UNIVERSIDADE FEDERAL DA PARAÍBA
Centro de Ciências e Tecnologia
Coordenação de Pós-Graduação em Informática

DISSERTAÇÃO DE MESTRADO

Projeto EaD:
Uma Ferramenta para Auxiliar o Projeto de Cursos a Distância Suportados pela CMC

Rodrigo Bonifácio de Almeida

Campina Grande -PB
2001

Rodrigo Bonifácio de Almeida

Projeto EaD:
Uma Ferramenta para Auxiliar o Projeto de Cursos a Distância Suportados pela CMC

Dissertação apresentada ao curso de Mestrado em Informática da Universidade Federal da Paraíba, em cumprimento às exigências para obtenção do grau de Mestre.

Área de Concentração: Ciência da Computação

Orientadores: Francisco Vilar Brasileiro
Marcelo Alves de Barros

Campina Grande - PB
2001

ALMEIDA, Rodrigo Bonifácio de

A447P

Projeto EaD: Uma Ferramenta para Auxiliar o Projeto de Cursos a Distância Suportados pela CMC.

Dissertação (Mestrado), Universidade Federal da Paraíba, Centro de Ciências e Tecnologia, Coordenação de Pós-Graduação em Informática, Campina Grande - Pb, Agosto de 2001.

86 p. Il.

Orientadores: Dr. Francisco Vilar Brasileiro
Dr. Marcelo Alves de Barros

Palavras-Chave:

1. Redes de Computadores
2. Ferramenta de Auxílio ao Projeto
3. Ensino à Distância
4. Comunicação Mediada por Computador

CDU - 621.391

Este trabalho é dedicado aos meus pais,
a minha irmã e a Suzana Simões Ramos.
O incentivo de vocês foi fundamental para
a realização deste trabalho.

Agradecimentos

Participar de um programa de mestrado não é fácil. Muitas vezes você tem que abrir mão da convivência com a sua família; mudar de residência por um certo período; se manter com uma bolsa que mal dá para cobrir os seus gastos com moradia e alimentação; e perceber o quão está sendo heróica a tarefa de “fazer ciência” no Brasil. Apesar de todas essas dificuldades, só tenho que agradecer ao Departamento de Sistemas e Computação por ter me dado essa oportunidade. Os conhecimentos adquiridos durante esse período me fizeram crescer muito. Nunca aprendi tanto em tão pouco tempo. Em especial, gostaria de expressar meu agradecimento às seguintes pessoas:

A Francisco Vilar Brasileiro (Fubica), meu orientador, que aceitou participar deste projeto e apenas com a sua confiança e motivação foi possível realizar este trabalho. Creio que se não fosse por ele, esta dissertação não teria sido concluída.

A Marcos Sampaio e Jacques Sauvé, pelo nível das aulas apresentadas em suas disciplinas. Tenho um grande respeito e admiração pelos mesmos.

A Aninha, Verônica, Zeneide e demais funcionários do Departamento de Sistemas e Computação (UFPB) pela amizade, carinho e atendimentos recebidos.

Eduardo, Alex e Antônio que, apesar das discussões (normais quando se divide um apartamento), tornaram a minha estadia em Campina Grande a mais agradável possível.

A Hilmer, Alberto, André, Edeyson, Tarig e demais colegas do curso. Espero que nós nos encontremos em breve (em Brasília, em Sergipe ou, quem sabe, na Turquia).

A Sávio, Patrocínia e Deise pela motivação e por sempre segurarem a barra nos momentos que me ausentei do trabalho para cumprir alguma atividade desta dissertação.

A Márcio (Gandaia), Rodrigo (Scheyman) e Valdeci (vulgo Valdécia) que suportaram minhas alternâncias de humor durante esse último ano em Recife.

A todos meus amigos da cidade maravilhosa (João Pessoa). Mesmo morando tão próximo fica difícil não sentir saudades. Espero um dia poder voltar a morar lá.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Quadros

Resumo

O ensino a distância vem sendo utilizado há várias décadas com o intuito de prover formação às pessoas que, por algum motivo (tempo disponível, distância dos centros acadêmicos, limitações físicas, etc.), não têm acesso aos meios convencionais de educação. Durante esse período, meios como o sistema postal, canais de televisão/rádio e tapes audiovisuais foram utilizados para efetivar esta modalidade de ensino. Mesmo ainda sendo utilizados, esses recursos dão maior ênfase na disponibilização do material didático a ser lecionado, deixando de lado um aspecto fundamental no processo de ensino-aprendizagem: A interação entre os participantes dos cursos. Com a queda nos preços dos computadores e a expansão da Internet, as características de interação, abrangência e facilidade de manutenção oferecida pela Comunicação Mediada por Computador (CMC) passaram a ser exploradas na educação a distância.

A CMC suporta um conjunto de recursos tecnológicos (videoconferência, hipertexto, correio eletrônico, etc.) que podem ser utilizados para disponibilizar os componentes pedagógicos de uma lição. Cada um desses recursos tem características próprias (sincronismo, capacidade de interação, facilidade de uso, etc.), que os tornam mais indicados para determinadas atividades de ensino. Dessa forma, um educador, durante a fase de projeto de um curso suportado pela CMC, além de se preocupar com o planejamento pedagógico, tem que avaliar quais recursos tecnológicos devem ser utilizados. Pensando nisso, essa dissertação apresenta o projeto e a implementação de uma aplicação para a Web (*Projeto EaD*) cujo objetivo é auxiliar os educadores, não especialistas em informática, a selecionar os componentes pedagógicos/tecnológicos utilizados nas lições de um curso a distância suportado pela CMC.

Abstract

Distance learning has been used during many decades in order to provide knowledge to people that, for some reason (avaliable time, distance of academic centers, physics limitations, etc.) do not have access to conventional ways of education. During this period, medias like mail system, television/radio channels and audiovisual tapes have been used to provide this kind of education. Although they still are in use, these resources emphasize the avaliability of the didactic media to be teached, despising a fundamental aspect of the education process: The interaction among the class participants. With the falling of computer prices and expansion of the Internet, features like interaction, widespreadness and easily maintance offered by Computer Media Communication (CMC) start to be explored in distance learnig.

The CMC supports a set of technological resources (videoconference, hypertext, eletronic mail, etc.) that can be used to provide pedagogical components of a lesson. Each of those resources has its own features (synchronism, interaction capabilities, easy of use, etc.) tha turn it more appropriate to be used in a given kind of teaching activity. In this way, an educator, during the project phase of a course supported by CMC, should not only worry about pedagogical planning, but what kind of technological resource should be used. Keeping that in mind, this work presents the project and implementation of a Web applicatoin (*Projeto EaD*) with the aim of helping the educators, informatics non-experts, to select the pedagogical/technological components used in the lessons of a distance learnig course supported by CMC.

1 INTRODUÇÃO

A partir dos anos sessenta, a educação a distância (EaD) começou a distinguir-se como uma modalidade não convencional de ensino capaz de atender, com grande perspectiva de eficiência, eficácia e qualidade, aos anseios de universalização do ensino; bem como um meio apropriado à permanente atualização dos conhecimentos, gerados de forma cada vez mais intensa pela ciência e cultura humana. Inicialmente associado ao estudo por correspondência, o ensino a distância surgiu como uma alternativa para o aumento do nível cultural das populações, facilitando o acesso à educação àquelas pessoas que, por algum motivo, não possuíam condições de obter instrução pelos métodos convencionais.

Os avanços na EaD sempre estiveram relacionados à popularização dos meios de comunicação utilizados para sua efetivação. Até a década de sessenta, os cursos por correspondência eram a única modalidade de ensino a distância disponível, sendo caracterizada por uma interação mínima entre alunos/professores e realizada de forma lenta e ineficiente através do sistema postal tradicional. O passo seguinte foi a utilização do rádio e da televisão como ambientes de EaD. Apesar desses meios disponibilizarem as informações diretamente nas residências dos alunos, aumentando a diversidade de material e facilitando o acesso aos mesmos, a comunicação continuava essencialmente unidirecional. Com isso, as alternativas para efetivação do EaD focavam apenas a disponibilização do material didático a ser lecionado, deixando de lado um aspecto essencial no processo de ensino-aprendizagem: a interação entre os participantes dos cursos.

Com a queda no preço dos ambientes computacionais e a expansão das redes de computadores (mais particularmente da *Internet*), surgiu o conceito de Comunicação Mediada por Computador (CMC – *Computer Mediated Communication*). Segundo [OTSUKA 97], CMC é qualquer sistema capaz de apresentar e/ou transportar informações no sentido computador-pessoa ou pessoa-pessoa através de computadores. Características como abrangência, flexibilidade e interatividade tornaram a CMC uma alternativa promissora para o suporte de cursos a distância.

1.1 Educação a Distância

O Projeto de Ensino a Distância da Califórnia (http://www.otan.dni.us/cdlp/cdlp.html) disponibiliza uma série de definições sobre EaD; quase todas se referindo à disponibilização de recursos pedagógicos para estudantes remotos e abrangendo tanto o *ensino a distância* (o papel do professor no processo de educação) quanto a aprendizagem a distância (o papel dos estudantes)[1]. O projeto propõe os seguintes elementos-chave para caracterizar o ensino a distância:

- A separação, tanto espacial quanto temporal, do educador e do estudante durante a maior parte das atividades instrucionais;
- A utilização de meios de comunicação para disponibilizar o conteúdo pedagógico dos cursos e garantir a interação entre os estudantes e os educadores;
- O processo de ensino centrado no aluno, ou seja, o estudante é quem escolhe o ritmo de aprendizagem e a estratégia pedagógica que lhe for mais conveniente.

A utilização da EaD se apresenta como uma alternativa promissora para a universalização do ensino, diminuindo restrições de tempo e espaço para as pessoas que almejam ter acesso ao conhecimento. Vale ressaltar que a palavra “acesso” no ensino a distância abrange o acesso à instrução no momento que for mais conveniente para os estudantes; o acesso a especialistas em locais onde os mesmos não estão disponíveis; o acesso a um material didático de qualidade, etc.

Para que a adoção do ensino a distância seja efetiva, faz-se necessária a criação de uma política de incentivos para esta modalidade de ensino, com o objetivo de integrar órgãos públicos e privados para que o ensino a distância esteja ao alcance de qualquer camada social.

[1] Sempre que for conveniente, este documento tratará *educação a distância* e *ensino a distância* como sinônimos

1.2 A Utilização da CMC no Ensino a Distância

A maior vantagem de se utilizar a Comunicação Mediada por Computador como meio de suporte à EaD é a alta capacidade de interação presente nas tecnologias de CMC. Essas tecnologias podem ser classificadas de acordo com o sincronismo (síncrona/assíncrona) e a mídia utilizada, variando das tecnologias baseadas em texto plano às tecnologias que combinam áudio, gráficos e vídeo.

As tecnologias de CMC podem ser utilizadas para disponibilizar diferentes atividades pedagógicas de um curso a distância. Dependendo de um conjunto de fatores, como, por exemplo, os recursos presentes na instituição, o canal de comunicação utilizado para disponibilizar os cursos e a forma como os participantes estão geograficamente dispersos, diferentes alternativas de projeto podem ser apresentadas para a implantação de um curso a distância.

Além da maior capacidade de interação, a utilização da CMC para disponibilizar um curso a distância apresenta as seguintes vantagens:

- *Acessibilidade→ A acessibilidade é a principal vantagem de se oferecer cursos a distância suportados pela CMC. O termo acessibilidade engloba o acesso a cursos e instrutores qualificados em regiões onde os mesmos não estão disponíveis; acesso a cursos proibitivos por razões políticas, orçamentárias, etc; acesso à instrução a qualquer momento; e o acesso a informações remotas.*
- *Flexibilidade quanto ao processo de aprendizagem→ Cursos a distância suportados pela CMC apresentam flexibilidade quanto ao horário de estudo e quanto à abordagem utilizada para o ensino. Como o conteúdo de um curso pode ser estruturado e disponibilizado em diversas mídias, um estudante pode optar pela que melhor se adeque ao seu estilo de aprendizagem e/ou às suas limitações físicas .*
- *Facilidade de manutenção dos cursos→ Como a maior parte do conteúdo didático de um curso a distância suportado pela CMC fica centralizada em servidores, o esforço necessário para sua atualização é bem inferior ao presente nos outros meios de suporte ao ensino a distância.*
- *Globalização dos cursos→ A capacidade dos estudantes estarem interagindo com outros alunos geograficamente dispersos, em um curso a distância suportado pela*

CMC, tem uma importância muito grande para a troca de experiências (culturais, políticas, etc.) e formação de pessoas mais adaptadas a um mundo globalizado.

Apesar da alternativa de se utilizar a CMC como meio de suporte ao ensino a distância ser bastante promissora, existem as seguintes desvantagens que precisam ser consideradas:

- *Custo de Desenvolvimento/Implantação/Acesso→ Um fator primário que pode inviabilizar a implantação de um curso a distância suportada pela CMC são os elevados custos em termos de horas trabalhadas e aquisição de ferramentas/equipamentos. Mais ainda, o acesso a cursos suportados por essas tecnologias exige a utilização de computadores e canais de comunicação (geralmente linhas telefônicas). Como um dos objetivos do EaD é permitir que pessoas menos favorecidas tenham acesso ao conhecimento, é necessário que o governo estabeleça incentivos para disponibilizar esse serviço nas comunidades carentes (*[MORLEY 2000] *apresenta uma discussão sobre esse tema).*
- *O Tempo de Acesso→ O tempo de resposta de uma solicitação feita pela CMC é bastante dependente da tecnologia de transmissão que está sendo utilizada. Com isso, ambientes que não possuem uma infra-estrutura de comunicação adequada podem comprometer a motivação do aluno em relação a um curso suportado pela CMC.*
- *Gerenciamento → Existem uma série de tarefas que devem ser executadas para manter o funcionamento de um curso a distância suportado pela CMC. Entre estas tarefas, podemos citar a implantação de restrições de acesso ao conteúdo dos cursos, a atualização do material didático disponibilizado e a manutenção das bases de dados. Isto requer que seja contratada (ou terceirizada) uma equipe responsável pelo gerenciamento dos serviços do sistema de apoio ao EaD.*

1.3 Motivação

O projeto de um curso a distância suportado pela CMC envolve a análise de componentes sócio-psicológicos (perfil, receptividade com a tecnologia de CMC, etc.) dos participantes, componentes pedagógicos (objetivos do curso, atividades pedagógicas, certificação, etc.) e componentes tecnológicos (canais de comunicação, escolha das tecnologias para efetivar as atividades pedagógicas do curso, etc) .

Devido à pouca familiaridade de grande parte dos educadores (excetuando os das áreas técnicas) com os recursos tecnológicos suportados pela CMC, vários programas de EaD não tiram proveito de toda a capacidade de interação que o ambiente oferece. Com base nisso, essa dissertação apresenta o projeto e a implementação do *Projeto EaD*, que é uma ferramenta para auxiliar os educadores a projetar um curso a distância suportado pela CMC, levando em consideração os componentes pedagógico e tecnológicos inerentes ao domínio do problema.

O *Projeto EaD* está inserido em uma fase intermediária do ciclo de desenvolvimento de um curso a distância suportado pela CMC. Como seu objetivo é auxiliar a tomada de decisão sobre quais atividades pedagógicas/recursos tecnológicos devem ser utilizados para a disponibilização de uma lição, faz-se necessário que o pedagogo tenha um conhecimento prévio sobre a missão do curso, as lições necessárias para atingir os objetivos, o perfil dos alunos, etc. Após a utilização do *Projeto EaD*, sugere-se que sejam utilizadas ferramentas de autoria, para a criação dos conteúdos didáticos, e ferramenta para a construção de ambientes de aprendizagem como o *e-Group* [MENDES 2000] ou o *AulaNet* (http://www.les.inf.puc-rio.br/aulanet), que facilitam a disponibilização e administração desses conteúdos para os estudantes terem acesso.

Com a utilização de um ambiente integrado para facilitar a especificação de um curso a distância suportado pela CMC, os educadores podem tornar mais efetivo o processo de ensino-aprendizagem (uma vez que os mesmos vão poder avaliar diferentes tecnologias para a disponibilização das atividades pedagógicas) e melhorar a comunicação com a equipe técnica das instituições de ensino.

1.4 Organização do Documento

O objetivo desta dissertação é o projeto e a implementação de um ambiente para auxiliar os educadores, não especialistas em informática, a projetar um curso de EaD suportado pela CMC. Como discutido anteriormente, tanto os aspectos pedagógicos quanto os aspectos tecnológicos devem ser levados em consideração durante o projeto de um curso a distância, dificultando o trabalho dos educadores e produzindo resultados nem sempre satisfatórios.

O próximo capítulo apresenta a complexidade envolvida no projeto de um curso a distância, descrevendo os aspectos pedagógicos e tecnológicos que devem ser analisados. Como a CMC suporta uma gama de ferramentas (hipertexto, animações, vídeo conferência, etc.) que podem ser utilizadas para implementar uma atividade pedagógica (aula expositiva, aula de discussão, aula prática, etc.), esse capítulo discute as características a serem avaliadas na hora de se optar por uma determinada alternativa de projeto.

O capítulo 3 apresenta os esquemas de dados (nível conceitual e interno) que descrevem as relações entre as estratégias de ensino, os recursos pedagógicos e os recursos tecnológicos associados a um curso a distância suportado pela CMC. A facilidade na definição/alteração das características pertinentes aos componentes pedagógicos e tecnológicos de um curso suportado pela CMC e a portabilidade dos dados entre diferentes plataformas são dois requisitos importantes da solução que está sendo proposta. Com base nisto, foi feita a opção pela tecnologia *eXtensible Markup Language* (XML) para instanciar o esquema conceitual proposto.

O Capitulo 4, por sua vez, descreve as atividades e os artefatos construídos (visão do sistema, diagramas de casos de uso, diagramas de classes, diagramas de pacotes, classes Java, etc.) durante o desenvolvimento do Projeto EaD. Como processo de desenvolvimento, foi seguida a proposta de adequação do Processo Unificado (Apêndice A) para Pequenos Projetos apresentada em [POLLICE 2000].

No último capítulo são apresentadas as conclusões deste trabalho, enfatizando as contribuições deste estudo e as propostas para pesquisas futuras, que poderão estender os resultados desta dissertação.

Bibliografia

[MENDES 2000] MENDES, F., "*E-group: Um Ambiente para Suporte à Aprendizagem Colaborativa Baseada na Web*", Dissertação de Mestrado, UFPB/DSC, 2000.

[MORLEY 2000] Morley, J., "*Falling Through the Web, Inequality to access in Distance Education*", Education at Distance Magazine, 01-2000. **on-line:** http://www.usdla.org/15_publications.htm

[OTSUKA 97] OTSUKA, J., "*Fatores Determinantes na Efetividade de Ferramentas de Comunicação Mediada Por Computador no Ensino à Distância***"**, Trabalho Individual, Pós-Graduação em Ciência da Computação, Instituto de Informática, UFRGS, 1997.

[POLLICE 2000] POLLICE, G., Using The Rational Unified Process for Small Projects: Expanding Upon eXtreme Programming, Rational Software White Paper, 2000 **on-line**: http://www.rational.com/whitepapers

2 Projetos de Cursos a Distância Suportados pela CMC

O projeto de cursos a distância suportados pela CMC exige, por parte dos educadores, um esforço considerável para a integração de componentes bastante heterogêneos. A seleção da estratégia de ensino para a disponibilização de um curso, a escolha das tecnologias de CMC a serem utilizadas e o levantamento do perfil dos alunos a que o curso se destina são apenas alguns dos componentes que devem ser levados em consideração durante o projeto de um curso a distância suportado pela CMC. Ou seja, durante a especificação de um curso a distância suportado pela CMC, faz-se necessário a análise de aspectos pedagógicos, aspectos tecnológicos e aspectos de natureza social, legal e psicológica.

A primeira sessão deste capítulo apresenta considerações sobre os aspectos pedagógicos do projeto de um curso a distância suportado pela CMC. Inicialmente são descritos os papéis que os educadores e os estudantes devem exercer para que as atividades pedagógicas tenham êxito. Em seguida, são apresentadas algumas considerações relacionadas às estratégias de ensino de um curso a distância, que devem focar a educação colaborativa, incentivando o aluno a contribuir no processo de educação.

A segunda sessão desse capítulo apresenta os aspectos tecnológicos do projeto de um curso a distância suportado pela CMC. Nessa sessão é discutida a importância do planejamento tecnológico para tornar o ambiente de aprendizagem mais efetivo. Em relação às tecnologias que podem ser usadas para o suporte das atividades pedagógicas, são descritas suas características em relação ao sincronismo, mídia utilizada, custos para a instituição/estudantes e capacidade de interação; e apresentada uma proposta para classificá-las.

Aspectos que não têm natureza pedagógica nem tecnológica são apresentados na última sessão desse capítulo. Essas questões devem ser levadas em consideração durante a especificação de um curso a distância suportado pela CMC para garantir a autenticação dos participantes, integridade e segurança do material utilizado e fazer com que o uso da tecnologia no processo de ensino não comprometa psicologicamente os estudantes.

2.1 Aspectos Pedagógicos de um Curso a Distância Suportado pela CMC

Os avanços nas áreas de computação e telecomunicações e o uso cada vez maior de recursos tecnológicos em ambientes de ensino exigem que novas abordagens no processo de ensino/aprendizagem sejam empregadas. Especificamente para ambientes de educação a distância suportados pela CMC, as seguintes questões pedagógicas podem ser formuladas:

- *Qual o papel dos educadores?*
- *Qual o papel dos estudantes?*
- *Quais estratégias de ensino podem ser utilizadas?*
- *Quais formas de avaliação podem ser empregadas?*

As seguintes subseções trazem considerações sobre cada uma das questões acima.

2.1.1 O papel dos educadores

O educador assume o papel de tutor em ambientes de educação a distância, deixando de ser o *facilitador do conhecimento* para ser o *facilitador do ambiente de aprendizagem* [CASEY 98]. São de responsabilidade do tutor o apoio, a motivação e a avaliação da progressão dos alunos; a elaboração da documentação que os estudantes têm acesso e a aplicação dos exercícios e trabalhos necessários para o desenvolvimento do espírito crítico e da capacidade de raciocínio dos alunos [MENDES 2000]. É essencial que o tutor possua o embasamento pedagógico que o torne capaz de identificar quais são os objetivos esperados com a realização de um determinado curso.

Durante o projeto de um curso a distância suportado pela CMC, é necessário que o tutor faça um levantamento prévio sobre o perfil do público alvo a que o curso se destina, verifique quais são os resultados esperados pelos alunos e avalie as possibilidades da sua certificação junto a algum órgão competente. Se a instituição já oferecer um curso em outra modalidade, seja esta presencial ou não, deve-se manter o mesmo padrão de qualidade quando o curso for disponibilizado da forma *on-line*.

As tecnologias de CMC permitem que o educador utilize diferentes metodologias para o ensino/acompanhamento dos alunos em ambientes de educação a distância. Com a utilização de diferentes mídias (vídeo, som, texto, etc.) em um mesmo ambiente, o tutor

pode cobrir diferentes estilos de aprendizagem, fazendo com que o aluno utilize àquela que melhor lhe convir.

2.1.2 O papel dos estudantes

A CMC permite que os estudantes desenvolvam as atividades tanto sozinhos quanto em grupo (aprendizagem colaborativa). O processo de aprendizagem no EaD é centrado no aluno, pois, ao contrário do que acontece no ensino presencial, onde o aluno tem que acompanhar o ritmo de aprendizagem dos colegas, em ambientes de educação à distância o aluno tem sua vida facilitada, podendo aprender ao seu próprio ritmo de estudo.

O grau de interação suportado por algumas tecnologias de CMC permite que os estudantes controlem a seleção dos tópicos e enviem conclusões/contribuições sobre a temática que está sendo apresentada, tornando-os responsáveis pela própria aquisição/geração dos conhecimentos (ao invés de serem receptores passivos do processo de aprendizagem). O envolvimento com o processo de aprendizagem encoraja os alunos a fazerem reflexões pessoais sobre o material de um curso e o relacionar com conhecimentos adquiridos previamente. Isto está de acordo com a teoria construtivista, segundo a qual, a aprendizagem deve ser um processo ativo e não um processo de absorção [CARLOS 97].

Em um ambiente de ensino dirigido pelos estudantes, é essencial o uso simultâneo de diferentes estratégias de ensino. Os educadores devem utilizar estratégias que permitam aos estudantes selecionar uma alternativa de ensino que seja mais adequada ao seu estilo de aprendizagem.

2.1.3 Estratégias de ensino

A estratégia de ensino determina como devem ser desenvolvidas as lições para um determinado tipo de curso. Uma lição, por sua vez, é definida como uma unidade auto-suficiente de estudo que pode consistir de várias atividades pedagógicas (exposição, discussão, laboratório, etc.). Os cursos a distância suportados pela CMC podem ser compostos por várias lições; existindo um conjunto de tecnologias (hipertexto, correio eletrônico, videoconferência, etc) que podem ser utilizadas para a efetivação das atividades pedagógicas [MENDES 2000].

A seguir são apresentados alguns dos recursos que podem compor uma lição:

- Exposição: Recurso pedagógico que consiste na apresentação dos conteúdos didáticos aos alunos. Segundo [CARLOS 97], a aula expositiva constitui uma estratégia de ensino adequada quando os objetivos de ensino se referem aos níveis iniciais do domínio cognitivo.

- Discussão: Atividade que vem sendo cada vez mais praticada nas modalidades de ensino e que favorece a reflexão acerca dos conhecimentos adquiridos em leituras ou aulas expositivas; desenvolve novos conhecimentos mediante a utilização de conhecimentos e experiências anteriores; possibilita o enfoque de um assunto sobre diferentes pontos de vista.

- Laboratório ou Simulação: Recurso que permite aos alunos aplicarem os conhecimentos adquiridos em situações práticas ou reais. Estimular a reflexão sobre um determinado problema; auxiliar a fixação de conteúdos didáticos; realizar treinamentos; e desenvolver habilidades específicas são os principais objetivos das atividades de laboratório/simulação.

- Trabalhos Individuais: Atividade que permite ao estudante refletir sobre seu domínio em relação a um determinado assunto. Detectadas possíveis carências, os estudantes podem realizar leituras adicionais e/ou solicitar auxílio a monitores ou professores.

- Trabalhos em Grupo: Recurso utilizado para permitir que os alunos desenvolvam sua capacidade de trabalhar em equipe, possibilitando uma melhor reflexão sobre o conteúdo lecionado e desenvolvendo habilidades interpessoais que são necessárias em outras situações na vida dos sujeitos.

[CARLOS 97] apresenta um maior detalhamento sobre as atividades pedagógicas apresentadas acima, discutindo os pontos positivos e negativos sobre cada uma delas. É importante frisar que esses recursos podem ser combinados para a apresentação de uma lição, sendo a natureza do curso um fator determinante na escolha da estratégia de ensino.

Ou seja, quando um curso se apresenta essencialmente teórico, pode ser utilizada uma abordagem composta por uma aula expositiva, para difundir um conteúdo aos alunos, seguida por uma aula discursiva, para que os alunos reflitam sobre o assunto. Já em cursos que apresentam natureza prática, é essencial que atividades de simulação/laboratório sejam utilizadas.

Algumas considerações devem ser feitas antes de se utilizar a CMC como ferramenta de suporte ao ensino a distância. Primeiro, o ensino a distância suportado pela CMC deve ser encarado como um novo paradigma de ensino, pois será frustrada a tentativa de se adaptar esta modalidade de ensino aos métodos convencionais de educação. Isso ocorre porque nos métodos convencionais de educação, os estudantes não assumem nenhuma responsabilidade sobre o processo de ensino. Como os alunos em um curso a distância podem se dispersar facilmente, é imprescindível que sejam oferecidos meios para que eles contribuam com a geração de conhecimentos. Segundo, existem estratégias de ensino que, por exigirem maior grau de interação que os suportados pelas tecnologias disponíveis pela CMC, não devem ser utilizadas no desenvolvimento de cursos a distância baseados nesse ambiente. Estratégias de ensino que propõem, por exemplo, explanações dos educadores sobre algum material e o uso de meios para que os estudantes formulem questões/enviem contribuições, podem ser utilizados com sucesso em cursos a distância suportados pela CMC [CASEY 98].

2.1.4 Formas de avaliação

Um aspecto importante a ser levado em consideração em ambientes de ensino é a forma de avaliação utilizada nos cursos. [LUCKESY 95] traça um paralelo entre a "**avaliação da aprendizagem**", que busca alterar a estratégia de ensino sempre que for detectada alguma desmotivação/queda de rendimento dos estudantes; e a "**avaliação seletiva**", utilizada para verificar o desempenho do aluno em determinado conteúdo e classificá-lo como "aprovado" ou "reprovado". Como no ensino a distância suportado pela *CMC* os estudantes são os responsáveis pelo seu próprio ritmo de estudo, existindo pouca ou nenhuma interação face-a-face, deve ser dada uma prioridade ainda maior à avaliação da aprendizagem. [PALLOF et al 99] propõe que esta forma de avaliação seja feita durante todo o curso, tomando como base as práticas propostas e a quantidade/qualidade do material enviado pelos alunos durante as discussões.

2.2 Aspectos Tecnológicos de um Curso a Distância Suportado pela CMC

O planejamento tecnológico é outro aspecto importante no desenvolvimento de cursos a distância suportados pela CMC. É ele quem define a infraestrutura necessária para suportar os recursos pedagógicos relacionados a um determinado curso. [MENDES 2000] sugere que a mesma importância dada à fase de levantamento dos recursos pedagógicos deve ser dada à fase de especificação dos recursos tecnológicos.

Uma série de fatores deve ser levada em consideração para que as estratégias de ensino suportadas pela CMC sejam usadas de modo efetivo. A garantia de um serviço de qualidade (tempo de resposta aceitável, escalabilidade, facilidade de uso, etc.) é um fator determinante para o sucesso ou fracasso de um curso a distância suportado pela CMC. Não adianta a utilização dos mais avançados recursos pedagógicos suportados pela Web se, por exemplo, o canal de comunicação utilizado não fornecer uma largura de banda capaz de transmitir o conteúdo em um tempo satisfatório. Por outro lado, a não utilização/integração dos vários recursos pedagógicos suportados (resultado de uma tentativa de minimizar os custos) pode fazer com que a CMC seja utilizada como um mero meio de armazenamento de informações.

Durante o planejamento tecnológico, é necessário identificar quais recursos pedagógicos suportados pela CMC (videoconferência, apresentação multimídia, lista de discussão, etc.) se adequam a uma determinada estratégia de ensino. Encontradas as possíveis soluções, deve ser feita uma análise dos custos/benefícios obtidos com a adoção de uma alternativa particular. Esta análise visa identificar se a melhoria no processo de ensino-aprendizagem justifica os custos (recursos de *software*, recursos de *hardware*, recursos de horas trabalhadas, etc.) de se utilizar uma determinada solução.

[LAWHEAD et al 97] propõe a utilização de uma matriz bidimensional – *Uppsala Grid*– para a análise dos custos/benefícios relacionados às tecnologias de CMC que podem ser utilizadas na disponibilização de um curso a distância. A matriz *Uppsala* é composta pelos seguintes eixos:

- Eixo horizontal: Corresponde a uma lista, não exaustiva, dos vários componentes pedagógicos de uma lição (aula de exposição, aula de discussão, aula de laboratório, etc).

- Eixo Vertical: Corresponde a uma lista, também não exaustiva, dos recursos tecnológicos que podem ser associados aos componentes pedagógicos de uma lição.

Para cada ponto de intercessão entre a tecnologia de CMC e o recurso pedagógico, [LAWHEAD et al 97] sugere que seja feita a análise dos custos para a instituição/estudantes[1] e do benefício a nível de ensino/aprendizagem de se utilizar uma alternativa particular.

2.2.1 Recursos tecnológicos suportados pela CMC

A CMC suporta um conjunto de ferramentas que podem ser utilizadas para viabilizar as atividades pedagógicas associadas a uma estratégia de ensino. [HARTLEY et al 96] agrupa estas ferramentas seguindo duas dimensões. A primeira utiliza como critério de classificação o sincronismo da tecnologia; enquanto que, a segunda, classifica as tecnologias de acordo com a *mídia* suportada.

- Tecnologias Síncronas/Assíncronas: Caracterizam o sincronismo entre os participantes de uma atividade pedagógica. Apesar das tecnologias síncronas serem mais interativas que as tecnologias assíncronas, elas exigem que os estudantes/tutores acessem a atividade pedagógica em um horário preestabelecido.

- Tecnologias Baseadas em Texto/Tecnologias Baseadas em Recursos Multimídia: Caracterizam as ferramentas de acordo com a mídia utilizada, variando das tecnologias baseadas em texto às tecnologias que combinam diferentes mídias.

Outros critérios de classificação também serão utilizados, no presente trabalho, para agrupar os recursos tecnológicos suportados pela CMC. Esses critérios levam em

[1] Os custos para instituição correspondem aos gastos com desenvolvimento, aquisição de equipamentos, implantação, tempo despendido, etc. Por sua vez, os custos para os estudantes compreendem os gastos com a aquisição do *hardware*/*software* necessário e a utilização de um canal de comunicação para o acesso ao curso.

consideração os custos para a instituição e para o estudante, o grau de capacitação necessário para trabalhar com a tecnologia (tanto de desenvolvimento quanto de utilização) e o grau de interação suportado (interação com os participantes e com o ambiente de aprendizagem).

A seguir é apresentada uma lista com as tecnologias de CMC bem como as atividades pedagógicas que as mesmas podem suportar.

- HTML Estático: Tecnologia assíncrona que pode combinar texto e gráficos em um mesmo ambiente. O HTML estático apresenta baixa capacidade de interação, podendo ser utilizado para disponibilizar informações sobre os cursos e o material didático para os estudantes. A sua utilização apresenta baixo custo, pois a maior parte dos editores de texto atuais apresenta funcionalidades de conversão para HTML, além de existirem servidores e navegadores Web gratuitos.

- Aplicações Web do Lado Servidor: Possibilitam maior interação com o ambiente WWW que o HTML estático. Podem ser utilizadas, entre outras coisas, para a aplicação de exercícios de fixação e exigem que a equipe de desenvolvimento tenha conhecimento em programação com tecnologias como CGI, ASP, Servlets, etc.

- Aplicações Web do Lado Cliente: Estendem a capacidade de interação que o HTML estático oferece, provendo recursos que permitem a visualização de conceitos, a aplicação de exercícios e simulações. Exigem que a equipe de desenvolvimento tenha conhecimentos de tecnologias como SMIL, APPLETS Java, JavaScript, etc.

- Correio Eletrônico: Tecnologia assíncrona, que possui grande popularidade com os usuários da *Internet*. Pode ser utilizado, entre outras coisas, para o envio de questionamentos, esclarecimentos de dúvidas e para o estabelecimento de uma discussão sobre determinado assunto. Apresenta baixo custo para a instituição (já que existem servidores SMTP[2] gratuitos para a maior parte das plataformas) e para os estudantes.

- Grupo de Interesse: Permite o estabelecimento de uma discussão sobre determinado tópico. É uma tecnologia assíncrona, baseada em texto, que armazena

[2] *Simple Mail Transfer Protocol*

as mensagens recebidas para que possam ser acessadas pelos membros do grupo. Da mesma forma que o correio eletrônico, essa tecnologia também apresenta baixos custos para a instituição e para os estudantes.

- *Internet Real Chat* (IRC): Ferramenta síncrona, baseada em texto, que permite estabelecer uma discussão, em tempo real, sobre determinado assunto. A interação de uma atividade pedagógica utilizando o IRC pode envolver várias pessoas simultaneamente, sendo uma tecnologia que apresenta baixo custo para a instituição e para estudantes.

- *Virtual Reality Markup Language* (VRML): Linguagem de descrição de mundos virtuais e simulações interativas com vários participantes se comunicando através da Internet. É uma tecnologia síncrona, que combina gráficos e imagens, podendo ser utilizada para a visualização de conceitos e aplicações de aulas práticas/simulações. Apresenta baixo custo de desenvolvimento e utilização, porém, dependendo da complexidade do mundo virtual a ser construído, exige alto grau de especialização por parte da equipe de desenvolvimento [AMES 97, OTSUKA 97].

- Videoconferência: Tecnologia síncrona que envolve a transmissão de áudio e vídeo em tempo real entre os participantes dos cursos. Podem ser utilizados dois modelos de vídeo conferência para a educação a distância:

    - Videoconferência pessoal→ Os participantes das atividades pedagógicas podem participar da videoconferência em locais distintos. Constitui o suporte ideal para o ensino a distância, em que aulas e seminários podem ser assistidos por alunos separados geograficamente. A qualidade audiovisual resultante não é a ideal, por outro lado, os equipamentos necessários para suportar a videoconferência pessoal apresentam baixo custo.

    - Videoconferência em sala→ Os estudantes têm acesso à atividade pedagógica em uma sala compartilhada[3], onde recebem e enviam informações audiovisuais de alta qualidade. Exige que os estudantes

[3] É possível existirem várias salas, em locais distintos, sincronizada na mesma conferência.

estejam presentes em locais preestabelecidos e apresenta elevados custos em termos de equipamentos e canais de comunicação.

2.2.2 Classificação das Tecnologias

A Tabela 2-1 ilustra a classificação das tecnologias apresentadas de acordo com os critérios de sincronismo, mídia utilizada, grau de interação, custo e capacitação para desenvolvimento e utilização.

Tabela 2-1 Classificação das tecnologias CMC

Tecnologia	Critérios de classificação					
	Sincronismo	Mídia Utilizada (*)	Interação		Custo	
			Ambiente	Participantes	Instituição	Estudante
HTML Estático	Assíncrona	T, G, A	Sem interação	Sem interação	Baixo custo	Baixo custo
Aplicações Web do Lado Servidor	Assíncrona	T, G, A	Média interação	Sem interação	Baixo custo	Baixo custo
Aplicações Web do Lado Cliente	Assíncrona	T, G, A	Alta interação	Sem interação	Baixo custo	Baixo custo
Correio Eletrônico	Assíncrona	T, G, A	Sem interação	Média interação	Baixo custo	Baixo custo
Grupos de Interesse	Assíncrona	T	Sem interação	Média interação	Baixo custo	Baixo custo
Internet Real Chat	Síncrona	T	Sem interação	Alta interação	Baixo custo	Baixo custo
Virtual Reality Markup Language	Síncrona	T, G, A, V	Alta interação	Alta interação	Baixo custo	Baixo custo
Vídeo Compactado	Assíncrona	T, G, A, V	Baixa interação	Baixa interação	Alto custo	Baixo custo
Videoconferência Pessoal	Síncrona	T, A, V	Alta interação	Alta interação	Baixo custo	Baixo custo
Videoconferência em Sala	Síncrona	T, A, V	Alta interação	Alta interação	Alto custo	Alto custo

Tecnologia	Critérios de classificação	
	Capacitação de Desenvolvimento	Capacitação de Utilização
HTML Estático	Baixo grau de capacitação	Baixo grau de capacitação
Aplicações Web do Lado Servidor	Médio grau de capacitação	Baixo grau de capacitação
Aplicações Web do Lado Cliente	Médio grau de capacitação	Baixo grau de capacitação
Correio Eletrônico	Baixo grau de capacitação	Baixo grau de capacitação
Grupos de Interesse	Baixo grau de capacitação	Baixo grau de capacitação
Internet Real Chat	Baixo grau de capacitação	Baixo grau de capacitação
Virtual Reality Markup Language	Alto grau de capacitação	Médio grau de capacitação
Vídeo Compactado	Baixo grau de capacitação	Baixo grau de capacitação
Videoconferência Pessoal	Baixo grau de capacitação	Baixo grau de capacitação
Videoconferência em Sala	Médio grau de capacitação	Baixo grau de capacitação

(*)
T- Texto
A- Áudio
G- Gráfico
V- Vídeo

A Tabela 2-2 apresenta uma síntese sobre a possibilidade de se utilizar uma determinada tecnologia de CMC como ferramenta de suporte às atividades pedagógicas. É importante ressaltar que diferentes recursos pedagógicos podem ser utilizados para apresentação de uma lição. Por exemplo, em uma disciplina sobre compressão de dados, pode ser utilizada uma atividade de exposição, para a descrição de um determinado algoritmo, seguido de uma atividade de simulação que permita a visualização da sua execução. A primeira atividade poderia ser feita utilizando HTML Estático enquanto que a segunda poderia ser implementada utilizando um APPLET Java.

Tabela 2-2 Relacionamento entre os componentes pedagógicos/tecnológicos

Tecnologia	Recurso Pedagógico				
	Exposição	Discussão	Laboratório	Trab. Individual	Trab. em Grupo
HTML Estático	X				
Aplicações Web do Lado Servidor	X			X	
Aplicações Web do Lado Cliente	X		X	X	X
Correio Eletrônico		X			
Grupos de Interesse		X			X
Internet Real Chat		X			X
Virtual Reality Markup Language		X	X	X	X
Videoconferência Pessoal	X	X			X
Videoconferência em Sala	X	X			X

2.3 Demais Aspectos Relacionados a EaD Suportado pela CMC

Além dos aspectos pedagógicos e tecnológicos, um conjunto de fatores também deve ser levado em consideração durante o projeto de um curso a distância suportado pela CMC. As subsessões seguintes descrevem alguns desses aspectos.

2.3.1 Integridade, Segurança, Direitos Autorais e Ética

Em um curso a distância suportada pela CMC, pode existir pouca ou nenhuma interação face-a-face entre os estudantes e os educadores da instituição. A maior parte da comunicação é estabelecida utilizando recursos computacionais e a Internet. Dessa forma, é imprescindível a implantação de mecanismos para autenticar os estudantes e as instituições que disponibilizam os cursos.

Além da autenticação, é necessário que exista segurança dos dados armazenados nos servidores, impedindo que os trabalhos submetidos pelos estudantes sejam acessados ou violados por terceiros.

Em relação aos direitos autorais, é importante que, durante as fases de projeto e implantação de um curso a distância, sejam estabelecidas normas para a utilização tanto do material didático oferecido pela instituição quanto dos trabalhos produzidos pelos estudantes. Entre os objetivos dessas normas, podemos destacar:

- Impedir que sejam disponibilizados materiais didáticos que violem propriedades intelectuais.
- Impedir que sejam apropriados os trabalhos dos estudantes sem a devida permissão e reconhecimento.
- Restringir acesso ao material do curso aos estudantes registrados.

Ainda existem estudos sobre como estabelecer limites em relação ao que é permitido ou não em uma comunidade virtual. O estabelecimento de discussões *on-line* pode fazer com que surjam pontos de vistas antagônicos, devendo existir respeito sobre a opinião dos diferentes participantes.

2.3.2 Perfil dos Estudantes

Um outro aspecto a ser levado em consideração quando se deseja disponibilizar um curso baseado na CMC é o perfil dos estudantes. [PALLOF et al 99] apresenta um estudo realizado pelo Projeto de Educação a Distância da Califórnia que traça o perfil dos estudantes que obtêm melhores resultados em programas de EaD. Segundo esse trabalho, esses estudantes apresentam as seguintes características:

- São motivados, disciplinados e possuem grandes expectativas em relação aos cursos.
- Procuram de forma voluntária a educação.
- Geralmente possuem idade mais avançada que a média dos estudantes.

O grau de familiaridade com a tecnologia utilizada também é um fator importante a ser levantado no projeto de um curso a distância. Sugere-se que sejam utilizados recursos mais simples, com uma interface bastante amigável, nos primeiros contatos com a tecnologia por parte dos estudantes. Quando estes se mostrarem mais confortáveis com o uso dos recursos de CMC, ferramentas com maior riqueza tecnológica podem ser empregadas.

Existem controvérsias sobre as vantagens obtidas, em termos de relacionamento interpessoal, quando utilizamos a CMC. De acordo com Craig Brod, citado em [PALLOF et al 99], com o aumento do uso da comunicação eletrônica, diminuímos e alteramos o senso sobre nós mesmos, criando novas barreiras para a intimidade, continuidade e comunidade. Por outro lado, outros autores apresentam casos de pessoas tímidas que, em atividades pedagógicas face-a-face, não se sentiam confortáveis para interagir com os demais estudantes e professores. Quando esses mesmos alunos participavam de atividades *on-line*, eles levantavam questionamentos e contribuíam para o processo de ensino-aprendizagem. É imprescindível que os educadores tornem o ambiente educacional bastante dinâmico, fazendo com que sejam estabelecidos interações entre os estudantes com finalidades que estendam as acadêmicas.

2.4 Conclusões

Este capítulo apresentou os componentes que devem ser levados em consideração durante o projeto de um curso a distância suportado pela CMC. Em relação aos aspectos pedagógicos, foram definidos os papéis dos educadores e estudantes; e apresentada a necessidade da adoção de estratégias de ensino que foquem a educação colaborativa. Como existe pouca interação face-a-face no EaD, os alunos podem se tornar dispersos, caso eles não assumam um papel ativo no processo de ensino-aprendizagem. Em relação aos aspectos tecnológicos, foi apresentada uma proposta para a classificação de diferentes tecnologias que podem ser utilizadas para a educação a distância suportada pela CMC.

O próximo capítulo apresenta os esquemas de dados (níveis conceitual e físico) que estabelecem o relacionamento entre as estratégias de ensino, as atividades pedagógicas e os recursos tecnológicos associados a um curso a distância suportado pela CMC.

Bibliografia

[AMES 97] AMES, A., "*The VRML 2.0 Source Book*", John Wiley & Sons, Inc. 1997.

[CARLOS 97] CARLOS, A., "*Metodologia do Ensino Superior*", São Paulo, São Paulo: Atlas, 1997.

[CASEY 98] CASEY, D., "*Learning From or Though the Web: Models of Web Based Education*", In: Integrating Technology into Computer Science Education – ACM, Proceedings… p. 51-54, 1998.

[HARTLEY et al 96] HARTLEY, S., "*Enhancig Teaching Using The Internet*", In.: Integrating Technology into Computer Science Education –ACM. Proceedings… p. 218-228, 1996.

[LAWHEAD et al 97] LAWHEAD, P., ALPERT, CARSWEL, CIZMAR, DEWITT, FAHRAEUS, SCOTT, "*The Web and Distance Learning: What is Appropriate and What is Not*". In.: Integrating Technology into Computer Science Education –ACM. Proceedings… p. 27-37, 1997.

[LUCKESY 95] LUCKESY, C., "Avaliação da Aprendizagem: Um Ato Amoroso", In: Avaliação da Aprendizagem Escolar, p. 168-180. São Paulo, São Paulo: Cortez, 1995.

[MENDES 2000] MENDES, F., "*E-group: Um Ambiente para Suporte à Aprendizagem Colaborativa Baseada na Web*", Dissertação de Mestrado, UFPB/DSC, 2000.

[PALLOF et al 99] PALLOF, R., PRATT, "*Building Learning Communities in Cyberspace, Effective Strategies for The Online Classroom*", San Francisco, California: Jossey-Bass, 1999.

3 Esquemas de Dados para o Projeto de EaD Suportado pela CMC

Este capítulo descreve os esquemas de dados utilizados para representar os relacionamentos entre os componentes relevantes durante o projeto de um curso a distância suportado pela CMC.

A implantação de níveis de abstração entre o ambiente desenvolvido e as informações armazenadas nas bases de dados é o principal objetivo dos esquemas de dados. Diferentes níveis de abstração favorecem a reutilização e a flexibilidade, uma vez que componentes das camadas inferiores podem ser alterados sem que sejam necessárias mudanças nas camadas superiores. Para a modelagem de dados, geralmente são utilizados três níveis de abstração: o nível externo, que define as diferentes visões que as aplicações têm acesso; o nível conceitual, que descreve, de forma independe dos recursos computacionais, o sistema como um todo; e o nível interno, que representa o sistema em termos de estruturas computacionais (arquivos de dados, formas de acesso, indexação, etc.).

Inicialmente, é estabelecido um paralelo entre modelos e esquemas de dados. Em seguida são apresentados os esquemas do nível conceitual, criado utilizando o modelo entidade relacionamento; e o esquema interno, construído utilizando a tecnologia eXtensible Markup Language (XML).

3.1 Modelos de Dados x Esquemas de Dados

Para estabelecer o paralelo entre modelos e esquemas de dados, vamos utilizar o conceito de **extensão/intenção** apresentado em [SCHIEL 98] como uma das dimensões de um sistema de informação[1]. Podemos idealizar o conceito de extensão/intenção fazendo a seguinte analogia com a orientação a objetos: uma classe X está para o conceito de intenção da mesma forma que as instâncias (objetos) da classe X estão para o conceito de extensão. Os esquemas de dados são a extensão de um modelo de dados particular, utilizados para representar um sistema específico.

Em [SCHIEL 98] são identificados os níveis de *Dados de Aplicativos*, *Dicionário de Dados*, *Modelos de Dicionário de Dados* e *Meta Modelo de Dicionário de Dados* para a dimensão

[1] Segundo [SCHIEL 98], um sistema de informação computadorizado possui duas dimensões ortogonais: A *Dimensão do Ponto de Vista* (formada pelos níveis Externo, Conceitual e Interno) e a *Dimensão de Extensão/Intenção* (formado pelos níveis de Dados Aplicativos, Dicionário de Dados, Modelos de Dicionário de Dados e Meta Modelo de Dicionário de Dados).

Extensão/Intenção de um sistema de informação. Essas camadas apresentam a característica de que cada nível inferior é a extensão do nível imediatamente superior.

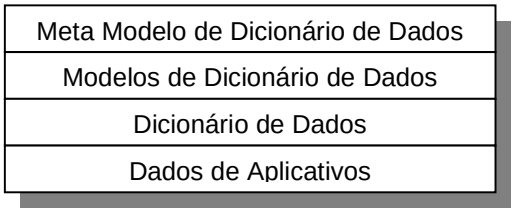


Figura 3-1 Níveis da Dimensão Extensão/Intenção

Dados de Aplicativos: Contêm os dados armazenados em um determinado instante. É a extensão real, física, do banco de dados.

Dicionário de Dados: Contém as estruturas de dados resultantes de um projeto do sistema modelado. Ex.: Esquema de dados seguindo uma notação de modelagem (Entidade Relacionamento, UML, etc.).

Modelos de Dicionário de Dados: Também conhecidos como modelos de dados, são formados por um conjunto de meios lingüísticos (ou gráficos) para descrever um esquema de dados.

Meta Modelo do Dicionário de Dados: Meta-linguagem que descreve as características específicas de um modelo de dicionário de dados. Essa meta-linguagem pode servir para descrever diferentes modelos de dicionários de dados.

3.2 Esquema de Dados: Nível Conceitual

O esquema conceitual de dados é responsável pela representação do sistema de forma independente dos recursos computacionais utilizados para armazenar as informações. Apesar de ser utilizado o *Rational Unified Process* em conjunto com os diagramas UML[2] no desenvolvimento da ferramenta de auxílio ao projeto de EaD, o esquema de dados aqui apresentado foi construído com base no modelo entidade-relacionamento (ER). A escolha pelo modelo ER se deve ao fato de querermos, neste momento, representar apenas as entidades relevantes no projeto de um curso a distância suportado pela CMC e seus relacionamentos; além disso, o modelo ER é um modelo de dados conceitual bastante utilizado dentro e fora do meio acadêmico.

[2] Essas tecnologias serão apresentadas no próximo capítulo.

3.2.1 Modelo Entidade-Relacionamento

O modelo ER descreve os dados em termos de entidades, atributos e relacionamentos. Uma entidade é um conceito do mundo real cuja existência não depende de outros elementos, podendo ser um objeto tangível – como um estudante, um meio físico de comunicação, etc. - ou pode ser um conceito abstrato – como uma lição de um curso a distância suportado pela CMC.

Os atributos são propriedades particulares que descrevem uma entidade. Ou seja, uma entidade *Curso* poderia ter os atributos *descrição*, *objetivos*, *quantidade de créditos*, *público alvo*, etc. Um conjunto de entidades que possuem os mesmos atributos dão origem a um *tipo de entidade*. As seguintes classes de atributos são suportados pelo modelo ER:

- *Simples x Composto:* Atributos compostos podem ser divididos em partes menores, representando informações básicas com significados independentes. Ex: O atributo Endereço do Aluno poderia ser constituído pelas informações Rua, Número, Bairro, Cidade, Estado e CEP. Os atributos que não podem ser subdivididos são chamados de simples ou atômicos.
- *Único x Multivalorado:* Atributos que possuem mais de um valor para a mesma entidade são chamados de atributos multivalorados. Ex.: O atributo Objetivo de um Curso pode ter mais de um valor para a mesma entidade. Os atributos que só admitem um valor por entidade são chamados de únicos.
- *Armazenado x Derivado:* Atributos que podem ser calculados a partir de outros atributos são chamadas de atributos derivados. Ex.: A idade de um estudante pode ser calculada a partir da sua data de aniversário. Os atributos que não são deriváveis a partir de outros atributos são classificados como armazenados.

As associações entre duas ou mais entidades recebem o nome de *relacionamento* no modelo ER. Sejam A e B dois tipos de entidades. Uma relação R entre A e B pode ser definida como um subconjunto do produto cartesiano entre A e B. [SCHIEL 98] descreve três tipos de relacionamento entre entidades:

- *Relacionamentos um-para-um:* Cada elemento de A está associado com exatamente um elemento de B (e vice-versa).

- *Relacionamentos um-para-muitos:* Cada elemento de A pode estar associado a vários elementos de B. Os elementos de B só podem estar associados a um único elemento de A.
- *Relacionamentos muitos-para-muitos:* Um elemento de A pode estar associado a vários elementos de B. Cada elemento de B pode estar associado a vários elementos de A.

3.2.2 Diagrama ER

O esquema apresentado na Figura 3-2 foi construído para descrever os relacionamentos entre as lições, as estratégias de ensino, as atividades pedagógicas e os recursos tecnológicos utilizados para a disponibilização de um curso a distância suportado pela CMC. Com essa premissa, entidades como Estudante e Instituição de Ensino, que fazem parte do contexto de um curso a distância, não estão presentes nesse modelo.

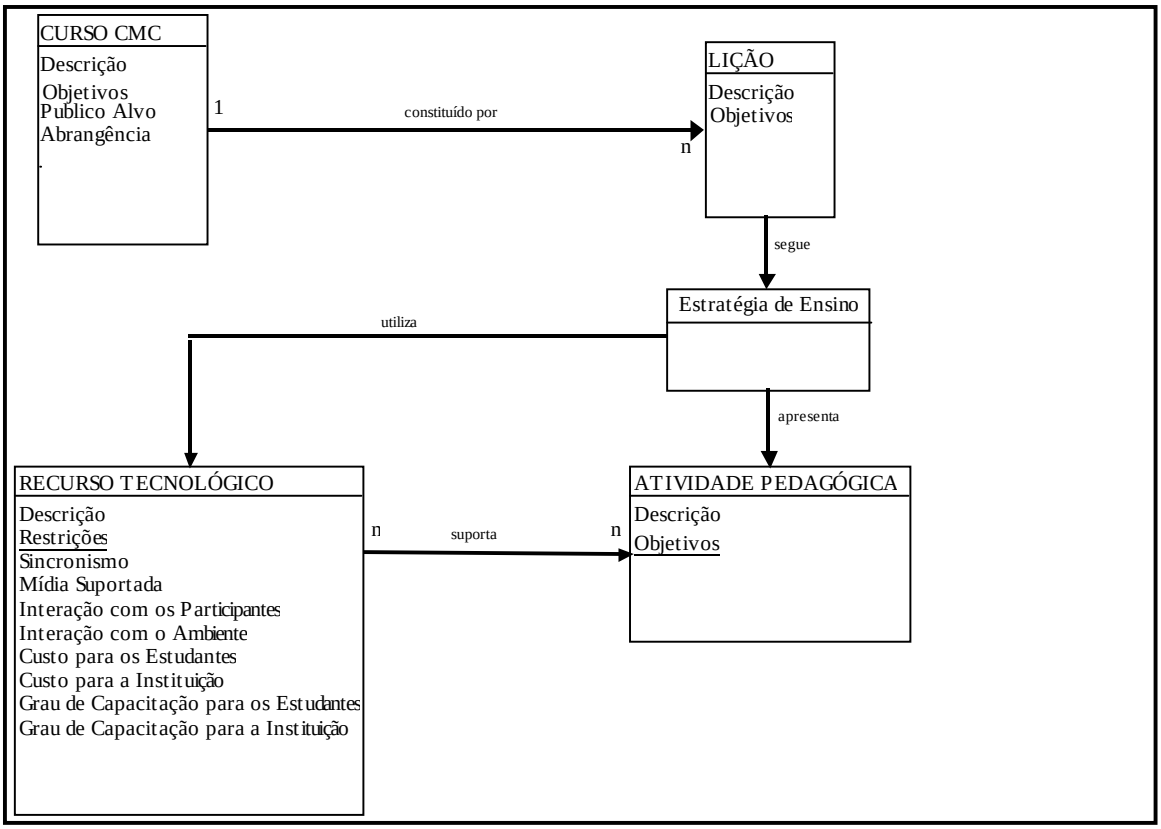


Figura 3-2 Esquema Entidade-Relacionamento

As seguintes entidades estão presentes no modelo:

- Curso CMC: Representa um curso a distância suportado pela CMC. Um curso possui uma descrição (Ex.: Programação Orientada a Objetos, Sistemas Operacionais I, etc.); um conjunto de objetivos (Ex.: aprender conceitos básicos da Orientação a Objetos, traçar o paralelo entre as atividades de Análise e Projeto, etc.); um público alvo (Ex.: alunos de ciência da computação que tenham cursado introdução a programação) e a abrangência do curso (disponibilizado globalmente, disponibilizado para algumas instituições, etc.). Os cursos podem ser disponibilizados em várias lições.
- Lição: Representa uma lição de um curso a distância suportado pela CMC. Cada lição possui uma descrição e um conjunto de objetivos; sendo apresentada seguindo a estratégia de ensino que seja mais adequada para o processo de ensino-aprendizagem.
- Estratégia de Ensino: Corresponde a decisão de se apresentar uma lição de um curso utilizando um conjunto de atividades pedagógicas efetivadas pelos recursos tecnológicos apropriados.
- Atividade Pedagógica: Representa uma atividade pedagógica (aula expositiva, prática, discussão, etc.) de um curso a distância suportado pela CMC. As atividades pedagógicas são suportadas por pelo menos um recurso tecnológico.
- Recurso Tecnológico: Representa um recurso tecnológico (correio eletrônico, lista de discussão, videoconferência, etc.) que pode ser utilizado para efetivar as atividades pedagógicas de um curso a distância suportado pela CMC. Os recursos tecnológicos possuem um conjunto de restrições que podem ser satisfeitas ou não pela instituição de ensino. Como exemplo de restrições de recursos tecnológicos poderíamos citar a largura do canal de comunicação, o grau de capacitação da equipe de desenvolvimento e a infraestrutura necessária para a sua utilização.

3.3 Esquema de Dados: Nível Físico

Nesta sessão é apresentado o esquema de dados, baseado na tecnologia XML (*eXtensible Markup Language*) [BRAY et al 2000][AMBLER 2000][PFEIFFER 2000], que estabelece a relação entre as estratégias de ensino, os recursos pedagógicos e os recursos tecnológicos de um curso a distância suportado pela CMC. A opção pela tecnologia XML é decorrente da sua portabilidade, flexibilidade e legibilidade.

3.3.1 Extensible Markup Language

A eXtensible Markup Language (XML) é um padrão industrial, independente de plataforma, utilizado para compartilhar dados. Apesar da linguagem XML, da mesma forma que a linguagem HTML (*Hiper Text Markup Language*), delimitar as informações com o auxílio de marcadores (*tags*), existem grandes diferenças entre as duas tecnologias. Em primeiro lugar, enquanto os marcadores HTML especificam como exibir as informações, os marcadores XML especificam o significado das mesmas. Em segundo lugar, o HTML suporta um conjunto limitado de marcadores, que foram predefinidos pelas especificações do padrão; enquanto que os marcadores XML são extensíveis, permitindo que os desenvolvedores das aplicações criem novos marcadores para descrever o significado dos dados. O exemplo de documento XML apresentado no Quadro 3-1, exibe construções que podem ser utilizadas para manter uma agenda de endereços de correio eletrônico:

Quadro 3-1 Exemplo de documento XML para uma agenda de correio eletrônico

```
<?xml version=”1.0”?>
<agendaEndereco>
  <pessoa>
    <nome>
      <ultimoNome>Almeida</ultimoNome><primeiroNome>Rodrigo</primeiroNome>
    </nome>
      <endereco>rba@dsc.ufpb.br</endereco>
  </pessoa>
  <pessoa>
    <nome>
      <ultimoNome>Almeida</ultimoNome><primeiroNome>Janaina</primeiroNome>
    </nome>
      <endereco>jana_jpa@xxx.com.br</endereco>
  </pessoa>
</agendaEndereco>
```

O documento `agendaEndereco`, além de ser bastante legível, pode servir como fonte de dados em diferentes ambientes e aplicativos que reconheçam seu formato. Além da portabilidade, outras informações (como endereço residencial, telefone, sexo, etc.) poderiam ser facilmente acrescentadas, bastando, para isso, incluir novos marcadores no formato do documento.

XML é um subconjunto simplificado da *Standard Generalized Markup Language* (SGML), baseada na *Generalized Markup Language* e padronizada em 1986. O padrão XML foi proposto para atender aos seguintes requisitos:

- Ter seu uso voltado para a Internet e para o compartilhamento de informações;
- Suportar uma grande variedade de aplicações;
- Ser compatível com o padrão SGML;
- Apresentar facilidades para a construção de programas que processam documentos XML (*parsers* XML);
- Apresentar facilidades para a criação de documentos XML;
- Os documentos XML devem ser razoavelmente claros e legíveis pelas pessoas.

A especificação XML é atualmente uma recomendação aceita pelo W3C (*World Wide Web Consortium*), aguardando uma aceitação formal de padrão pelo mesmo órgão.

3.3.2 Elementos e atributos dos documentos XML

Os documentos XML possuem uma estrutura de marcadores hierárquica. Isto é, para cada marcador de início, deve existir um marcador de fim correspondente. No vocabulário XML, um **elemento** corresponde a um par de marcadores de início e fim. Qualquer elemento deve estar corretamente delimitado por um outro elemento, existindo um único elemento raiz nos documentos XML. Utilizando o exemplo de documento para agenda de correio eletrônico apresentado anteriormente, temos que o elemento raiz é Agenda de Endereço, delimitado pelos marcadores `<agendaEndereco>` e `</agendaEndereco>`.

Os atributos são utilizados para associar pares *nome-valor* aos elementos. Cada elemento pode ter um conjunto de atributos; e cada atributo só pode ser descrito nos marcadores de início ou fim do elemento. Se desejássemos descrever os elementos Primeiro Nome e Último Nome como atributos do elemento Nome, a sintaxe seria a seguinte:

Quadro 3-2 Sintaxe para a criação de atributos

```
<pessoa>
  <nome ultimoNome=”Almeida” primeiroNome = “Rodrigo”>
  </nome>
  <endereco>rba@dsc.ufpb.br</endereco>
</pessoa>
```

A decisão sobre quando representar uma informação como elemento ou atributo é feita com base na hierarquia existente entre os elementos e na possibilidade de existir mais de uma ocorrência da informação associada a uma outra. Só pode haver uma ocorrência de um determinado atributo por elemento; enquanto que pode existir mais de uma ocorrência de um determinado sub-elemento.

3.3.3 Definição dos tipos de documento

A definição dos marcadores, necessários para um tipo particular de documento XML, é feita utilizando uma linguagem de esquema XML. Um esquema descreve a estrutura de um conjunto de documentos e define as restrições para o conteúdo dos mesmos. A linguagem de esquema mais utilizada para definir aplicações XML é a *Document Type Definition Language* [BUCK 2001][BOURRET 2000]. Os esquemas definidos nessa linguagem são chamados de *Document Type Definition* (DTD).

Um documento XML é válido se o seu conteúdo obedece às regras gramaticais definidas no DTD correspondente. Essa validação permite que as aplicações chequem se os dados XML estão completos, se estão formatados corretamente e se possuem os valores apropriados para os atributos.

Um DTD define uma gramática que descreve quais marcadores e atributos são válidos em um documento XML, e em que contexto. Gramáticas para linguagens são descritas com EBNF (*Extended Backus-Naur Form*), utilizando regras de produção, onde o lado esquerdo da regra representa uma construção e o lado direito representa seu conteúdo. Utilizando EBNF, poderíamos indicar que uma pessoa contém um nome e, opcionalmente, um e-mail da seguinte forma:

```
pessoa ::= (nome e-mail*)
```

Uma declaração do elemento *pessoa*, equivalente, em um DTD seria:

```
<!ELEMENT pessoa (nome, e-mail*)>
```

A Tabela 3-1 apresenta os símbolos e as palavras reservadas utilizadas nas construções DTDs para especificar as regras dos elementos XML.

Tabela 3-1 Símbolos e palavras reservadas das construções DTDs

Construção	Significado
A?	O elemento A é opcional
A+	Uma ou mais ocorrências do elemento A
A*	Zero ou mais ocorrências do elemento A
A | B	Ocorrência do elemento A ou do elemento B (não de ambos)
A, B	Ocorrência do elemento A seguido do elemento B (nesta ordem)
(A, B)+	Uma ou mais ocorrências do elemento A seguido do elemento B.
#PCDATA	Ocorrência de uma dada string.

O Quadro 3-3 descreve o DTD utilizado na definição dos documentos XML para a agenda de correio eletrônico:

Quadro 3-3 DTD utilizado para a definição dos documentos XML de agenda eletrônica

```
<!ELEMENT agendaEndereco (pessoa)+>
<!ELEMENT pessoa (nome, e-mail*)>
<!ELEMENT nome (segundoNome, primeiroNome)>
<!ELEMENT segundoNome (#PCDATA)>
<!ELEMENT primeiroNome (#PCDATA)>
<!ELEMENT e-mail (#PCDATA)>
```

As regras para a declaração de atributos em DTD apresentam estrutura semelhante às utilizadas para definição dos elementos. Por exemplo, para a definição de um atributo *sexo*, associado ao elemento pessoa, poderíamos ter:

Quadro 3-4 Exemplo de definição do atributo sexo associado ao elemento pessoa

```
<!ELEMENT pessoa (nome, e-mail*)>
<!ATTLIST pessoa sexo(masculino , feminino) #IMPLIED>
```

A construção acima representa que o elemento *pessoa* possui um atributo enumerado, cujo nome é *sexo* e que pode assumir os valores “masculino” ou “feminino”. A palavra reservada `#IMPLIED` indica que este atributo é opcional (pode estar presente ou não em um elemento *pessoa*).

A Tabela 3-2 apresenta os símbolos e palavras reservadas utilizadas para estabelecer as regras dos atributos XML.

Tabela 3-2 Símbolos e palavras reservadas para a definição dos atributos XML

Construção	Significado
att CDATA	O atributo *att* pode assumir qualquer valor.
att (val1, val2, ..., valn)	O atributo *att* pode assumir um dos valores *val1*, *val2*, ..., *valn*.
att CDATA #FIXED 'val1'	O atributo *att* possui o valor fixo '*val1*'.
att CDATA #IMPLIED	O atributo *att* é opcional.
att CDATA #REQUIRED	O atributo *att* é requerido.
att CDATA ID #REQUIRED	O atributo *att* é requerido e possui valor único.
att CDATA IDREF	O atributo *att* deve corresponder com um atributo ID de algum outro elemento XML.

3.3.4 Definição dos esquemas de dados

Agora que já apresentamos as construções dos documentos XML, descreveremos os esquemas propostos para o projeto de cursos a distância suportados pela CMC. Esses modelos devem estabelecer as relações entre os seguintes componentes:

- Lições que compõem um curso a distância suportado pela CMC;
- Estratégias de ensino selecionadas para a apresentação das lições;
- Atividades pedagógicas associadas às estratégias de ensino; e
- Recursos tecnológicos escolhidos para suportar as atividades pedagógicas.

Dois esquemas (DTDs) são sugeridos nessa dissertação. Os documentos que obedecem ao primeiro esquema (documentos ***relacaoAPRT***) devem ser produzidos de forma conjunta pelos educadores e pela equipe técnica das instituições; servindo como base de conhecimentos que associa as atividades pedagógicas aos recursos tecnológicos. Já os documentos referentes ao segundo esquema (documentos ***cursoEDCMC***), são produzidos como resultado final do projeto de um curso a distância suportado pela CMC.

As seguintes premissas foram definidas para o primeiro DTD:

- Cada atividade pedagógica possui uma descrição, um conjunto de objetivos e pode ser efetivada por, pelo menos, um recurso tecnológico. Por exemplo, a atividade pedagógica *aula de discussão* possui como principais objetivos:
    - Desenvolver novos conhecimentos mediante a utilização de conhecimentos e experiências anteriores;
    - Possibilitar o enfoque de um assunto sobre diferentes pontos de vista.

Além disso, a atividade *aula de discussão* podem ser efetivadas através de vídeo conferência, sala de bate papo, etc.

- Cada recurso tecnológico possui uma descrição, uma lista de restrições que podem estar satisfeitas ou não pela instituição, e um conjunto de atributos enumerados, que o qualificam de acordo com os critérios de sincronismo, mídia utilizada, interação com os participantes, interação com o ambiente, etc.

Seguindo essas premissas, o seguinte esquema XML, definido utilizando DTD, está sendo proposto:

Quadro 3-5 DTD para os documentos XML relacaoAPRT

```
<!ELEMENT relacaoAPRT (atividadePedagogica+, recursoTecnologico+)>
<!ELEMENT atividadePedagogica (objetivoAP)+>
<!ATTLIST atividadePedagogica codigoAP ID #REQUIRED>
<!ATTLIST atividadePedagogica descricaoAP CDATA #REQUIRED>
<!ELEMENT objetivoAP (#PCDATA)>
<!ELEMENT recursoTecnologico (restricaoRecursoTecnologico*)>
<!ATTLIST recursoTecnologico descricaoRT CDATA #REQUIRED>
<!ATTLIST recursoTecnologico codigoAP IDREFS #REQUIRED>
<!ATTLIST recursoTecnologico sincronismo (Síncrona | Assíncrona) #REQUIRED>
<!ATTLIST recursoTecnologico midia (Texto | Multimídia) #REQUIRED>
<!ATTLIST recursoTecnologico interacaoPart (Ausente | Média | Alta) #REQUIRED>
<!ATTLIST recursoTecnologico interacaoAmb (Ausente | Média | Alta) #REQUIRED>
<!ATTLIST recursoTecnologico custoEst (Baixo | Médio | Alto) #REQUIRED>
<!ATTLIST recursoTecnologico custoInst (Baixo | Médio | Alto) #REQUIRED>
<!ATTLIST recursoTecnologico capacitacaoDes (Baixo | Médio | Alto) #REQUIRED>
<!ATTLIST recursoTecnologico capacitacaoUtil (Baixo | Médio | Alto) #REIRED>
<!ELEMENT restricaoRecursoTecnologico EMPTY>
<!ATTLIST restricaoRecursoTecnologico descricaoRestricao CDATA #REQUIRED>
<!ATTLIST restricaoRecursoTecnologico satisfeita (Sim | Não) #REQUIRED>
```

Para o segundo DTD, as seguintes premissas foram definidas:

- Um curso possui uma descrição, um conjunto de objetivos e é constituído por várias lições;
- Cada lição possui uma descrição, um objetivo e é apresentada seguindo alguma estratégia de ensino;
- As estratégias de ensino possuem uma descrição e são compostas por um conjunto de atividades pedagógicas;

- As atividades pedagógicas possuem uma descrição e podem ser apresentadas por um conjunto de recursos tecnológicos.

De acordo com essas premissas, o seguinte DTD está sendo proposto:

Quadro 3-6 DTD para os documentos XML cursoEDCMC

```
<!ELEMENT cursoEDCMC(objetivoCurso+, licaoCurso+)>
<!ATTLIST cursoEDCMC descricaoCurso CDATA #REQUIRED>
<!ATTLIST cursoEDCMC publicoAlvo CDATA #REQUIRED>
<!ATTLIST cursoEDCMC abrangencia CDATA #REQUIRED>
<!ELEMENT objetivoCurso(#PCDATA)>
<!ELEMENT licaoCurso (objetivoLicao+, estrategiaEnsino)>
<!ATTLIST licaoCurso descricaoLicao CDATA #REQUIRED>
<!ELEMENT objetivoLicao(#PCDATA)>
<!ELEMENT estrategiaEnsino (atividadePedagogica)+>
<!ELEMENT atividadePedagogica (recursoTecnologico)+>
<!ATTLIST atividadePedagogica descricaoAP CDATA #REQUIRED>
<!ELEMENT recursoTecnologico (restricaoRT)*>
<!ATTLIST recursoTecnologico descricaoRT CDATA #REQUIRED>
<!ELEMENT restricaoRT EMPTY>
<!ATTLIST restricaoRT descricaoRestricao CDATA #REQUIRED>
<!ATTLIST restricaoRT satisfeito (SIM|NÃO) #REQUIRED>
```

3.3.4.1 Exemplo de documento XML relacaoAPRT

O exemplo de documento XML relacaoAPRT, apresentado a seguir, descreve os relacionamentos entre as atividades pedagógicas *Exposição*, *Discussão*, *Simulação*, *Trabalho Individual* e *Trabalho em Grupo* e os recursos tecnológicos *HTML Estático*, *Aplicação Web do Lado Servidor* e *Videoconferência em Sala*. As tabelas Tabela 3-3 e Tabela 3-4 descrevem, respectivamente, os qualificadores das atividades pedagógicas e dos recursos tecnológicos.

Tabela 3-3 Atividades Pedagógicas do documento relacaoAPRT

Atividades Pedagógicas		
Atividade Pedagógica	Código	Objetivos
Exposição	AP1	– Apresentar conteúdos didáticos aos alunos
Discussão	AP2	– Favorecer a reflexão acerca dos conhecimentos adquiridos em leituras/aulas expositivas; – Desenvolver novos conhecimentos mediante a utilização de conhecimentos anteriores; – Possibilitar o enfoque de um assunto sob diferentes pontos de vista.

Simulação	AP3	– Aplicar os conhecimentos adquiridos em situações práticas ou reais; – Estimular a reflexão sobre determinado problema.
Trabalhos individuais	AP4	– Permitir aos estudantes refletir sob seu domínio em um determinado assunto.
Trabalhos em grupo	AP5	– Possibilitar aos alunos refletir sob o conteúdo didático lecionado; – Desenvolver a capacidade dos estudantes trabalharem em equipe. – Desenvolver habilidades interpessoais que são importantes em situações além das acadêmicas.

Tabela 3-4 Recursos tecnológicos do documento relacaoAPRT

Recursos Tecnológicos			
Recurso Tecnológico	Cd. Atividades Pedagógicas	Atributos	Restrições
HTML Estático	AP1	– sincronismo = Assíncrona – mídia = Multimídia – interacaoPart = Ausente – interacaoAmb = Ausente – custoEst = Baixo – custoInst = Baixo – capacitacaoUtil = Baixo – capacitacaoDes = Baixo	-
Aplicação Web do lado Servidor	AP1 AP4	– sincronismo = Assíncrona – mídia = Multimídia – interacaoPart = Ausente – interacaoAmb = Média – custoEst = Baixo – custoInst = Baixo – capacitacaoUtil = Baixo – capacitacaoDes = Médio	– Conhecimentos em CGI, Servlets, etc

Vídeo conferência em Sala	AP1 AP2 AP5	– sincronismo = Síncrona – mídia = Multimídia – interacaoPart = Alta – interacaoAmb = Baixa – custoEst = Alto – custoInst = Alto – capacitacaoUtil = Baixo – capacitacaoDes = Médio	– Canal de comunicação com mínimo de 300 kbps – Sala disponível com equipamentos audiovisuais.

Quadro 3-7 Exemplo do documento relacaoAPRT

```
<?xml version=”1.0” ?>
<!DOCTYPE relacaoAPRT SYSTEM “relacaoAPRT.dtd”>
<relacaoAPRT>
   <atividadePedagogica codigoAP = “AP1” descricaoAP = “Exposição”>
      <objetivoAP>
         Apresentar conteúdos didáticos aos alunos.
      </objetivoAP>
   </atividadePedagogica>
   <atividadePedagogica codigoAP = “AP2” descricaoAP = “Discussão”>
      <objetivoAP>
         Favorecer a reflexão acerca dos conhecimentos adquiridos em leituras/aulas expositivas.
      </objetivoAP>
      <objetivoAP>
         Desenvolver novos conhecimentos mediante a utilização de conhecimentos anteriores.
      </objetivoAP>
      <objetivoAP>
```

```
        Possibilitar o enfoque de um assunto sob diferentes pontos de vista.
    </objetivoAP>
</atividadePedagogica>
<atividadePedagogica codigoAP = “AP3” descricaoAP = “Simulação”>
    <objetivoAP>
        Aplicar os conhecimentos adquiridos em situações práticas ou reais.
    </objetivoAP>
    <objetivoAP>
        Estimular a reflexão sobre determinado problema.
    </objetivoAP>
</atividadePedagogica>
<atividadePedagogica codigoAP = “AP4” descricaoAP = “Trabalhos Individuais”>
    <objetivoAP>
        Permitir aos estudantes refletir sobre seu domínio em um determinado assunto.
    </objetivoAP>
</atividadePedagogica>
<atividadePedagogica codigoAP = “AP5” descricaoAP = “Trabalhos em Grupo”>
    <objetivoAP>
        Possibilitar aos alunos refletir sobre o conteúdo lecionado.
    </objetivoAP>
    <objetivoAP>
        Desenvolver a capacidade dos estudantes trabalharem em equipe.
    </objetivoAP>
    <objetivoAP>
        Desenvolver habilidades interpessoais que são importantes em situações além das acadêmicas.
    </objetivoAP>
</atividadePedagogica>
```

```
<recursoTecnologico descricaoRT = ”HTML Estático”
                    codigoAP = ”AP1”
                    sincronismo = ”Assíncrona”
                    midia = ”Multimídia”
                    interacaoPart = ”Ausente”
                    interacaoAmb = ”Ausente”
                    custoEst  = ”Baixo”
                    custoInst = ”Baixo”
                    capacitacaoDes = ”Baixo”
                    capacitacaoUil = ”Baixo”>
</recursoTecnologico>
<recursoTecnologico descricaoRT = ”Aplicação Web do Lado Servidor”
                    codigoAP = ”AP1 AP4”
                    sincronismo = ”Assíncrona”
                    midia = ”Multimídia”
                    interacaoPart = ”Ausente”
                    interacaoAmb = ”Média”
                    custoEst  = ”Baixo”
                    custoInst = ”Baixo”
                    capacitacaoDes = ”Médio”
                    capacitacaoUil = ”Baixo”>
   <restricaoRecursoTecnologiaco descricaoRestricao = "Conhecimentos em tecnologias como CGI, ASP, Servlets, etc.”
                                 Satisfeita = ”Não”>
   </restricaoRecursoTecnologico>
</recursoTecnologico>
```

```
   <recursoTecnologico descricaoRT = ”Videoconferência em Sala”
                       codigoAP = ”AP1 AP2 AP5”
                       sincronismo = ”Síncrona”
                       midia = ”Multimídia”
                       interacaoPart = ”Alta”
                       interacaoAmb = ”Baixa”
                       custoEst  = ”Alto”
                       custoInst = ”Alto”
                       capacitacaoDes = ”Médio”
                       capacitacaoUil = ”Baixo”>
      <restricaoRecursoTecnologico descricaoRestricao = "Canal de comunicação com mínimo de 300Kbps”
                                   Satisfeita = ”SIM”>
      </restricaoRecursoTecnologico>
      <restricaoRecursoTecnologiaca descricaoRestricao = "Sala disponível com equipamentos audiovisuais”
                                   Satisfeita = ”SIM”>
   </recursoTecnologico>
</relacaoAPRT>
```

3.3.5 Exemplo de documento XML para o DTD cursoEADCMC

Seja o curso a distância "Métodos Avançados de Programação" suportado pela CMC, com as características descritas na Tabela 3-5:

Tabela 3-5 Exemplo para o curso Métodos Avançados de Programação

Métodos Avançados de Programação			
Objetivos	– Apresentar uma visão histórica das técnicas que visam a reutilização de software. – Destacar o papel dos Padrões de Projeto. – Apresentar os conceitos evolvidos no desenvolvimento de software baseado em componentes. – Comparar arquiteturas de componentes disponíveis no mercado.		
Público Alvo	– Alunos de ciência da computação que tenham cursado disciplina introdutória sobre orientação a objetos.		
Abrangência	– Distribuído geograficamente.		
Lição			
Facetas da Reutilização de Software	Objetivos	– Apresentar as dificuldades no desenvolvimento de software – Contextualizar a reutilização como saída para o desenvolvimento de software de qualidade (atenda às expectativas dos usuários cumprindo o planejamento de custos e prazos).	
	Estratégia de Ensino	Atividade Pedagógica	Recurso Tecnológico
		Exposição	Videoconferência pessoal
		Discussão	Sala debate-papo
Design Patterns	Objetivos	– Abordar os Padrões de Projeto como técnica de reutilização. – Apresentar a estrutura de um padrão de projeto – Propor que os alunos desenvolvam apresentações (em HTML) sobre os Padrões de Projeto não apresentados pelo professor.	
	Estratégia de Ensino	Atividade Pedagógica	Recurso Tecnológico
		Exposição	Aplicação Web do lado cliente
		Atividade em Grupo	Sala debate-papo
Desenvolvimento de software baseado em componentes	Objetivos	– Apresentar o desenvolvimento de software baseado em componentes. – Estabelecer o paralelo *Client Components x Server Components* – Comparar as diferentes arquiteturas de componentes.	
	Estratégia de Ensino	Atividade Pedagógica	Recurso Tecnológico
		Exposição	Aplicação Web do lado cliente
		Atividade Individual	Aplicação Web do lado servidor

Quadro 3-8 Exemplo do documento relacaoAPRT para o curso Métodos Avançados de Programação

```
<?xml version=”1.0” ?>
<!DOCTYPE cursoEDMC SYSTEM “cursoEDMC.dtd”>
<cursoEDMC descricaoCurso = “Métodos Avançados de Programação”
           publicoAlvo = “Alunos de Ciência da Computação que tenham cursado Introdução a OO”
           abrangencia = “Curso distribuído geograficamente”>
<objetivoCurso>Apresentar uma visão histórica das técnicas que visam a reutilização de software.</objetivoCurso>
<objetivoCurso>Destacar o papel dos Padrões de Projeto.</objetivo>
<objetivoCurso>Apresentar os conceitos evolvidos no desenvolvimento de software baseado em componentes</objetivoCurso>
<objetivoCurso>Comparar arquiteturas de componentes disponíveis no mercado.</objetivoCurso>
  <licaoCurso descricaoLicao = “Facetas da Reutilização de Software”>
     <objetivoLicao>Apresentar as dificuldades no desenvolvimento de software</objetivoLicao>
     <objetivoLicao> Contextualizar a reutilização como saída para o desenvolvimento de
                     software de qualidade.
    </objetivoLicao>
     <estrategiaEnsino>
        <atividadePedagogica descricaoAP = “Exposição”>
           <recursoTecnologico descricaoRT = “Vídeo Conferência pessoal”></recursoTecnologico>
        </atividadePedagogica>
        <atividadePedagogica descricaoAP = “Discussão”>
           <recursoTecnologico descricaoRT = “Sala de Bate-Papo”></recursoTecnologico>
        </atividadePedagogica>
     </estrategiaEnsino>
  </licaoCurso>
  <! Demais lições deveriam ser adicionadas aqui -->
</cursoEDMC>
```

3.4 Conclusões

Este capítulo apresentou as estruturas de dados (tanto a nível conceitual quanto físico) utilizadas para relacionar os componentes relevantes de um projeto de curso a distância suportado pela CMC. O próximo capítulo descreve o processo e os artefatos construídos durante o desenvolvimento do ambiente proposto nessa dissertação.

Bibliografia

[AMBLER 2000] AMBLER, S., "*XML in The Real World*", Thinking Objectively, Software Development, setembro de 2000.
on-line: http://www.sdmagazine.com/documents/sdm0009h/

[BOURRET 2000] BOURRET, R., "*Declaring Elements and Attributes in an XML DTD*", 2000, **on-line:** http://www.rpbourret.com/xml/xmldtd.htm

[BRAY et al 2000] BRAY, T., PAOLI, MALER, "*Extensible Markup Language*", W3C Recommendation, 2000. **on-line:** http://www.w3.org/XML/

[BUCK 2001] BUCK, L., "*Modeling Relational Data in XML* ", Extensibility Report, 2001. **on-line:** http://www.extensibility.com/main_modeling.htm

[ELMASRI 2000] ELMASRI, R., SHAMKANT, "*Fundamentals Of Database Systems*", Addison-Wesley, 2000.

[PFEIFFER 2000] PFEIFFER, R., "*XML: Tutorials for Programmers*", IBM-Online Courses, 2000. **on-line:** http://www-106.ibm.com/developerworks/xml/

[SCHIEL 98] SCHIEL, U., "*Elementos de Sistemas de Informações e Banco de Dados*", monografia, UFPB/DSC, 1998.

4 Projeto EaD

Este capítulo apresenta a ferramenta de auxílio para o projeto de cursos de educação a distância. Como metodologia de desenvolvimento da ferramenta, foi seguida a adequação do Processo Unificado (Apêndice A) para Pequenos Projetos apresentada em [POLLICE 2000]. O Processo Unificado se sustenta em três características: ser iterativo e incremental; ser orientado pelos casos de uso; e ser focado em uma arquitetura. A seguir serão apresentadas as iterações utilizadas durante o desenvolvimento, bem como os artefatos (diagramas UML, classes Java, etc.) construídos durante as disciplinas de análise, projeto, implementação e testes do sistema.

4.1 A Solução

Como foi apresentado no capítulo 2, o projeto de um curso a distância suportado pela CMC exige a especificação de componentes heterogêneos (pedagógicos, tecnológicos, sociais, etc.). Muitas vezes os educadores não têm conhecimentos sobre os recursos tecnológicos de CMC que podem ser utilizados para o EaD. Com base nisso, foi desenvolvido uma aplicação cujo objetivo é ajudar os educadores a projetar um curso a distância suportado pela CMC. O restante desse capítulo apresenta os artefatos construídos no desenvolvimento do *Projeto EaD*.

4.1.1 Artefatos da Fase de Iniciação

Os seguintes artefatos foram construídos durante a fase de iniciação:

a) Visão do Sistema: *A visão do sistema é um artefato construído na fase de iniciação que descreve, em um alto nível, o objetivo, os usuários e as principais restrições de uma aplicação. Para pequenos projetos, é normal que a visão do sistema tenha um ou dois parágrafos* [POLLICE 2000].

Visão do Sistema
O objetivo do sistema *Projeto EaD* é auxiliar pedagogos, não especialistas em informática, a especificar um curso a distância suportado pela CMC. Para isso, faz-se necessário a criação de um banco de dados que estabeleça as relações entre as características dos cursos suportados pela CMC, os recursos pedagógicos e os recursos tecnológicos. A interação com o sistema deve ser feita utilizando um navegador Web. Após o educador percorrer as etapas necessárias para a especificação, o sistema deve gerar um relatório indicando a estrutura do curso e os recursos pedagógicos/tecnológicos utilizados, apontando possíveis carências na infraestrutura da instituição.

Figura 4-1 Visão do Sistema

b) Lista dos Principais Riscos: *A lista contemplando os principais riscos do desenvolvimento serve para planejar as etapas de desenvolvimento. Quanto mais cedo os riscos forem identificados e tratados, menores serão as chances do projeto fracassar.*

Riscos
1) Período de desenvolvimento de um mês (talvez insuficiente para cobrir todas as funcionalidades). 2) Pouco conhecimento na integração das tecnologias Java (utilizada no desenvolvimento) e XML (utilizada como fonte de dados da aplicação).

Figura 4-2 Riscos do Projeto

c) Diagramas de Casos de Uso: *Os diagramas de caso de uso são utilizados para capturar os requisitos funcionais da aplicação, levando em conta a perspectiva de "para quem" ou "para o que" tal funcionalidade vai produzir um resultado de valor. Cada caso de uso corresponde a uma seqüência de atividades que o sistema deve disponibilizar para oferecer um resultado significativo para um ator ("para quem" ou "para o que" serve a funcionalidade). A Figura 4.3 apresenta o diagrama de casos de uso para o Projeto EaD.*

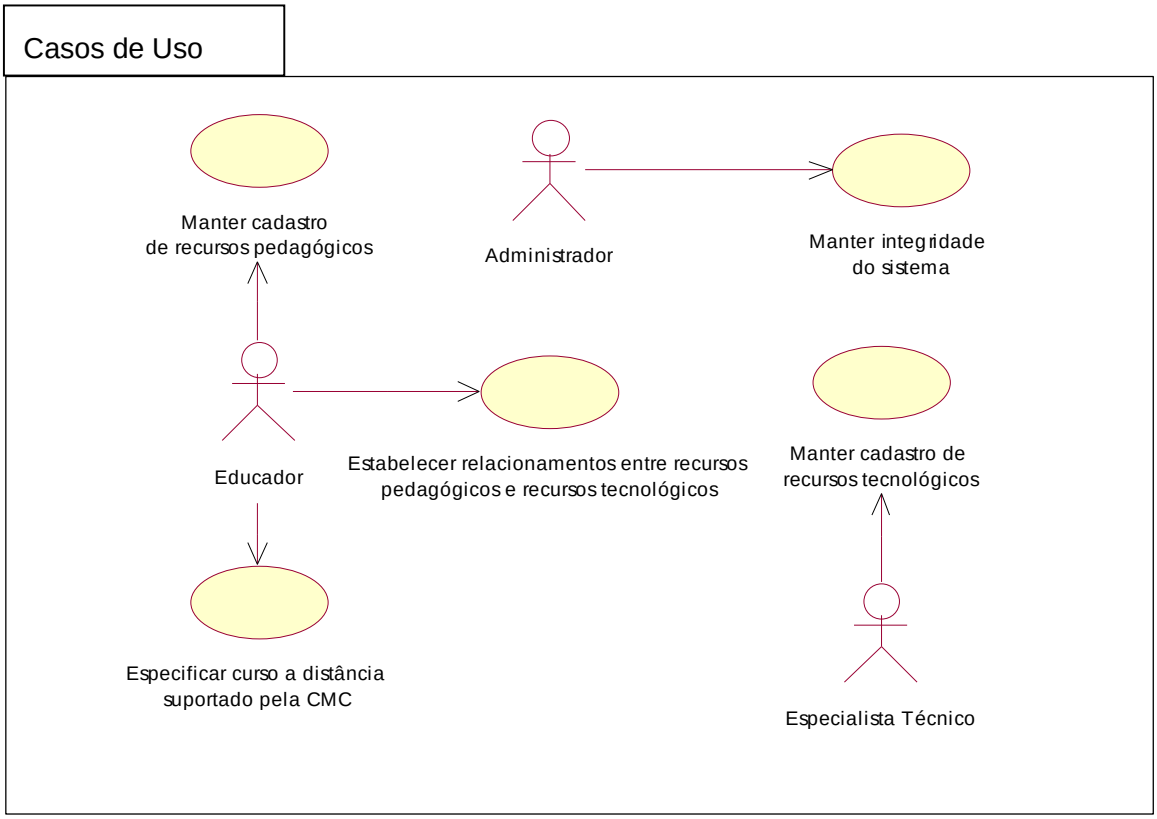


Figura 4-3 Diagrama de Casos de Uso

A seguir são descrito os atores e os casos de uso do *Projeto EaD*:

Atores:

1) Educador: *Papel que representa os educadores da instituição que desejam especificar um curso a distância utilizando recursos de CMC.*
2) Especialista Técnico: Papel que representa os especialistas na área técnica e que são responsáveis por manter as informações sobre os recursos tecnológicos.
3) Administrador: *Papel que representa os responsáveis por manter a integridade do sistema (políticas de acesso, arquivos XML, etc.).*

Casos de Uso

1) Especificar curso a distância suportado pela CMC

Ator: Educador

Descrição: Os educadores iniciam o caso de uso fornecendo informações (nome do curso, público alvo, forma de acesso, etc.) sobre o curso. Em seguida, o educador deve selecionar, para cada lição do curso, a estratégia de ensino e as tecnologias de CMC a serem utilizadas na apresentação das atividades pedagógicas. Com base na infraestrutura e nos recursos disponíveis na instituição, o educador pode ser alertado da não satisfação de algum requisito tecnológico, sendo necessária a decisão em manter a especificação ou reiterar o processo.

2) Manter cadastro de recursos pedagógicos

Ator: Educador

Descrição: Permite que sejam feitas manutenções (inserções, alterações, remoções) das informações sobre os recursos pedagógicos.

3) Estabelecer relacionamentos entre recursos pedagógicos e recursos tecnológicos

Ator: Educador

Descrição: Permite que o educador, em conjunto com um especialista na área técnica, estabeleça os relacionamentos entre os recursos pedagógicos (exposição, laboratório, trabalhos individuais, etc.) e os recursos tecnológicos (videoconferência, lista de discussão, correio eletrônicos, etc.).

4) Manter cadastro de recursos tecnológicos

Ator: Especialista Técnico

Descrição: Permite que sejam feitas as manutenções (inserções, alterações, remoções) das informações sobre os recursos tecnológicos.

5) Manter integridade do sistema

Ator: Administrador

Descrição: Permite que o administrador configure o acesso dos usuários, verifique a consistência dos arquivos, etc.

4.1.2 Artefatos da Fase de Elaboração

O desenho arquitetural de um sistema consiste em modelos a nível de requisitos, analise, projeto e implementação, que contemplam os aspectos mais significativos da aplicação. Em termos de funcionalidade e riscos de desenvolvimento, o caso de uso "Especificar curso a distância suportado pela CMC" se apresenta como o mais significativo para o projeto, servindo, dessa forma, como ponto de partida para a criação do desenho arquitetural.

a) Projeto arquitetural (Visão de Análise)

1) Diagrama de Classes: As classes na disciplina de análise representam abstrações de uma ou mais classes (ou subsistemas) que surgem na disciplina de projeto e focam sempre os requisitos funcionais da aplicação. A especificação dessas classes é sempre em um alto nível de abstração, com uma linguagem próxima ao domínio do problema. As classes nos modelos de análise sempre estão associadas aos estereótipos *boundary* (modelam as classes associadas à interação entre o sistema e os atores), *entity* (modelam as classes do domínio do problema que são persistentes) e *control* (modelam as classes responsáveis pelo controle de um caso de uso específico) [JACOBSON 1998]. A Figura 4.4 apresenta o diagrama de classes participantes, na visão de análise, do caso de uso "Especificar curso a distância suportado pela CMC".

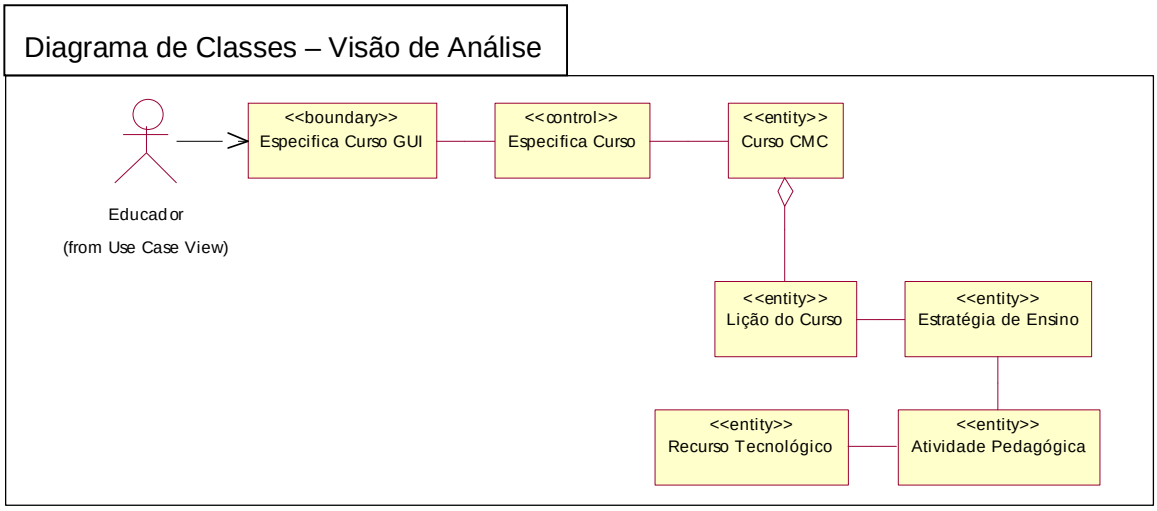


Figura 4-4 Diagrama de classes (visão de análise)

A seguir são descritas as classes apresentadas na Figura 4-4.

Classes de Análise

1) Especifica Curso GUI

   Estereótipo: *Boundary*

   Descrição: Encapsula a interface que o usuário utiliza para interagir com o sistema. Foi feita a opção de se desenvolver o *Projeto EaD* como uma aplicação para Web. As páginas HTML vistas a seguir apresentam a seqüência de passos que devem ser percorridos para especificar um curso a distância suportado pela CMC. A primeira página solicita que o educador forneça a descrição do curso, o público alvo e a forma de acesso (Figura 4.5). A segunda página exibe o detalhamento do curso, apresentando os objetivos e as lições que compõem o mesmo (Figura 4.6). Essa página disponibiliza *links* para que o educador adicione um novo objetivo ou uma nova lição ao curso. Após a criação de uma lição, é exibida a página que apresenta os objetivos/atividades pedagógicas da lição (Figura 4.7). Nesta página, o educador pode adicionar um novo objetivo (página semelhante a que permite adicionar um objetivo ao curso) ou adicionar uma nova atividade pedagógica à lição (exige que o educador selecione a atividade pedagógica e o recurso tecnológico utilizado para a sua efetivação - Figura 4.8).

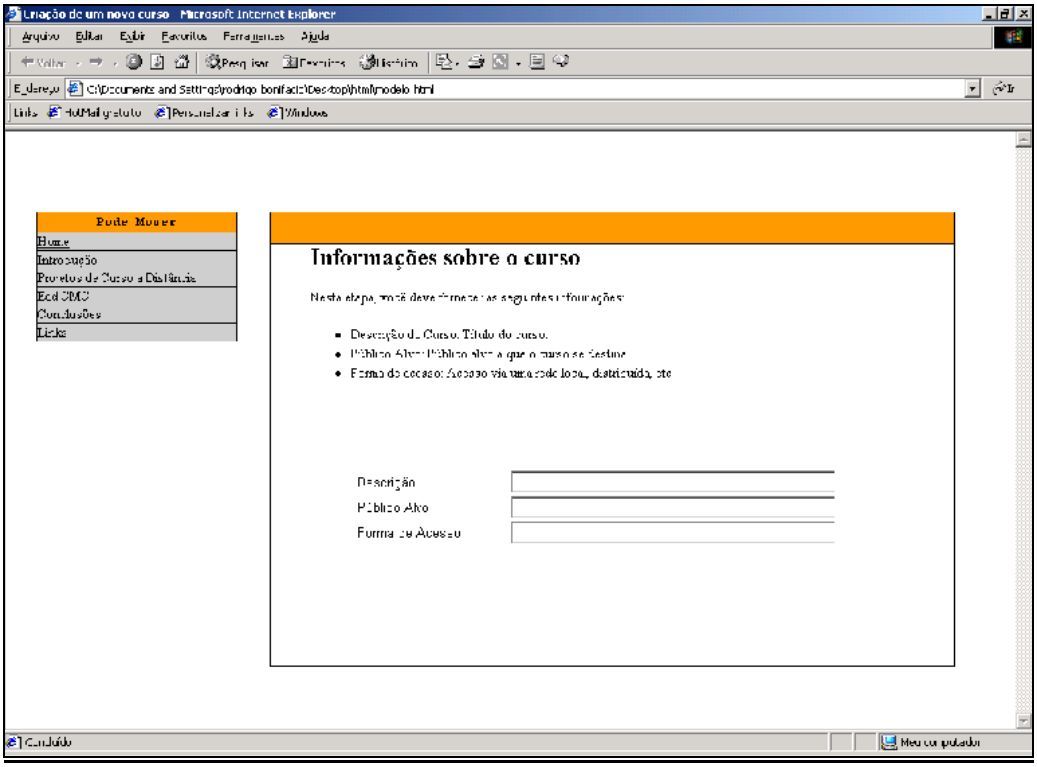


Figura 4-5 Página de informações sobre o curso

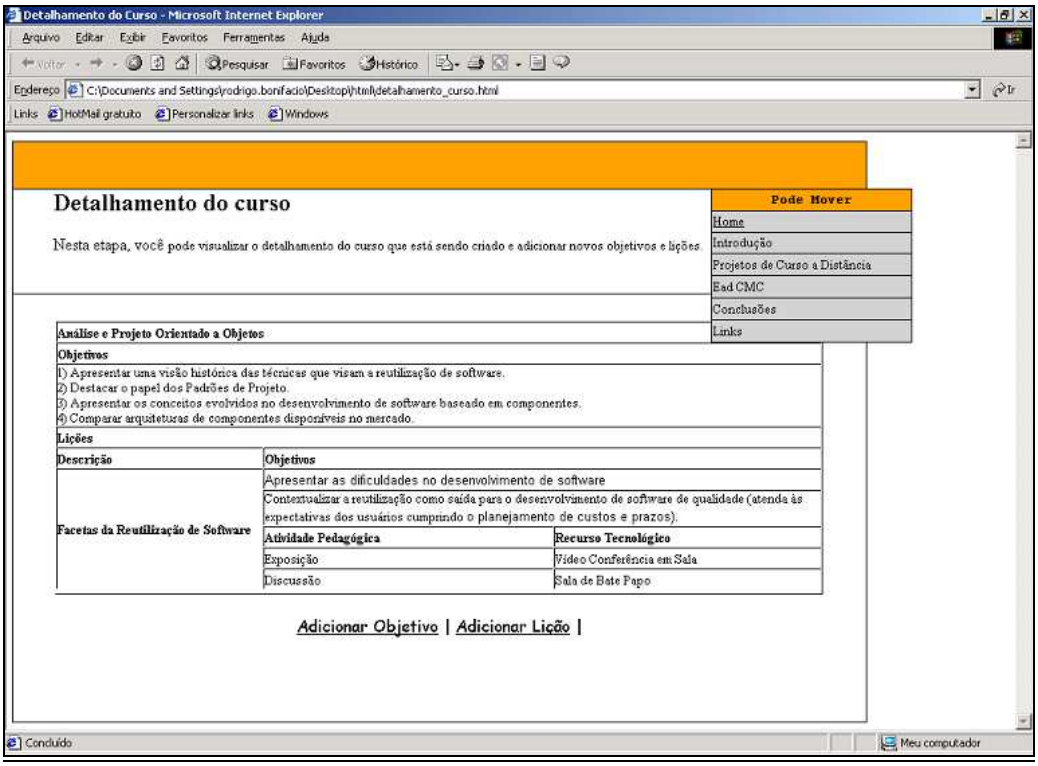


Figura 4-6 Página de detalhamento do curso

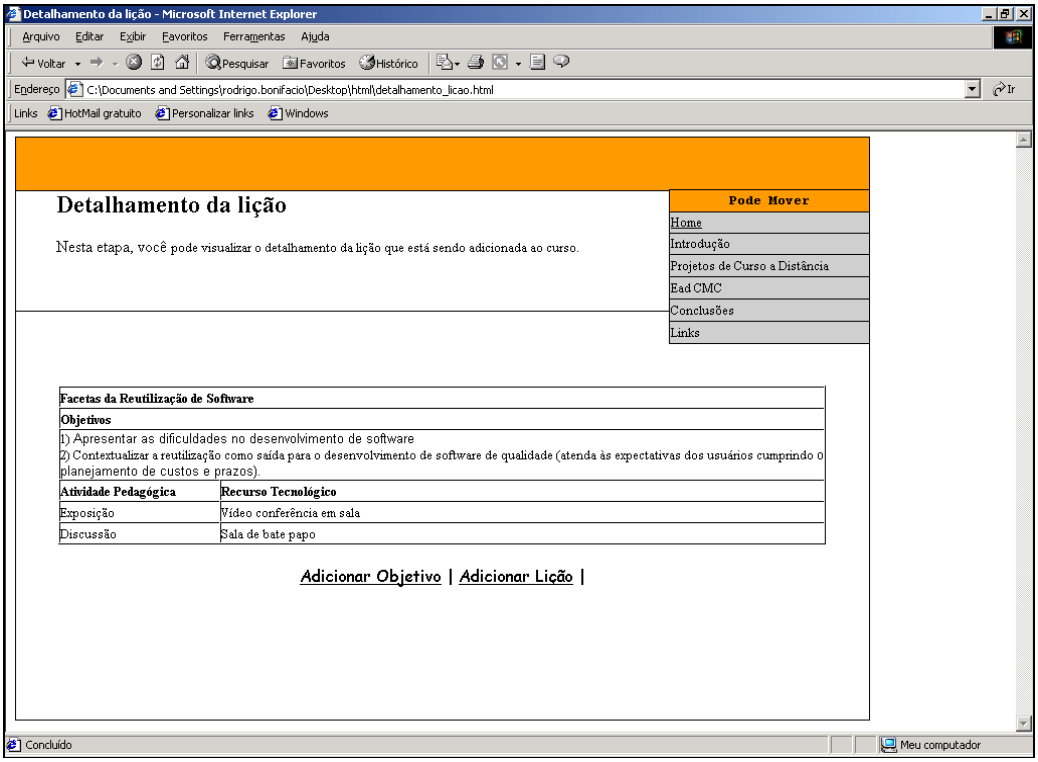


Figura 4-7 Página de detalhamento da lição

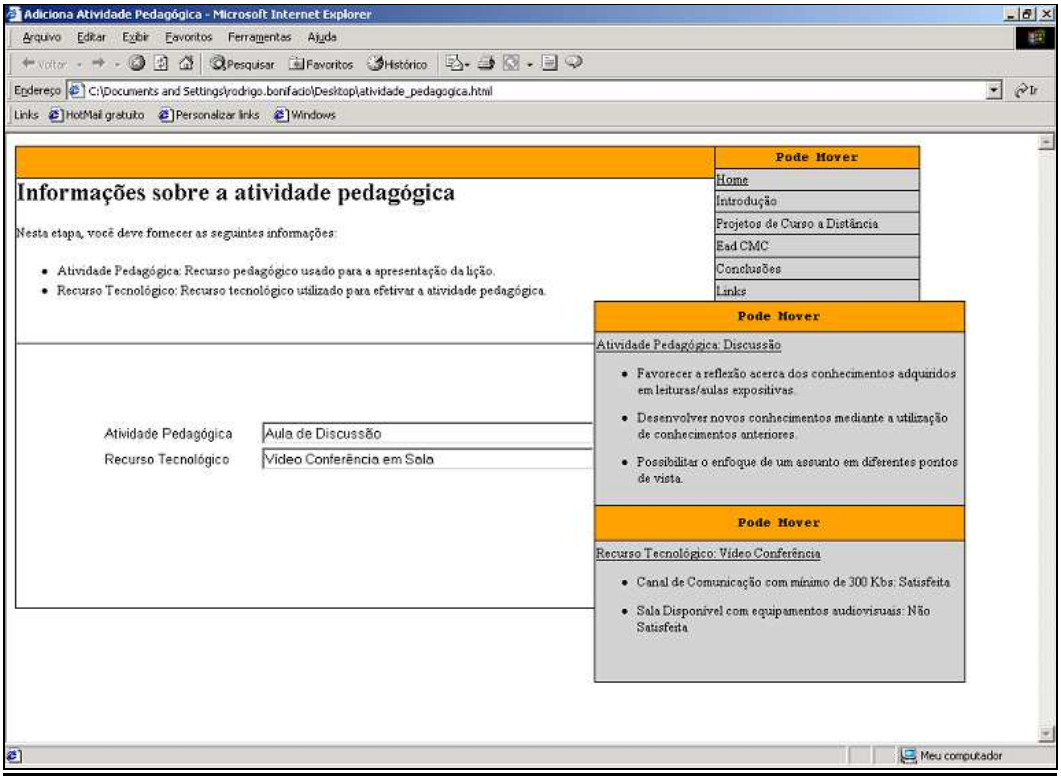


Figura 4-8 Página que adiciona atividade pedagógica

2) Especifica Curso

Estereótipo: *Control*

Descrição: Encapsula a lógica responsável por auxiliar os educadores na especificação de um curso suportado pela CMC. A partir da seqüência de passos necessária para a criação de um curso, esta classe fica responsável por adicionar novos objetivos/lições ao curso; consultar a base de dados para estabelecer os relacionamentos entre as atividades pedagógicas e os recursos tecnológicos; e gerar os arquivos de saída.

3) Curso CMC

Estereótipo: *Entity*

Descrição: Classe que representa um curso a distância suportado pela CMC. Um curso pode ser constituído por várias lições.

4) Lição do Curso

Estereótipo: *Entity*

Descrição: Classe que representa uma unidade básica de ensino de um curso. As lições são apresentadas aos alunos seguindo uma estratégia de ensino particular.

5) Estratégia de Ensino

Estereótipo: *Entity*

Descrição: Classe que representa o modo como as lições são apresentadas. Corresponde às associações entre uma atividade pedagógica e um recurso tecnológico, visando disponibilizar uma lição aos alunos.

6) Atividade Pedagógica

Estereótipo: *Entity*

Descrição: Classe que representa uma atividade pedagógica (aula expositiva, aula de discussão, exercícios individuais, etc.).

7) Recurso Tecnológico

Estereótipo: *Entity*

Descrição: Classe que representa um recurso tecnológico (hipertexto, correio eletrônico, vídeo conferência em sala, etc.) que pode ser utilizado para suportar uma atividade pedagógica.

b) Projeto arquitetural (Visão de Projeto)

1) Diagrama de Classes

Os diagramas na visão de projeto apresentam classes cujo nível de abstração está próximo à implementação. Além de uma visão mais detalhada das classes relacionadas ao domínio do problema, as classes responsáveis pela interface com o usuário (controles gráficos, formulários, páginas HTML), pela garantia da persistência das informações (acesso ao sistema de arquivos, banco de dados, documentos XML, etc.) e pelo cumprimento dos requisitos não funcionais (reusabilidade, flexibilidade, etc.) são descritas durante a etapa de projeto.

Para favorecer a flexibilidade e o reúso da aplicação; e agrupar os componentes presentes nos diagramas de classes da visão de projeto, foram criados as camadas[1] apresentadas na Figura 4-9:

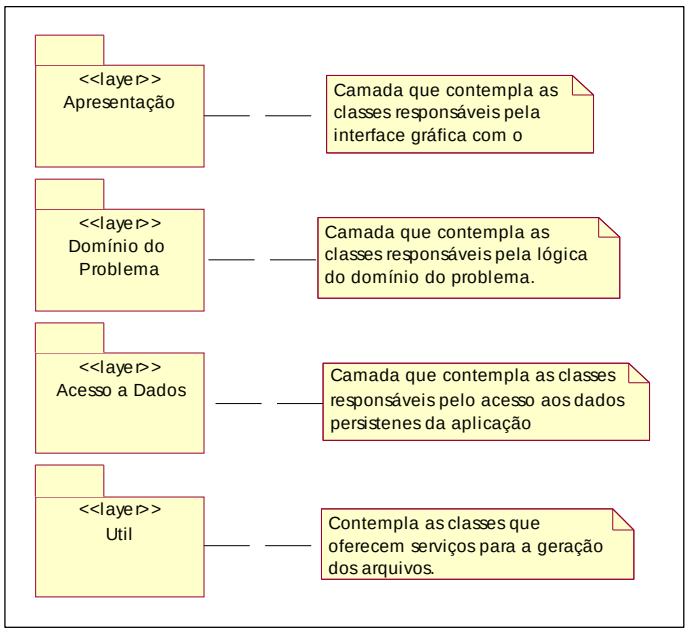


Figura 4-9 Organização do sistema em camadas

Camada do Domínio do Problema

[1] As camadas são modeladas como pacotes UML com estereótipo Layer.

Contempla as classes relacionadas com o domínio do problema e que encapsulam a lógica da aplicação.

Camada Domíno do Problema

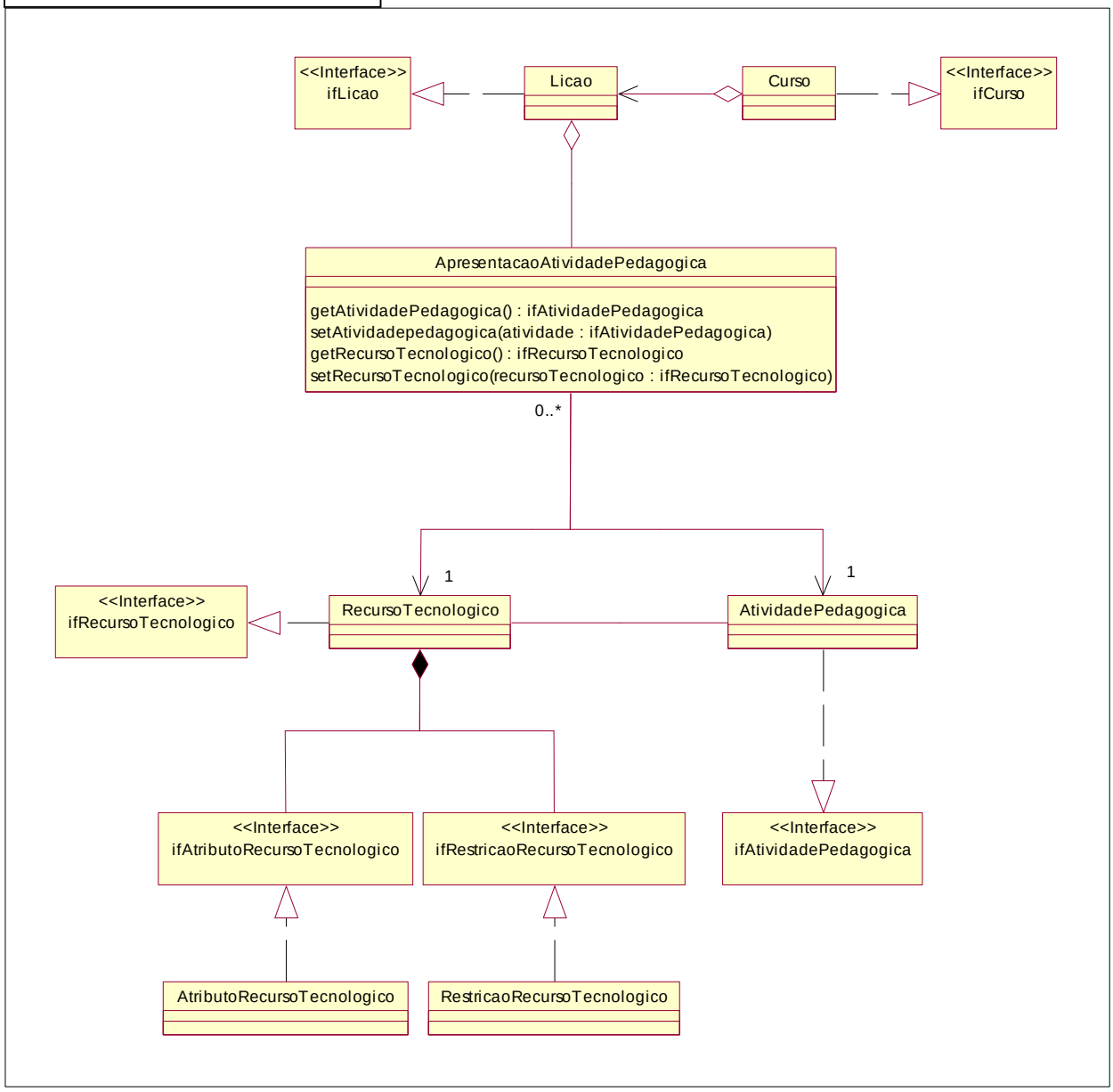


Figura 4-10 Diagrama de classes da camada Domínio do Problema

A Figura 4.10 apresenta o diagrama de classes da camada do domínio do problema. A abstração fundamental deste pacote é a classe `Curso`, que representa um curso a distância suportado pela CMC. Um curso possui uma descrição, um conjunto de objetivos e um conjunto de lições. Uma lição (representada pela classe `Licao`) corresponde a uma unidade de ensino. Por exemplo, em um curso sobre orientação a objetos, poderíamos ter uma lição que apresentasse os conceitos básicos da Orientação a Objetos (polimorfismo, herança, encapsulamento), uma outra lição que apresentasse técnicas de análise, uma outra que apresentasse técnicas de projeto e assim sucessivamente. Cada lição pode ser apresentada utilizando uma estratégia de ensino (seqüência de atividades pedagógicas – representada pela classe `ApresentacaoAtividadePedagogica`) que seja mais adequada ao processo de ensino-aprendizagem. Cada apresentação de atividade pedagógica corresponde à associação entre uma atividade pedagógica (aulas de exposição, aulas de laboratório, exercícios de fixação, etc.) e um recurso tecnológico (páginas HTML, chats, listas de discussão, videoconferência, etc.) utilizado para a sua efetivação. As atividades pedagógicas possuem uma descrição e um conjunto de objetivos que justificam a sua utilização. Exemplos de objetivos da atividade pedagógica Aula Discursiva seriam: "*Favorecer a reflexão sobre conhecimentos adquiridos em leituras/aulas expositivas*", "*Desenvolver novos conhecimentos mediante a utilização de conhecimentos anteriores*" e "*Possibilitar o enfoque de um assunto sob diferentes pontos de vista"*. Os recursos tecnológicos possuem uma descrição, um conjunto de atributos (pares nome-valor que qualificam um recurso tecnológico) e um conjunto de restrições para a sua utilização. A Tabela 4-1 apresenta exemplos de atributos/restrições do recurso tecnológico Vídeo Conferência em Sala.

Tabela 4-1 Atributos e restrições do recurso tecnológico vídeo conferência em sala

Vídeo Conferência em Sala		
Atributos	Nome	Valor
	Sincronismo	Síncrona
	Mídia	Multimídia
	Interação com os participantes	Alta
	Interação com o ambiente	Baixa
Restrições	Descrição	Satisfeita pela Instituição
	Canal de comunicação com mínimo de 300 kbps	Sim
	Sala disponível com equipamentos audiovisuais	Não

As interfaces presentes no diagrama da Figura 4.10 foram definidas no pacote Domínio do Problema/Interfaces (Figura 4.11).

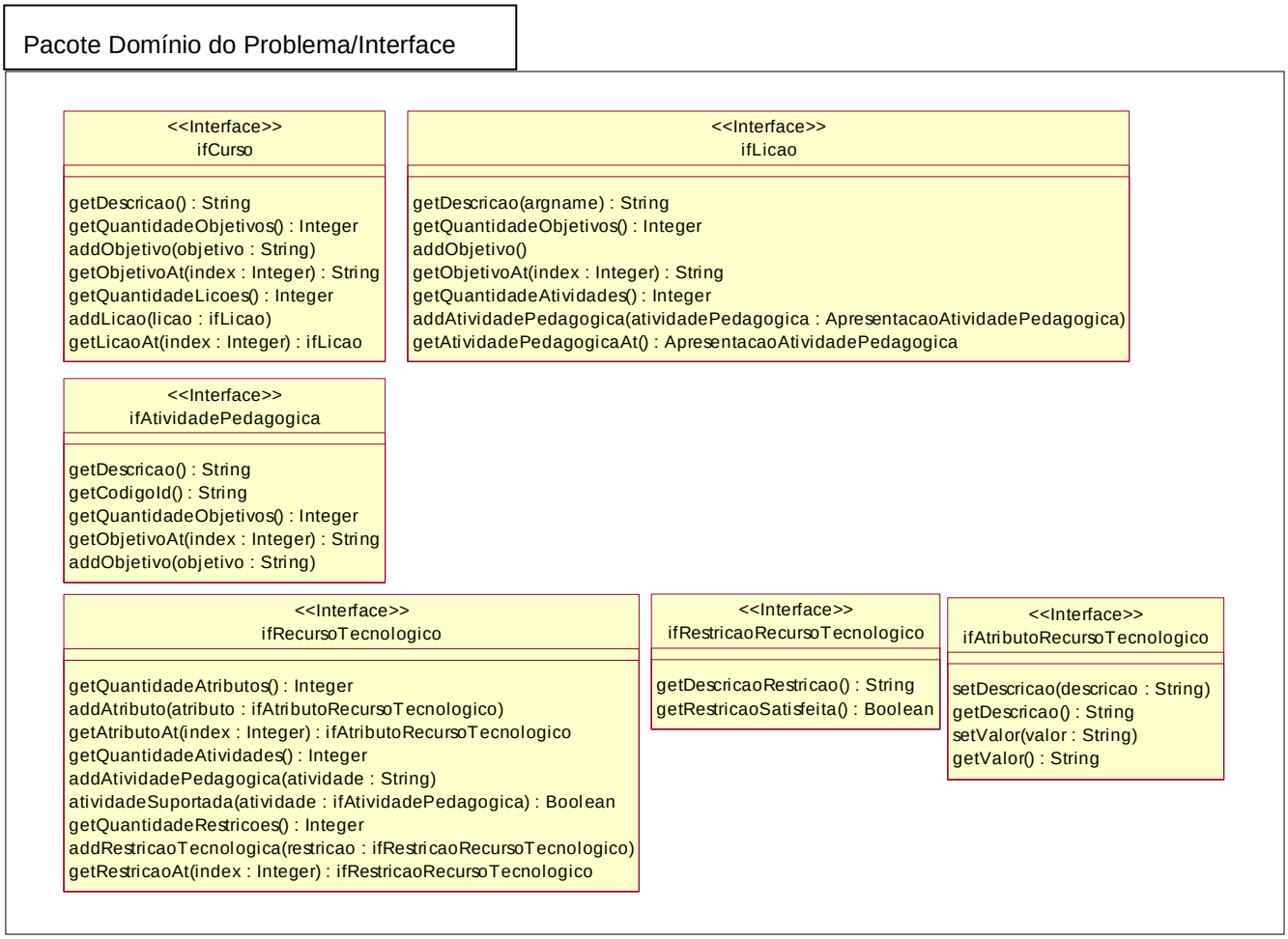


Figura 4-11 Diagrama de classes (pacote Domínio do Problema/Interfaces)

Uma interface corresponde ao conjunto de requisições que podem ser enviadas a um objeto. A definição das interfaces é importante pois, apesar de serem a única forma de acesso aos objetos, nenhum detalhe de implementação é fornecido. Ou seja, na aplicação foi definida a interface `ifCurso` como o conjunto de assinaturas de métodos a serem implementados pelos objetos de uma classe curso qualquer. O método `addLicao()`, da interface `ifCurso`, é utilizado para adicionar uma lição ao curso. Nenhum detalhe sobre as estruturas de dados utilizadas para armazenar as lições de um curso é apresentado. Dessa forma, poderíamos implementar uma classe `CursoArray` que utilizasse um arranjo para manter as lições de um curso; e, posteriormente, implementar uma classe `CursoHash` que utilizasse uma tabela hash

para a mesma finalidade. Apesar da mudança de implementação, desde que as classes `CursoArray` e `CursoHash` obedecessem a interface `ifCurso`, o código da aplicação não seria alterado.

Pacote Domínio do Problema/Interfaces

As interfaces que foram definidas no pacote Domínio do Problema/Interfaces podem ser vistas na Tabela 4-2:

Tabela 4-2 Interfaces do pacote Domínio do Problema/interfaces

ifCurso		
Método	Assinatura	
getDescricao	Retorna a descrição do curso.	
	Retorno	String contendo a descrição do curso.
getQuantidadeObjetivos	Retorna a quantidade de objetivos do curso.	
	Retorno	Inteiro contendo a quantidade de objetivos do curso.
AddObjetivo	Adiciona um objetivo ao curso.	
	Parâmetros	String contendo o objetivo a ser adicionado.
getObjetivoAt	Retorna o objetivo cujo índice é passado como parâmetro.	
	Parâmetros	Índice do objetivo desejado.
	Retorno	String contendo a descrição do objetivo desejado.
getQuantidadeLicoes	Retorna a quantidade de lições que compõem o curso.	
	Retorno	Inteiro contendo a quantidade de lições do curso.
AddLicao	Adiciona uma lição ao curso.	
	Parâmetros	Lição a ser adicionada.
getLicaoAt	Retorna a lição cujo índice é passado como parâmetro.	
	Parâmetros	Índice da lição desejada.
	Retorno	Lição cujo índice foi passado como parâmetro.

IfLicao		
Método		
getDescricao	Retorna a descrição da lição.	
	Retorno	String contendo a descrição da lição.
getQuantidadeObjetivos	Retorna a quantidade de objetivos da lição.	
	Retorno	Inteiro contendo a quantidade de objetivos da lição.
addObjetivo	Adiciona um objetivo à lição.	
	Parâmetro	String contendo a descrição do objetivo.
getObjetivoAt	Retorna o objetivo cujo índice é passado como parâmetro.	
	Parâmetros	Índice do objetivo desejado.
	Retorno	String contendo a descrição do objetivo desejado.
getQuantidadeAtividades	Retorna a quantidade de atividades pedagógicas da lição.	
	Retorno	Quantidade de atividades pedagógicas da lição.
addAtividadePedagogica	Adiciona uma apresentação de atividade pedagógica à lição.	
	Parâmetro	Apresentação de atividade pedag. a ser adicionada.
getAtividadeAt	Retorna a apresentação de atividade cujo índice é passado como parâmetro.	
	Parâmetros	Índice da atividade desejada.
	Retorno	Apresentação de atividade pedagógica cujo índice é passado como parâmetro.

IfAtividadePedagogica		
Método		
getDescricao	Retorna a descrição da atividade pedagógica.	
	Retorno	String contendo a descrição da atividade pedagógica.
getQuantidadeObjetivos	Retorna a quantidade de objetivos da atividade pedagógica.	
	Retorno	Inteiro contendo a quantidade de objetivos da atividade.
addObjetivo	Adiciona um objetivo à atividade pedagógica.	
	Parâmetro	String contendo a descrição do objetivo.
getObjetivoAt	Retorna o objetivo cujo índice é passado como parâmetro.	
	Parâmetros	Índice do objetivo desejado.
	Retorno	String contendo a descrição do objetivo desejado.

IfRecursoTecnologico		
Método		
getDescricao	Retorna a descrição do recurso tecnológico.	
	Retorno	String contendo a descrição do recurso tecnológico.
getQuantidadeAtributos	Retorna a quantidade de atributos do recurso tecnológico.	
	Retorno	Inteiro contendo a quantidade de atributos do recurso.
addAtributo	Adiciona um atributo ao recurso tecnológico.	
	Parâmetro	Atributo de recurso tecnológico a ser adicionado.
getAtributoAt	Retorna o atributo do recurso tecnológico cujo índice é passado como parâmetro.	
	Parâmetros	Índice do atributo desejado.
	Retorno	Atributo de recurso tecnológico desejado.

IfRecursoTecnologico (continuação)		
Método	Assinatura	
getQuantidadeAtividades	Retorna a quantidade de atividades pedagógicas suportado pelo recurso tecnológico.	
	Retorno	Quantidade de recursos tecnológicos suportados.
addAtividadePedagogica	Adiciona uma atividade pedagógica ao recurso tecnológico.	
	Parâmetro	Atividade pedagógica a ser adicionada.
atividadeSuportada	Verifica se uma atividade pedagógica é suportada pelo recurso tecnológico.	
	Parâmetros	Atividade pedagógica a ser analisada.
	Retorno	Verdadeiro: Atividade pedagógica suportada. Falso: Atividade pedagógica não suportada.
getQuantidadeRestricoes	Retorna a quantidade de restrições do recurso tecnológico.	
	Retorno	Quantidade de restrições do recurso tecnológico.
addRestricaoTecnologica	Adiciona uma restrição ao recurso tecnológico.	
	Parâmetro	Restrição do recurso tecnológico a ser adicionada.
getRestricaoAt	Retorna a restrição tecnológica cujo índice é passado como parâmetro.	
	Parâmetros	Índice da restrição tecnológica desejada.
	Retorno	Restrição do recurso tecnológico desejado.

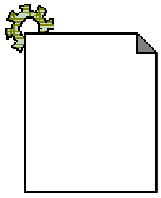
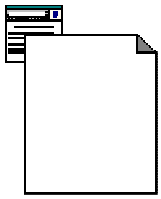
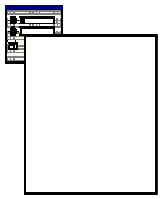
ifRestricaoRecursoTecnologico		
Método	Assinatura	
getDescricaoRestricao	Retorna a descrição da restrição do recurso tecnológico.	
	Retorno	String contendo a descrição da restrição tecnológica.
restricaoSatisfeita	Retorna um valor lógico indicando se a restrição é satisfeita ou não pela instituição.	
	Retorno	Verdadeiro: Restrição satisfeita Falso: Restrição não satisfeita

ifAtributoRecursoTecnologico		
Método	Assinatura	
getDescricao	Retorna a descrição do atributo.	
	Retorno	String contendo a descrição do atributo.
setDescricao	Altera a descrição do atributo.	
	Parâmetros	String contendo a descrição do atributo.
getValor	Retorna o valor do atributo.	
	Retorno	String contendo o valor do atributo.
setDescricao	Altera o valor do atributo.	
	Parâmetros	String contendo o valor do atributo.

Camada de Apresentação

Contempla as classes responsáveis pela interface com o usuário. Como o padrão UML não foi proposto para atender perfeitamente a todos os tipos de aplicações, faz-se necessário a utilização de extensões para adaptá-lo a essas situações. Conceitos como páginas HTML, formulários HTML, hiperlinks, etc., que não estão definidos no padrão UML, precisam ser modelados nas aplicações Web. Dessa forma, os diagramas de classes desta camada utilizam a Extensão UML para Aplicações Web (*Web Application Extension - WAE*) apresentada em [CONALLEN 2000]. A Tabela 4-3 apresenta os estereótipos propostos na WAE e que são utilizados no pacote interface com usuário.

Tabela 4-3 Estereótipos da WAE

Estereótipo	Descrição	Figura
Server Page	Representa um recurso do servidor que, quando requisitado, retorna uma página HTML construída dinamicamente para o usuário. Essas classes são as responsáveis por receber as requisições dos usuários, submetidas via formulários, hiperlinks, etc., e por interagir com as classes de acesso aos dados, lógica de negócio, aplicações externas, etc. Servlets, CGIs, código JSP, ASP, etc. são exemplos de conceitos que podem ser modelados com este estereótipo.	
Client Page	Utilizado para representar as páginas HTML. Páginas HTML são uma mistura de dados, formatação e, eventualmente, scripts que podem ser interpretados pelo navegadores Web. Páginas cliente podem ter associações com outras páginas (tanto do lado cliente quanto do lado servidor).	
Form	Uma classe representada com o estereótipo `<<form>>` corresponde a um conjunto de campos de entrada de dados que fazem parte de uma página cliente (estão relacionados via composição com uma página cliente). Uma classe `<<form>>` é mapeada diretamente com o marcador HTML *form* e seus atributos representam os campos de entrada (área de texto, caixas de checagem, botões, etc.).	
Link	Estereótipo utilizado nas associações entre uma página cliente e uma outra página (cliente ou servidora). Corresponde aos *links* HTML.	`<<link>>`
Submit	Estereótipo utilizado nas associações entre uma classe <<form>> e uma classe <<server page>>. A associação <<submit>> corresponde às requisições feitas aos servidores quando os usuários submetem (enviam) as informações do formulário para o servidor.	`<<submit>>`
Build	Estereótipo utilizado nas associações entre uma classe <<server page>> e uma classe <<client page>>. As páginas servidoras existem apenas no lado servidor, sendo usadas para a criação das páginas clientes.	`<<build>>`

A camada de apresentação é composta por dois pacotes: o pacote *servlets*, que descreve as classes responsáveis pela geração das páginas HTML necessárias para a especificação de um curso; e o pacote *controles HTML*, que descreve os formulários e controles visuais utilizados para o envio das informações dos usuários para o servidor HTTP. Será utilizada uma abordagem *bottom-up* para descrever a camada de apresentação. Inicialmente serão descritos os diagramas de classes dos pacotes *servlets* e *controles HTML* para, só então, descrever o diagrama de classe que mostra as associações entre os elementos dos mesmos.

Pacotes Apresentação/Controles HTML

Contempla as classes que definem os formulários e os controles HTML utilizados na interface com os usuários. A Figura 4.12 apresenta o diagrama de classes do pacote Controles HTML

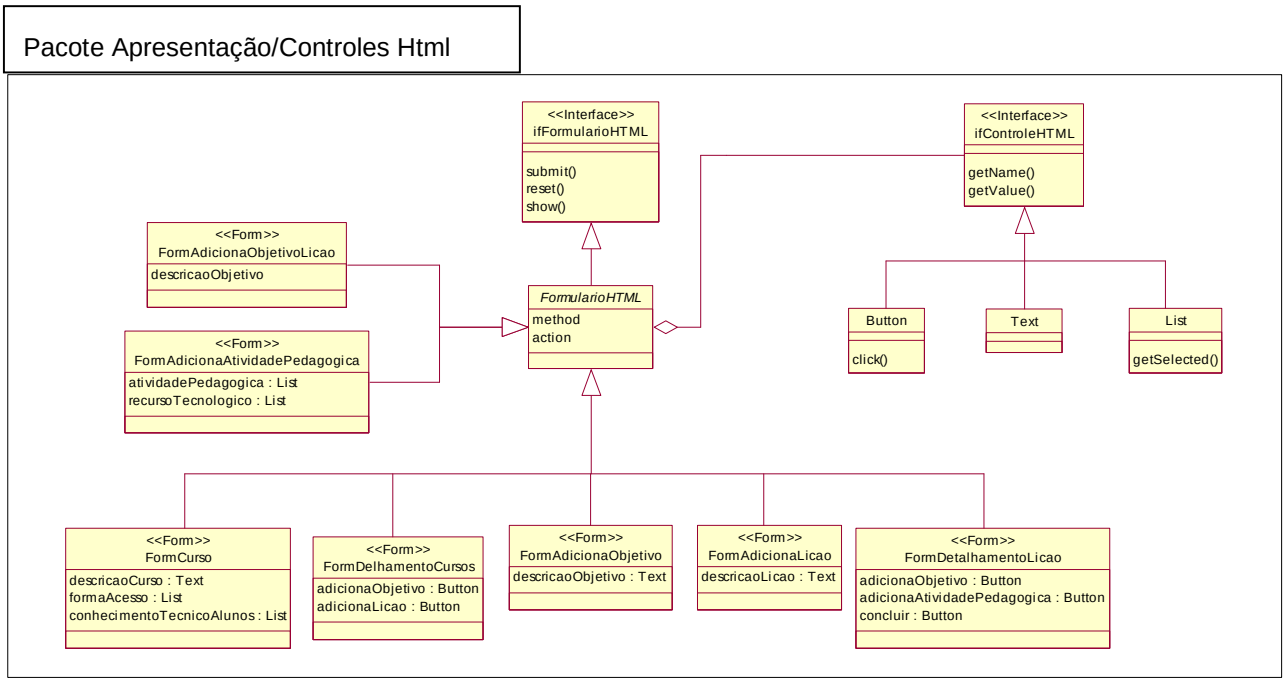


Figura 4-12 Diagrama de classes (pacote Apresentação/Controles HTML)

Neste pacote são apresentadas as interfaces *ifControleHTML* e *ifFormularioHTML*. A interface *ifControleHTML* define os métodos *getName()* e *getValue()* utilizados para acessar, respectivamente, o nome e o valor dos controles HTML. Os controles HTML são utilizados para a entrada e envio dos dados dos usuários. Nessa aplicação, foi necessária a utilização dos controles *Button*, *Text* e *List*, todos implementando a interface *ifControleHTML*. Além dos métodos *getName()* e *getValue()*, cuja assinatura foi definida na interface *ifControleHTML*, o controle *Button* possui o método *click()*, disparado quando o usuário dá um clique no botão; e o controle *List* possui o método *getSelected()* que retorna o índice do elemento selecionado.

Todo formulário HTML, utilizado na aplicação, implementa os métodos definidos na interface *ifFormularioHTML*. Formulários HTML são utilizados para enviar informações aos servidores HTTP; estes, por sua vez, ficam responsáveis por processar as informações recebidas e retornar uma página, construída dinamicamente, para o usuário que fez a requisição.

A interface *ifFormularioHTML* define os métodos *submit()*, *reset()* e *show()* que são utilizados, respectivamente, para enviar as informações ao servidor HTTP, limpar os dados dos controles e exibir o formulário. A classe abstrata *FormularioHTML* encapsula os atributos *method* (define o método de requisição como `Get` ou `Post`) e *action* (define a URL a ser chamada quando o formulário é submetido) e estabelece o relacionamento de composição com a interface *ifControleHTML* (indicando que um formulário é constituído por vários controles HTML). Os formulários apresentados na Tabela 4-4 são utilizados para o projeto de um curso a distância suportado pela CMC:

Tabela 4-4 Formulários HTML utilizados

FormCurso Formulário que permite aos pedagogos fornecerem as informações sobre descrição do curso, forma de acesso e perfil dos alunos.		
Atributo method	Post	
Atributo action	http://localhost/servlet/ServletCurso	
Controles	descricao	Controle tipo Text com a descrição do curso.
	formaAcesso	Controle tipo List com a forma de acesso do curso.
	perfilAluno	Controle tipo Text com o perfil dos alunos.

FormAdicionaObjetivo Formulário que permite aos pedagogos adicionarem um objetivo ao curso.		
Atributo method	Post	
Atributo action	http://localhost/servlet/ServletCurso	
Controles	descricaoObjetivo	Controle tipo Text com a descrição do objetivo.

FormAdicionaLicao Formulário que permite aos pedagogos adicionarem uma lição ao curso.		
Atributo method	Post	
Atributo action	http://localhost/servlet/ServletLicao	
Controles	descricaoLicao	Controle tipo Text com a descrição da Lição.

FormAdicionaAtividadePedagogica Formulário que permite aos pedagogos adicionarem uma atividade pedagógica à lição.		
Atributo method	Post	
Atributo action	http://localhost/servlet/ServletLicao	
Controles	atividadePedagogica	Controle tipo List com as atividades pedagógicas que podem ser utilizadas nas lições de um curso suportado pela CMC.
	recursoTecnologico	Controle tipo List com os recursos tecnológicos que podem ser utilizados para a apresentação da atividade pedagógica selecionada.

FormAdicionaObjetivoLicao Formulário que permite aos pedagogos adicionarem um objetivo à lição.		
Atributo method	Post	
Atributo action	http://localhost/servlet/ServletLicao	
Controles	descricaoObjetivo	Controle tipo Text com a descrição do objetivo da Lição.

Pacotes Apresentação/Servlets

Contempla as classes responsáveis pela geração dinâmica das páginas HTML. Um *servlet* é uma classe Java que pode ser carregada dinamicamente para estender as funcionalidades de um servidor qualquer (apesar de ser mais utilizado para adaptar o serviço HTTP) [HUNTER 2000].

O serviço HTTP apresenta a característica de não preservar o estado da conexão entre as sucessivas requisições dos usuários. Ou seja, a conexão é encerrada sempre que uma solicitações ao serviço HTTP (um acesso a uma página HTML, por exemplo) é atendida. Tecnologias como *Servlets*, *Common Gateway Interface*, *Active Server Page*, etc. são utilizadas para gerar páginas HTML dinâmicas (seguindo informações enviadas pelos usuários ou

recuperadas de um banco de dados) e preservar o estado de uma conexão HTTP entre as requisições.

A Sun disponibiliza um conjunto de classes que oferecem suporte à tecnologia *Servlets*. A classe abstrata `HttpServlet` (pacote `javax.servlet.http`), por exemplo, serve como um *framework* para o desenvolvimento de *servlets* HTTP. Se quisermos ampliar as funcionalidades desse serviço, basta estendermos esta classe reimplementando alguns dos seus métodos. A Figura 4.13 apresenta o diagrama das classes *servlets* da aplicação.

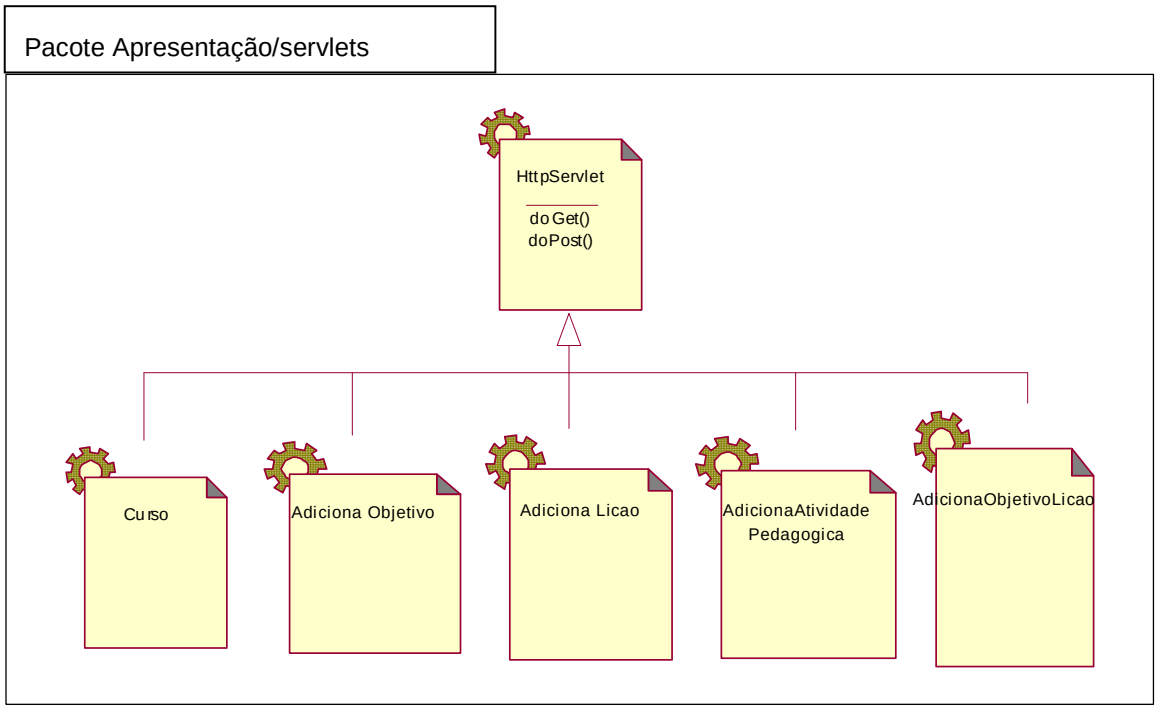


Figura 4-13 Diagrama de classes (pacote Apresentação/Servlets)

O diagrama da Figura 4.13 apresenta a classe abstrata `HTTPServlet`, que define um conjunto de métodos (além do `doGet()` e `doPost()` presentes na figura) que são chamados em respostas às requisições HTTP. O método `doGet()` é associado às requisições HTTP do tipo GET; enquanto que o método `doPost()` é associado às requisições do tipo POST.

As demais classes servlets herdam (*estendem* no mundo Java) as definições (atributos e métodos) da classe `HTTPServlet` e reimplementam o método `doPost()` para a geração das páginas HTML em resposta às informações enviadas pelos usuários.

O estado entre as requisições é preservado utilizando a interface `HTTPSession`. Um servlet obtém um objeto `HTTPSession` através do método `getSession()` do objeto `req` (passado como parâmetro nos métodos `doGet()` e `doPost()`).

A interface `HTTPSession` define os métodos `putValue()` e `getValue()` que podem ser utilizados, respectivamente, para adicionar e recuperar um objeto cujo status deve persistir entre as requisições. A Tabela 4-5 apresenta a descrição dos servlets utilizados nessa aplicação.

Tabela 4-5 Descrição das classes Servlets

Servlet	Curso
Ativação	i. Submissão do formulário Curso.
	ii. Submissão do formulário Adiciona Objetivo.
	iii. Clique no botão btnConcluido (página detalhamento da licao).
Tarefas	Ativação i.: Adiciona uma instância da classe curso à sessão do usuário. Ativação ii. Adiciona um objetivo ao curso.
	Exibe a página com o detalhamento (objetivos /lições/atividades pedagógicas) do curso.
Servlet	AdcionaObjetivo
Ativação	i. Clique no botão btnAdicionaObjetivo (página detalhamento curso).
Tarefas	Exibe a página que permite adicionar um objetivo ao curso.
Servlet	AdcionaLicao
Ativação	i. Clique no botão btnAdicionaLicao (página detalhamento curso).
Tarefas	Exibe a página que permite adicionar uma lição ao curso.
Servlet	Licao
Ativação	i. Submissão do formulário Adiciona Lição ii. Submissão do formulário Adiciona Objetivo da Lição. iii. Submissão do formulário Adiciona Atividade Pedagógica.
Tarefas	Ativação i.: Adiciona a lição ao curso. Ativação ii.: Adiciona o objetivo à lição. Ativação iii.: Adiciona a atividade pedagógica à lição.
	Exibe a página com o detalhamento (objetivo/atividades pedagógicas) da lição.

A Figura 4.14 apresenta o diagrama de classes que mostra as associações existentes entre os elementos dos pacotes `Controles HTML` e `Servlets`.

Camada de Apresentação

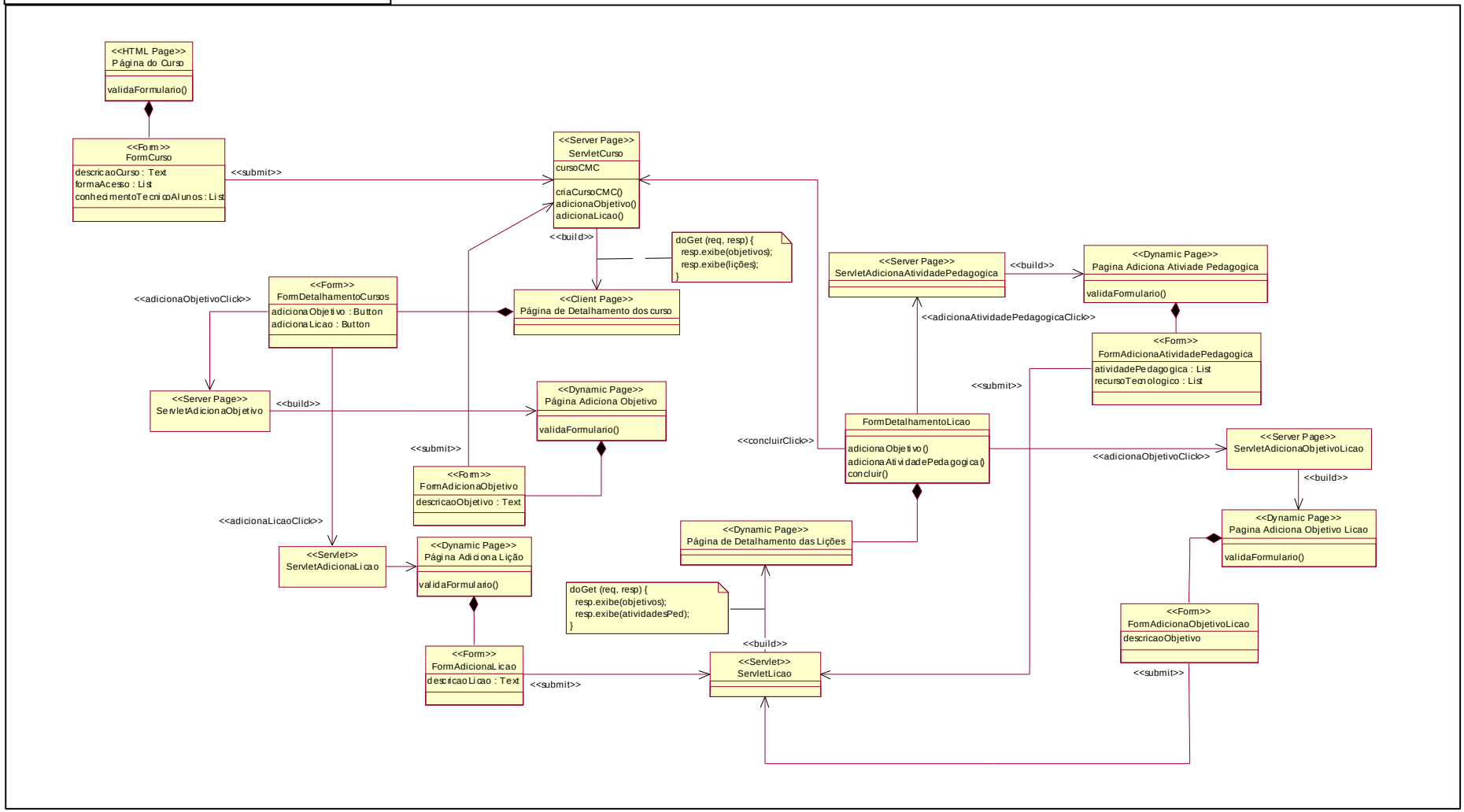


Figura 4-14 Diagrama de classes (Camada de Apresentação)

Camada de Acesso a Dados

Encapsula as classes utilizadas para carregar e armazenar as informações que estabelecem o relacionamento entre as atividades pedagógicas e os recursos tecnológicos. O diagrama de classes exibido na Figura 4.15 define o pacote Acesso a Dados/XML.

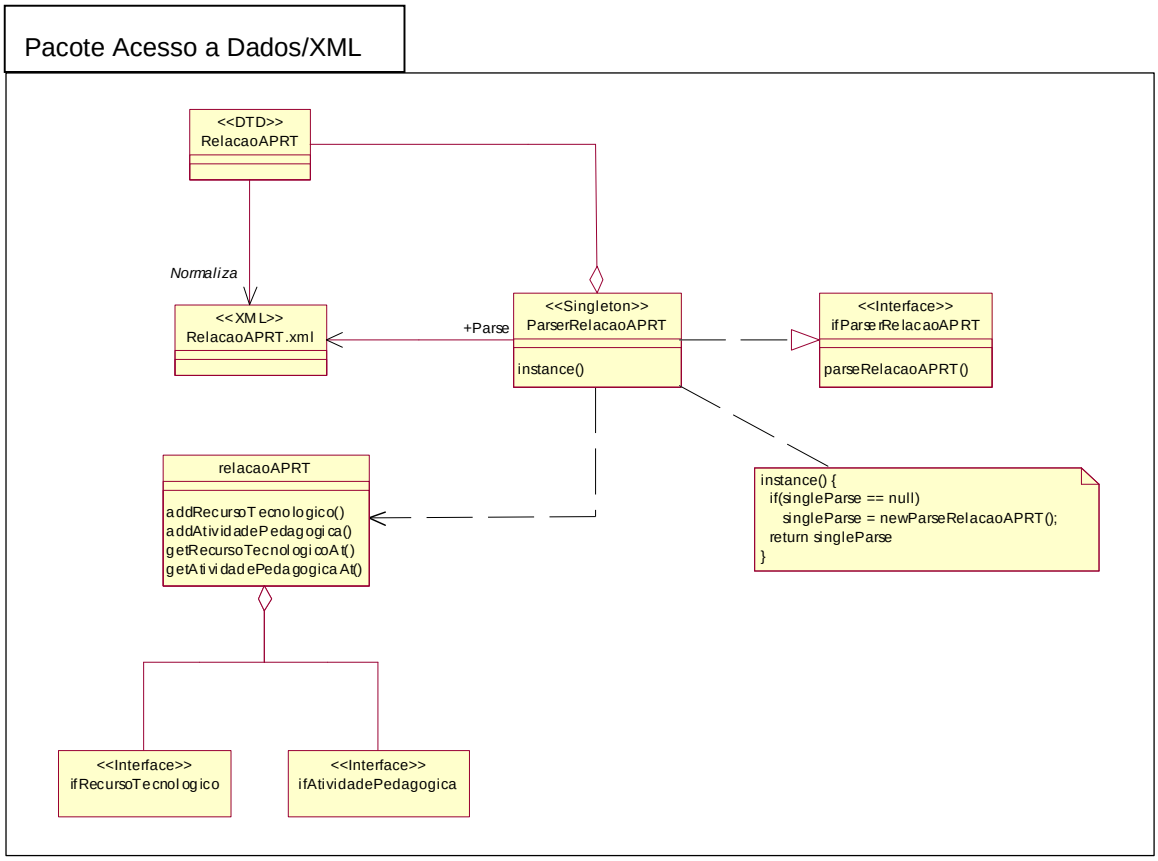


Figura 4-15- Diagrama de classes (pacote Acesso a Dados/XML)

Neste diagrama são apresentadas as classes `DTDRelacaoAPRT` (estereótipo DTD – *Data Type Definition*), que define as regras gramaticais dos arquivos `XML RelacaoAPRT` (representados pela classe `XMLRelacaoAPRT`)[2]. A interface `ifParserRelacaoAPRT` define o método `parseRelacaoAPRT()` que processa um arquivo `XML RelacaoAPRT` e retorna uma instância da classe `RelacaoAPRT`. A classe `ParserRelacaoAPRT` é responsável pela implementação do método `parseRelacaoAPRT()`. O estereótipo

[2] Para maiores informações sobre o arquivo XML Relação APRT e a gramática associada, consultar o capítulo 3.

`singleton` (padrão de projeto *Singleton*) indica que existe uma única instância da classe `ParseRelacaoAPRT` acessada através do método `instance()`. A classe RelacaoAPRT possui um conjunto de métodos para adicionar e recuperar as atividades pedagógicas/recursos tecnológicos presentes no arquivo processado.

Camada Útil

Define as classes responsáveis pela geração do resultado da especificação de um curso nos formatos HTML, XML ou LaTeX. Foi feita a opção por encapsular cada um dos algoritmos de geração dos arquivos em classes distintas, seguindo o padrão de projeto *Estrategy* [GAMMA et. al 1994]. A proposta de se implementar diferentes algoritmos em classes distintas apresenta as seguintes vantagens:

- Eliminar estruturas condicionais que tornam o código menos legível;
- Permitir que uma nova estratégia possa ser utilizada sem modificar o código cliente (classe curso);
- Favorecer a utilização de diferentes alternativas de implementação de um algoritmo em tempo de execução.

A Figura 4-16 apresenta o diagrama de classes da Camada Útil.

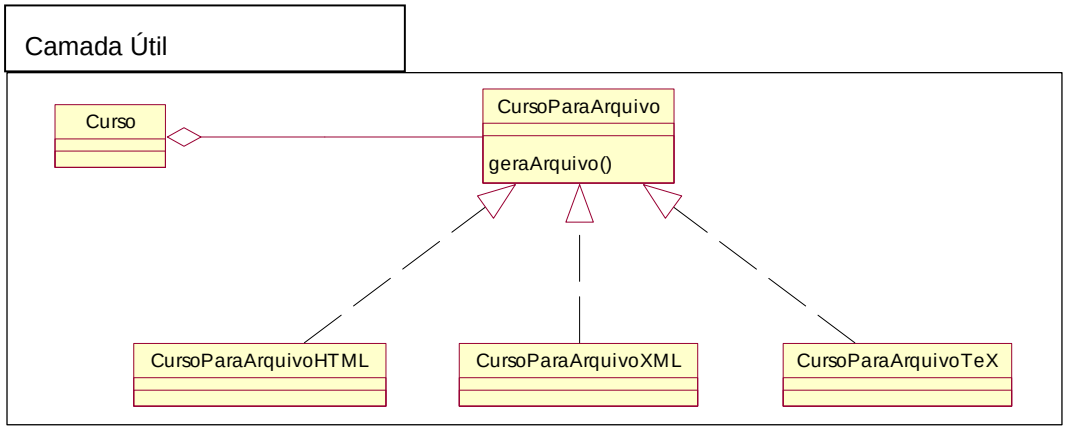


Figura 4-16 Diagrama de classes (camada util)

c) Projeto arquitetural (Visão de Implementação)

Os modelos que surgem na visão de implementação contemplam os componentes (arquivos fonte, arquivos executáveis, bibliotecas de ligação dinâmica, etc.) que são construídos a partir da especificação resultante das atividades de projeto.

O ambiente tecnológico utilizado para o desenvolvimento do *Projeto EaD* consistiu de um servidor HTTP (no caso o *Internet Information Service* da Microsoft), *o JRun Server* e o *Java Development Kit*. O *JRun Server* é um container que se conecta ao servidor HTTP interceptando alguns tipos de requisições (requisições aos *Servlets*, requisições aos documentos JSP, etc.).

O mapeamento entre os diagramas da linguagem UML, utilizados na visão de projeto, e as estruturas suportadas pela linguagem Java é bastante direto. A Tabela 4-6 apresenta o mapeamento das estruturas utilizadas na fase de projeto e as estruturas utilizadas na fase de implementação.

Tabela 4-6 Mapeamento entre as estruturas UML e as estruturas Java

Estrutura da visão de projeto UML	Estrutura de visão de Implementação Java	Sintaxe Java
Pacotes UML	Pacotes Java	`package` *`nome_pacote`*
Interfaces UML	Interfaces Java	`interface` *`nome_interface`*
Classes UML	Classes Java	`class` *`nome_clsse`*
Relacionamento de realização (classe implementa interface)	Relacionamento de implementação (classe implementa interface)	`implemnts` *`interface`*
Relacionamento de herança (classe herda as definições de outra classe)	Relacionamento de herança (classe herda as definições de outra classe)	`extends` *`classe`*
Relacionamento de composição (classe faz parte de outra classe)	Relacionamento de composição (membro de uma classe é instância de outra)	*`classe nomeAtributo`*

A Figura 4-17 apresenta como as camadas da visão de projeto (Figura 4-9) foram mapeados em pacotes Java.

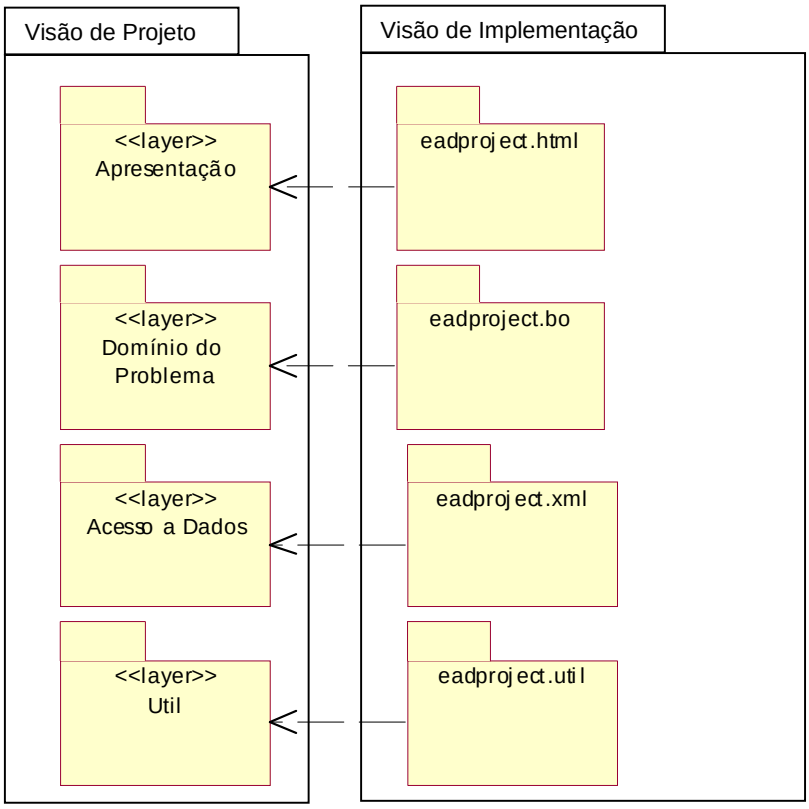


Figura 4-17 Mapeamento entre os pacotes da Visão de Projeto e os pacotes da Visão de Implementação

Os pacotes na linguagem Java correspondem a agrupamentos de classes e interfaces inter-relacionadas, sendo utilizados para favorecer o controle de acesso e evitar conflito de nomes. Para indicar que uma classe ou interface é membro de um pacote, coloca-se o *statement* `package` no início do arquivo fonte no qual a mesma é definida. Uma classe Java é definida em um arquivo cujo nome corresponde ao nome da classe (para as classes públicas isto é obrigatório. Para as demais classes, isto é desejável) e que possui a extensão "*.java*". Os arquivos fonte devem ser salvos em uma estrutura de diretórios que corresponda ao pacote do qual a classe faz parte. Ou seja, a classe Curso, pertencente ao pacote `eadproject.bo`, deve ser salva em um arquivo `Curso.java` no diretório `eadproject/bo`.

A Figura 4-18 apresenta o mapeamento das classes presentes na camada Domínio do Problema (Figura 4-10) para as classes Java do pacote `eadproject.bo` (*business objects*).

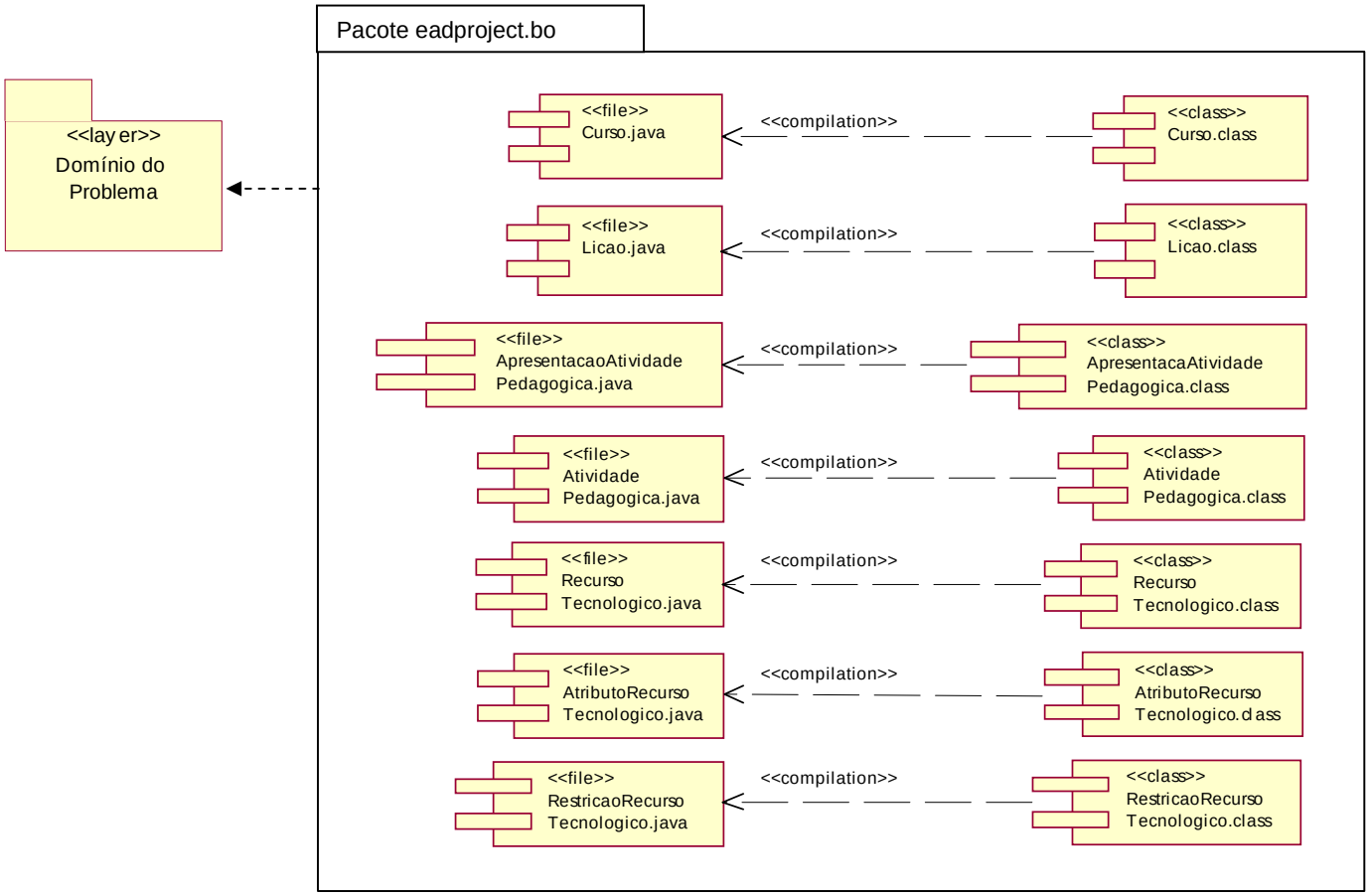


Figura 4-18 Mapeamento (Classes de Projeto x Componentes de Implementação)

A dependência `<<compilation>>` corresponde ao processo de compilação dos arquivos "*.java*" para gerar os arquivos de código de bytes (*bytecode*) correspondentes. Esses arquivos de código de bytes são os responsáveis pela independência de plataforma da tecnologia Java. Uma vez gerados os arquivos de código de bytes, eles podem ser interpretados em qualquer ambiente que possua uma Máquina Virtual Java [ARNOLD 1998]; [HORSTMANN 1999].

As Figuras 4-19 e 4-20 apresentam, respectivamente, o código fonte da interface `IfCurso` (definida no pacote Lógica de Negócio/Interfaces – visão de projeto) e da classe `Curso` (definida no pacote Lógica do Negócio – visão de projeto).

```
package eadproject.bo.interfsce;

/**
 * Interface que define os métodos a serem implementados pelas classes que
 * representam um curso a distância suportado pela CMC.
 * @date: Setembro de 2001
 * @author: Rodrigo Bonifácio de Almeida
 */
public interface IFCurso {

  /**
   * Retorna uma String que corresponde à descrição do curso.
   * @return Descrição do curso.
   */
  public String getDescricao();

  /**
   * Retorna uma String que corresponde ao público alvo a que o curso se destina.
   * @return Público alvo a que o curso se destina.
   */
  public String getPublicoAlvo();

  /**
   * Retorna uma String que corresponde à forma de acesso ao curso.
   * @return Forma de acesso ao curso.
   */
  public String getFormaAcesso();

  public int getQuantidadeLicoes();

  /**
   * Retorna uma Licao que corresponde à Licao cujo índice foi passado como parâmetro.
   * @param index Índice da Lição desejado.
   * @return Lição cujo índice foi passado como parâmetro.
   */
  public String getLicaoAt(int index) throws IndexOfBoundsException;

  /**
   * Adiciona uma Lição ao curso.
   * @param Licao Lição a ser adicionada.
   */
  public void addLicao(IFLicao licao);

}
```

Figura 4-19 Código fonte da interface IfCurso

A interface IFCurso define as assinaturas de métodos que devem ser implementados pelas classes que representam um curso a distância suportado pela CMC. O código `package eadprojecto.bo.interface` indica que a interface `IFCurso` pertence ao pacote `eadproject.bo.interface`. A linha `public interface IfCurso` é responsável por declarar a interface. As demais linhas de código presentes na Figura 4-19 correspondem aos comentários (quando delimitadas pelos caracteres /* */) e às definições dos métodos.

```
package eadcmc.bo;

import java.util.List;
import java.util.ArrayList;

import eadcmc.bo.interface.*;

/**
 * Classe que representa um curso a distância suportado pela CMC.
 * @date: Setembro de 2001
 * @author: Rodrigo Bonifácio de Almeida
 */
public class Curso implements IFCurso {
  /** Descrição do curso */
  private String descricao;
  /** Público alvo */
  private String publicoAlvo;
  /** Forma de acesso */
  private String formaAcesso;

  /** Quantidade de lições */
  private int quantidadeLicoes;

  /** Lista utilizada para manter as lições do curso */
  private List licoesCurso;

  /**
   * Construtor da classe Curso.
   * @param descricao Descrição do curso
   * @publicoAlvo Público alvo
   * @formaAcesso Forma de acesso
   */
  public Curso(String descricao, String publicoAlvo, String formaAcesso) {
    this.descricao = descricao;
    this.publicoAlvo = publicoAlvo;
    this.formaAcesso = formaAcesso;
    this.licoesCurso = new ArrayList();
    this.quantidadeLicoes = 0;
  }

 /**
   * Retorna uma String que corresponde à descrição do curso.
   * @return Descrição do curso.
   */
  public String getDescricao() {
    return descricao;
  }

 /**
   * Retorna uma String que corresponde ao público alvo a que o curso se destina.
   * @return Público alvo a que o curso se destina.
   */
  public String getPublicoAlvo() {
    return publicoAlvo;
  }

 /**
   * Retorna uma String que corresponde à forma de acesso ao curso.
   * @return Forma de acesso ao curso.
   */
  public String getFormaAcesso() {
    return formaAcesso;
  }
```

```
  /**
   * Retorna um inteiro que corresponde a quantidade de lições do curso.
   * @return Quantidade de lições do curso.
   */
  public int getQuantidadeLicoes() {
    return quantidadeLicoes;
  }

 /**
   * Retorna a Licao que corresponde à Licao cujo índice foi passado como parâmetro.
   * @param index Índice da Licao desejado.
   * @return Licao cujo índice foi passado como parâmeto.
   */
 public IfLicao getLicaoAt(int index) throws IndexOfBoundsException {
    if (index < quantidadeLicoes)
      return (IFLicao)licoesCurso[index];
    throw new IndexOutOfBoundsException("Índice da lição fora da faixa");
 }

 /**
  * Adiciona uma Licao ao curso.
  * @param Licao Licao a ser adicionada.
  */
 public void addLicao(IfLicao Licao) {
    quantidadeLicoes++;
    licoesCurso.add(licao);
  }

}
```

Figura 4-20 Código fonte da classe Curso

A classe Curso representa um curso a distância suportado pela CMC. O código

"*`public class Curso implements IFCurso`*" é responsável pela declaração da classe Curso e indica ao compilador que a mesma deve implementar todos os métodos definidas na interface IFCurso.

Com exceção do pacote Interface com Usuário, os demais pacotes pertencentes à visão de projeto foram mapeados de forma análoga ao pacote Lógica do Negócio. Ao invés de mapear as classes da visão de projeto em classes Java, as classes do pacote Interface com Usuário seguiram as regras mostradas na Tabela 4-7:

Tabela 4-7 Mapeamento entre as estruturas do pacote Interface com o Usuário e as estruturas Java

Visão de Projeto	Visão de Implementação
Classes com estereótipo <<`Form`>>	Formulários HTML presentes nos arquivos JSP
Classes com estereótipo <<`Server Page`>>	Código Java presente nos arquivos JSP
Classes com estereótipo <<`Client Page`>>	Código HTML presente nos arquivos JSP

A Figura 4-21 apresenta o mapeamento entre o pacote Interface com Usuário (visão de projeto) e o pacote `eadproject.html`.

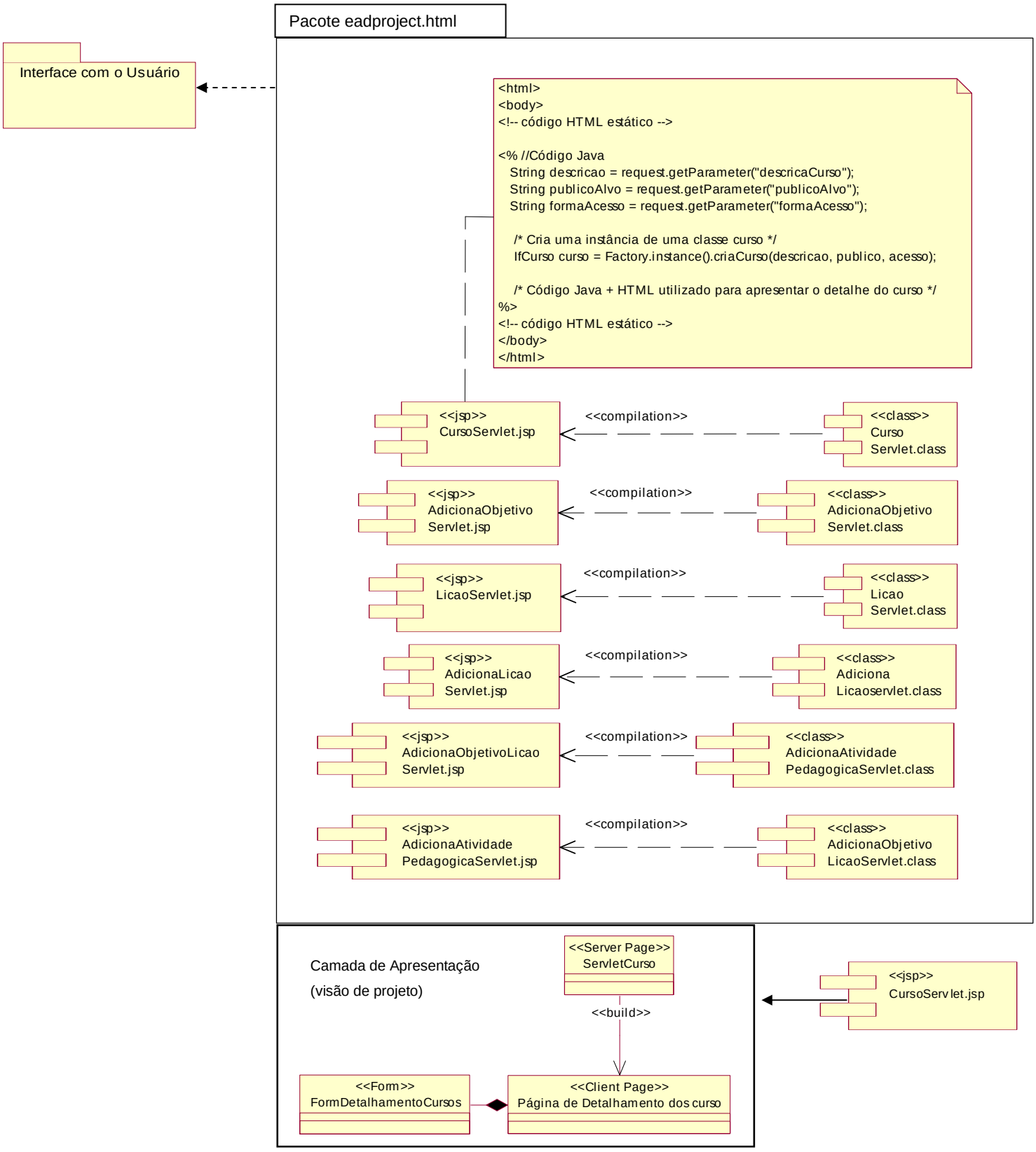


Figura 4-21 Mapeamento da Camada de Apresentação/Estruturas Java

A Figura 4-21 mostra que as classes da camada de apresentação foram mapeadas em páginas JSP (*Java Server Page*). Mais ainda, mostra que um único documento JSP pode ser utilizado para implementar várias classes da fase de projeto (diferente do mapeamento *um-para-um* utilizado nos demais pacotes).

Os documentos JSP são arquivos no formato texto que combinam dados estáticos, conhecidos como *template data*, expressos nos formatos HTML, WML, XML, etc. e dados dinâmicos escritos utilizando a linguagem Java. Quando é feita a primeira requisição a um arquivo JSP, o container *JRun* executa um *parser* no documento e gera um código fonte Java. Em seguida é feita a compilação deste arquivo e uma classe Servlet é gerada, carregada e executada. Os seguintes passos são realizados quando um documento JSP é requisitado pela primeira vez:

1. É realizado um *parser* no documento JSP produzindo um código fonte Java;
2. O código fonte Java é compilado em uma classe Servlet;
3. A classe Servlet é carregada na memória do servidor HTTP;
4. O Servlet é executado.

Nas requisições subseqüentes, se o arquivo JSP não tiver sido modificado após a geração da classe Servlet, os passos 1 e 2 não são executados. No melhor caso, se a classe Servlet estiver carregada na memória do servidor, apenas o último passo é realizado.

4.1.3 Artefatos da Fase de Construção

Os artefatos que surgem na fase de construção correspondem aos modelos de análise, projeto, implementação e testes das funcionalidades que não fizeram parte da arquitetura do sistema. Entre essas funcionalidades estão os casos de uso Manter Cadastro de recursos Pedagógicos/Tecnológicos, Estabelecer Relacionamentos entre Recursos Pedagógicos e Recursos Tecnológicos e Manter a Integridade do Sistema.

Apesar de serem importantes para facilitar a utilização/configuração do *Projeto EaD* esses casos de uso não apresentam a mesma importância do caso de uso Especificar Curso a Distância (seriam basicamente entradas de dados cadastrais que iriam alterar as bases de dados) que é o foco da dissertação. Os mesmos podem ser implementados como um trabalho futuro da dissertação.

4.1.4 Artefatos da Fase de Transição

Na fase de Transição o sistema é colocado em produção, testado e integrado com os demais sistemas com os quais o mesmo deve colaborar. O *Projeto EaD* deve ser inserido em um ambiente que interaja com sistemas que auxiliem o educador nas seguintes tarefas:

- Especificação dos aspectos educacionais como missão do curso, lições necessárias para atingir os objetivos, perfil dos alunos, etc.
- Criação, disponibilização e administração dos conteúdos didáticos aos quais os estudantes têm acesso.

Outra atividade que poderia ser realizada na fase de Transição do *EaD Project* consiste na validação, junto aos educadores, da melhoria no processo de ensino-aprendizagem obtida com o uso da ferramenta. Essas tarefas também estão sugeridas como trabalhos futuros.

4.2 Conclusões

Este capítulo apresentou as fases de desenvolvimento do *Projeto EaD*, cujo objetivo é auxiliar os educadores, não especialistas em informática, no projeto de um curso a distância suportado pela CMC. Foi seguido o Processo Unificado de Desenvolvimento de Software (RUP - *Rational Unified Software*) que é constituído pelas fases de Iniciação, Elaboração, Construção e Transição. As duas primeiras fases foram concluídas (por estarem mais alinhadas com os objetivos desta dissertação). As demais foram sugeridas como trabalhos futuros.

O foco desta versão do *Projeto EaD* está no projeto da infraestrutura necessária para disponibilizar cursos em um ambiente de CMC. Um passo anterior, que consiste em auxiliar os educadores na especificação da missão do curso, do público alvo, das métricas de qualidade, etc. fará parte de um outro trabalho.

Bibliografia

[ARNOLD 1998] ARNOLD, K., GOSLING, HOLMES, The Java Programming Language, Addison-Wesley, 1998.

[BERGSTEN 2000] BERGSTEN, H., JavaServer Pages, O'Reilly, 2000.

[CONALLEN 2000] CONALLEN, J., Modeling Web-Tier Components, Software Development Magazine, Janeiro de 2001, **on-line:** http://www.sdmagazine.com

[GAMMA et. al 1994] GAMMA, E., HELM, JOHNSON, VLISSIDES, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, 1994.

[HORSTMANN 1999] HORSTMANN, C., CORNELL, Core Java 2, Volume 1: Fundamentals, Prentice-Hall, 1999

[HUNTER 2000] HUNTER, J., CRAWFORD, Java Servlet Programming, O'Reilly, 2000.

[JACOBSON 1992] JACOBSON, I., CHRISTERSON., JONSSON., ÖVERGAARD, , Object Oriented Software Engineering: A Use Case Driven Approach, Addison-Wesley, 1992.

[JACOBSON 1998] JACOBSON, I., BOOCH, RUMBAUGH, The Unified Software Development Process, Addison-Wesley, 1998

[KRUCHTEN 1998] KRUCHTEN, P., The Rational Unified Process: An Introduction, Addison-Wesley, 1998.

[POLLICE 2000] POLLICE, G., Using The Rational Unified Process for Small Projects: Expanding Upon eXtreme Programming, Rational Software White Paper, 2000 **on-line**: http://www.rational.com/whitepapers

[POMPILHO 1995] POMPILHO, S., Análise Essencial - Guia Prático de Análise de Sistemas, Livraria e Editora Infobook S/A, 1995.

[ROGERS 1997] ROGERS, G., Framework-Based Software Development in C++, Prentice-Hall, 1997.

[SZYPERSKI 1998] SZYPERSKI, C., Component Software: Beyond Object-Oriented Programming, Addison-Wesley, 1998.

5 Conclusões

O uso da Comunicação Mediada por Computador vem se mostrando como uma alternativa promissora para a efetivação do ensino a distância. Através da CMC é possível cobrir o importante aspecto da interação entre os participantes de um curso a distância; aspecto este deixado de lado pelos outros meios utilizados para prover o EaD.

Durante o projeto de um curso a distância suportado pela CMC, faz-se necessária a avaliação de aspectos interdisciplinares (aspectos pedagógicos, psicológicos, sociais e tecnológicos). Partindo do pressuposto que a utilização de ferramentas para auxiliar os educadores durante as etapas de projeto/disponibilização de um curso a distância poderiam melhorar o processo de ensino aprendizagem, esta dissertação apresentou o *Projeto EaD*, uma aplicação Web voltada para facilitar a tomada de decisão sobre quais componentes pedagógicos/tecnológicos devem ser utilizados para apresentar as lições de um curso.

O *Projeto EaD* se enquadra em uma fase intermediária do ciclo de desenvolvimento de um curso a distância suportado pela CMC. Antes da utilização do Projeto EaD, o educador já deve ter definido a missão do curso, as lições necessárias para atingir os objetivos, o público alvo, etc. Após essas variáveis terem sido bem delimitadas, o Projeto EaD pode ser utilizado para, a partir da infra-estrutura presente na instituição e da natureza do curso (definida na fase anterior), facilitar a seleção dos componentes pedagógicos/tecnológicos a serem utilizados. Uma fase posterior à utilização do *Projeto EaD* consiste na utilização de ferramentas como o e-Group ou o AulaNet que facilitam a geração, a disponibilização e a administração dos conteúdos didáticos apresentados aos alunos.

As seguintes contribuições são frutos desse trabalho:

- Levantamento bibliográfico sobre o estado da arte no projeto de cursos a distância suportado pela CMC. Neste aspecto foi destacado o processo interdisciplinar envolvido durante a especificação de um curso a distância; traçado o perfil dos estudantes/educadores; explicitado a necessidade de estratégias de ensino que favoreçam a participação ativa dos estudantes no processo de ensino-aprendizagem; e proposto um conjunto de critérios (sincronismo, mídia, interação, custos, etc.) para classificar as tecnologias de CMC.

- Especificação e implementação de um esquema de dados para estabelecer os relacionamentos entre os componentes pedagógicos e os componentes tecnológicos de um curso a distância suportado pela CMC. Para criar uma camada de abstração, foram propostos esquemas de dados para o nível conceitual (utilizando o modelo entidade relacionamento) e para o nível físico (utilizando a tecnologia XML). A utilização da tecnologia XML é justificada pela flexibilidade que a mesma apresenta. Com a adoção dessa solução, os atributos que qualificam os componentes pedagógicos/recursos tecnológicos podem ser facilmente modificados. Isto é importante para adaptar o *Projeto EaD* às diferentes realidades das instituições de ensino.
- Especificação e implementação do *Projeto EaD*, uma ferramenta para auxiliar os educadores no projeto de um curso a distância suportado pela CMC. O *Projeto EaD* foi desenvolvido seguindo a Processo Unificado de Desenvolvimento de Software. Neste processo são definidas as fases de Iniciação, Elaboração, Construção e Transição. Os artefatos construídos nas duas primeiras fases foram apresentados. Eles são responsáveis por delimitar o sistema descrevendo as funcionalidades em termos de casos de uso (fase de Iniciação) e por seguir o ciclo de desenvolvimento (análise, projeto, implementação e testes) para os casos de uso mais significativos e que apresentam maiores riscos de desenvolvimento (fase de Elaboração).

Como proposta de trabalhos futuros podemos destacar:

- Concluir as funcionalidades que não fizeram parte da arquitetura do *Projeto EaD*. Essas funcionalidades correspondem aos casos de uso que facilitam a manutenção das bases de dados e a configuração do sistema.
- Realizar testes de unidade do Projeto EaD. Os testes de unidade correspondem a atividade de avaliar se as funcionalidades dos componentes executáveis (classes Java, bibliotecas de ligação dinâmica, arquivos executáveis, etc.) estão de acordo com os requisitos da aplicação.
- Realizar testes de integração com ferramentas que cobrem as etapas de elaboração didática do curso e criação/disponibilização dos conteúdos didáticos. Esses testes de integração são importantes porque quanto mais suave (se

possível até transparente) for a transição entre as ferramentas, melhores serão os resultados.

- Realizar testes de utilidade para avaliar se o uso do *Projeto EaD*, em conjunto com ferramentas voltadas para as demais fases do ciclo de desenvolvimento, resulta em melhorias no processo de ensino aprendizagem.

Referências Bibliográficas

[AMBLER 2000] AMBLER, S., "*XML in The Real World*", Thinking Objectively, Software Development, setembro de 2000. **on-line:** http://www.sdmagazine.com/documents/sdm0009h/

[AMES 97] AMES, A., "*The VRML 2.0 Source Book*", John Wiley & Sons, Inc. 1997.

[ARNOLD 1998] ARNOLD, K., GOSLING, HOLMES, The Java Programming Language, Addison-Wesley, 1998.

[BERGSTEN 2000] BERGSTEN, H., JavaServer Pages, O'Reilly, 2000.

[BOURRET 2000] BOURRET, R., "*Declaring Elements and Attributes in an XML DTD*", 2000, **on-line:** http://www.rpbourret.com/xml/xmldtd.htm

[BRAY et al 2000] BRAY, T., PAOLI, MALER, "*Extensible Markup Language*", W3C Recommendation, 2000. **on-line:** http://www.w3.org/XML/

[BUCK 2001] BUCK, L., "*Modeling Relational Data in XML* ", Extensibility Report, 2001. **on-line:** http://www.extensibility.com/main_modeling.htm

[CARLOS 97] CARLOS, A., "*Metodologia do Ensino Superior*", São Paulo, São Paulo: Atlas, 1997.

[CASEY 98] CASEY, D., "*Learning From or Though the Web: Models of Web Based Education*", In: Integrating Technology into Computer Science Education – ACM, Proceedings… p. 51-54, 1998.

[CONALLEN 2000] CONALLEN, J., Modeling Web-Tier Components, Software Development Magazine, Janeiro de 2001, **on-line:** http://www.sdmagazine.com

[ELMASRI 2000] ELMASRI, R., SHAMKANT, "*Fundamentals Of Database Systems*", Addison-Wesley, 2000.

[GAMMA et. al 1994] GAMMA, E., HELM, JOHNSON, VLISSIDES, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, 1994.

[HARTLEY et al 96] HARTLEY, S., "*Enhancig Teaching Using The Internet*", In.: Integrating Technology into Computer Science Education –ACM. Proceedings… p. 218-228, 1996.

[HORSTMANN 1999]	HORSTMANN, C., CORNELL, Core Java 2, Volume 1: Fundamentals, Prentice-Hall, 1999
[HUNTER 2000]	HUNTER, J., CRAWFORD, Java Servlet Programming, O'Reilly, 2000.
[JACOBSON 1992]	JACOBSON, I., CHRISTERSON., JONSSON., ÖVERGAARD, , Object Oriented Software Engineering: A Use Case Driven Approach, Addison-Wesley, 1992.
[JACOBSON 1998]	JACOBSON, I., BOOCH, RUMBAUGH, The Unified Software Development Process, Addison-Wesley, 1998
[KRUCHTEN 1998]	KRUCHTEN, P., The Rational Unified Process: An Introduction, Addison-Wesley, 1998.
[LAWHEAD et al 97]	LAWHEAD, P., ALPERT, CARSWEL, CIZMAR, DEWITT, FAHRAEUS, SCOTT, "*The Web and Distance Learning: What is Appropriate and What is Not*". In.: Integrating Technology into Computer Science Education –ACM. Proceedings… p. 27-37, 1997.
[LUCKESY 95]	LUCKESY, C., "Avaliação da Aprendizagem: Um Ato Amoroso", In: Avaliação da Aprendizagem Escolar, p. 168-180. São Paulo, São Paulo: Cortez, 1995.
[MENDES 2000]	MENDES, F., "*E-group: Um Ambiente para Suporte à Aprendizagem Colaborativa Baseada na Web*", Dissertação de Mestrado, UFPB/DSC, 2000.
[MORLEY 2000]	Morley, J., "*Falling Through the Web, Inequality to access in Distance Education*", Education at Distance Magazine, 01-2000. **on-line:** http://www.usdla.org/15_publications.htm
[OTSUKA 97]	OTSUKA, J., "*Fatores Determinantes na Efetividade de Ferramentas de Comunicação Mediada Por Computador no Ensino à Distância***"**, Trabalho Individual, Pós-Graduação em Ciência da Computação, Instituto de Informática, UFRGS, 1997.
[PALLOF et al 99]	PALLOF, R., PRATT, "*Building Learning Communities in Cyberspace, Effective Strategies for The Online Classroom*", Jossey-Bass, 1999.
[PFEIFFER 2000]	PFEIFFER, R., "*XML: Tutorials for Programmers*", IBM-Online Courses, 2000. **on-line:** http://www-106.ibm.com/developerworks/xml/
[POLLICE 2000]	POLLICE, G., Using The Rational Unified Process for Small Projects: Expanding Upon eXtreme Programming, Rational Software White Paper, 2000 **on-line**: http://www.rational.com/whitepapers

[POMPILHO 1995] POMPILHO, S., Análise Essencial - Guia Prático de Análise de Sistemas, Livraria e Editora Infobook S/A, 1995.

[ROGERS 1997] ROGERS, G., Framework-Based Software Development in C++, Prentice-Hall, 1997.

[SCHIEL 98] SCHIEL, U., "*Elementos de Sistemas de Informações e Banco de Dados*", monografia, UFPB/DSC, 1998.

[SZYPERSKI 1998] SZYPERSKI, C., Component Software: Beyond Object-Oriented Programming, Addison-Wesley, 1998.

Apêndice A
O Processo Unificado de Desenvolvimento de Software

Introdução

O aumento da competitividade gerado pela globalização, os avanços dos recursos computacionais e a utilização da Internet para a troca de informações dos mais diferentes tipos exigem uma demanda por sistemas de software cada vez mais complexos. Se por um lado a sofisticação dos requisitos aumenta, os usuários das aplicações requerem prazos cada vez menores para a disponibilização de novas ferramentas ou atualização de aplicações já existentes. Para atender a essas expectativas, conceitos e tecnologias como orientação a objetos, objetos distribuídos, padrões de projeto, desenvolvimento baseado em componentes e *frameworks* [GAMMA et. al 1994][ROGERS 1997][SZYPERSKI 1998] estão se tornando cada vez mais evidentes. Além disso, se faz necessário a adoção de novas metodologias (processos) para o desenvolvimento de software. Um processo de desenvolvimento identifica *quem faz o que*, *como* e *quando* para transformar os requisitos de uma aplicação em um produto que satisfaça as expectativas (atuais e futuras) dos usuários, de uma forma que sejam minimizados os custos e o tempo de desenvolvimento.

O processo unificado é um *framework* que foi concebido em paralelo com o desenvolvimento da *Unified Model Language* (UML), e pode ser estendido para se adaptar ao desenvolvimento de aplicações de diferentes domínios.

Ou seja, o UP[1] propõe que seja seguido um conjunto de fases (Iniciação, Elaboração, Construção e Transição), disponibilizando, através de diagramas UML, artefatos que, dependendo da complexidade e do domínio da aplicação, podem ou não ser construídos.

[1] Unified Process.

Histórico do Processo Unificado

No início dos anos noventa, ficou evidente a necessidade de se desenvolver uma linguagem de modelagem padrão para o desenvolvimento de software orientado a objetos. Esse esforço unificou as metodologias *Booch*, *Object Modeling Technique* e *Objectory*, propostas respectivamente por Grady Booch, James Rumbaugh e Ivar Jacobson (conhecidos como "*Os três amigos"*); e incluiu propostas de desenvolvimento de empresas como a IBM, a HP e a Microsoft. Em novembro de 1997, a versão 1.1 da *Unified Model Language* foi promulgada padrão pelo *Object Management Group*.

Nessa época, "Os três amigos" estavam trabalhando na Rational Software Corporation, que adotou a UML em todos os modelos do seu processo de desenvolvimento (Rational Objectory Process- ROP). Através de fusões e parcerias com empresas especialistas em gerência de requisitos (*Requisite Inc.*), desenvolvimento de casos de testes (*SQA Inc.*), engenharia de dados (*Vigortech*), etc.; o ROP passou a cobrir aspectos de todo o ciclo de desenvolvimento, unificando uma grande variedade de contribuições. Em junho de 1998, foi disponibilizada uma nova versão do seu produto, dessa vez intitulado Rational Unified Process, que se tornou disponível ao público através do livro [JACOBSON 1998].

Características do Processo Unificado

O UP está fundamentado em três características: ser dirigido por casos de uso; ser iterativo e incremental; e ser focado na arquitetura do software a ser implementado. Estar de acordo com os requisitos dos usuários é o aspecto mais importante para identificar o sucesso ou o fracasso de uma aplicação. O UP utiliza os diagramas de Casos de Uso (*Use Cases* [KRUCHTEN 1998][JACOBSON 1992]) para capturar e validar os requisitos funcionais de um sistema de software. Os diagramas de caso de uso apresentam os requisitos sob a ótica de *para quem* ou *para o que* uma funcionalidade vai produzir um resultado interessante.

O UP é um processo dirigido pelos casos de uso porque eles são utilizados durante as disciplinas de análise, projeto, implementação e testes do sistema. Na disciplina de análise, são criados diagramas (de classes, de colaboração, de interação, de estado e de atividade) em um nível de abstração que reflete o domínio do problema e que servem para **especificar** os casos de uso.

O resultado da disciplina de projeto é um conjunto de modelos, cujo nível de abstração está mais próximo da implementação, que **realiza** os casos de uso. A estrutura estática da aplicação, na fase de projeto, é representada em termos de subsistemas, classes e interfaces que visam garantir requisitos não funcionais (flexibilidade, manutenibilidade, reutilização, etc.) e especificar o ambiente tecnológico de implementação (plataforma de objetos distribuídos, gerenciador de banco de dados, pacote de interface gráfica, etc.); enquanto que os aspectos dinâmicos são representados através das colaborações entre esses componentes. Na disciplina de implementação, é criado um modelo do sistema que inclui código fonte (classes Java, arquivos C, etc.), arquivos executáveis (*bytecode*, arquivos com extensão "exe", bibliotecas de ligação dinâmica, etc.) e componentes reusáveis (*Java Beans*, ActiveX, repositórios de componentes EJB, etc.) que **implementam** os casos de uso. Durante a disciplina de testes do sistema, os casos de uso são utilizados para **definir** os casos de teste da aplicação (*Test Cases* [JACOBSON 1998]), que consistem na elaboração de cenários (dados de entrada, condições de execução e resultados esperados) para a aplicação dos testes.

Além dos aspectos relacionados aos requisitos funcionais (capturados pelos casos de uso), o UP também é orientado por uma perspectiva técnica - a arquitetura de desenvolvimento. A descrição da arquitetura contempla tanto os **elementos estruturais significantes** da aplicação (subsistemas, classes, componentes, etc.), quanto a colaboração desses elementos estabelecidas através de interfaces. A arquitetura do sistema é apresentada nas diferentes visões construídas durante o desenvolvimento. Ou seja, existe a visão arquitetural do modelo de casos de uso, a visão arquitetural do modelo de análise, a visão arquitetural do modelo de projeto, a visão arquitetural do modelo de implementação, a visão arquitetural do modelo de implantação, etc. Por outro lado, apenas os casos de uso mais significantes compõem a visão arquitetural do modelo de casos de uso; da mesma forma que apenas os diagramas mais importantes do modelo de análise compõem a visão arquitetural do modelo de análise; e assim sucessivamente para as demais visões arquiteturais. A opção por construir a arquitetura com base apenas nos elementos mais importantes se deve ao fato de que, dessa forma, existe uma minimização nas mudanças durante o ciclo de desenvolvimento. Todos os elementos de um nível conceitual que não estão presentes na arquitetura são construídos de forma a se adaptar aos subsistemas, classes, componentes e interfaces da arquitetura proposta. Quanto mais cedo for definida a arquitetura, melhor será o entendimento do sistema (complexidade de comportamento, ambiente operacional,

etc.), a organização de desenvolvimento (os subsistemas, que obedecem a interfaces preestabelecidas, podem ser atribuídos a diferentes equipes de desenvolvimento) e o favorecimento do reuso e da evolução do sistema (como as colaborações entre os subsistemas, classes e componentes são feitas através de interfaces, esses elementos podem ser alterados desde que obedeçam à mesma interface).

O UP organiza o desenvolvimento do sistema nas fases de Iniciação, Elaboração, Construção e Transição (ver seção 4.1.3). Cada uma dessas fases pode ser constituída por várias iterações; enquanto que estas, por sua vez, seguem todo o fluxo de atividades (captura de requisitos, análise, projeto, implementação e testes) de uma ou mais funcionalidades do sistema (casos de uso). Diferentemente do ciclo de desenvolvimento proposto na análise essencial [POMPILHO 1995], cuja geração de código só ocorre após ter sido concluída a análise e o projeto do sistema como um todo, o UP modifica ou acrescenta um conjunto de artefatos (diagramas de casos de uso, diagramas de classe, diagramas de colaboração, código executável, etc.) a cada iteração. Os incrementos correspondem às mudanças existentes entre duas iterações sucessivas. Ao término de cada iteração, o gerente do projeto pode checar se o desenvolvimento do sistema está dentro das expectativas de custo e prazo estimadas e os usuários podem avaliar se os requisitos da aplicação estão sendo atendidos. As principais vantagens de se utilizar uma proposta de desenvolvimento iterativa e incremental são as seguintes:

- Redução dos riscos do projeto: Os riscos são variáveis que dificultam ou impedem o sucesso de um projeto. Existem riscos relacionados a questões técnicas (tecnologias não dominadas pela equipe de desenvolvimento, requisitos de desempenho, escalabilidade, robustez, etc.) e não técnicas (falta de experiência no domínio da aplicação, cronogramas e investimentos apertados, etc.). As iterações são identificadas e priorizadas para que os riscos do projeto sejam tratados (eliminados do projeto, confinados a um determinado escopo ou monitorados durante o desenvolvimento) o mais cedo possível. Se algum risco para o projeto não puder ser tratado, deve ser tomada a decisão sobre o prosseguimento ou não do desenvolvimento. Como essa decisão é tomada nas fases iniciais, pouco tempo e recursos foram despendidos.

- Garantia da conformidade dos requisitos: As necessidades e as tecnologias das organizações estão mudando de uma forma cada vez mais rápida. Se os usuários de uma aplicação só tiverem contato com um produto executável após ter sido concluída a análise e o projeto do sistema como um todo, pode ser que os requisitos implementados não correspondam às necessidades vigentes dos usuários. O UP propõe que a cada iteração se faça um pouco de análise, um pouco de projeto, um pouco de implementação, um pouco de testes e um pouco de implantação do sistema. Ao término das iterações, os usuários podem validar se os incrementos resultantes estão de acordo com os requisitos. Caso positivo, pode ser iniciada uma nova iteração para acrescentar funcionalidades ao produto que está sendo construído. Caso contrário, é iniciada uma nova iteração cujo objetivo é realizar correções para garantir a conformidade do produto com os requisitos dos usuários.
- Disponibilidade de melhores subsídios para a gerência do projeto: Ao término de cada iteração, o gerente de projeto pode avaliar se os objetivos foram cumpridos obedecendo aos prazos e os custos estimados. Se algo estiver fora das expectativas, carências podem ser supridas nas próximas iterações. Quanto mais cedo forem identificadas mudanças no planejamento do desenvolvimento, menor será o impacto nos custos e prazos do projeto.

Fases do Processo Unificado

O UP subdivide o ciclo de desenvolvimento nas fases de Iniciação, Elaboração, Construção e Transição. Durante a fase de Iniciação, é desenvolvido o Caso de Negócio (*Business Case*) que responde a questões como: *O que o sistema deve disponibilizar para seus principais usuários? Quais elementos deverão compor a arquitetura? Qual o plano de trabalho e qual o custo para o desenvolvimento do produto?* A primeira questão é respondida com um modelo de casos de uso que contém as funções mais críticas da aplicação. A segunda questão é respondida com um esboço da arquitetura que contempla os principais subsistemas do projeto. Como resposta para a terceira questão, são identificados e priorizados os riscos mais importantes; são feitos um planejamento detalhado da fase de elaboração e um levantamento superficial dos custos e prazos para o sistema como um todo.

Durante a fase de Elaboração, a maior parte dos casos de uso é especificada em detalhes e o desenho da arquitetura é construído. Os demais artefatos construídos ao longo do desenvolvimento devem se ajustar à arquitetura proposta, que consiste nos elementos mais importantes do modelo de casos de uso, dos diagramas de análise, dos diagramas de projetos e dos diagramas de implementação. No final da fase de elaboração, o gerente de projeto possui os subsídios necessários para planejar as atividades e estimar os recursos requeridos para completar o projeto. A questão nessa fase é: *Estão os riscos sobre razoável controle e os casos de uso, a arquitetura e o planejamento estáveis o suficiente para propor o desenvolvimento do software na forma de um contrato?*

Durante a fase de Construção, a visão do sistema evolui para um produto a ser transferido ao cliente. Apesar da arquitetura se encontrar estável, algumas mudanças são requeridas quando os desenvolvedores encontram melhores soluções para estruturar o software. No final desta fase, o produto contempla todos os casos de uso que gerentes e usuários concordaram em implementar. Alguns erros ainda existem e devem ser corrigidos apenas na fase de transição. A questão neste ponto é: *O produto satisfaz os requisitos dos usuários de tal forma que ele possa ser disponibilizado?*

Durante a fase de Transição, uma versão de avaliação do produto é disponibilizada para um pequeno número de usuários. Esses usuários experimentam a aplicação, reportando deficiências e erros para a equipe de desenvolvimento. A equipe de desenvolvimento, por sua vez, fica responsável pela geração de uma nova versão do produto, disponibilizada a um maior número de usuários, acrescida de algumas funcionalidades propostas e com os erros reportados corrigidos. O treinamento dos usuários, a assistência *on-line* e a correção dos defeitos detectados após a disponibilização são outras atividades da fase de transição.

[ARNOLD 1998] ARNOLD, K., GOSLING, HOLMES, The Java Programming Language, Addison-Wesley, 1998.

[BERGSTEN 2000] BERGSTEN, H., JavaServer Pages, O'Reilly, 2000.

[CONALLEN 2000] CONALLEN, J., Modeling Web-Tier Components, Software Development Magazine, Janeiro de 2001, **on-line:** http://www.sdmagazine.com

[GAMMA et. al 1994] GAMMA, E., HELM, JOHNSON, VLISSIDES, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, 1994.

[HORSTMANN 1999] HORSTMANN, C., CORNELL, Core Java 2, Volume 1: Fundamentals, Prentice-Hall, 1999

[HUNTER 2000] HUNTER, J., CRAWFORD, Java Servlet Programming, O'Reilly, 2000.

[JACOBSON 1992] JACOBSON, I., CHRISTERSON., JONSSON., ÖVERGAARD, , Object Oriented Software Engineering: A Use Case Driven Approach, Addison-Wesley, 1992.

[JACOBSON 1998] JACOBSON, I., BOOCH, RUMBAUGH, The Unified Software Development Process, Addison-Wesley, 1998

[KRUCHTEN 1998] KRUCHTEN, P., The Rational Unified Process: An Introduction, Addison-Wesley, 1998.

[POLLICE 2000] POLLICE, G., Using The Rational Unified Process for Small Projects: Expanding Upon eXtreme Programming, Rational Software White Paper, 2000 **on-line**: http://www.rational.com/whitepapers

[POMPILHO 1995] POMPILHO, S., Análise Essencial - Guia Prático de Análise de Sistemas, Livraria e Editora Infobook S/A, 1995.

[ROGERS 1997] ROGERS, G., Framework-Based Software Development in C++, Prentice-Hall, 1997.

[SZYPERSKI 1998] SZYPERSKI, C., Component Software: Beyond Object-Oriented Programming, Addison-Wesley, 1998.