tarefas para o LMS, este continuará gerenciando os *learning services* cadastrados no sistema, independente de corresponderem a serviços legados ou novos *web services*. Eventualmente, esse gerenciamento pode gerar o primeiro esforço de implementação no sistema de *e-Learning*. Neste sentido, o LMS precisa conceber a noção de agrupamento de *learning services* nos chamados domínios de serviços, permitindo que o LMS estabeleça uma comunicação com o *framework* apenas com base nesse conceito mais abstrato. O objetivo é que o LMS redirecione tarefas para o *framework* através da classe responsável por um determinado domínio, que encapsula a lógica de decisão sobre qual serviço especifico deve ser utilizado.

Ainda analisando a comunicação do LMS com o *framework*, a camada de visualização dos serviços é a parte do LMS mais afetada com relação a flexibilidade dos *learning services*. Isso ocorre porque a interface gráfica disponibilizada pelo LMS para um determinado serviço será sobrecarregada para o domínio ao qual esse serviço é classificado. O objetivo é utilizar o conceito de domínios de serviços e permitir que o LMS tenha uma camada de visualização única e robusta para todos os *learning services* que compõem determinado domínio. Dessa forma, dois fatores devem ser observados durante o *refactoring* da camada de visualização: a nova interface de comunicação genérica para o domínio do serviço e as funcionalidades específicas de determinados *learning services*.

O primeiro ponto refere-se a característica da nova modelagem proposta pelo *framework*, onde cada domínio de serviços no LMS possui uma interface de comunicação pré-definida, permitindo que a camada de visualização realize qualquer comunicação através dessa interface, abstraindo as peculiaridades de cada *learning service*.

O segundo ponto está relacionado às inevitáveis diferenças entre as implementações dos serviços, onde alguns possuem funcionalidades específicas e exclusivas. Neste caso, a camada de visualização do LMS poderá considerar essas

funcionalidades exclusivas, mas precisará receber a informação se deve ou não disponibilizar tal funcionalidade para os usuários. Essa informação também será disponibilizada para o LMS através da interface de comunicação do domínio ao qual o serviço está incluído.