Proposta para Integração de Sistemas Legados para Aprendizado a Distância: Estudo de Caso em Planejamento de Sistemas Móveis Celulares

A. M. L. Miranda[1], C. R. L. Francês[2], G. P. S. Cavalcante[3], J. C. W. A. Costa[4], L. A. G. Oliveira[5], *et al*.
[1,2,3,4] Laboratório de Eletromagnetismo Aplicado - UFPA- Rua Augusto Corrêa, No. 01, Guamá - Caixa Postal 8619 - CEP 66.075-900 – Belém – Pará.
[5]Departamento de Engenharia de Computação e Automação – UFRN - Caixa Postal 1524 – 59.072-970- Natal - RN - Brasil

***Resumo* – Um desafio para o oferecimento de educação continuada, em engenharia especialmente, é certamente a integração dos sistemas legados, pois o cenário de soluções disponibilizadas é usualmente formado por um ambiente totalmente heterogêneo (diversos sistemas operacionais e hardwares). Desse modo, torna-se interessante a busca de soluções que facilitem a interoperabilidade entre esses sistemas legados, de modo a permitir a integração, entre essas diversas soluções já disponibilizadas, intra ou inter instituições. A proposta apresentada neste trabalho mostra como se pode realizar essa integração utilizando-se tecnologias como UML, MVC, CORBA e SOAP para aplicativos desenvolvidos na UFPA para planejamento de sistemas móveis celulares.**

***Palavras-Chaves* – Integração de Sistemas, Sistemas Legados, Sistemas Distribuídos, Objetos Distribuídos, CORBA, Sistemas Móveis.**

I. INTRODUÇÃO

A constante evolução tecnológica é um fator que contribui para a fragmentação de sistemas em qualquer corporação, fazendo com que as mesmas se deparem com falta de integração entre seus sistemas legados [1]. Em especial, na área educacional é comum encontrar aplicações legadas em diversas instituições de ensino que poderiam ser utilizadas de forma integrada a outras aplicações com o propósito de formar uma grande solução computacional para um determinado problema.

Esse aspecto é marcante em engenharia elétrica, área na qual inúmeros softwares já estão desenvolvidos para importantes aplicações, nas mais diversas plataformas de programação, por diversos grupos de pesquisadores do Brasil.

Uma forma de tratar esse problema seria reescrever todo o código das aplicações legadas, elegendo linguagens e sistemas operacionais que possam perdurar; porém isso demandaria muito tempo e dinheiro, além de desprezar o *know-how* já estabelecido para as aplicações em uso. Uma possível alternativa, que é a apresentada nesse artigo, seria adicionar entre as aplicações uma camada intermediária chamada *middleware*, a qual seria responsável por tratar toda a heterogeneidade do ambiente, fazendo então com que os mesmos conseguissem interagir.

A. M. L. Miranda, andremoacir@unama.br, C. R. L. Francês, rfrances@ufpa.br, G. P. S. Cavalcante, gervasio@ufpa.br, J. C. W. A. Costa, jweyl@ufpa.br, L. A. G. Oliveira, affonso@dca.ufrn.br, A. M. Cavalcante, amc@ufpa.br, G. H. S Carvalho, ghsc@ufpa.br, Tel +55-91-2111302, Fax +55-91-2111634.

Este trabalho foi parcialmente financiado pelo CNPq.

Outra motivação deste trabalho é a crescente demanda pelo desenvolvimento de soluções para aumentar a oferta de ações para aprendizagem a distância na área tecnológica. Entretanto, mesmo sendo a educação a distância uma ferramenta com potencial indiscutível, esta também se ressente da falta de homogeneidade de plataforma e aplicativos.

Em função desse contexto, este trabalho propõe uma abordagem para tratar o problema de heterogeneidade de aplicativos, linguagens de programação e sistemas operacionais, elaborando-se uma estratégia para integrar sistemas legados com o enfoque de educação a distância no domínio da área de engenharia elétrica.

Desta forma, são apresentados os aspectos fundamentais referentes à integração de três sistemas legados e à execução dos mesmos via *Web*. Esses sistemas têm a função de realizar algumas tarefas referentes planejamento de sistemas móveis celulares, e são módulos de um sistema maior chamado CELLP [2], que foi desenvolvido na UFPA.

Originalmente, os módulos que compõem o CELLP não interoperavam e eram executados em um ambiente totalmente heterogêneo. A fim de possibilitar a integração entre esses módulos, utilizou-se a arquitetura CORBA (*Common Object Request Broker Arquitecture*) para "mascarar" toda a heterogeneidade e complexidade do ambiente, pois CORBA é uma arquitetura que possui vários recursos para integrar sistemas legados, juntamente com ampla variedade de hardwares e softwares heterogêneos [3].

Para que isso fosse possível, houve a necessidade de utilizar a UML (*Unified Modeling Language*) para fazer a engenharia reversa do CELLP, baseando-se no padrão de projeto MVC (*Model View Controller*), a fim de facilitar o processo de integração dos três sistemas e facilitar também a fatoração dos mesmos em alguns objetos distribuídos. Esses processos, posteriormente, agilizaram as modificações necessárias no código, para enfim possibilitar a integração e execução via *Web* através do protocolo SOAP (*Simple Object Access Protocol*). Ao adicionar ao CELLP uma nova filosofia multicamada, implementou-se a versão WebCELLP [4]. Desta forma, o ponto-chave da proposta está na concepção e implementação do *middleware* (discutido na próxima seção).

No estudo de caso foi criado um ambiente para *Web*, o qual disponibiliza, de forma prática e didática, um referencial teórico sobre planejamento de sistemas móveis celulares e a possibilidade de execução do sistema WebCELLP. Esse ambiente servirá de apoio para a disciplina de Sistemas

Móveis Celulares do curso de Engenharia Elétrica da UFPA. O ambiente tem como seus principais objetivos:

Em relação ao ensino-aprendizagem:

- Facilitar o aprendizado, através de uma *interface* interativa, reforçando o conhecimento adquirido na sala de aula tradicional;
- Disponibilizar o conhecimento para aprendizado e revisões complementares das disciplinas envolvidas;
- Disponibilizar um protótipo para o uso da comunidade acadêmica e empresarial, deixando-o preparado para o acoplamento de novos módulos;
- Possibilitar, por um período de tempo integral, a utilização do ambiente, levando em conta a disponibilidade e o ritmo do aluno;

Em relação aos sistemas utilizados:

- Possibilitar a integração de sistemas que foram desenvolvidos em linguagens de programação diferentes;
- Possibilitar a interoperabilidade de sistemas operacionais e arquiteturas de hardware diferentes;
- Disponibilizar a utilização do CELLP em formato "universal" de navegação, via Web;
- Ampliar a tolerância a falhas do sistema, pela adoção da arquitetura de três camadas.

II. *Middleware*

Middleware é a camada de software situada entre o sistema operacional e os componentes de aplicação que tem como finalidade facilitar a comunicação e a coordenação desses componentes que estão distribuídos nos diversos computadores de uma rede [5]. Portanto, caso seja desejado que um determinado aplicativo possa ser utilizado em uma quantidade razoável de máquinas e sistemas operacionais, uma possível alternativa é o uso de *middleware*.

Como o propósito do WebCELLP é executar em um ambiente distribuído, portanto heterogêneo, foram criados cinco objetos que são gerenciados por dois servidores que executam em hardwares e sistemas operacionais diferentes. Para que fossem geradas as implementações dos cinco objetos especificados no *middleware* CORBA, juntamente com seus stubs e skeletons (elementos que encapsulam a comunicação) foram definidas quatro IDLs (*Interface Definition Language*), as quais foram submetidas a dois compiladores diferentes. Assim, quando um cliente emitir uma requisição será utilizada uma chamada remota para invocar os métodos (forma de comunicação entre objetos) dos objetos.

Essa requisição é enviada através de uma mensagem para o servidor que gerencia o objeto. O servidor ao receber a requisição, ativa o objeto para que o mesmo possa executar o método solicitado. Quando o objeto tiver posse do resultado, ele informa o servidor, que por sua vez retorna esse resultado ao cliente em uma outra mensagem.

III. Modelagem do Sistema

Dos três módulos que compõe o WebCELLP, dois (o *software* Predict e o Traffic Design) foram desenvolvidos no Delphi 7® e executam sobre o sistema operacional Windows®, e o terceiro (o *software* Radio Link) foi desenvolvido no Kylix 3® e executa sobre o sistema operacional Linux®.

O módulo Predict foi desenvolvido principalmente para realizar estudos estatísticos comparativos entre os modelos clássicos de rádio-propagação [2]. A fig. 1., mostra uma janela desse *software, a* qual é dividida nas seguintes abas:

- Parâmetros: nela o usuário deverá fornecer os dados referentes aos parâmetros do sistema (potência transmitida, freqüência, etc) que foram utilizados na campanha de medições de referência. O usuário também deve selecionar nesta aba, os modelos que serão colocados sob avaliação assim como os parâmetros urbanos necessários para a aplicação dos mesmos;
- Base de Dados: nela o usuário deverá fornecer a base de dados das campanhas de medições realizadas na região de interesse, ele terá também a opção de editar os dados e depois salvá-los em um arquivo;
- Análise Gráfica: nela são gerados gráficos da potência recebida (dBm) *versus* distância (m), baseando-se nos modelos colocados sob avaliação e nos dados da campanha de medições;
- Análise Estatística: nela são apresentados os desempenhos dos modelos testados frente às medidas. O desempenho é especificado sobre três aspectos: Desvio Absoluto (dB), Erro Médio Absoluto (dB) e Erro RMS (dB). O usuário ainda tem a opção de gerar um relatório apresentando os resultados obtidos e de salvar esses resultados em um determinado projeto.

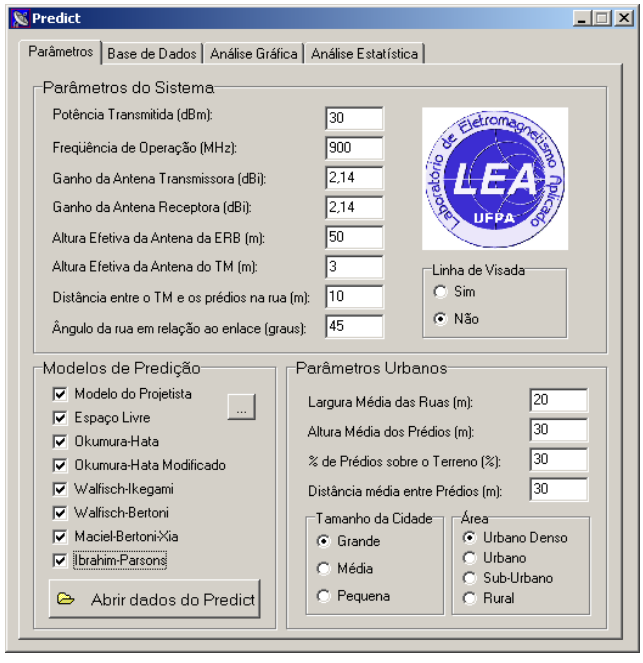

Fig. 1. Janela inicial do Predict.

Já o *software* Radio Link foi desenvolvido para realizar o balanço de potência do sistema, determinando o raio máximo de cobertura de um determinado modelo para os dois enlaces básicos (descida e subida). A janela desse módulo é dividida

em duas abas, uma denominada “Dados de Entrada”, onde o usuário fornecerá os parâmetros do sistema e escolherá o modelo que deseja realizar o estudo, e outra denominada “Resultados”, onde o usuário poderá verificar os resultados dos cálculos realizados, gerar relatório contendo os resultados obtidos e salvar esses resultados em um determinado projeto.

O *software* Traffic Design realiza o projeto de tráfego baseado em um mapa que demonstra o tráfego previsto para uma determinada região. O mapa é dividido em quadrículas que representam o tráfego estimado na área delimitada pela mesma. Nesse módulo o usuário define alguns parâmetros do mapa e o raio das células, para que seja determinado o número de canais necessários para atender tal tráfego. O usuário ainda tem a possibilidade de gerar um relatório do resultado dessa contagem e de salvar esse resultado em um determinado projeto.

Na Fig. 2. é apresentado um caso de uso que mostra a interação entres os módulos do WebCELLP. Inicialmente, o usuário deverá visitar um site que contém um material teórico sobre planejamento de sistemas móveis celulares e um *link* para a execução do sistema. Ao clicar nesse *link*, o usuário deverá informar seu *login* de acesso, para que seja feita a sua validação, e então o usuário poderá executar qualquer módulo do sistema.

Após a validação, caso o usuário escolha executar o módulo Predict, ele terá como resultado a definição do modelo de predição ideal para a região em estudo. Caso o usuário decida executar o módulo Radio Link, será necessária uma interação com o módulo Predict, para que seja feita a captura de alguns parâmetros e somente após isso é que será feito o cálculo do raio máximo de cobertura da célula.

Na execução do Traffic Design também será necessária uma interação para capturar parâmetros, porém com o módulo Radio Link, para então ser feito o cálculo da contagem de tráfego na célula.

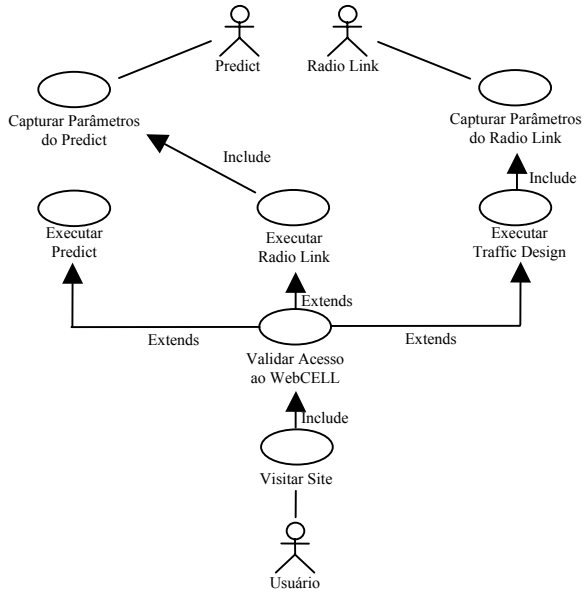

Fig. 2. Caso de uso geral do WebCELLP.

IV. MVC

A escolha do padrão de projeto MVC deu-se pelo fato dele ser ideal para projetar aplicações *Web* [6]. Durante a fase de implementação, é importante deixar registrado os seguintes aspectos referentes ao MVC:

- Na Vista (interface gráfica do sistema) utilizou-se a interface gráfica já existente no software legado, fazendo-se algumas adaptações, como a retirada das regras de negócios da própria interface.
- No Modelo, onde estão armazenadas as regras de negócios, construíram-se dois servidores de aplicação, um implementado em Delphi 7® e outro em Kylix 3®, sendo que em ambos foram criados objetos utilizando os recursos de programação distribuída do padrão CORBA. Além dos servidores de aplicação, criou-se também um banco de dados no InterBase 6.5®, o qual armazena além das tabelas algumas regras de negócios.
- O elemento que faz o papel de Controlador é o próprio CORBA, que serve de intermediador entre o cliente e os objetos do Modelo.

Na Fig. 3. pode-se visualizar o modelo MVC do WebCELLP.

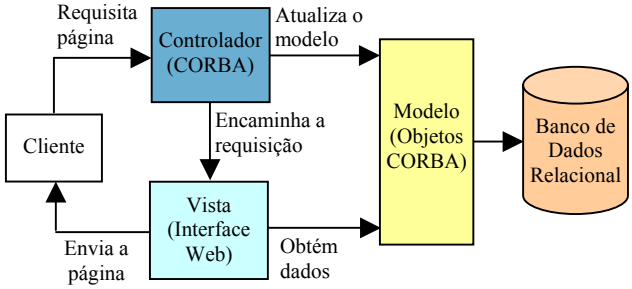

Fig. 3. Modelo MVC do WebCELLP.

V. Plataforma Heterogênea

No desenvolvimento do WebCELLP foram utilizados dois ambientes de programação visual que usam a linguagem de programação *Object Pascal*: o Delphi® e o Kylix®, que foram construído para serem executados no sistema operacional Windows® e Linux®, respectivamente. Através do Delphi/Kylix pode-se desenvolver desde aplicações genéricas, que não acessam banco de dados, até aplicações Cliente/Servidor com n-camadas que integram várias linguagens, como é o caso desse sistema.

A Fig. 4. apresenta a arquitetura desenvolvida para que o WebCELLP seja acessado via *Web* e opere sobre diversas plataformas e sistemas operacionais. Para isso necessitou-se utilizar os recursos de computação distribuída do Delphi/Kylix, os quais permitem a criação de aplicações com n-camadas, que são compostas por um conjunto de componentes e outras tecnologias como CORBA, DCOM, SOAP e TCP/IP, permitindo assim a construção de uma variedade de aplicações com acesso a banco de dados.

Especificamente no sistema em questão, utilizou-se o CORBA para prover a comunicação entre a aplicação cliente

e a aplicação servidora. Porém, para ser flexível o bastante a ponto de operar sobre o protocolo HTTP, houve a necessidade de utilizar a tecnologia *WebServices*, baseado no protocolo SOAP.

Sendo assim, o cliente *Web* faz a requisição através do SOAP, que encaminha a requisição ao respectivo objeto CORBA no servidor, para que sejam obtidos os dados do banco de dados. De posse desses dados, o servidor os envia para o cliente no formato XML, conforme é mostrado na Fig. 4. Essa divisão de responsabilidade permite que o cliente visualize, pagine, busque e ordene os dados, localmente, atualizando a tela conforme necessário sem fazer novas conexões ao servidor.

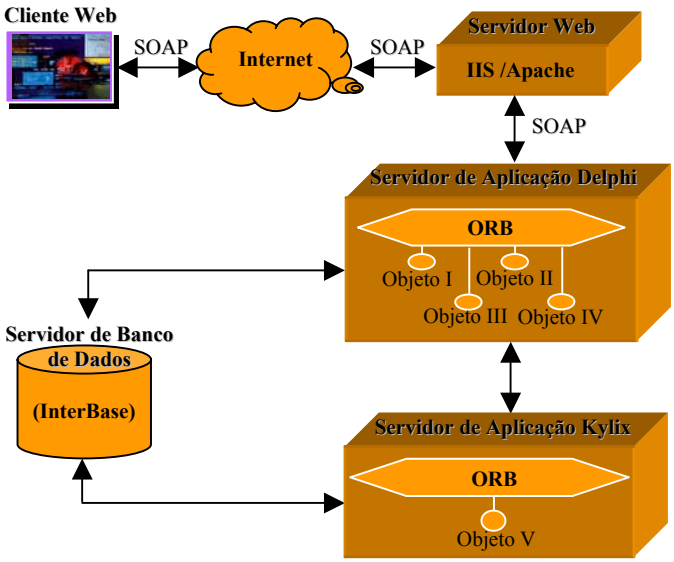

Fig. 4. Arquitetura do WebCELLP.

Analisando a Fig. 4., percebe-se que no Servidor de Aplicação Delphi estão quatro objetos CORBA pertencentes ao Predict e ao Traffic Design, e que no Servidor de Aplicação Kylix está um objeto pertencente ao Radio Link. Percebe-se também nos dois servidores de aplicação a presença do ORB (*Object Request Broker*), que é o núcleo da arquitetura CORBA, e tem com principais funções: localização do objeto, ativação do objeto se necessário, envio da requisição do cliente para o objeto e envio da resposta do servidor para o cliente [7].

Em relação à base de dados remota, pode-se afirmar que os acessos a essa base poderiam ser mais rápidos, caso fossem feitos diretamente ao banco de dados. Porém, por questão de segurança, os acessos foram feitos pelos objetos CORBA a um servidor remoto, pois esses objetos têm implementado todos os métodos necessários para acessar a base de dados do sistema. Outra vantagem, além da segurança, é a reutilização e a padronização dos códigos de acesso à base de dados, facilitando futuras manutenções.

Outro aspecto relevante neste trabalho é o fato de que a maioria das aplicações *Web* baseadas em dados, sofrem com a baixa velocidade de comunicação. Isso se dá pelo fato de que cada requisição do cliente ao servidor precisa conter todas as informações de estado, que são necessárias para satisfazer tal requisição. Portanto, o servidor precisa executar uma nova consulta à base de dados, ao invés de reutilizar os conjuntos de resultados previamente estabelecidos.

No WebCELLP tal fato não ocorre porque, utilizando-se CORBA, pode-se fazer com que as aplicações *Web* melhorem o desempenho e o tempo de resposta ao cliente, reduzindo-se o número de requisições que cada cliente faz ao servidor e a quantidade de trabalho que o servidor precisa executar.

Uma outra solução para realizar a integração do CELLP que foi citada neste artigo, seria reescrever todo o código da aplicação, porém com certeza seria muito mais trabalhosa.

Os comentários finais que devem ser pontuados com relação a esta proposta são com relação a possíveis dificuldades que podem ser encontradas para replicá-la:

- Fatoração dos módulos do CELLP em objetos distribuídos: os módulos do CELLP operavam isoladamente, com nenhum grau de integração. Precisou-se então fatorá-los em objetos CORBA, deixando-os inclusive prontos para serem reutilizados;
- Configuração do sistema operacional: para possibilitar a comunicação do CORBA com o Linux® foi preciso configurar algumas variáveis de ambiente do Linux®. Portanto, num projeto como este é necessária a participação de profissionais que dominem tanto a área de sistemas operacionais quanto a de redes de computadores;
- Configuração do Kylix®: foi preciso também incluir alguns arquivos do Kylix® no Linux®, para que fosse possível a comunicação do CORBA com o servidor de aplicação Kylix.

VI. Conclusão

Um ganho proporcionado por este trabalho foi a mudança na forma de ensino da disciplina Sistemas Móveis Celulares, dentro do curso de Engenharia Elétrica da Universidade Federal do Pará. Ao longo dos últimos anos, esta disciplina vem sendo ministrada de forma teórica e totalmente presencial, o que limitava em parte um aprendizado mais abrangente. Com o desenvolvimento desse sistema (dizer o nome do sistema), surgiu a possibilidade de transformar parte da carga horária teórica em prática, além da possibilidade de torná-la semi-presencial. Segundo a legislação vigente, pode-se ter até 20% da carga horária em regime não presencial, o que, via de regra, ainda não é utilizado de maneira eficiente em cursos de Engenharia.

Assim, atualmente, estão sendo utilizadas duas abordagens: presencial e a distância. Na parte presencial é apresentado o conteúdo teórico da disciplina, como já vem acontecendo ao longo dos anos; e na parte a distância será utilizado o sistema WebCELLP, como ferramenta de apoio, para que o aluno possa aplicar os conhecimentos adquiridos em sala de aula, através da criação de um projeto de planejamento de sistema móvel celular.

Neste artigo foram apresentados alguns pontos relevantes a aplicações distribuídas e integração de sistemas legados, procurando viabilizar a interoperabilidade entre ambientes heterogêneos, o que se constitui em um problema do estado-

da-arte de corporações de diferentes setores e, em especial, a área de educação.

Através do projeto e da implementação do WebCELLP, foi possível se efetivar uma solução viável para problemas de integração de sistemas legados heterogêneos, aplicando-se a estratégia baseada em middleware, no domínio de ensino de engenharia elétrica.

Além da utilização do sistema, há alguns outros ganhos indiretos na utilização dos princípios de engenharia de software para desenvolver sistemas de software. Por exemplo, percebeu-se que através do padrão de projeto MVC, facilmente identificaram-se as classes e instâncias participantes, seus papéis, colaborações e a distribuição de responsabilidade. A utilização desse padrão tornou a aplicação menos complexa e agilizou a fase de implementação, pois o padrão MVC é ideal para ser utilizado em aplicações com três camadas, como o WebCELLP. A abordagem utilizada permite uma facilidade maior no aumento da escalabilidade dos módulos do sistema, além de facilitar a adição de outros aplicativos que podem interagir com o WebCELLP. Além disso, a abordagem pode ser generalizada para outros aplicativos disponíveis em diferentes instituições ou mesmo dentro do DEEC da UFPA.

REFERÊNCIAS

[1] CUMMINS, F. A., "Integração de Sistemas: arquite-turas para integração de sistemas e aplicações corporativas", Rio de Janeiro: Campus, 2002.

[2] CAVALCANTE, A. M. *et al*., "Software Educacional para Dimensionamento de Sistemas Móveis Celulares", Trabalho apresentado no X Simpósio Brasileiro de Microondas e Optoeletrônica - SBMO. Recife, 2002.

[3] COULOURIS, G., DOLLIMORE, J., *et al*., "Distributed Systems: Concepts and Design," England: Pearson Education, 2001.

[4] MIRANDA, A. M. L., "Integração de Sistemas Legados para aprendizado a Distância: Estudo de Caso em Planejamento de Sistemas Móveis Celulares", Dissertação de Mestrado, 2003. Programa de Pós-Graduação em Engenharia Elétrica. Universidade Federal do Pará, 2003.

[5] EMMERICH, W., "Software Engineering and Middleware: A Roadmap. In The Future of Software Engineering", 22nd Int. Conf. On Software Engineering (ICSE2000), pages 117 –129. ACM Press, May 2000.

[6] BROWN S. *et al*., "Professional JSP", 2nd Edition. [s.l.]: Wrox, 2001.

[7] OMG, Object Management Group. Common Object Request Broker Architecture: Core Specification Version 3.0, OMG Document, December 2002. Disponível em: <http://www.omg.org/technology/documents/corb_spec_catalog.htm>. Acessado em: 09/02/2003.