UNIVERSIDADE FEDERAL DE SANTA CATARINA - UFSC

SISTEMA DE CONTROLE DE ATIVIDADES EMPRESARIÁIS PARA FEDERAÇÃO CATARINENSE DE MUNICÍPIOS

Trabalho de Conclusão de Curso
apresentado ao Curso de Bacharelado
em Sistemas de Informação da
Universidade Federal de Santa Catarina
como requisito parcial para obtenção do
grau de bacharel em Sistemas de Informação

Orientador: Prof. João Bosco Mangueira Sobral, Dr.

Florianópolis
2005

KLEYTON WEBER DA SILVA
SERGIO FERREIRA DE MENDONÇA

SISTEMA DE CONTROLE DE ATIVIDADES EMPRESARIÁIS PARA FEDERAÇÃO CATARINENSE DE MUNICÍPIOS

Florianópolis, 01 de Junho de 2005.

BANCA EXAMINADORA

Prof. João Bosco Sobral Universidade Federal de Santa Catarina **Orientador**	**Prof. Bernardo Gonçalves Riso** Universidade Federal de Santa Catarina
Prof. Frank Augusto Siqueira Universidade Federal de Santa Catarina	**Ronaldo dos Santos Mello** Universidade Federal de Santa Catarina

AGRADECIMENTOS

Agradecemos a Deus que nos proporcionou saúde e inteligência durante esta caminhada.

A família, que nos apoiou e acolheu em momentos difíceis.

Ao corpo docente pela competência no exercício da profissão e forma afetuosa que trata seus alunos.

Ás amizades constituídas durante o curso, que esperamos ser apenas uma semente plantada em nossa juventude que crescerá e dará frutos pro resto de nossas vidas.

SUMÁRIO

LISTA DE FIGURAS

RESUMO

Ferramentas para desenvolvimento de atividades operacionais estão em constante estado de evolução e cada vez mais presentes nas organizações. Contudo, a singularidade de cada empresa exige um conjunto de características que dificilmente podem ser acopladas em uma única ferramenta.

Este trabalho tem como objetivo prover o desenvolvimento de uma ferramenta de controle de atividades empresariais para atender inicialmente o caso específico da FECAM – Federação Catarinense dos Municípios, suprindo as principais limitações operacionais encontradas nesta entidade.

Palavras-Chave: FECAM, WEB, frameworks, Struts, Hibernate, JSP, Java 2 Enterprise Edition (J2EE), MYSQL, plataforma Java 2 Standard Edition (J2SE).

1. INTRODUÇÃO

1.1. Contextualização

O emprego da tecnologia da informação como ferramenta de trabalho nas empresas deixou de ser exceção para firmar-se como a regra, seja em pequenos escritórios ou em grandes empresas. Ao menos naquelas onde as tarefas fins constituem produto ou serviço relativamente similar, ou em que o processo de fabricação da mercadoria ou prestação do serviço segue determinado padrão. A utilização da tecnologia da informação como agilização e controle dessas tarefas é corriqueira.

Tendo como exemplo a Federação Catarinense de Municípios – FECAM, entidade representativa dos interesses municipais de Santa Catarina, que muito embora seja uma associação sem fins econômicos, precisa fazer uso da tecnologia da informação para melhorar seus serviços e reduzir os custos operacionais.

1.2. Objetivos do Trabalho

1.2.1. Objetivo Geral

O objetivo geral deste trabalho é desenvolver uma aplicação para controle de atividades empresariais voltado ao ambiente Web, utilizando a plataforma J2EE e os frameworks de código aberto Struts e Hibernate, suprindo as necessidades operacionais encontradas pela FECAM.

1.2.2. Objetivos Específicos

- Estudar a arquitetura da plataforma J2EE;
- Estudar a arquitetura do framework Strutrs;
- Estudar a arquitetura do framework Hibernate;
- Estudar mecanismos de segurança aplicados à plataforma J2EE;
- Estudar o banco de dados Mysql.
- Desenvolver um Sistema de Controle Empresarial utilizando as tecnologias acima citadas.

1.2.3. Justificativa do Trabalho

Existem diversos fatores consideráveis para a elaboração desta ferramenta de controle, dentre as quais o fato de estar servindo a comunidade, considerando que o usuário beneficiado é uma instituição sem fins lucrativos que presta serviço aos municípios de Santa Catarina. Além de cumprir o papel de cidadão existe o fator científico da descoberta de conhecimentos pois o desenvolvimento do referido sistema implica em um considerável acréscimo de conhecimento acadêmico.

1.2.4. Escopo

Este trabalho consiste no levantamento dos requisitos e necessidades operacionais da FECAM, especificando um conjunto características relevantes ao sistema. Tendo os pontos de atuação definidos serão estudadas e empregadas as seguintes tecnologias: arquitetura J2EE; framework opensource Struts e Hibernate e possíveis Desing Patterns que surgirão no desenvolvimento do sistema.

1.2.5. Meta

Oferecer uma ferramenta que resolva com rapidez e eficiência os problemas levantados pelo futuro usuário, satisfazendo suas necessidades operacionais do dia-a-dia.

Atingir o objetivo acadêmico de um trabalho deste porte que é expandir o conhecimento técnico, pessoal e profissional compartilhando descobertas e gerando o mais valioso patrimônio da atualidade: A informação.

2. CLIENTE ALVO (FECAM)

2.1. Institucional

A FECAM - Federação Catarinense de Municípios, fundada no dia 16 de julho de 1980, tem sede em Florianópolis, Praça XV de Novembro, 270 – Centro. É uma entidade municipalista privada, de natureza civil, sem fins lucrativos, que tem como objetivo atender os interesses comuns dos Municípios Catarinenses. Com a finalidade de integrá-los e representá-los, objetivando a valorização e fortalecimento do Municipalismo.

Atua através de projetos básicos voltados ao desenvolvimento institucional das Associações Municipais e dos Municípios Associados, capacitação dos agentes públicos municipais, elaboração de planos e projetos de desenvolvimento regional, consultorias especializadas, além de gestão da informação e aprimoramento tecnológico. Atua em parceria com outras entidades representativas do Municipalismo brasileiro, como a CNM - Confederação Nacional de Municípios e Associações de Municípios.

2.2. Metodologia da Empresa:

A FECAM estabelece um Plano de Trabalho anual apresentando projetos onde são traçadas linhas de estratégia para ações de cada ano vigente, contando com as associações dos municípios pois devem trabalhar em parceria na organização e promoção de eventos, capacitação dos agentes políticos e servidores municipais, repasse de orientações técnicas, encaminhamento de reivindicações, elaboração de projetos e mobilização dos municípios em torno de ações de interesse comum.

Consultas destinadas ao TCE/SC e ao Colegiado de Contadores podem ser feitas pelo sistema da FECAM, identificando a intenção de remeter a pergunta ao Tribunal, sendo que o prazo de resposta se sujeita à elaboração do parecer por parte do Tribunal de Contas do Estado.

2.3. Serviços Prestados:

2.3.1. Consultorias Especializadas:

A FECAM disponibiliza atualmente consultorias especializadas em Licitação Pública e Contabilidade Pública, que podem ser utilizadas pelos municípios filiados e pelas associações de municípios gratuitamente.

O prazo de resposta à consulta é de 02 (dois) dias úteis, podendo variar em casos excepcionais.

2.3.2. Consulta de Receitas:

È disponibilizada uma consulta de receitas de todos os municípios vinculados a FECAM, possibilitando acesso publico ás finanças dos principais municípios de Santa Catarina.

2.3.3. Cursos e treinamentos:

A capacitação dos agentes políticos e servidores públicos municipais é um dos mais destacados serviços oferecidos pela FECAM, sempre em parceria com as Associações.

Nos últimos anos, milhares de técnicos e autoridades municipais receberam treinamento, orientações e esclareceram suas dúvidas em seminários e cursos, realizados em diversas regiões do estado, abrangendo temas como controle interno, previdência, licitações, contabilidade pública, finanças municipais, lei de responsabilidade fiscal, elaboração dos orçamentos, estatuto das cidades etc.

2.3.4. Colegiado de Contadores:

O Colegiado de Contadores é um órgão vinculado à FECAM, formado por até dois contabilistas de cada associação de municípios e um coordenador, este necessariamente é indicado pela FECAM.

O objetivo principal é a uniformização de procedimentos bem como a aproximação dos responsáveis pela contabilidade pública municipal com o Tribunal de Contas do Estado, Governo Estadual e demais órgãos dos governos estadual e federal, visando fortalecer e valorizar os contabilistas que dedicam sua vida trabalhando na administração pública.

2.3.5. Acompanhamento de processos de análise das contas no TCE/SC:

Por meio da Assessoria de Comunicação da FECAM, todas as análises das contas municipais pelo TCE/SC são registradas e os resultados enviados aos interessados.

Aos Municípios ou Associações que necessitarem de ajuda nos processo junto ao TCE/SC ou mesmo queiram informações detalhadas sobre seus processos, podem solicitar à consultoria contábil da FECAM.

2.3.6. Provedor de Acesso aos Municípios Catarinense:

Disponibiliza para os municípios e associações serviços tecnológicos de última geração, provedor de Internet banda larga e servidor de correio eletrônico. O objetivo com isso é oferecer o que há de mais moderno no mercado, com preços diferenciados e excelente qualidade. Conheça a seguir as tecnologias e serviços disponíveis.

2.3.7. Autenticação ADSL:

É a liberação de uma senha e login para que a conexão com a Internet possa ser efetuada.

2.3.8. Webmail aos municípios:

Oferece serviços de webmail e agregados de segurança para prover contas seguras aos usuários por preços acessíveis.

2.4. Especificações Técnicas dos Recursos Físicos e de Software:

Servidor equipado com processador Xeon® de 3,06 GHz, 512 MB de memória RAM, disco SCSI de 36,6 GB, disco IDE-ATA com 80 GB espelhado, link de 1 Mbps FULL, OS Linux Redhat 7.3.

3. LEVANTAMENTO DE REQUISITOS

A FECAM atualmente emprega tecnologia da informação apenas em parte de seus trabalhos, que em alguns casos encontram-se defasadas e, conseqüentemente, de pouco auxílio e importância. Pensando nisso será apresentada abaixo uma proposta de reestruturação do controle de atividades e de documentos da FECAM, no intuito de propiciar aos responsáveis pela realização destes trabalhos um melhor controle de processo.

3.1. Caracterização dos serviços e ações da FECAM

A FECAM presta seus trabalhos às Associações de Municípios e aos Municípios de Santa Catarina de acordo com os seis projetos de atuação da entidade, a saber:

- *Representação e Defesa Institucional da FECAM, Associações e Municípios Catarinenses;*
- *Consultoria Especializada as Associações de Municípios;*
- *Planos e Projetos de Desenvolvimento Regional;*
- *Gestão da Informação e Desenvolvimento Tecnológico;*
- *Programa de Capacitação de Agentes Políticos e de Servidores Públicos Municipais; e*
- *Gestão de Qualidade da Administração Pública.*

Todas as linhas de atuação da entidade são desempenhadas com o apoio de órgãos públicos e outros parceiros, como a Confederação Nacional de Municípios, formando um círculo das entidades com as quais a FECAM se relaciona. Com algumas exceções, pode-se dizer que os “clientes” da FECAM compõem uma comunidade determinável e permanente.

Quanto ao desempenho das atividades pelos profissionais da FECAM, justamente por tratar externamente com as mesmas entidades, a utilização de padrões é perfeitamente aplicável, já que a rotina dos trabalhos não sofre muitas variações.

Por tratar-se de uma associação representativa dos interesses municipalistas, é muito importante à FECAM demonstrar os resultados de suas atividades aos seus

“clientes”, ou seja, uma prestação de contas. Dessa sorte, é essencial registrar os trabalhos realizados e inserir as conseqüências dos mesmos para que se possa apresentar aos interessados e justificar a própria existência da FECAM.

A partir dessas considerações preliminares, é possível descrever as maneiras de como a FECAM relaciona-se com seus parceiros, relacionando os instrumentos e estabelecendo padrões para tais atividades. Feito isso, a determinação do Controlador de Atividades – CATE, personalizado à FECAM, será facilmente realizada.

3.2. Módulos do sistema

Com o estudo de caso foi possível constatar inicialmente 3 módulos referentes aos processos da prestação de serviços, acompanhamento de resultados e apresentação aos associados – 21 clientes diretos (associações de municípios) e 293 indiretos (municípios de Santa Catarina).

3.2.1. Módulo 1 – Controle de Documentos

O primeiro módulo é referente ao controle dos documentos elaborados pelos colaboradores da empresa para envio aos clientes. Este módulo tornou-se necessário em função da quantidade de documentos gerados dentro da Fecam e da falta de uma padronização para armazenamento, classificação e localização dos mesmos.

Desta forma, cada documento deverá ser armazenado em um repositório gerenciado pelo sistema e classificado com atributos que permitam facilitar a localização, o acompanhamento do envio para os clientes e a geração de relatórios.

Sendo assim, foram levantados os seguintes atributos para cada documento:

- Modalidade do Documento:
  - Ofícios (Normal ou Circular)
  - Comunicados
  - Pareceres
  - Editais

  - Resoluções
- Responsável:
  - Funcionário que foi o autor do documento.
- Assinante:
  - Funcionário em nome do qual o documento será enviado.
- Destinatários:
  - Órgãos/Entidades ou pessoas físicas que receberão o documento.
- Número Identificador:
  - Número único por modalidade de documento, para controle e identificação. Deve ser gerado automaticamente pelo sistema.
- Data de elaboração do documento.
- Assunto do documento.
- Forma de envio do documento.
- Data de envio do documento.

Um documento poderá ser cadastrado inicialmente sem a data e a forma de envio, permanecendo assim num status de Não Enviado. A partir do momento que estes dados forem informados, o mesmo passará ao status de Enviado.

Para gerenciamento dos documentos faz-se necessário uma ferramenta de localização, facilitando a busca dos registros pelos seus atributos. A partir desta ferramenta de localização deve ser permitida a geração de relatórios sobre os documentos.

3.2.2. Módulo 2 – Controle de Atividades Externas

Os funcionários da FECAM são constantemente solicitados para elaboração e acompanhamento de diversas atividades externas ao ambiente da empresa, como cursos, palestras ou reuniões.

As atividades devem ser cadastradas com as seguintes características, sendo que após este cadastro as mesmas permanecerão com status de Agendada:

- Responsável:
  - Funcionário responsável pela organização e realização da atividade.
- Apoiadores:
  - Funcionários (0 ou mais) que apoiarão o responsável na realização da atividade
- Título da atividade.
- Espécie da atividade:
  - Reunião
  - Evento
  - Outros
- Local onde ocorrerá a atividade.
- Data de realização.
- Horário de realização.
- Duração da atividade.
- Pauta da atividade:
  - Deve-se informar um texto resumido sobre a pauta e também cadastrar um arquivo no sistema com a pauta completa.
- Projeto de atuação:
  - A atividade deve estar relacionada a um dos projetos de atuação da FECAM.
- Conselho de representação:
  - A atividade pode ou não estar relacionada a um dos conselhos de representação da FECAM.

Após a realização de uma atividade, é gerado um documento contendo um resumo de tudo o que ocorreu na mesma. Este documento também deverá ser cadastrado no sistema, juntamente com uma breve explanação a seu respeito. Assim que este cadastro for realizado, a atividade em questão receberá o status de Concluída.

Para gerenciamento das atividades faz-se necessário uma ferramenta de localização, facilitando a busca dos registros pelos seus atributos. A partir desta ferramenta de localização deve ser permitida a geração de relatórios sobre as atividades.

3.2.3. Módulo 3 – Cadastros Básicos

Para uma utilização mais flexível do sistema, faz-se necessário este módulo contendo os cadastros, alterações e exclusões dos dados básicos utilizados no sistema. Estes dados encontram-se relacionados a seguir:

- Tipos de documentos:
- Responsáveis por documentos:
- Assinantes de documentos:
- Destinatários de documentos:
- Formas de envio de documentos.
- Responsáveis por atividades:
- Apoiadores de atividades:
- Espécies de atividades:
- Projetos de atuação da FECAM:
- Conselhos de representação e suas respectivas gestões:

4. DESENVOLVIMENTO DO SISTEMA DE CONTROLE DE ATIVIDADES EMPRESARIAIS.

4.1. Arquitetura do sistema.

O sistema utilizará uma arquitetura composta de quatro camadas:

4.1.1. Camada de apresentação

Trata-se da camada de interação com o usuário. Nesta camada foram utilizadas páginas JSP (descritos na seção 4.2.1.2) e scripts na linguagem JavaScript.

4.1.2. Camada de controle

Esta camada determina o fluxo da apresentação servindo como uma camada intermediária entre a camada de apresentação e a lógica de negócio. É composta pelos Forms e pelas Actions do framework Struts (descrito na seção 5).

4.1.3. Camada de negócio

Trata-se da camada que contém a lógica da aplicação. Nesta camada fica concentrada toda a complexidade do sistema, composta pelo Session EJB (descrito na seção 4.8).

4.1.4. Camada de persistência

Camada responsável pelo mapeamento objeto-relacional. Simplifica o acesso aos dados para a camada de negocio, fornecendo um modelo de objetos independente do mecanismo de banco de dados utilizado. Para tal tarefa é utilizado o framework Hibernate (descrito na seção 6).

A figura 1 representa a arquitetura descrita acima:

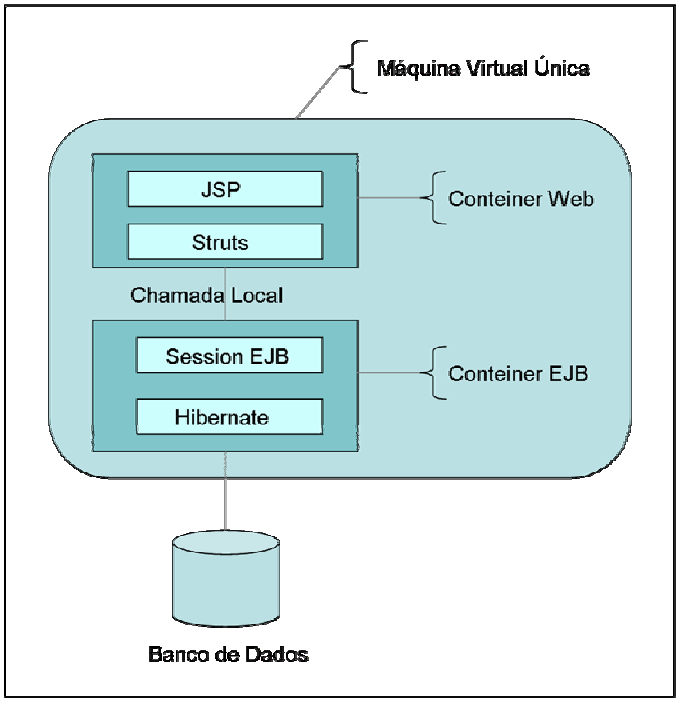


Figura 1. Arquitetura do sistema

Para alcançar esta estrutura, foi utilizado o servidor de aplicação JBoss versão 4.0.2 (http://www.jboss.org). Este servidor de aplicação possui tanto o contêiner Web quanto o contêiner EJB, permitido que ambos possam rodar na mesma máquina virtual, utilizando acesso local em detrimento do acesso via RMI, resultando numa melhor performance.

4.2. Desenvolvimento do Sistema

O objetivo do sistema é gerenciar as atividades relacionadas à FECAM, com isso identificamos dois elementos chave no projeto (Documento / Atividade). Para organizar e acompanhar o andamento de fatos e atividades de interesse da empresa e associados é necessário a geração de arquivos descritivos com detalhes e características que facilitem a busca das informações.

Podemos dividir as operações em duas seqüências distintas: Geração de atividades e criação de documentos.

A figura 2 ilustra o fluxo do cadastro de uma atividade:

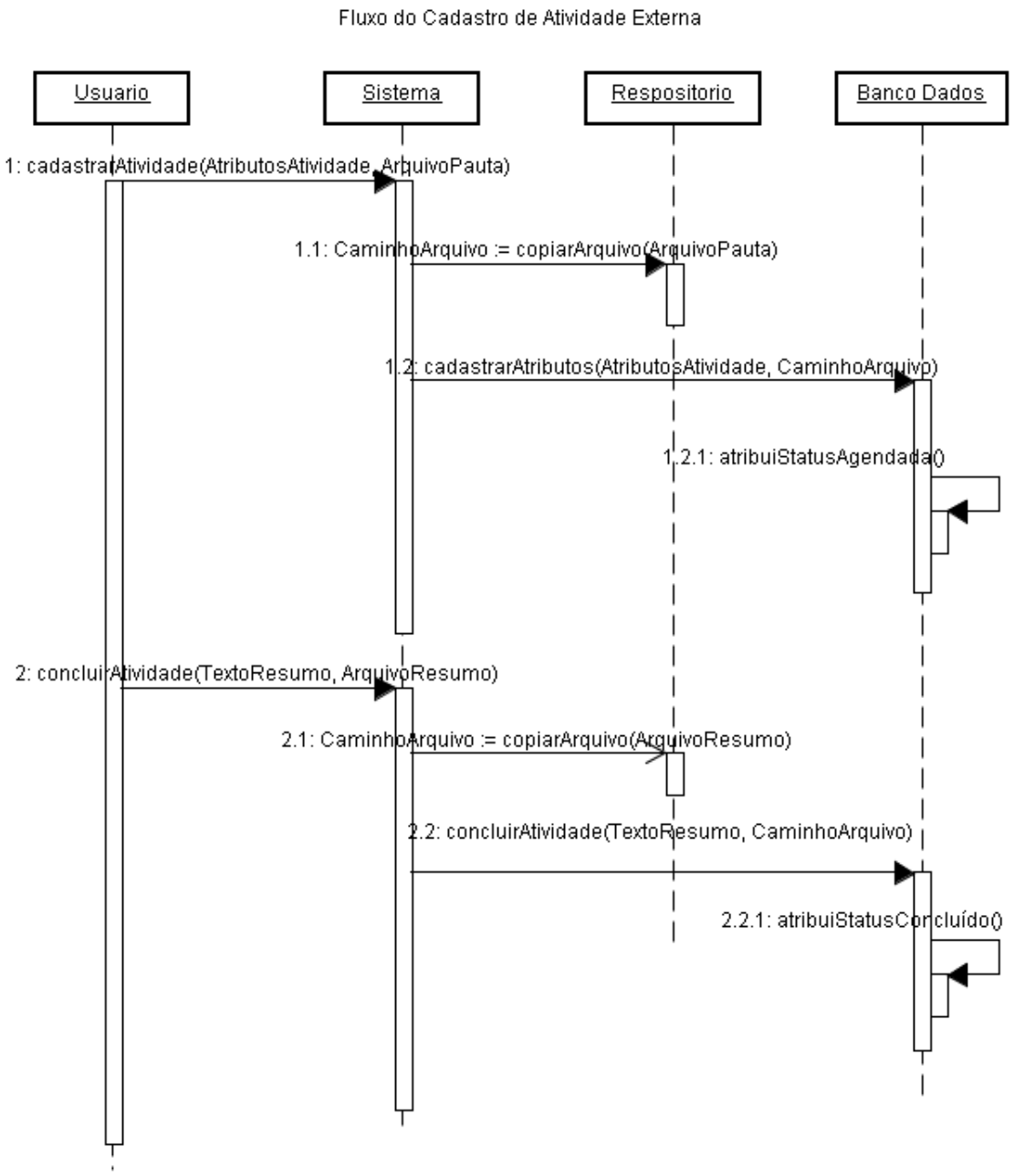


Figura 2. Diagrama de Seqüência do Cadastro de Atividade Externa.

O conselho de representação é composto por duas pessoas (Titular e Suplente), servindo como porta-voz dos conselhos de municípios nas decisões e tomadas de atitudes.

As atividades são pautadas pelos conselhos de representação e gerenciadas por um responsável escolhido em senso comum. Este responsável deve manter a o alinhamento dos objetivos da atividade e fazer cumprir as datas e metas.

O personagem Apoiador fica com a parte operacional do processo de gerenciamento das atividades, organizando arquivos e criando os documentos necessários para o registro das atividades. Suas atividades são acompanhadas diretamente pelo responsável.

A figura 3 ilustra o fluxo do cadastro de documento:

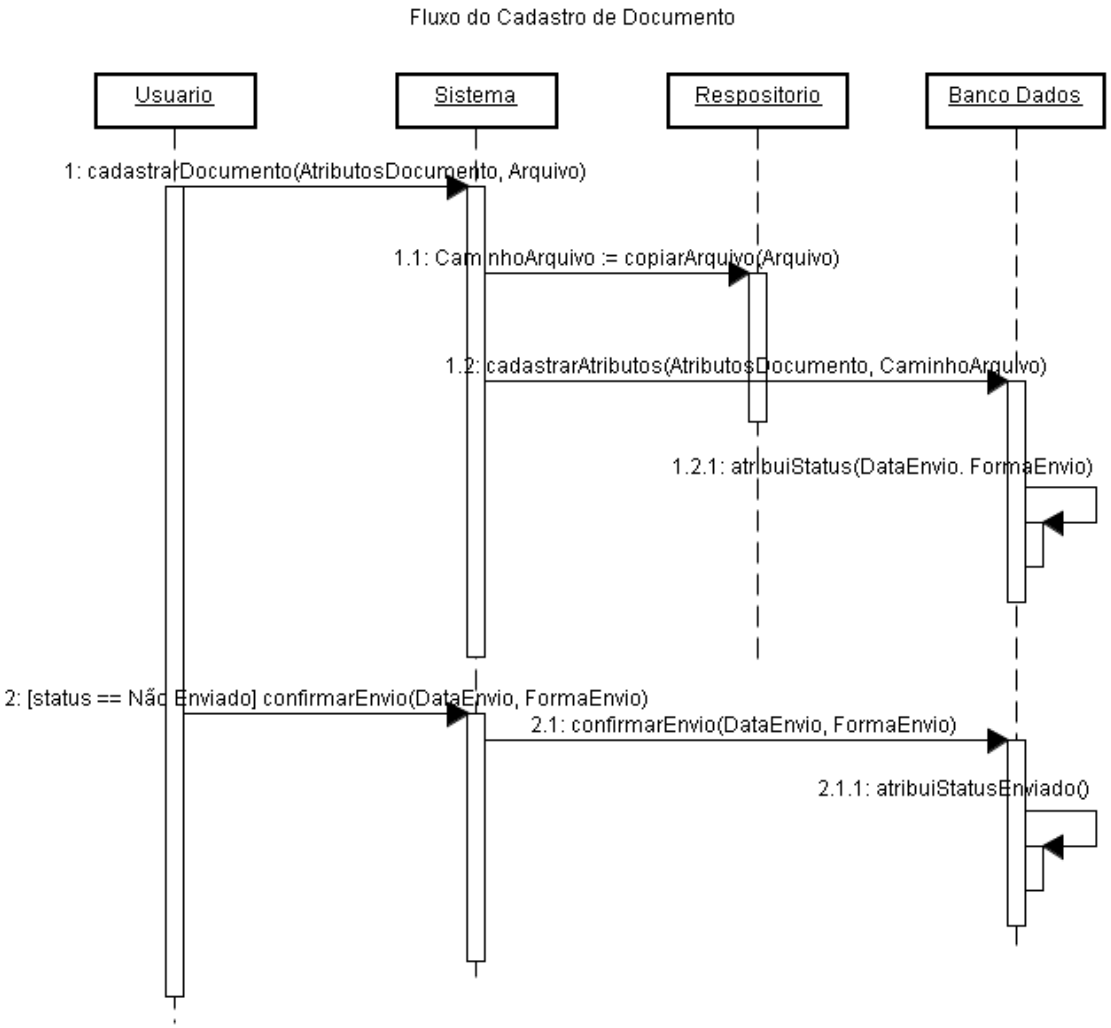


Figura 3. Diagrama de Seqüência do Cadastro de Documento

O Autor elabora o documento e salva no sistema, sendo necessária a assinatura do responsável para posterior envio aos destinatários.

4.2.1. Base de dados.

Com o estudo dos dados coletadas nas fases de levantamento de requisitos chegou-se a seguinte base de dados relativo ao envio de documentos:

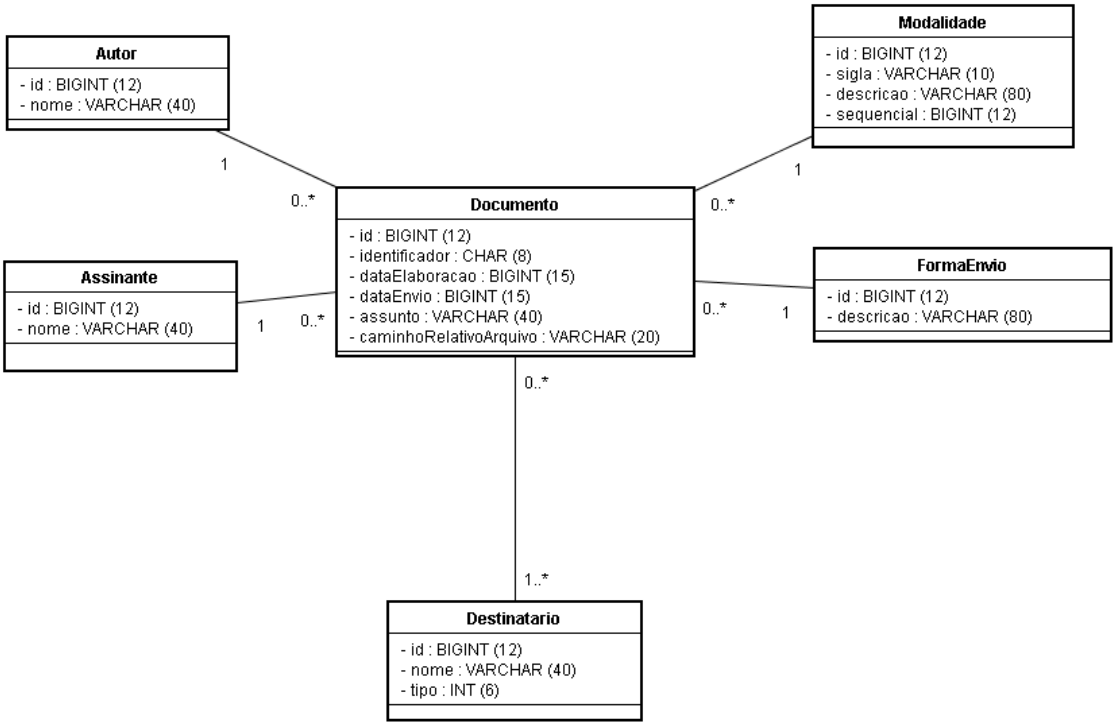


Figura 4. Tabela Documento e seus relacionamentos diretos

Documento é o elemento chave do sistema, haja vista que o sistema tem como objetivo gerenciar e facilitar a elaboração de arquivos, possibilitando que agentes chamados de autores criem arquivos de uma determinada modalidade e encaminhem para os destinatários por uma determinada forma de envio, sendo essa operação autenticada por um assinante.

4.2.2. Classes de Persistência

A persistência implementada em Hibernate torna-se um espelho da base de dados, com a criação de beans que representam as tabelas como abordado na descrição da tecnologia de Persistência de Dados Hibernate descrita neste trabalho.

A figura 5 apresenta o diagrama de classes envolvendo o Objeto principal Documento e os elementos ligados diretamente a ele.

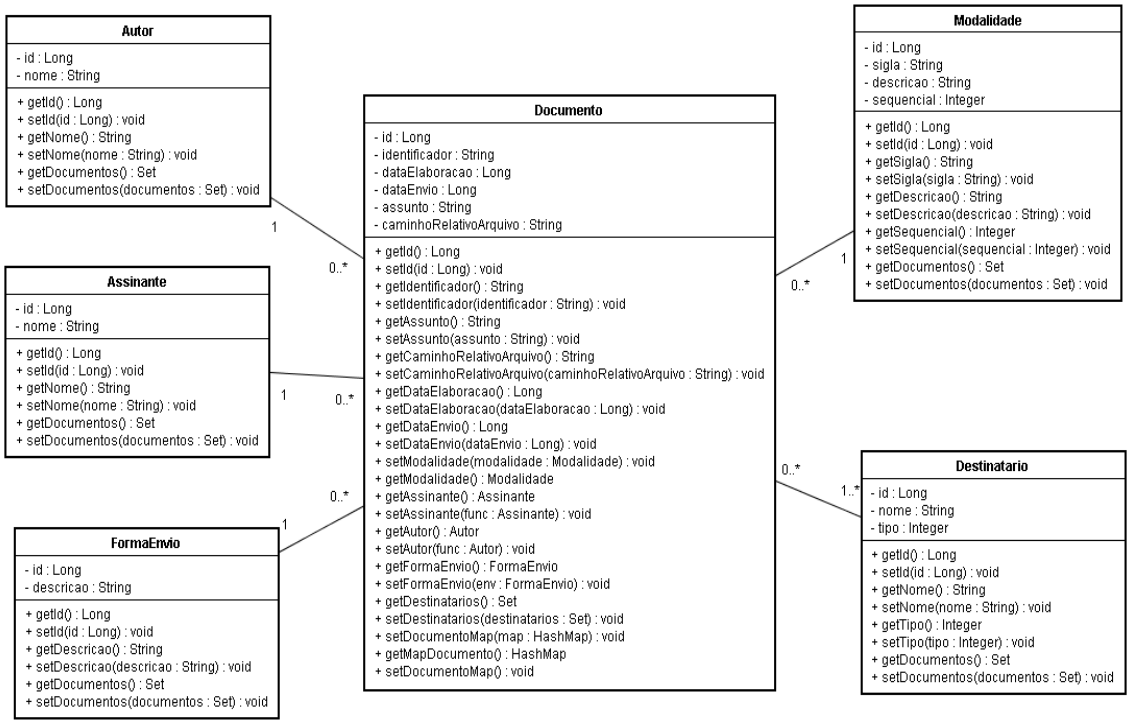


Figura 5. Classe de persistência Documento e demais elementos relacionados.

Com o diagrama é possível constatar o tratamento das tabelas como objetos persistidos no banco de dados, devido a semelhança da base de dados com o diagrama de classes da camada de persistência da aplicação.

A classe documento possui atributos simples como data de envio do arquivo e atributos na forma de objetos relacionados como por exemplo Autor. Para tal é necessária a construção de métodos de acesso para tais objetos, mais conhecidos como GETTERS e SETTERS.

A implementação da persistência partiu da escrita das classes a representarem o banco de dados, sendo o mesmo gerado automaticamente pelas TAGS xdoclet inseridas no código como mostra o exemplo:

(implementação do relacionamento na classe Documento)

```
/**
 * @hibernate.many-to-one column="id_forma_envio" not-null="true"
 *                        class="br.ufsc.cate.hibernate.FormaEnvio"
 *
 */
public FormaEnvio getFormaEnvio() {
      return formaEnvio;
}
```

Esta declaração define que um documento possui uma forma de envio, satisfazendo a regra de negocio.

Mas não podemos esquecer que um relacionamento de constituído por duas entidades, sendo necessário implementa-lo também na classe FormaEnvio:

(implementação do relacionamento na classe FormaEnvio)

```
  /**
   *@hibernate.set role="documentos" inverse="true" lazy="false"
   *@hibernate.collection-key column="id_forma_envio"
   *@hibernate.collection-one-to-many class="br.ufsc.cate.hibernate.Documento"
   * @return
   */
  public Set getDocumentos()
  {
     return documentos;
  }
```

Lembrando que a classe FormaEnvio possui um atributo chamado documentos do tipo java.util.Set que representa uma coleção de documentos relacionados com a FormaEnvio.

A figura 6 apresenta o diagrama de classes completo referente aos objetos de persistência com o banco.

Diagrama completo

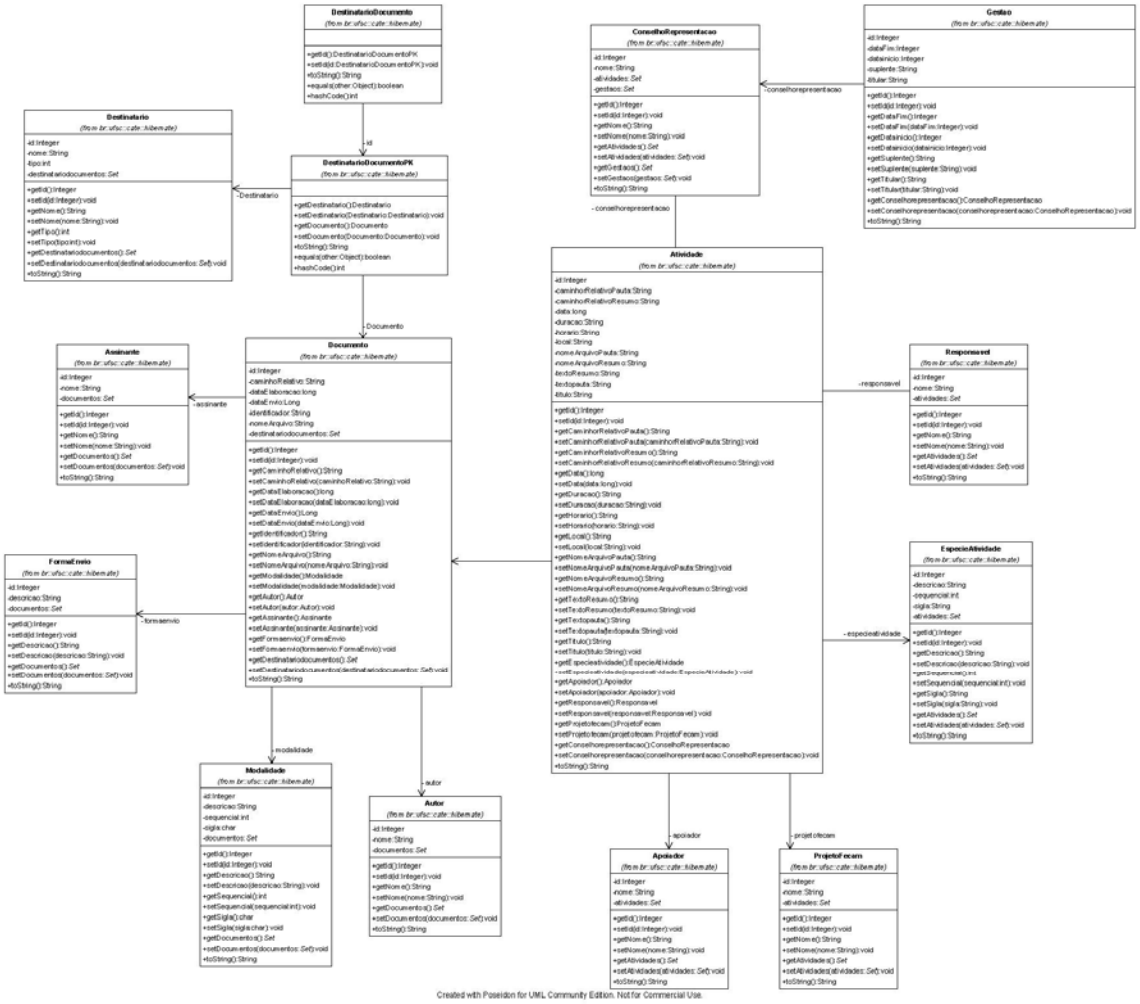


Figura 6. Diagrama das classes de Persistência

4.2.3. Interface com o usuário

Neste capítulo serão apresentados os aspectos relevantes da interface da aplicação com o usuário.

As necessidades do sistema apontaram para um layout simples, primando pela eficiência e desempenho, gerando um site com boa navegabilidade.

Estes resultados foram alcançados com o uso de tecnologias como JavaScript para validação de campos; taglibs do Struts para montagem dinâmica dos campos de formulários, taglig open-source displaytag para geração de relatórios. Tais características perceptíveis nas telas demonstradas a baixo:

Figura 7. Cadastro de atividade externa.

Figura 8. Pesquisa de atividade externa.

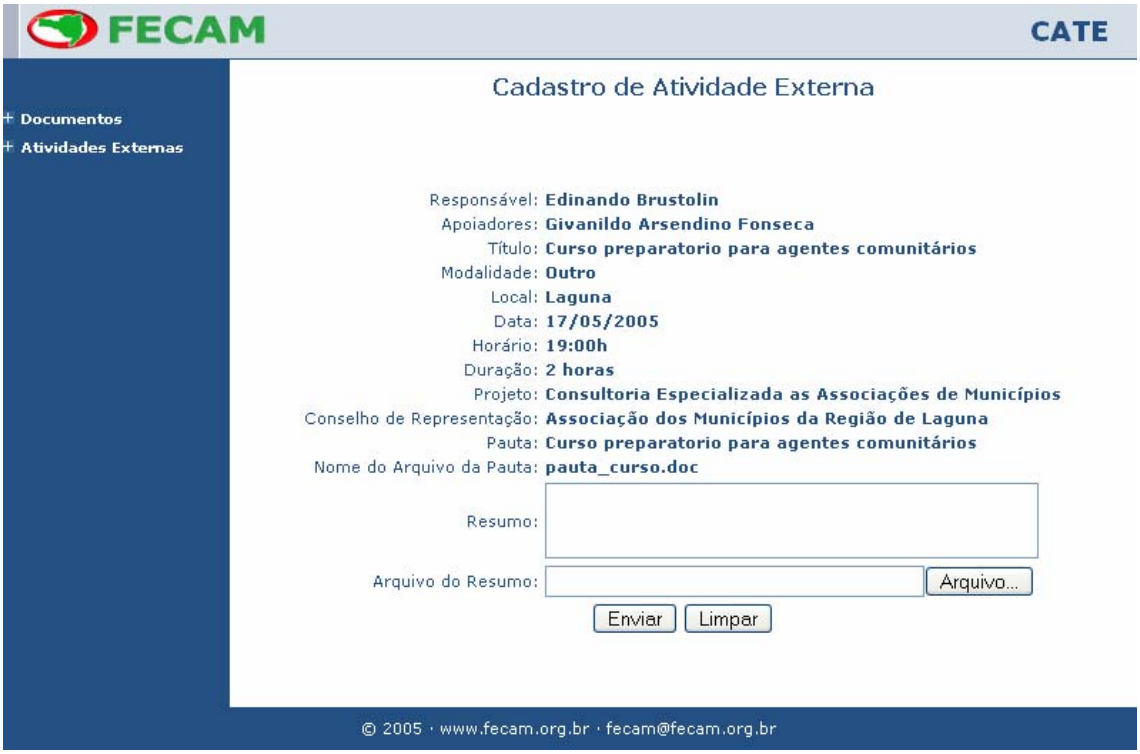


Figura 9.Conclusão de atividade externa.

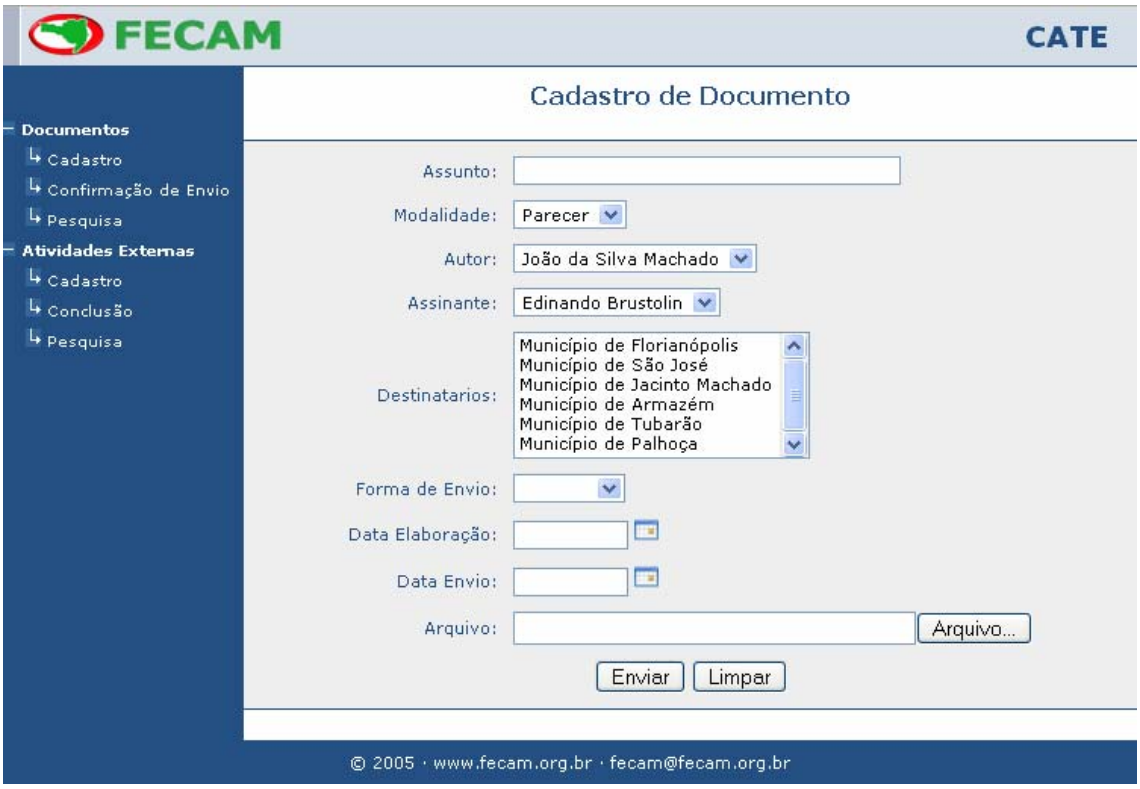


Figura 10. Cadastro de documento.

Figura 11. Pesquisa de documento

Figura 12. Confirmação do envio de documento.

4.3. Testes

Para testar o sistema foram utilizados dois frameworks de testes, um para a camada de persistência e outro para testes de aceitação. Ambos são extensões do framework de teste JUnit (http://www.junit.org).

4.3.1. Testes da camada de persistência.

Para os testes da camada de persistência utilizamos o framework DbUnit. Este framework utiliza arquivos XML para definir os dados que devem constar no banco de dados no momento em que o teste é executado. Assim é possível escrever testes para a camada de persistência tendo certeza de quais dados constarão no banco de dados.

Maiores detalhes sobre o DbUnit podem ser encontrados em http://www.dbunit.org/ .

4.3.1.1. Exemplo de utilização:

Código do arquivo XML para teste da classe Assinante que representa a tabela assinantes no banco de dados:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<dataset>
      <ASSINANTE id=”1” nome=”Edinando Brustolin” />
      <ASSINANTE id=”2” nome=”Kleyton Weber da Silva” />
      <ASSINANTE id=”3” nome=”Sergio Ferreira de Mendonça” />
</dataset>
```

Trecho de código da classe de teste da classe assinante:

```
public void testGetByPrimaryKey() {
      Session session = HibernateUtil.currentSession();
      Assinante assinante = (Assinante) session.get(Assinante.class,
Integer.valueOf(1));
      assertEquals(“validando o id do assinante”, Integer.valueOf(1),
assinante.getId());
```

```
        assertEquals(“validando o nome do assinante”, “Edinando Brustolin”,
    assinante.getNome());
    }
```

Desta forma, sabemos que no banco de dados estarão os dados contidos no arquivo XML e assim podemos verificar se a busca pela chave primária do Assinante está correta. Buscamos o Assinante com a chave primária 1 e validamos se o objeto retornado realmente possui os valores esperados.

4.3.2. Testes de aceitação

Para os testes de aceitação utilizamos o framework Struts Test Case for JUnit (STC). Este framework provê uma série de facilidades para testes em aplicações baseadas no framework Struts. O STC pode ser utilizado de duas formas, uma que utiliza o framework MockObjects para simular o contêiner Web, podendo assim realizar o teste fora do mesmo e outra que utiliza o framework Cactus para realizar o teste dentro do contêiner Web.

Como o STC está sendo utilizado para testes de aceitação, utilizamos a versão Cactus para realizar o teste dentro do contêiner. Desta forma podemos testar situações reais de uso do sistema, um dos principais objetivos do teste de aceitação.

Maiores informações sobre os frameworks citados em: STC - http://strutstestcase.sourceforge.net/ , MockObjects - http://www.mockobjects.com/ e Cactus - http://jakarta.apache.org/cactus/ .

Exemplo de utilização:

Trecho do código da classe de testes da action de cadastro de atividades externas.

```
1 public void testFaltandoParametros() {
2       setRequestPathInfo("/cadastroAtividadeExterna");
3       addRequestParameter("assunto","Assunto Teste");
4       actionPerform();
5       verifyForward("cadastro.atividade.externa");
6       verifyActionErrors(new String[]  {"erro.parametros.obrigatorios"});
7 }
```

Na linha 2, setamos qual action será chamada, no caso a de cadastro de atividades externas. Em seguida, na linha 3, adicionamos o parâmetro *assunto* à requisição. A linha quatro faz com que a action seja executada. Como adicionamos apenas um parâmetro à requisição, a action deve retornar para a página de cadastro e informar o erro ocorrido. Na linha 5 verficamos se a action retornou para a página de cadastro e na linha 6 verficamos se ela informou o erro.

5. A ARQUITETURA J2EE

5.1. Introdução

Segundo (YUNG, 2004), os sistemas de software sofreram grandes impactos quanto à forma e metodologia de desenvolvimento, necessárias frente ao avanço da tecnologia, assim, as aplicações precisaram trabalhar com grau cada vez maior de complexidade, passando a trabalhar com vários usuários, múltiplas máquinas, e até acesso distribuído.

Para resolver cada aspecto envolvido nessa crescente complexidade, era necessário desenvolver além da aplicação desejada, todos os aspectos de consistência, transação, acesso a dados, controle de sessão e muitos outros tão complexos quanto.

A abordagem da tecnologia Java para este problema foi um modelo arquitetural aberto, composto de camadas, onde teríamos a descrição por funcionalidades de elementos de tecnologias equivalentes e que seguissem uma forma padronizada de funcionamento e comunicação, que poderiam facilmente ser substituídos por outro equivalente. Desta abordagem, temos como resultado a plataforma Java 2 Enterprise Edition (J2EE).

Mais do que simplesmente ser uma linguagem Java no lado do servidor, a especificação J2EE define como uma aplicação criada faz para ter acesso à determinada funcionalidade ou serviço da especificação J2EE, que deve entender esta estrutura de acesso.

A tecnologia J2EE baseia-se em provedores de funcionalidades, os chamados “contêineres”, que são aplicações servidoras. Essas aplicações ou servidores de aplicação irão cuidar de fornecer serviços e infra-estrutura, são elas que se encarregam de garantir todos os aspectos necessários para que tenhamos uma aplicação 100% funcional: controle de transação, persistência de objetos, comunicação com outras aplicações, segurança, controle de acesso, etc.

Então a especificação não é uma nova linguagem, e sim uma nova abordagem que visa simplificar o modelo de desenvolvimento de aplicações mais complexas. De fato, a plataforma J2EE é executada sobre a plataforma Java 2 Standard Edition (J2SE); além disto, todos os componentes desenvolvidos para a plataforma J2EE são desenvolvidos em Java e compilados para *bytecode* Java.

5.2. Tecnologias Envolvidas no J2EE

Segundo (YUNG, 2004), a plataforma J2EE se utiliza de tecnologias padronizadas Java, tanto para o desenvolvimento de aplicações, as chamadas tecnologias primárias, quanto para prover suporte ao acesso às funcionalidades, chamadas de tecnologias de suporte.

5.2.1. Tecnologias Primárias

Define-se como tecnologia primária aquelas utilizadas diretamente pelo desenvolvedor para gerar a aplicação desejada. São elas:

5.2.1.1. Sevlets

São componentes que interagem com outros componentes clientes seguindo um modelo de requisição/resposta, ou seja, a aplicação cliente irá invocar um método específico de um Servlet, que irá disparar a execução de uma série de atividades. O uso mais comum é para acesso a clientes Web. Seu uso pode ser tanto para a apresentação da aplicação , quanto para a lógica de negócios (YUNG, 2004).

5.2.1.2. JavaServer Pages (JSP)

São páginas em texto formatado, contendo porções que serão convertidas e entendidas como código Java. Estas porções são compiladas em uma aplicação Java, no momento que se necessita do seu uso. O uso mais comum é em paginas dinâmicas, onde criam-se páginas mesclando a linguagem HTML com porções de código Java para ter acesso a componentes e dados de aplicações Java. Seu uso é recomendado para a apresentação da aplicação (YUNG, 2004).

5.2.1.3. Enterprise JavaBeans (EJB)

São componentes flexíveis voltados para aplicações de negócio distribuídas ou de maior complexidade. Podem representar uma entidade de negócio (Entity Beans), uma lógica de negócio (Session Beans) ou uma comunicação assíncrona (Message-Driven Beans). Neles está encapsulado o funcionamento de aspectos intrínsecos do sistema, tais como o gerenciamento de instâncias simultâneas, suporte a execução do componente através de ambientes distribuídos, etc (YUNG, 2004).

5.2.2. Tecnologias de Suporte

As tecnologias de suporte ou serviços são aquelas usadas para prover acesso a uma determinada funcionalidade desejada, sejam recursos, transação, comunicação, outros sistemas ou autenticação. Existem inúmeras tecnologias possíveis de serem usadas como tecnologia de suporte à plataforma J2EE. A seguir, serão citadas as de uso mais comum:

5.2.2.1. Java DataBase Connectivity (JDBC)

Trata-se de uma API para conexão com Banco de Dados. Ela que permite executar comandos SQL a partir de métodos de linguagem de programação Java. Quando a persistência dos dados não é gerenciada pelo contêiner, a API JDBC é utilizada para executar os comando SQL no banco de dados.

Ela possui duas partes: uma interface em nível de aplicação usada pelos componentes da aplicação para acessar um banco de dados e uma interface provedora de serviço para anexar um driver JDBC à plataforma J2EE (SUN, 2003).

5.2.2.2. Java Transaction API (JTA)

Trata o suporte à transação, ou seja, a garantia que uma atividade seja realizada do início ao fim em caso de sucesso, e ainda, que seja desfeito aquilo que foi realizado em caso de insucesso de alguma etapa entre o início e o fim do bloco definido (YUNG, 2004).

5.2.2.3. Java Message Service (JMS)

Provê suporte a comunicação assíncrona entre componentes. Os componentes irão receber comunicação gerada por uma aplicação fonte utilizando um servidor de mensagens, que se encarrega de guardar e distribuir a mensagem para os diversos clientes associados (YUNG, 2004).

5.2.2.4. Java Naming and Directory Interface (JNDI)

Trata-se de uma interface unificada para acessar diferentes tipos de serviços de nome e diretório. A JNDI é utilizada para registrar e consultar componentes de negócios e outros objetos orientados a serviço em um ambiente J2EE. Ele inclui suporte para Lightweight Directory Access Protocol (LDAP), para CORBA Object Services (COS), serviço de nome e o Java RMI Registry (BODOFF, 2002) (SUN, 2003). Isso permite que as aplicações J2EE coexistam com aplicações e sistemas legados.

5.2.2.5. JavaMail API

Utilizada para enviar notificações de correio eletrônico. A API JavaMail tem duas partes: uma interface em nível de aplicação usada pelos componentes da aplicação para enviar mensagens de correio e uma interface de provedor de serviços.

5.2.2.6. JavaBeans Activation Framework (JAF)

Proporciona serviços padrão para determinar o tipo de uma parte arbitrária de dados, encapsular o acesso a ela, descobrir as operações disponíveis para ela e criar o componente JavaBeans apropriado para executar essas operações (BODOFF et. al, 2002).

5.2.2.7. Java API for XML Processing (JAXP)

Fornece suporte ao processamento de documentos XML usando Document Object Model (DOM), Simple API for XML (SAX) e Extensible Styleshet Language Transformations (XSLT) (SUN, 2003) (W3C, 2003).

O JAXP permite que as aplicações analisem e transformem documentos XML independente de uma implementação de processamento XML específica. Uma aplicação J2EE pode usar XML para produzir relatórios e as diferentes empresas que recebem esses relatórios podem tratar os dados da maneira mais adequada às suas necessidades. Uma empresa poderia processar os dados XML em um programa para converter o XML em HTML de forma a publicar os relatórios na web, outra poderia processar os dados XML em uma ferramenta para criar uma apresentação de marketing e, ainda, outra empresa poderia ler os dados XML para processá-los em uma aplicação J2EE.

5.2.2.8. Java Connector Architecture (JCA)

Suporta o acesso a sistemas de informações diversos, permite a conexão entre aplicações Java com aplicações que geram dados, utilizando uma implementação JCA adequada. Pode ser tratado desta forma o acesso a informações de dados gerados por sistema de folhas de pagamento ou de gerência de relacionamento com cliente, por exemplo. (YUNG, 2004).

5.2.2.9. Java Authentication and Autorization Service (JAAS)

É um pacote Java que permite aos serviços autenticar e reforçar o controle de acesso de usuários. É um mecanismo de autenticação e autorização completamente separado do código da aplicação. O mecanismo mais simples é uma combinação nome/senha do usuário, mas a especificação também proporciona suporte para dispositivos como cartões magnéticos, leitores de digitais ou leitores de retina sem modificações na aplicação (OTN, 2004).

5.2.2.10. JavaServer Faces (JSF)

A tecnologia JavaServer Faces simplifica a construção de interfaces para aplicações JavaServer. Desenvolvedores podem construir aplicações Web rapidamente da seguinte forma: montando componentes de interface reusáveis numa página; conectando esses componentes a fonte de dados da aplicação; e associando eventos gerados pelo cliente a eventos de tratamento no lado do servidor.

5.3. Tecnologia de Componentes

Um componente é uma unidade de software em uma camada da aplicação. Juntamente com componentes JavaBeans, que fazem parte da plataforma J2SE, a plataforma J2EE especifica os seguintes componentes (SINGH, e STEARNS e JOHNSON, 2002):

- Clientes da aplicação e applets são componentes que são executados no cliente.
- Componentes da tecnologia Java Servlet e JavaServer Pages (JSP) são componentes Web que são executados no servidor.
- Componentes Enterprise JavaBeans (EJB) são componentes de negócios que são executados no servidor.

Todos os componentes J2EE, em tempo de execução, dependem de uma entidade chamada do contêiner. Os contêineres permitem o gerenciamento do ciclo de vida dos componentes, segurança, publicação e controle de concorrência. A arquitetura J2EE é composta das seguintes camadas:(ALUR e CRUPI e MALKS, 2002)

- *Cliente* - A camada do cliente interage com o usuário e exibe as informações do sistema para o usuário. A plataforma J2EE suporta diferentes tipos de clientes, incluindo clientes HTML, applets Java e aplicações Java.
- *Web* - A camada da Web gera uma lógica de apresentação e aceita as respostas do usuário a partir dos clientes da apresentação, que são a partir dos clientes da apresentação, que são normalmente clientes HTML, Applets Java e outros clientes da Web. Baseada na solicitação recebida do cliente, a camada de apresentação gera a resposta apropriada para uma solicitação recebida do cliente. Na plataforma J2EE, os servlets e JSPs em um contêiner da Web implementam essa camada.
- *Negócios* - Essa camada trata da principal lógica de negócios da aplicação. Ela fornece as interfaces necessárias aos componentes de serviços de negócios subjacentes. Os componentes de negócios são, normalmente, implementados como componentes EJB com suporte a partir de um contêiner EJB, facilitando o ciclo de vida do componente e gerência à persistência, as transações e a alocação de recursos.
- *EIS (Enterprise Information Systems)* - Esta camada é responsável pelo sistema de informações da empresa, incluindo os sistemas de banco de dados, o de processamento de transação, os sistemas legados e os sistemas de planejamento de recurso da empresa. A camada EIS é o ponto onde as aplicações J2EE se integram com os sistemas que não são J2EE e sistemas legados.

A figura 13 mostra como as camadas e contêineres são organizados na arquitetura J2EE.

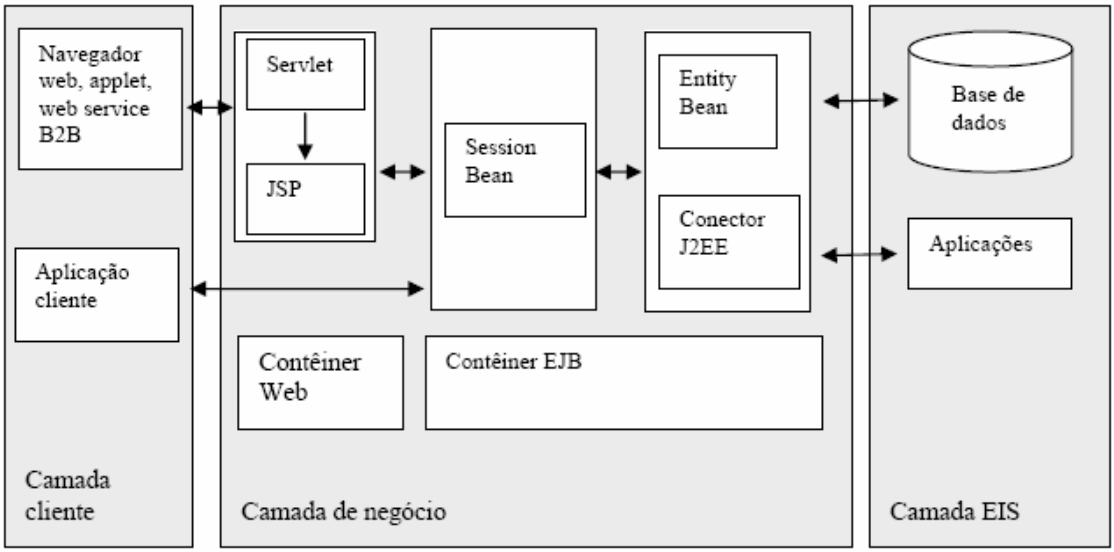


Figura 13. Arquitetura J2EE básica (SUN, 2004)

5.4. Infra-Estrutura J2EE

Segundo (YUNG, 2004), o modelo de arquitetura definido na J2EE implica no uso de uma infra-estrutura de suporte que irá conter o código de aplicação criado pelo desenvolvedor, os chamados "Contêineres". Estes contêineres são aplicações provedoras de serviços, que irá implementar as APIs definidas para o tipo de serviço oferecido.

Assim teremos diferentes contêineres apropriados para cada tipo de serviço demandado na J2EE:

- *Contêiner Web* – Provê uma JVM para as aplicações Java destinadas a atender requisições Web, tais como Servlets e JavaServer Pages, e ainda podem atender a requisições de web Services.
- *Contêiner EJB* – Destinado a suportar as funcionalidades demandas pelos componentes EJB, provêem um ambiente de execução para suportar as aplicações distribuídas, controle transacional, persistência, pool de objetos e etc.

- *Outros Contêineres* – Podemos ter outros contêineres que irão prover serviços tais como: serviços de mensagens, serviços de e-mail, controle transacional, serviços de segurança, serviços de acesso, serviços de diretório, etc.
- A grande quantidade de serviços usados em conjunto resulta no uso de diversos contêineres específicos para cada tecnologia a ser utilizada ou suportada, no entanto a especificação J2EE define muito claramente as maneiras de interações e os papéis de cada uma na solução como um todo, desta maneira, podemos usar contêineres diversos (até mesmo de fabricantes distintos) para compormos um ambiente J2EE.

5.5. Aplicações Clientes

Segundo (YUNG, 2004), as aplicações clientes que irão acessar os componentes J2EE não precisam ser necessariamente clientes web ou clientes Java. A arquitetura é ampla e o acesso através de diversos protocolos permite o uso de clientes mais variados tais como:

- *Navegadores* – Aplicações irão utilizar os protocolos de rede tais como HTTP e FTP para acessarem Servlets ou JSPs que irão atender as requisições de contudo.
- *Applets* – Código Java em formato de Applet, distribuído como uma parte de uma página Web. Pode acessar uma aplicação Java no servidor, e ainda requisitar dados e serviços.
- *Aplicação Java WebStart* – Código Java que irá utilizar a JRE (ambiente de execução Java) associada ao sistema operacional e não presa ao navegador.
- *Aplicações Java e Não-Java Distribuídas* – Aplicações podem acessar aplicações J2EE, seja por meio de protocolos Internet (http, etc), seja por meio de *sockets* de rede ou ainda, através de protocolos de aplicações distribuídas (RMI, CORBA, Web Services, etc).
- *Aplicações J2EE* – Qualquer aplicação J2EE pode conversar com outra aplicação J2EE, seja através de protocolos de aplicações distribuídas, seja por meio de criação de *clusters* de contêineres.

5.6. Servlets

Devido à importância da geração de conteúdo dinâmico para o desenvolvimento da web, foi natural que a Sun propusesse extensões para Java em seu domínio. Da mesma forma que a Sun introduziu applets como pequenas aplicações baseadas em Java para adicionar funcionalidade interativa aos navegadores da web, em 1996 a Sun introduziu servlets como pequenas aplicações baseadas em Java para adicionar funcionalidade dinâmica a servidores da web. Os servlets de Java têm um modelo de programação similar aos scripts de CGI, na medida em que eles recebem uma solicitação de HTTP de um navegador da web como input e espera-se que localizem e/ou construam o conteúdo apropriado para a resposta do servidor (FIELDS e KOLB, 2000).

Diferentemente dos programas tradicionais baseados em CGI que necessitam da criação de um novo processo para tratar de cada nova solicitação, todos os servlets associados com um servidor da web rodam dentro de um único processo. Este processo roda uma Java Virtual Machine (JVM), que é o programa específico de plataforma para rodar programas de Java compilados (compatível com várias plataformas). Ao invés de criar um novo processo a cada solicitação, a JVM cria um encadeamento de Java para tratar de cada solicitação de servlet.

Em essência, servlets fornecem uma metodologia baseada em Java para mapear solicitações de HTTP em respostas HTTP. A geração de conteúdo da web dinâmico usando-se servlets é, então, realizada através de código Java que fornece a HTML (ou outros dados) representando aquele conteúdo. No caso dos dados de HTML, uma abordagem é fazer com que o código de Java construa cadeias contendo o texto de marcação apropriado e depois imprima estas cadeias no fluxo de saída associado com a resposta de HTML. Isto é, freqüentemente, chamado de abordagem out.println, porque uma parte significativa do código resultante consiste de linhas que começam com esta seqüência de caracteres (ou uma seqüência bem similar).

Um servlet é uma classe da linguagem de programação Java usada para estender as capacidades dos servidores que hospedam aplicações acessadas via um modelo de programação do tipo solicitação, eles são mais usados para estender as aplicações

hospedadas em servidores Web. Para tais aplicações, a tecnologia Java Servlet define classes servlet específicas para http (BODOFF et al, 2002).

5.7. JSP

Um componente JavaServer Pages (JSP) é um tipo de Java servlet que é projetado para exercer o papel de interface com o usuário em uma aplicação Java para Web. Os desenvolvedores Web escrevem JSPs como um arquivo texto que combina código HTML ou XHTML, elementos XML e ações e comandos JSP embutidos. JSPs são originalmente parecidos com o modelo de ferramentas de script do lado servidor, como a tecnologia ASP, da Microsoft Corporation. JSPs estão focados principalmente em elementos XML, incluindo elementos ou *tags* customizadas como o principal método de geração dinâmica de conteúdo web.

Um compilador JSP automaticamente converte o documento texto em um servlet. O contêiner Web cria uma instância do servelt e o deixa disponível para manipular requisições. Estas ações são transparentes para o desenvolvedor, que não precisa se preocupar com esta tradução do código fonte em servlet.

Desta forma o desenvolvedor preocupa-se apenas na dinâmica do JSP e com os elementos e tags customizadas utilizadas para gerar as respostas. Desenvolver JSPs como um documento de texto ao invés código Java permite que disigners possam trabalhar na parte gráfica do HTML, deixando as tags XML e o conteúdo dinâmico para os desenvolvedores. (PERRY, 2004)

5.8. EJB

Segundo (ARMSTRONG, 2004), enterprise beans são components J2EE™ que implementam a tecnologia Enterprise JavaBeans™ (EJB™). Enterprise beans rodam dentro de um container EJB, um ambiente contido no servidor J2EE. Embora transparente para o desenvolvedor da aplicação, o container EJB provê para os EJBs serviços de nível de sistema, como transações. Estes serviços permitem aos EJBs um rápido desenvolvimento e instalação, e também faz com que os mesmos tornem-se o núcleo de aplicações J2EE transacionais.

Escritos na linguagem de programação Java, enterprise beans são os componentes do lado servidor que encapsulam as regras de negócio de uma aplicação. A lógica de negócios é a parte do código que representa o propósito da aplicação.

5.8.1. Benefícios dos Enterprise Beans

Por várias razões, os enterprise beans simplificam o desenvolvimento de aplicações grandes e distribuídas. Primeiro, porque o container EJB provê e é resposável pelos serviços de nível de sistema (como controle transações e autenticações de segurança) permitindo ao desenvolvedor se concentrar em resolver problemas da lógica de negócios.

Segundo, porque os beans contêm a lógica de negócio da aplicação, deixando o desenvolvedor da camada cliente focar-se na apresentação. O desenvolvedor da camada cliente não precisa codificar as rotinas que implementam as regras de negócio ou o acesso a bases de dados. O resultado disto é uma camada cliente mais enxuta, um benefício importante para clientes que rodam em pequenos dispositivos.

Terceiro, porque os enterprise beans são componentes portáveis que podem ser reutilizados na construção de novas aplicações.

5.8.2. Quando usar Enterprise Beans

Deve-se considerar a utilização de enterprise beans se a aplicação possui algum dos seguintes requerimentos:

- A aplicação deve ser escalável. Para acomodar um número crescente de usuários, é necessário distribuir os componentes da aplicação em múltiplas máquinas. Os enterprise bean não só podem rodar em diferentes máquinas, como sua localização fica transparente para a camada cliente.
- Transações são requeridas para garantir a integridade dos dados. Os enterprise beans suportam transações, os mecanismos que controlam o acesso concorrente a objetos compartilhados.
- A aplicação terá uma diversidade de clientes. Com apenas poucas linhas de código clientes remotos podem facilmente localizar enterprise beans. Estes clientes podem ser pequenos, diversos e numerosos.

5.8.3. Tipos dos Enterprise Beans

Abaixo segue um resumo dos três tipos de EJBs existentes:

- Session Bean – Representa um cliente único dentro do servidor J2EE. Para acessar uma aplicação instalada no servidor, a camada cliente invoca os métodos do session bean. O session bean realiza o trabalho para a camada cliente, protegendo-a da complexidade da lógica de negócio contida dentro do servidor.
- Entity Bean - Representa um objeto de negócio em um mecanismo de armazenamento persistente. Tipicamente, cada classe de entity bean possui uma tabela correspondente em uma base de dados relacional e cada instancia desta classe corresponde a uma linha desta tabela.
- Message-Driven Bean – É um enterprise bean que permite às aplicações J2EE processar mensagens de forma assíncrona. Ele atua como uma *listener* de mensagens JMS, similar a um *listener* de eventos com a exceção de que ele recebe mensagens ao invés de eventos. Estas mensagens podem ser enviadas por qualquer componente J2EE – uma aplicação cliente, outro enterprise bean ou um componente web – ou por um sistema ou aplicação JMS que não utiliza a tecnologia J2EE.

6. A ARQUITETURA DO STRUTS

6.1. Visão Geral do Struts

A Struts Framework é um projeto open source mantido pela Apache Software Foundation. É uma implementação do design pattern MVC (Model-View-Controller) para aplicações java com internet. O objetivo do pattern MVC é separar de maneira clara a camada de apresentação (View) da camada de Negócio (Model).

A arquitetura MVC - Model-View-Controller (Modelo-Visualização-Controle) é um padrão que separa de maneira independente o Modelo, que representa os objetos de negócio (Model) da camada de apresentação, que representa a interface com o usuário ou outro sistema (View); e o Controle de fluxo da aplicação (Controller).

A Struts Foi escrita por Craig McClanahan em Maio de 2000, e desde então vem sendo melhorado pela comunidade open-source. Foi desenvolvida com o objetivo de fornecer uma framework para facilitar o desenvolvimento de aplicações para web.

6.1.1. Motivos para utilizar a Struts Framework

- Tornou-se um padrão de mercado;
- Garantia de que alguém (Apache Group) irá manter a framework (correção de bugs e novos releases);
- Integração com a maioria das IDEs de mercado
- Não reinventar a roda, focando os seus esforços em regras de negócio;
- Separar a camada de negócio da camada de apresentação;
- Já incorpora diversos design patterns
- Criação de aplicações padronizadas, facilitando a manutenção;
- Criação de Aplicações Internacionalizadas;
- Possibilidade de gerar a saída de acordo com o dispositivo usado (HTML, SHTML, WML, etc.);
- Aumentar a produtividade.

6.1.2. Detalhes do funcionamento

O usuário faz uma solicitação através de uma url no browser.

Ex: http://localhost:8080/cadastro/listUsers.do. Note que no final da url tem um sufixo **.do** que será usado para invocar (na verdade mapear) o servlet controller da struts.

Se for a primeira solicitação que o container recebeu para esta aplicação, ele irá invocar o método init() da ActionServlet (controller da Struts) e irá carregar as configurações do arquivo struts-config.xml em estruturas de dados na memória. Vale lembrar que esta passagem só será executada uma única vez, pois nas solicitações subseqüentes, a servlet consulta estas estruturas na memória para decidir o fluxo a ser seguido.

Baseado no fluxo definido no arquivo struts-config.xml, e que neste momento já se enc0ontra carregado em estruturas na memória, o ActionServlet identificará qual o ActionForm (classe para a validação dos dados) irá invocar. A classe ActionForm através do método **validate** irá verificar a integridade dos dados que foram recebidos na solicitação que vem do browser.

O controle da aplicação é retomado pelo ActionServlet, que verifica o resultado da verificação do ActionForm.

Se faltou alguma coisa (campo não preenchido, valor inválido, etc.), o usuário recebe um formulário html (geralmente o mesmo que fez a solicitação), informando o motivo do não atendimento da solicitação, para que o usuário possa preencher corretamente os dados para fazer uma nova solicitação.

Se não faltou nenhuma informação, ou seja, todos os dados foram enviados corretamente, o controller passa para o próximo passo (Action).

O ActionServlet, baseado no fluxo da aplicação (estruturas já carregadas em memória) invoca uma classe Action. A classe Action passará pelo método **execute** que irá delegar a requisição para a camada de negócio.

A camada de negócio irá executar algum processo (geralmente popular um bean, ou uma coleção). O resultado da execução deste processo (objetos já populados) será usado na camada de apresentação para exibir os dados.

Quando o controle do fluxo da aplicação voltar ao Action que invocou o processo da camada de negócio, será analisado o resultado, e definido qual o mapa adotado para o fluxo da aplicação. Neste ponto, os objetos que foram populados na camada de negócio serão inseridos como atributos na seção do usuário.

Baseado no mapeamento feito pelo o Action, o Controller faz um forward para o JSP para apresentar os dados.

Na camada de apresentação (View), os objetos que foram setados como atributos da sessão do usuário serão consultados para montar o html para o browser.

Chega o html da resposta requisitada pelo usuário.

O Controller já vem implementado na Struts, embora, caso seja possível estendê-lo a fim de adicionar funcionalidade. O fluxo da aplicação é programado em um arquivo XML através das ações que serão executadas. As ações são classes base implementadas pela framework seguindo o padrão MVC. Assim devemos estendê-las a fim de adicionar a funcionalidade desejada.

A geração da interface é feita através de custom tags, também já implementadas pela Struts, evitando assim o uso de Scriptlets (códigos java entre <%> e <%>), deixando o código JSP mais limpo e fácil de manter.

7. A ARQUITETURA DO HIBERNATE

7.1. Visão Geral do Hibernate

Trabalhar com dados relacionais implica em adaptações dos modelos de objetos, aumentando o esforço aplicado no processo de persistência manual dos objetos no banco de dados e obrigando os desenvolvedores de aplicações a utilizar a linguagem SQL para acessos aos dados. Isto significa uma redução considerável na qualidade do produto final, a construção de uma modelagem orientada a objetos inconsistente e um desperdício considerável de tempo na implementação manual da persistência.

Uma solução adotada por muitos desenvolvedores na implementação da persistência de objetos se dá através da inclusão de instruções SQL na codificação da lógica da aplicação. Tal fato, muitas vezes implica no acoplamento do sistema ao SGBD utilizado. Assim, alterações na estrutura das tabelas ou do próprio banco de dados implicam em alterações nas instruções SQL, devendo, portanto, a aplicação ser reescrita, recompilada e testada novamente.

Uma outra solução adotada permite diminuir este acoplamento. A idéia é retirar as instruções SQL das classes da aplicação e incluí-las em classes especializadas, denominadas Classes de Acesso a Dados (*Data Access Classes)*. Deste modo, as alterações no modelo de dados vão requer modificações apenas nas classes de acesso a dados, o que restringe o impacto das mudanças no sistema como um todo. O código fica mais limpo e mais fácil de manter, mais ainda não é a solução ideal visto que os dois mundos (objetos e dados relacionais) ainda estão presentes e intimamente ligados.

Uma camada de abstração de acesso a dados deve garantir aos desenvolvedores de software independência entre o modelo de objetos e o esquema do banco de dados, permitindo, ainda, o armazenamento dos dados em mecanismos de persistência diferentes dos relacionais, tais como SGBD orientados a objetos, objeto-relacionais ou arquivos XML.

A fim de contornar estes problemas, foram propostas diversas idéias para permitir um processo de mapeamento entre sistemas baseados em objetos e bases de dados relacionais. Tais propostas convergiram para o conceito de Camadas de Persistência.

7.1.1. Camadas de Persistência

Por definição, uma camada de persistência de objetos é um conjunto de programas (uma biblioteca) que permite, de forma transparente, a realização do processo de persistência, ou seja, o armazenamento e manutenção do estado de objetos em algum meio não volátil. A independência entre a camada de persistência e o repositório utilizado permite o gerenciamento da persistência de um modelo de objetos em diversos tipos de repositórios. A camada de persistência dá ao desenvolvedor a capacidade de trabalhar como se estivesse em um sistema puramente orientado a objetos, utilizando-se de métodos para incluir, alterar ou remover objetos a partir de uma linguagem de consulta para SGBD orientados a objetos, tal como a linguagem OQL.

A utilização de uma camada de persistência encapsula, ou seja, isola os acessos realizados diretamente ao banco de dados pela aplicação. Deste modo, as construções de consultas e operações de manipulação de dados (insert, update e delete) residem em uma camada de objetos inacessível ao programador. O encapsulamento garante maior confiabilidade às aplicações e permite que o próprio SGBD ou a estrutura de suas tabelas possam ser modificados sem grandes impactos à aplicação.

Além do encapsulamento e do suporte aos diversos tipos de mecanismos de persistência (SGDB relacional, arquivos XML, etc.) citados acima, uma camada de persistência deve implementar as seguintes características:

- Possibilidade de manipulação de conjuntos de objetos, listas de objetos que são obtidas ou retornadas da base de dados.
- Implementar controle de transações.
- Permitir a adição de novas classes ao esquema e uma fácil modificação do mecanismo de persistência.
- Implementar algoritmos de geração de chaves de identificação de modo a garantir que aplicação trabalhe com objetos com identidade única.
- Implementar sincronização entre o banco de dados e a aplicação.
- Implementar técnicas (cursores, proxies, etc.) que garantam que os atributos somente serão carregados quando necessários e que mantenham controle quanto à

posição dos objetos no banco de dados. Estas características são importantes para aplicações que utilizam objetos grandes.

- Dar suporte a mecanismos de geração de relatórios.
- Implementar pool de conexões que garanta a utilização simultânea do sistema por vários usuários sem queda de performance.
- Implementar um controle de acessos concorrentes.

Hibernate é um mecanismo simples e poderoso que permite a persistência de objetos em banco de dados relacionais de maneira transparente para qualquer tipo de aplicação Java. Esta ferramenta, que pode ser baixada gratuitamente da Internet através do endereço http://hibernate.sf.net.

O Hibernate suporta alguns dos principais bancos de dados relacionais disponíveis no mercado, permitindo a utilização de meios nativos para geração de chaves primárias e *pessimistic locking*. O hibernate trabalha com os bancos de dados através de dialetos, conforme a tabela :

Banco de dados	**Dialeto**
DB2	cirus.hibernate.sql.DB2Dialect
MySQL	cirus.hibernate.sql.MySqlDialect
SAPDB	cirus.hibernate.sql.SAPDBDialect
Oracle	cirus.hibernate.sql.OracleDialect
Sybase	cirus.hibernate.sql.SybaseDialect
Progress	cirus.hibernate.sql.ProgressDialect
McKoiSQL	cirus.hibernate.sql.McKoiDialect
Interbase/Firebird	cirus.hibernate.sql.InterbaseDialect
Pointbase	cirus.hibernate.sql.PointbaseDialect
PostgreSQL	cirus.hibernate.sql.PostgreSQLDialect
HypersonicSQL	cirus.hibernate.sql.HSQLDialect
Microsoft SQL Server	cirus.hibernate.sql.SybaseDialect

Fonte: www.mundojava.com.br

O desenvolvimento usando Hibernate é um processo que pode ser dividido em cinco etapas. O primeiro passo é a construção do banco de dados com o qual a aplicação irá trabalhar, ou seja, criar as tabelas onde os objetos serão persistidos. Este banco de dados, com suas entidades, atributos e relacionamentos, poderá ser criado de forma tradicional ou, a critério do usuário, poderá ser utilizada a ferramenta *SchemaExport* que acompanha o Hibernate, Esta ferramenta gera o esquema do banco de dados baseado no relacionamento entre os objetos que se quer persistir. O segundo passo é criação dos objetos cujos estados vão ser persistidos, isto é, a construção de classes (beans) para cada entidade do banco de dados. Estas classes devem ser construídas seguindo o modelo JavaBeans, com métodos *get e set* para manipulação dos atributos. Neste ponto, o Hibernate difere-se de outros mecanismos de persistências como o JDO, visto que os beans utilizados pelo Hibernate não necessitam estender superclasses ou implementar interfaces. É necessária a implementação de um construtor *default* (construtor sem parâmetros) para todas as classes persistentes. O Hibernate faz a instanciação de objetos e o acesso às suas propriedades através de reflexão, assim métodos de acesso e construtores não necessitam ser declarados como públicos. Adicionalmente, deve ser criada uma propriedade "chave-primária", que fará às vezes de uma *primary key*, através da qual um objeto será unicamente identificado. Esta propriedade, não obrigatória, pode ser do tipo primitivo *int, long, char* ou mesmo uma *String.*

Após a criação das tabelas e dos beans é necessário criar meta-dados de modo a fornecer ao Hibernate informações sobre como relacionar os objetos com as entidades do banco de dados. Esta é a terceira etapa, a criação de arquivos XML que relacionam as propriedades de cada objeto aos campos das tabelas. Nestes arquivos de mapeamento deve-se informar, ainda, qual propriedade do objeto se relaciona com a chave-primária, bem com os relacionamentos entre entidades (inclusive os tipos de relacionamento: 1-1, 1-N ou N-N).

A quarta etapa refere-se à criação de um arquivo contendo as propriedades necessárias para que o Hibernate se conecte ao banco de dados. Existe um arquivo de configuração modelo (*hibernate.properties*) que poderá ser utilizado como base para que o usuário proceda à configuração. A quinta e última etapa é a criação de *Data Access Objects* (DAO), Tais mecanismos são *design pattern* úteis para separação da lógica de acesso a dados da lógica de negócios da aplicação. Estas classes conterão os métodos de inclusão, alteração, exclusão dos objetos, etc.

Em resumo, o Hibernate é uma ferramenta que permite trabalhar com persistência sobre banco de dados, sem necessidade da inclusão de instruções SQL em meio ao código Java, assim como elimina a necessidade de se mapear ResultSets e implementar configuração de pool de conexões, etc., o que torna o código mais legível e, conseqüentemente, mais fácil de manter. Contudo a implementação não é independente da fonte de dados, visto que o Hibernate trabalha apenas com alguns dos bancos de dados relacionais. Por outro lado, caso a aplicação exija consultas SQL complexas, há de se considerar a utilização da linguagem HQL, a Query Language do Hibernate.

A figura 14 ilustra a estrutura de funcionamento do Hibernate:

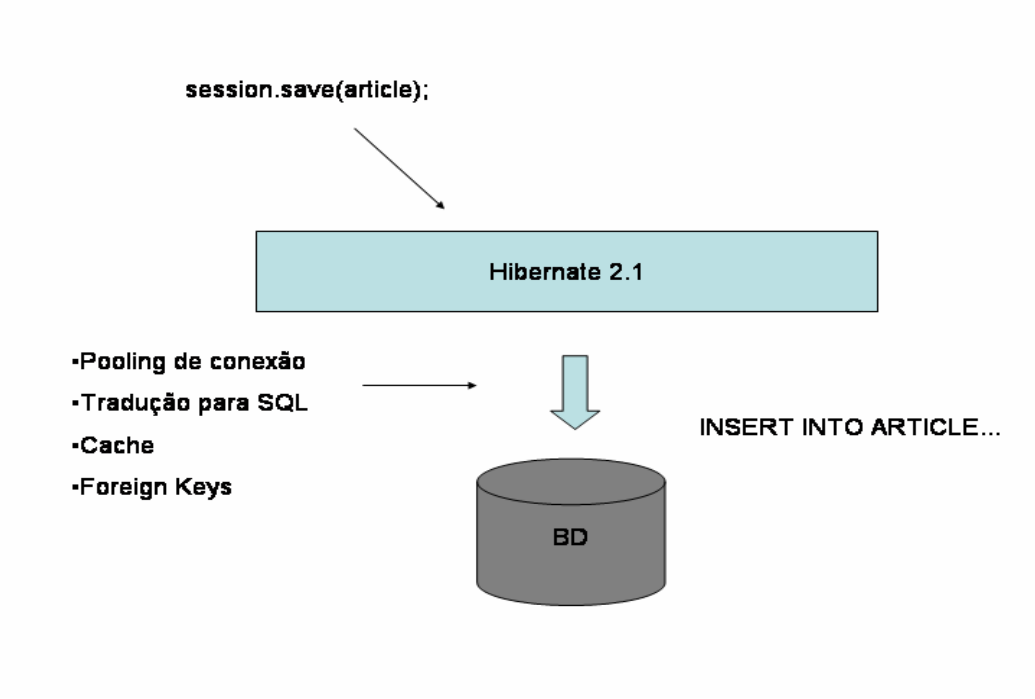


Figura 14. Funcionamento do Hibernate

7.2. Principais características

Entre as principais características do Hibernate podemos citar:

- Sua implementação é relativamente simples;
- Alta performance;
- Definição de mapeamento Objeto/Relacional (XML e Xdoclet);
- Permite a persistência dos objetos Java, incluindo polimorfismo, herança, composição, associação, etc, com grande facilidade;
- Possui uma linguagem de conexão (Hibernate Query Language) entre os mundos dos Objetos e Banco de dados relacionais simples;
- Suporta apenas banco de dados relacionais;
- É um software livre.

8. O BANCO DE DADOS MYSQL

8.1. Visão Geral do MySql

Segundo (MYSQL, 2005), o MySQL é o mais popular sistema de gerenciamento de banco de dados SQL Open Source. É desenvolvido, distribuído e tem suporte da MySQL AB. A MySQL AB é uma empresa comercial, fundada pelos desenvolvedores do MySQL, cujo negócio é fornecer serviços relacionados ao sistema de gerenciamento de banco de dados MySQL.

8.1.1. O MySQL é um sistema de gerenciamento de bancos de dados.

Um banco de dados é uma coleção de dados estruturados. Ele pode ser qualquer coisa desde uma simples lista de compras a uma galeria de imagens ou a grande quantidade de informação da sua rede coorporativa. Para adicionar, acessar, e processar dados armazenados em um banco de dados de um computador, você necessita de um sistema de gerenciamento de bancos de dados como o Servidor MySQL. Como os computadores são muito bons em lidar com grandes quantidades de dados, o gerenciamento de bancos de dados funciona como a engrenagem central na computação, seja como utilitários independentes ou como partes de outras aplicações.

8.1.2. O MySQL é um sistema de gerenciamento de bancos de dados relacional.

Um banco de dados relacional armazena dados em tabelas separadas em vez de colocar todos os dados um só local. Isso proporciona velocidade e flexibilidade. A parte SQL do nome MySQL significa “Structured Query Language - Linguagem Estrutural de Consultas". SQL é linguagem padrão mais comum usada para acessar banco de dados e é definida pelo Padrão ANSI/ISO SQL. (O padrão SQL vem evoluindo desde 1986 e existem

diversas versões. Neste manual, "SQL-92" se refere ao padrão liberado em 1992, "SQL-99" se refere ao padrão liberado em 1999, e "SQL:2003" se refere a versão do padrão liberado em 2003. O termo "o padrão SQL" é usado para indicar a versão atual do Padrão SQL em qualquer momento).

8.1.3. O é MySQL um software Open Source.

Open Source significa que é possível para qualquer um usar e modificar o programa. Qualquer pessoa pode fazer download do MySQL pela Internet e usá-lo sem pagar nada. Caso haja interesse, também é possível estudar o código fonte e alterá-lo para adequá-lo às suas necessidades. O MySQL usa a GPL (GNU General Public License - Licença Pública Geral GNU) http://www.fsf.org/licenses, para definir o que se pode e não se pode fazer com o software em diferentes situações. Também há uma versão comercial, licenciada, para casos de desconforto com a GPL ou de precisar embutir o MySQL em uma aplicação comercial.

8.2. Principais Características do MySQL

Segundo (MYSQL, 2005), a seguinte lista descreve algumas das características mais importantes do Programa de Banco de Dados MySQL.

8.2.1. Quanto a portabilidade:

- Escrito em C e C++.
- Testado com uma ampla faixa de compiladores diferentes.
- Funciona em diversas plataformas.
- Utiliza o GNU Automake, Autoconf, e Libtool para portabilidade.
- Possui APIs para C, C++, Eiffel, Java, Perl, PHP, Python, Ruby e Tcl..
- Suporte total a multi-threads usando threads diretamente no kernel. Isto significa que se pode facilmente usar múltiplas CPUs, se disponível.

- Fornece mecanismos de armazenamento transacional e não transacional.
- Tabelas em disco (MyISAM) baseadas em árvores-B extremamente rápidas com compressão de índices.
- É relativamente fácil se adicionar outro mecanismo de armazenamento. Isto é útil se você quiser adicionar uma interface SQL a um banco de dados caseiro.
- Um sistema de alocação de memória muito rápido e baseado em processo (thread).
- Joins muito rápidas usando uma multi-join de leitura única otimizada.
- Tabelas hash em memória que são usadas como tabelas temporárias.
- Funções SQL são implementadas por meio de uma biblioteca de classes altamente otimizada e com o máximo de performance. Geralmente não há nenhuma alocação de memória depois da inicialização da pesquisa.
- O código do MySQL foi testado com Purify (um detector comercial de falhas de memória) e também com o Valgrind, uma ferramenta GPL.
- Disponível como versão cliente/servidor ou embutida (ligada).

8.2.2. Quanto aos tipos de coluna:

- Aceita diversos tipos de campos: tipos inteiros de 1, 2, 3, 4 e 8 bytes com e sem sinal, FLOAT, DOUBLE, CHAR, VARCHAR, TEXT, BLOB, DATE, TIME, DATETIME, TIMESTAMP, YEAR, SET e ENUM.
- Registros de tamanhos fixos ou variáveis.

8.2.3. Quanto aos comandos e funções:

- Completo suporte a operadores e funções nas partes SELECT e WHERE das consultas. Por exemplo:
- mysql> SELECT CONCAT(first_name, " ", last_name)
- -> FROM nome_tbl
- -> WHERE income/dependents > 10000 AND age > 30;
- Suporte pleno às cláusulas SQL GROUP BY e ORDER BY. Suporte para funções de agrupamento (COUNT(), COUNT(DISTINCT ...), AVG(), STD(), SUM(), MAX() e MIN()).
- Suporte para LEFT OUTER JOIN e RIGHT OUTER JOIN com as sintaxes SQL e ODBC.
- Alias em tabelas e colunas são disponíveis como definidos no padrão SQL92.
- DELETE, INSERT, REPLACE, e UPDATE retornam o número de linhas que foram alteradas (afetadas). É possível retornar o número de linhas com padrão coincidentes configurando um parâmetro quando estiver conectando ao servidor.
- O comando específico do MySQL SHOW pode ser usado para devolver informações sobre bancos de dados, tabelas e índices. O comando EXPLAIN pode ser usado para determinar como o otimizador resolve a consulta.
- Nomes de funções não conflitam com nomes de tabelas ou colunas. Por exemplo, ABS é um nome de campo válido. A única restrição é que para uma chamada de função, espaços não são permitidos entre o nome da função e o '(' que o segue.
- Pode-se misturar tabelas de bancos de dados diferentes na mesma pesquisa (como na versão 3.22).

8.2.4. Quanto à segurança

- O MySQL possui um sistema de privilégios e senhas que é muito flexível, seguro e que permite verificação baseada em estações/máquinas. Senhas são seguras porque todo o tráfico de senhas é criptografado quando você se conecta ao servidor.

8.2.5. Quanto à escalabilidade e limites

- O MySQL lida com bancos de dados enormes. Usamos o Servidor MySQL com bancos de dados que contém 50.000.000 registros e sabemos de usuários que usam o Servidor MySQL com 60.000 tabelas e aproximadamente 5.000.000.000 de linhas.
- São permitidos até 32 índices por tabela. Cada índice pode ser composto de 1 a 16 colunas ou partes de colunas. O tamanho máximo do índice é de 500 bytes (isto pode ser alterado na compilação do MySQL). Um índice pode usar o prefixo de campo com um tipo CHAR ou VARCHAR.

8.2.6. Quanto à conectividade

- Os clientes podem se conectar ao servidor MySQL usando sockets TCP/IP, em qualquer plataforma. No sistema Windows na família NT (NT, 2000 ou XP), os clientes podem se conectar usando named pipes. No sistema Unix, os clientes podem se conectar usando arquivos sockets.
- A interface Connector/ODBC fornece ao MySQL suporte a programas clientes que usam conexão ODBC (Open-DataBase-Connectivity). Por exemplo, pode-se usar o MS Access para conectar ao seu servidor MySQL. Os clientes podem ser executados no Windows ou Unix. O fonte do Connector/ODBC está disponível. Todas as funções ODBC são suportadas, assim como muitas outras.

8.2.7. Quanto à localização

- O servidor pode apresentar mensagem de erros aos clientes em várias línguas.
- Suporte total para vários conjuntos de caracteres, que incluem ISO-8859-1 (Latin1), big5, ujis e mais. Por exemplo, os caracteres Escandinavos 'â', 'ä', 'ö' são permitidos em nomes de tabelas e colunas.
- Todos os dados são armazenados no conjunto de caracteres escolhido. Todas as comparações em colunas de seqüências caso-insensitivo.
- A ordenação é feita de acordo com o conjunto de caracteres escolhido (o modo sueco por padrão). É possível alterar isso quando o servidor MySQL é iniciado. Para ver um exemplo de várias ordenações avançadas, procure pelo código de ordenação Tcheca. O Servidor MySQL suporta diversos conjuntos de caracteres que podem ser especificados em tempo de compilação e execução.

8.2.8. Quanto aos clientes e às ferramentas

- O servidor MySQL foi construído com suporte para instruções SQL que verificam, otimizam e reparam tabelas. Estas instruções estão disponíveis a partir da linha de comando por meio do cliente myisamcheck, O MySQL inclui também o myisamchk, um utilitário muito rápido para realizar estas operações em tabelas MyISAM.
- Todos os programas MySQL podem ser chamados com as opções --help ou -? para obter ajuda online.

9. CONCLUSÃO

O desenvolvimento do sistema para controle de atividades empresariais envolveu uma série de desafios e novas tecnologias, mas ao fim deste jornada foi possível atingir os objetivos iniciais, fornecendo uma ferramenta de grande utilidade para a Federação Catarinense dos Municípios de Santa Catarina.

Além dos objetivos iniciais do cliente, chegou-se a um considerável nível de satisfação pessoal e acadêmica pois o resultado final foi muito além de uma simples ferramenta, transformou-se em conhecimento tecnológico e experiência no relacionamento entre desenvolvedor e cliente alvo.

Deficiências, limitações e dificuldades são normais a qualquer projeto, neste caso específico denotamos o tempo escasso devido ao retardo na escolha do tema final, mas foi de grande valia pois o escopo inicial foi atingido, faltando apenas a difusão efetiva da ferramenta junto ao cliente.

Em função da futura implantação no cliente (FECAM) novas necessidades surgiram, pois uma ferramenta de tal porte exige aprimoramento contínuo até atingir o nível ideal de estabilidade e funcionalidades, mas as metas iniciais do projeto foram atingidas com sucesso.

10. BIBLIOGRAFIA

AHMED, Khawar Zaman. UMRYSH, Cary E. *Desenvolvendo Aplicações Comerciais em Java com J2EE™ e UML*. Rio de Janeiro: Editora Ciência Moderna Ltda., 2002.

ALUR, Deepak e CRUPI, John e MALKS, Dan. Core J2EE Patterns. As Melhores práticas e estratégias de design. Editora Campus, 2002.

YUNG, Leandro. *Revista Mundo Java, Número 04, Ano 01*. Editora Mundo OO, 2004.

SUN Microsystems. Java™ 2 Platform Enterprise Edition Specification, v1.4. 2003

BROEMMER, Darren. J2EE Best Practices - Java Design Patterns, Automation and Performance. Addison-Wesley, 2003.

BODOFF, Stephanie e GREEN, Dale e HAASE, Kim e JENDROCK, Eric e PAWLAN, Monica e STEARNS, Beth. Tutorial do J2EE. Editora Campus, 2002.

SINGH, Inderjeet e STEARNS, Beth e JOHNSON, Mark. Desinging Enterprise Applications with J2EE Platform. Segunda Edição. Addison-Wesley, 2002.

W3C World Wide Web Consortium. http://www.w3c.org. Acesso em 21/11/2004.

OTN, Oracle Tecnology Network. http://www.oracle.com/technology/tech/java/index.html. Acesso em 29/11/2004

PERRY, Bruce W. Java Servlet &JSP. Cookbook. Primeira Edição. Editora O'Reilly, 2004

FIELDS, Duane K. e KOLB, Mark. Desenvolvendo na Web com JavaServer Pages. Editora Ciência Moderna, 2000.

MYSQL, Manual de Referência do. http://dev.mysql.com/doc/mysql/pt/. Acesso em 12/05/2005.

ARMSTRONG, Eric e BODOFF, Stephanie e CARSON, Debbie e EVANS, Ian e FISHER, Maydene e GREEN, Dale e HAASE, Kim e JENDROCK, Eric e PAWLAN, Monica e STEARNS, Beth. The J2EE™ 1.4 Tutorial

11. ANEXO I

11.1. Artigo Sobre o Trabalho

SISTEMA PARA CONTROLE DE ATIVIDADES EMPRESARIAIS PARA A FEDERAÇÃO CATARINENSE DOS MUNICÍPIOS

Kleyton Weber da Silva
Sergio Ferreira de Mendonça
Bacharelado em Sistemas de informação, 2005
Departamento de Informática e Estatística,
Universidade Federal de Santa Catarina (UFSC),

Resumo

Este trabalho tem como objetivo descrever os processos que envolveram a construção de uma ferramenta Web para controle de atividades empresariais voltada para a Federação Catarinense dos Municípios. Serão apresentadas as tecnologias empregadas, assim como os requisitos de projeto que levaram ao emprego de tais tecnologias, assim como o levantamento das peculiaridades do cliente alvo (FECAM).

O intuito da aplicação é automatizar os processos internos e externos da empresa, possibilitando um gerenciamento da atividades e melhor prestação de contas da empresa com seus clientes.

Palavras-Chave: FECAM, WEB, frameworks, Struts, Hibernate, JSP, Java 2 Enterprise Edition (J2EE), MYSQL, plataforma Java 2 Standard Edition (J2SE).

Introdução:

O mercado apresenta uma grande necessidade de ferramentas para desenvolvimento de atividades operacionais, estando essas em constante estado de evolução e cada vez mais presentes nas organizações. Contudo, a singularidade de cada empresa exige um conjunto de características que dificilmente podem ser acopladas em uma única ferramenta.

Tendo como intuito inicial resolver os problemas gerenciais referentes a FECAM - Federação Catarinense dos Municípios, foi desenvolvida uma ferramenta de controle empresarial baseada em tarefas e arquivos, suprindo as principais limitações operacionais encontradas nesta entidade.

Contextualização

A necessidade latente do emprego da tecnologia da informação como ferramenta de trabalho nas empresas pode ser constatada em pequenos escritórios ou em grandes empresas. Sendo possível constatar sua eficiência na automatização de processos

que seguem algum padrão ou fluxo definido de atividades

Tendo como exemplo a Federação Catarinense de Municípios – FECAM, entidade representativa dos interesses municipais de Santa Catarina, que muito embora seja uma associação sem fins econômicos, precisa fazer uso da tecnologia da informação para melhorar seus serviços e reduzir os custos operacionais.

Objetivos do Trabalho

Objetivo Geral

O objetivo geral deste trabalho é desenvolver uma aplicação para controle de atividades empresariais voltado ao ambiente Web, utilizando a plataforma J2EE e os frameworks de código aberto Struts e Hibernate, suprindo as necessidades operacionais encontradas pela FECAM.

Objetivos Específicos

- *Estudar a arquitetura da plataforma J2EE;*
- *Estudar a arquitetura do framework Strutrs;*
- *Estudar a arquitetura do framework Hibernate;*
- *Estudar mecanismos de segurança aplicados à plataforma J2EE;*
- *Estudar o banco de dados Mysql.*
- *Desenvolver um Sistema de Controle Empresarial utilizando as tecnologias acima citadas.*

Institucional

A FECAM - Federação Catarinense de Municípios, fundada no dia 16 de julho de 1980, tem sede em Florianópolis, Praça XV de Novembro, 270 – Centro. É uma entidade municipalista privada, de natureza civil, sem fins lucrativos, que tem como objetivo atender os interesses comuns dos Municípios Catarinenses. Com a finalidade de integrá-los e representá-los, objetivando a valorização e fortalecimento do Municipalismo.

LEVANTAMENTO DE REQUISITOS

A FECAM atualmente emprega tecnologia da informação apenas em parte de seus trabalhos, que em alguns casos encontram-se defasadas e, conseqüentemente, de pouco auxílio e importância. Pensando nisso será foram levantados os principais pontos para reestruturação do controle de atividades e de documentos da FECAM.

Atividades passíveis de automatização e/ou controle

A FECAM presta seus trabalhos às Associações de Municípios e aos Municípios de Santa Catarina de acordo com os seis projetos de atuação da entidade, a saber:

- *Representação e Defesa Institucional da FECAM, Associações e Municípios Catarinenses;*
- *Consultoria Especializada as Associações de Municípios;*
- *Planos e Projetos de Desenvolvimento Regional;*
- *Gestão da Informação e Desenvolvimento Tecnológico;*
- *Programa de Capacitação de Agentes Políticos e de Servidores Públicos Municipais; e*
- *Gestão de Qualidade da Administração Pública.*

A partir dessas atividades é possível projetar um sistema para controle e gerenciamento visando melhorara as atividades de relacionamento FECAM com seus parceiros, relacionando os instrumentos

e estabelecendo padrões para tais atividades, desta forma chegando a uma ferramenta denominada como CATE- Controlador de Atividades Empresariais personalizado à FECAM.

Módulos do sistema

De acordo com as peculiaridades das atividades foram diagnosticados três módulos para desenvolvimento:

Módulo 1 – Controle de Documentos

O primeiro módulo é referente ao controle dos documentos elaborados pelos colaboradores da empresa para envio aos clientes. Este módulo tornou-se necessário em função da quantidade de documentos gerados dentro da Fecam e da falta de uma padronização para armazenamento, classificação e localização dos mesmos.

Desta forma, cada documento deverá ser armazenado em um repositório gerenciado pelo sistema e classificado com atributos que permitam facilitar a localização, o acompanhamento do envio para os clientes e a geração de relatórios.

Módulo 2 – Controle de Atividades Externas

Os funcionários da FECAM são constantemente solicitados para elaboração e acompanhamento de diversas atividades externas ao ambiente da empresa, como cursos, palestras ou reuniões.

As atividades devem ser cadastradas com as seguintes características, sendo que após este cadastro as mesmas permanecerão com status de Agendada:

Após a realização de uma atividade, é gerado um documento contendo um resumo de tudo o que ocorreu na mesma. Este documento também deverá ser cadastrado no sistema, juntamente com uma breve explanação a seu respeito. Assim que este cadastro for realizado, a atividade em questão receberá o status de Concluída.

Para gerenciamento das atividades faz-se necessário uma ferramenta de localização, facilitando a busca dos registros pelos seus atributos. A partir desta ferramenta de localização deve ser permitida a geração de relatórios sobre as atividades.

Módulo 3 – Cadastros Básicos

Para uma utilização mais flexível do sistema, faz-se necessário este módulo contendo os cadastros, alterações e exclusões dos dados básicos utilizados no sistema.

Arquitetura do sistema.

O sistema utiliza uma arquitetura composta de quatro camadas:

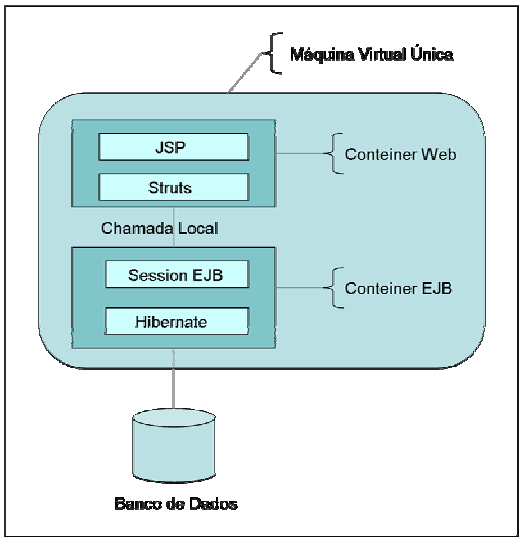
Camada de apresentação

Trata-se da camada de interação com o usuário. Nesta camada foram utilizadas páginas JSP (descritos na seção 4.2.1.2) e scripts na linguagem JavaScript.

Arquitetura do sistema.

O sistema utilizará uma arquitetura composta de quatro camadas:

Camada de apresentação
Camada de controle
Camada de negócio
Camada de persistência



Desenvolvimento do Sistema

O objetivo do sistema é gerenciar as atividades relacionadas à FECAM, com isso identificamos dois elementos chave no projeto (Documento / Atividade). Para organizar e acompanhar o andamento de fatos e atividades de interesse da empresa e associados é necessário a geração de arquivos descritivos com detalhes e características que facilitem a busca das informações.

Podemos dividir as operações em duas seqüências distintas: Geração de atividades e criação de documentos.

O fluxo do cadastro de uma atividade é mostrado na figura a seguir:

O conselho de representação é composto por duas pessoas (Titular e Suplente), servindo como porta-voz dos conselhos de municípios nas decisões e tomadas de atitudes.

As atividades são pautadas pelos conselhos de representação e gerenciadas por um responsável escolhido em senso comum. Este responsável deve manter a o alinhamento dos objetivos da atividade e fazer cumprir as datas e metas.

O personagem Apoiador fica com a parte operacional do processo de gerenciamento das atividades, organizando arquivos e criando os documentos necessários para o registro das atividades. Suas atividades são acompanhadas diretamente pelo responsável.

O fluxo do cadastro de documento é apresentado na figura a seguir:

O Autor elabora o documento e salva no sistema, sendo necessária a assinatura do responsável para posterior envio aos destinatários.

Tecnologias Empregadas

Linguagem de programação Java

Teve sua origem em 1994, na Sun MicroSystems por uma equipe liderada por James Gosling. Inicialmente era destinada á programação de aparelhos como celulares, relógios palms, mas devido à explosão da Internet a partir de 1995, identificou-se no Java uma linguagem ideal para aplicações na Internet. Hoje ela se transformou em uma plataforma.

A plataforma J2EE (Java 2 Enterprise Edition) é uma abordagem que oferece simplificação para desenvolvimento de aplicações mais complexas. Tal arquitetura define um conjunto de padrões para desenvolvimento de aplicações.

A Struts Framework é um projeto open source mantido pela Apache Software Foundation. É uma implementação do design pattern MVC (Model-View-Controller) para aplicações java com internet. O objetivo do pattern MVC é separar de maneira clara a camada de apresentação (View) da camada de Negócio (Model).

Hibernate é um mecanismo simples e poderoso que permite a persistência de objetos em banco de dados relacionais de maneira transparente para qualquer tipo de aplicação Java. Hibernate permite trabalhar com persistência sobre banco de dados, sem necessidade da inclusão de instruções SQL em meio ao código Java, assim como elimina a necessidade de se mapear ResultSets e implementar configuração de pool de conexões, etc., o que torna o código mais legível e, conseqüentemente, mais fácil de manter.

Segundo (MYSQL, 2005), o MySQL é o mais popular sistema de gerenciamento de banco de dados SQL Open Source. É desenvolvido, distribuído e tem suporte da MySQL AB. A MySQL AB é uma empresa comercial, fundada pelos desenvolvedores do MySQL, cujo negócio é fornecer serviços relacionados ao sistema de gerenciamento de banco de dados MySQL.

CONCLUSÃO

O desenvolvimento do sistema para controle de atividades empresariais propiciou o estudo de novas tecnologias e a formação de um conhecimento único ao elaborar um trabalho prático com participação direta de um cliente real, fato até o presente momento inexistente em nossa jornada acadêmica.

Deficiências, limitações e dificuldades são normais a qualquer projeto, neste caso específico denotamos o tempo escasso devido ao retardo na escolha do tema final.

Como o produto ainda não foi implantado no cliente novas funcionalidades e implementações surgirão pois uma ferramenta de tal porte exige aprimoramento contínuo até atingir o nível ideal de estabilidade e funcionalidades, mas as metas iniciais do projeto foram atingidas com sucesso.

Referências Bibliográficas

AHMED, Khawar Zaman. UMRYSH, Cary E. *Desenvolvendo Aplicações Comerciais em Java com J2EE™ e UML*. Rio de Janeiro: Editora Ciência Moderna Ltda., 2002.

ALUR, Deepak e CRUPI, John e MALKS, Dan. Core J2EE Patterns. As Melhores práticas e estratégias de design. Editora Campus, 2002.

YUNG, Leandro. *Revista Mundo Java, Número 04, Ano 01*. Editora Mundo OO, 2004.

SUN Microsystems. Java™ 2 Platform Enterprise Edition Specification, v1.4. 2003

BROEMMER, Darren. J2EE Best Practices - Java Design Patterns, Automation and Performance. Addison-Wesley, 2003.

BODOFF, Stephanie e GREEN, Dale e HAASE, Kim e JENDROCK, Eric e PAWLAN, Monica e STEARNS, Beth. Tutorial do J2EE. Editora Campus, 2002.

SINGH, Inderjeet e STEARNS, Beth e JOHNSON, Mark. Desinging Enterprise Applications with J2EE Platform. Segunda Edição. Addison-Wesley, 2002.

W3C World Wide Web Consortium. http://www.w3c.org. Acesso em 21/11/2004.

OTN, Oracle Tecnology Network. http://www.oracle.com/technology/tech/java/index.html. Acesso em 29/11/2004

PERRY, Bruce W. Java Servlet &JSP. Cookbook. Primeira Edição. Editora O'Reilly, 2004

FIELDS, Duane K. e KOLB, Mark. Desenvolvendo na Web com JavaServer Pages. Editora Ciência Moderna, 2000.

MYSQL, Manual de Referência do. http://dev.mysql.com/doc/mysql/pt/. Acesso em 12/05/2005.

ARMSTRONG, Eric e BODOFF, Stephanie e CARSON, Debbie e EVANS, Ian e FISHER, Maydene e GREEN, Dale e HAASE, Kim e JENDROCK, Eric e PAWLAN, Monica e STEARNS, Beth. The J2EE™ 1.4 Tutorial

12. ANEXO II

12.1. Código-Fonte

12.1.1. DocumentoConstants.java

```
package br.ufsc.cate.business.documento;

/**
 * @author Sergio
 *
 * TODO To change the template for this generated type comment go to
 * Window - Preferences - Java - Code Style - Code Templates
 */
public class DocumentoConstants {
    public static final String ID = "id";
    public static final String IDENTIFICADOR = "identificador";
    public static final String CODIGO_MODALIDADE = "modalidade";
    public static final String CODIGO_FORMA_ENVIO = "formaEnvio";
    public static final String CODIGO_ASSINANTE = "assinante";
    public static final String CODIGO_AUTOR = "autor";
    public static final String CODIGO_DESTINATARIO = "destinatario";
    public static final String DATA_ENVIO = "dataEnvio";
    public static final String DATA_ELABORACAO = "dataElaboracao";
    public static final String ASSUNTO = "assunto";
    public static final String ASSINANTE = "assinante";
    public static final String ARQUIVO = "arquivo";
    public static final String CAMINHO_RELATIVO = "caminhoRelativo";
}
```

12.1.2. DocumentoBusinessBean.java

```
package br.ufsc.cate.business.documento;

import java.io.File;
import java.rmi.RemoteException;
import java.util.HashMap;
import java.util.HashSet;
import java.util.List;
import java.util.Set;

import javax.ejb.EJBException;
import javax.ejb.SessionBean;
import javax.ejb.SessionContext;

import net.sf.hibernate.HibernateException;
import net.sf.hibernate.Session;
import net.sf.hibernate.Transaction;

import org.apache.log4j.Logger;

import br.ufsc.cate.hibernate.Destinatario;
import br.ufsc.cate.hibernate.Documento;
import br.ufsc.cate.hibernate.Modalidade;
```

```
import br.ufsc.cate.util.HibernateUtil;

/**
 * @author Kleyton Weber da Silva (kleyton@inf.ufsc.br) Sergio Ferreira de
 *         Mendonça (sfermend@inf.ufsc.br)
 * @version $Revision: 1.3 $<br>
 *          $Id: DocumentosBusinessBean.java,v 1.8 2005/05/03 00:55:57 sfermend
 *          Exp $
 *
 * @ejb:bean name="cate/DocumentosBusiness" display-name="DocumentosBusiness"
 *           type="Stateless" view-type="both"
 *           jndi-name="cate/DocumentosBusiness"
 *           jndi-local-name="cate/DocumentosBusinessLocal"
 *
 *
 * @ejb.transaction type="Supports"
 */
public class DocumentosBusinessBean implements SessionBean {
    private SessionContext context;

    public void setSessionContext(SessionContext context) throws EJBException,
                RemoteException {
            this.context = context;
    }

    public void ejbRemove() {
    }

    public void ejbActivate() {
    }

    public void ejbPassivate() {
    }

    public void ejbCreate() {
    }

    public void ejbPostCreate() {
    }

    /**
     * @ejb.interface-method view-type="both"
     */
    public void cadastrarDocumento(HashMap params) throws Exception {
            System.out.println("modalidade: " + params.get("modalidade"));
            System.out.println("autor: " + params.get("autor"));
            System.out.println("assinante: " + params.get("assinante"));
            System.out.println("destinatario: " + params.get("destinatario"));
            System.out.println("dataElaboracao: " + params.get("dataElaboracao"));
            System.out.println("dataEnvio: " + params.get("dataEnvio"));
            System.out.println("assunto: " + params.get("assunto"));
            System.out.println("formaEnvio: " + params.get("formaEnvio"));

            DocumentoUtil du = DocumentoUtil.getInstance();

            Session session = HibernateUtil.currentSession();
            Transaction t = session.beginTransaction();
            Modalidade mod = (Modalidade) session.get(Modalidade.class, Long
                            .valueOf((String)
params.get(DocumentoConstants.CODIGO_MODALIDADE)));

            File            arq            =            du.moverArquivoRepositorio((File)
params.get(DocumentoConstants.ARQUIVO), mod
                            .getSigla());
```

```
            Documento documento = new Documento();

            //setando destinatarios
            try {
                  documento.setDestinatarios(this.parseDestinatarios(params, session));
            } catch (HibernateException e) {
                  Logger.getLogger(this.getClass()).info("Exceção       setando       os
destinatarios:");
                  Logger.getLogger(this.getClass()).info(e);
            }

/*          {
                  //Buscando funcionario
                  Assinante assinante = new Assinante();
                  Long codigoAssinante = new Long((String) params
                              .get(DocumentoConstants.CODIGO_ASSINANTE));
                  assinante = assinanteDAO.findAssinante(codigoAssinante);
                  //setando forma de envio
                  documento.setAssinante(assinante);
            }

            {
                  //Buscando funcionario
                  Autor autor = new Autor();
                  Long codigoAutor = new Long((String) params
                              .get(DocumentoConstants.CODIGO_AUTOR));
                  autor = autorDAO.findAutor(codigoAutor);
                  //setando forma de envio
                  documento.setAutor(autor);
            }

            {
                  //Buscando forma de envio
                  FormaEnvio formaEnvio = new FormaEnvio();
                  Long codigoFormaEnvio = new Long((String) params
                              .get(DocumentoConstants.CODIGO_FORMA_ENVIO));
                  formaEnvio = formaEnvioDAO.findFormaEnvio(codigoFormaEnvio);
                  //setando forma de envio
                  documento.setFormaEnvio(formaEnvio);
            }

            {
                  //Buscando funcionario
                  Modalidade modalidade = new Modalidade();
                  String codigoModalidade = (String) params
                              .get(DocumentoConstants.CODIGO_MODALIDADE);
                  modalidade = modalidadeDAO
                              .findModalidade(new Long(codigoModalidade));
                  //setando forma de envio
                  documento.setModalidade(modalidade);
            }

            //#### SETANDO ATRIBUTOS SIMPLES DO DOCUMENTO ####\\

            documento.setDataElaboracao((Long) params
                        .get(DocumentoConstants.DATA_ELABORACAO));
            documento.setAssunto((String) params.get(DocumentoConstants.ASSUNTO));
            documento.setDataEnvio((Long) params.get(DocumentoConstants.DATA_ENVIO));
            documento.setNomeArquivo((String) params.get(DocumentoConstants.ARQUIVO));
            documento.setIdentificador((String) params
                        .get(DocumentoConstants.IDENTIFICADOR));

            //inserindo valor na base de dados
            System.out.println("inserindo documento");
            this.insert(documento);
```

```
            System.out.println("inseriu documento"); */

            t.commit();
            HibernateUtil.closeSession();
    }

    private   Set   parseDestinatarios(HashMap   params,   Session   session)   throws
HibernateException {
            Set dest = new HashSet();
            String[] destinatarios = (String[]) params
                            .get(DocumentoConstants.CODIGO_DESTINATARIO);
            Destinatario destinatario = null;
            for (int i = 0; i < destinatarios.length; i++) {
                    destinatario = (Destinatario)session.get(Destinatario.class, Long
                                    .valueOf(destinatarios[i]));
                    dest.add(destinatario);

            }
            return dest;
    }

    /**
     * @ejb.interface-method view-type="both"
     */
    public List filtrarDocumentos(HashMap params) throws Exception {
            System.out.println("modalidade: " + params.get("modalidades"));
            String[] modalidades = (String[]) params.get("modalidades");
            String query = "FROM br.ufsc.cate.hibernate.Documento d WHERE d.modalidade.id
= "
                            + modalidades[0];
            for (int i = 1; i < modalidades.length; i++) {
                    query = query.concat(" OR d.modalidade.id = " + modalidades[i]);
            }
            query = query.concat(" ORDER BY d.dataEnvio DESC");
    /*      DocumentoDAO docDao = new DocumentoDAO();
            List l = docDao.getList(query);
            Iterator it = l.iterator();
            ArrayList array = new ArrayList(l.size());
            while (it.hasNext()) {
                    array.add(((Documento) it.next()).getMapDocumento());
            }*/
            return null;//array;
    }


    /**
     * @ejb.interface-method view-type="both"
     */
    public List getModalidades() throws Exception {
            return HibernateUtil.list("FROM br.ufsc.cate.hibernate.Modalidade m");
    }

    /**
     * @ejb.interface-method view-type="both"
     */
    public List getDestinatarios() throws Exception {
            return HibernateUtil.list("FROM br.ufsc.cate.hibernate.Destinatario d");
    }

    /**
     * @ejb.interface-method view-type="both"
     */
    public List getAutores() throws Exception {
            return HibernateUtil.list("FROM br.ufsc.cate.hibernate.Autor a");
    }
```

```
    /**
     * @ejb.interface-method view-type="both"
     */
    public List getAssinantes() throws Exception {
           return HibernateUtil.list("FROM br.ufsc.cate.hibernate.Assinante a");
    }

    /**
     * @ejb.interface-method view-type="both"
     */
    public List getFormasEnvio() throws Exception {
           return HibernateUtil.list("FROM br.ufsc.cate.hibernate.FormaEnvio f");
    }
}
```

12.1.3. DocumentoUtil.java

```
package br.ufsc.cate.business.documento;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.nio.channels.FileChannel;
import java.text.DecimalFormat;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Properties;

import br.ufsc.cate.hibernate.Modalidade;

/**
 * @author Sergio
 *
 * Classe que implementa alguma utilidades referentes a documentos
 */
public class DocumentoUtil {

    private static Properties docProp = new Properties();

    private final static String DOCUMENTO_PROPERTIES = "documento.properties";
    private final static String DOCUMENTOS_BASEDIR_PROPERTY = "documentos.basedir";

    private static DocumentoUtil docUtil;

    private static void loadProperties(){
           try {
                  docProp.load(DocumentoUtil.class
                                .getResourceAsStream(DOCUMENTO_PROPERTIES));
           } catch (IOException e) {
                  System.err.println(e);
                  e.printStackTrace(System.err);
           }
    }

    private DocumentoUtil(){
           loadProperties();
    }

    public static synchronized DocumentoUtil getInstance(){
```

```
            if(docUtil == null){
                    docUtil = new DocumentoUtil();
            }
            return docUtil;
    }

    public File moverArquivoRepositorio(File arq, String siglaModalidade) throws
IOException{
            String basedir = docProp.getProperty(DOCUMENTOS_BASEDIR_PROPERTY);

            File destDir = new File(basedir + "/" + siglaModalidade);
            if(!destDir.exists()){
                    destDir.mkdirs();
            }
            File arqDest = new File(destDir, arq.getName());
            arqDest.createNewFile();

            FileInputStream in = new FileInputStream(arq);
            FileOutputStream out = new FileOutputStream(arqDest);
            FileChannel cin = in.getChannel();
            cin.transferTo(0, cin.size(), out.getChannel());

            in.close();
            out.close();

            return arqDest;
    }

    public String getCaminhoCompletoArquivo(String caminhoRelativo){
            return    docProp.getProperty(DOCUMENTOS_BASEDIR_PROPERTY)    +    "/"    +
caminhoRelativo;
    }

    public Long getTimestamp(String data) throws ParseException{
            SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");
            return new Long(sdf.parse(data).getTime());
    }

    /**
     * @param mod
     * @return
     */
    public String getIdUnicaDocumento(Modalidade mod) {
            DecimalFormat df = new DecimalFormat("000000");
            mod.setSequencial(mod.getSequencial() + 1);
            return mod.getSigla()+ "-" + df.format(mod.getSequencial());
    }
}
```

12.1.4. BusinessDelegate.java

```
package br.ufsc.cate.delegate;

import java.util.ArrayList;
import java.util.Collection;
import java.util.HashMap;
import java.util.List;

import br.ufsc.cate.business.documento.DocumentosBusiness;
import br.ufsc.cate.business.documento.DocumentosBusinessHome;
import br.ufsc.cate.business.documento.DocumentosBusinessLocal;
```

```
import br.ufsc.cate.business.documento.DocumentosBusinessLocalHome;
import br.ufsc.cate.util.ServiceLocator;


/**
 * @author  Kleyton Weber da Silva (kleyton@inf.ufsc.br)
 *          Sergio Ferreira de Mendonça (sfermend@inf.ufsc.br)
 * @version $Revision: 1.3 $<br>
 *          $Id: BusinessDelegate.java,v 1.3 2005/06/01 07:13:54 sfermend Exp $
 *
 */
public class BusinessDelegate {
    private static BusinessDelegate buss;
    private static ServiceLocator sl = ServiceLocator.getInstance();
    private      static      final      String      DOCUMENTO_BUSINESS_JNDI_NAME      =
"cate/DocumentosBusiness";
    private     static     final     String     DOCUMENTO_BUSINESS_LOCAL_JNDI_NAME     =
"cate/DocumentosBusinessLocal";

    public Collection getDestinatarios() {
           List retorno = new ArrayList();
           HashMap map = new HashMap();
           map.put("id", "1");
           map.put("nome", "Município de Florianópolis");
           retorno.add(map);
           map = new HashMap();
           map.put("id", "1");
           map.put("nome", "Município de São José");
           retorno.add(map);
           map = new HashMap();
           map.put("id", "1");
           map.put("nome", "Município de Jacinto Machado");
           retorno.add(map);
           map = new HashMap();
           map.put("id", "1");
           map.put("nome", "Município de Armazém");
           retorno.add(map);
           map = new HashMap();
           map.put("id", "1");
           map.put("nome", "Município de Tubarão");
           retorno.add(map);
           map = new HashMap();
           map.put("id", "1");
           map.put("nome", "Município de Palhoça");
           retorno.add(map);
           /*try {
                  retorno = this.getDocumentosBusinessLocal().getDestinatarios();
           } catch (Exception e) {
                  // TODO Auto-generated catch block
                  e.printStackTrace();
           }*/
           return retorno;
    }

    public Collection getAssinantes(){
           List retorno = new ArrayList();
           HashMap map = new HashMap();
           map.put("id", "1");
           map.put("nome", "Edinando Brustolin");
           retorno.add(map);
           /*try {
                  retorno = this.getDocumentosBusinessLocal().getAssinantes();
           } catch (Exception e) {
                  // TODO Auto-generated catch block
                  e.printStackTrace();
```

```
		}*/
		return retorno;
	}

	public Collection getAutores(){
		List retorno = new ArrayList();
		HashMap map = new HashMap();
		map.put("id", "1");
		map.put("nome", "João da Silva Machado");
		retorno.add(map);
		/*List retorno = null;
		try {
			retorno = this.getDocumentosBusinessLocal().getAutores();
		} catch (Exception e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}*/
		return retorno;
	}

	public Collection getFormasEnvio(){
		List retorno = new ArrayList();
		HashMap map = new HashMap();
		map.put("id", "1");
		map.put("descricao", "Correio");
		retorno.add(map);
		/*List retorno = null;
		try {
			retorno = this.getDocumentosBusinessLocal().getFormasEnvio();
		} catch (Exception e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}*/
		return retorno;
	}

	public List getModalidades(){
		List retorno = new ArrayList();
		HashMap map = new HashMap();
		map.put("id", "1");
		map.put("descricao", "Ofício Normal");
		retorno.add(map);
		map = new HashMap();
		map.put("id", "1");
		map.put("descricao", "Ofício Circular");
		retorno.add(map);
		map = new HashMap();
		map.put("id", "1");
		map.put("descricao", "Comunicado");
		retorno.add(map);
		map = new HashMap();
		map.put("id", "1");
		map.put("descricao", "Parecer");
		retorno.add(map);
		map = new HashMap();
		map.put("id", "1");
		map.put("descricao", "Edital");
		retorno.add(map);
		map = new HashMap();
		map.put("id", "1");
		map.put("descricao", "Resolução");
		retorno.add(map);
		/*List retorno = null;
		try {
			retorno = this.getDocumentosBusinessLocal().getModalidades();
```

```
            } catch (Exception e) {
                  // TODO Auto-generated catch block
                  e.printStackTrace();
            }*/
            return retorno;
      }

      public void cadastrarDocumento(HashMap params) throws Exception {
            this.getDocumentosBusinessLocal().cadastrarDocumento(params);
      }

      public List filtrarDocumentos(HashMap params) {
            List retorno = new ArrayList();
            HashMap map = new HashMap();
            map.put("identificador", "PAR-000001");
            map.put("assunto", "Parecer de Alteração do Tributos Fiscais");
            map.put("modalidade", "Parecer");
            map.put("autor", "João da Silva Machado");
            map.put("assinante", "Edinando Brustolin");
            map.put("destinatario", new String[] {"Município de São José", "Município de
Jacinto Machado"});
            map.put("formaEnvio", "");
            map.put("dataElaboracao", "06/02/2005");
            map.put("dataEnvio", "");
            map.put("arquivo", "parecer_alteracao_tributos_fiscais.doc");
            map.put("status", "Não Enviado");
            retorno.add(map);
            /*List retorno = new ArrayList();
            try {
                  retorno                                                    =
this.getDocumentosBusinessLocal().filtrarDocumentos(params);
            } catch (Exception e) {
                  // TODO Auto-generated catch block
                  e.printStackTrace();
            }*/
            return retorno;
      }

      private DocumentosBusiness getDocumentosBusiness() throws Exception {
            DocumentosBusinessHome                        home                 =
(DocumentosBusinessHome)sl.getRemoteHome(DOCUMENTO_BUSINESS_JNDI_NAME,
DocumentosBusinessHome.class);
            return home.create();
      }

      private DocumentosBusinessLocal getDocumentosBusinessLocal() throws Exception {
            DocumentosBusinessLocalHome                  home                 =
(DocumentosBusinessLocalHome)sl.getLocalHome(DOCUMENTO_BUSINESS_LOCAL_JNDI_NAME,
DocumentosBusinessLocalHome.class);
            return home.create();
      }

      public static BusinessDelegate getInstance() {
            if (buss == null) {
                  buss = new BusinessDelegate();
            }

            return buss;
      }
}
```

12.1.5. Apoiador.java

```
package br.ufsc.cate.hibernate;

import java.io.Serializable;
import java.util.Set;
import org.apache.commons.lang.builder.ToStringBuilder;


/** @author Hibernate CodeGenerator */
public class Apoiador implements Serializable {

    /** identifier field */
    private Integer id;

    /** persistent field */
    private String nome;

    /** persistent field */
    private Set atividades;

    /** full constructor */
    public Apoiador(Integer id, String nome, Set atividades) {
        this.id = id;
        this.nome = nome;
        this.atividades = atividades;
    }

    /** default constructor */
    public Apoiador() {
    }

    public Integer getId() {
        return this.id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public String getNome() {
        return this.nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }

    public Set getAtividades() {
        return this.atividades;
    }

    public void setAtividades(Set atividades) {
        this.atividades = atividades;
    }

    public String toString() {
        return new ToStringBuilder(this)
            .append("id", getId())
            .toString();
    }

}
```

12.1.6. Assinante.java

```
package br.ufsc.cate.hibernate;

import java.io.Serializable;
import java.util.Set;
import org.apache.commons.lang.builder.ToStringBuilder;


/** @author Hibernate CodeGenerator */
public class Assinante implements Serializable {

    /** identifier field */
    private Integer id;

    /** persistent field */
    private String nome;

    /** persistent field */
    private Set documentos;

    /** full constructor */
    public Assinante(Integer id, String nome, Set documentos) {
        this.id = id;
        this.nome = nome;
        this.documentos = documentos;
    }

    /** default constructor */
    public Assinante() {
    }

    public Integer getId() {
        return this.id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public String getNome() {
        return this.nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }

    public Set getDocumentos() {
        return this.documentos;
    }

    public void setDocumentos(Set documentos) {
        this.documentos = documentos;
    }

    public String toString() {
        return new ToStringBuilder(this)
            .append("id", getId())
            .toString();
    }

}
```

12.1.7. Atividade.java

```
package br.ufsc.cate.hibernate;

import java.io.Serializable;
import org.apache.commons.lang.builder.ToStringBuilder;


/** @author Hibernate CodeGenerator */
public class Atividade implements Serializable {

    /** identifier field */
    private Integer id;

    /** persistent field */
    private String caminhorRelativoPauta;

    /** nullable persistent field */
    private String caminhorRelativoResumo;

    /** persistent field */
    private long data;

    /** persistent field */
    private String duracao;

    /** persistent field */
    private String horario;

    /** persistent field */
    private String local;

    /** persistent field */
    private String nomeArquivoPauta;

    /** nullable persistent field */
    private String nomeArquivoResumo;

    /** nullable persistent field */
    private String textoResumo;

    /** persistent field */
    private String textopauta;

    /** persistent field */
    private String titulo;

    /** nullable persistent field */
    private br.ufsc.cate.hibernate.EspecieAtividade especieatividade;

    /** nullable persistent field */
    private br.ufsc.cate.hibernate.Apoiador apoiador;

    /** nullable persistent field */
    private br.ufsc.cate.hibernate.Responsavel responsavel;

    /** nullable persistent field */
    private br.ufsc.cate.hibernate.ProjetoFecam projetofecam;

    /** nullable persistent field */
```

```
    private br.ufsc.cate.hibernate.ConselhoRepresentacao conselhorepresentacao;

    /** full constructor */
    public Atividade(Integer id, String caminhorRelativoPauta, String
caminhorRelativoResumo, long data, String duracao, String horario, String local, String
nomeArquivoPauta, String nomeArquivoResumo, String textoResumo, String textopauta, String
titulo, br.ufsc.cate.hibernate.EspecieAtividade especieatividade,
br.ufsc.cate.hibernate.Apoiador apoiador, br.ufsc.cate.hibernate.Responsavel responsavel,
br.ufsc.cate.hibernate.ProjetoFecam projetofecam,
br.ufsc.cate.hibernate.ConselhoRepresentacao conselhorepresentacao) {
        this.id = id;
        this.caminhorRelativoPauta = caminhorRelativoPauta;
        this.caminhorRelativoResumo = caminhorRelativoResumo;
        this.data = data;
        this.duracao = duracao;
        this.horario = horario;
        this.local = local;
        this.nomeArquivoPauta = nomeArquivoPauta;
        this.nomeArquivoResumo = nomeArquivoResumo;
        this.textoResumo = textoResumo;
        this.textopauta = textopauta;
        this.titulo = titulo;
        this.especieatividade = especieatividade;
        this.apoiador = apoiador;
        this.responsavel = responsavel;
        this.projetofecam = projetofecam;
        this.conselhorepresentacao = conselhorepresentacao;
    }

    /** default constructor */
    public Atividade() {
    }

    /** minimal constructor */
    public Atividade(Integer id, String caminhorRelativoPauta, long data, String duracao,
String horario, String local, String nomeArquivoPauta, String textopauta, String titulo)
{
        this.id = id;
        this.caminhorRelativoPauta = caminhorRelativoPauta;
        this.data = data;
        this.duracao = duracao;
        this.horario = horario;
        this.local = local;
        this.nomeArquivoPauta = nomeArquivoPauta;
        this.textopauta = textopauta;
        this.titulo = titulo;
    }

    public Integer getId() {
        return this.id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public String getCaminhorRelativoPauta() {
        return this.caminhorRelativoPauta;
    }

    public void setCaminhorRelativoPauta(String caminhorRelativoPauta) {
        this.caminhorRelativoPauta = caminhorRelativoPauta;
    }

    public String getCaminhorRelativoResumo() {
```

```
    return this.caminhorRelativoResumo;
}

public void setCaminhorRelativoResumo(String caminhorRelativoResumo) {
    this.caminhorRelativoResumo = caminhorRelativoResumo;
}

public long getData() {
    return this.data;
}

public void setData(long data) {
    this.data = data;
}

public String getDuracao() {
    return this.duracao;
}

public void setDuracao(String duracao) {
    this.duracao = duracao;
}

public String getHorario() {
    return this.horario;
}

public void setHorario(String horario) {
    this.horario = horario;
}

public String getLocal() {
    return this.local;
}

public void setLocal(String local) {
    this.local = local;
}

public String getNomeArquivoPauta() {
    return this.nomeArquivoPauta;
}

public void setNomeArquivoPauta(String nomeArquivoPauta) {
    this.nomeArquivoPauta = nomeArquivoPauta;
}

public String getNomeArquivoResumo() {
    return this.nomeArquivoResumo;
}

public void setNomeArquivoResumo(String nomeArquivoResumo) {
    this.nomeArquivoResumo = nomeArquivoResumo;
}

public String getTextoResumo() {
    return this.textoResumo;
}

public void setTextoResumo(String textoResumo) {
    this.textoResumo = textoResumo;
}

public String getTextopauta() {
    return this.textopauta;
```

```
    }

    public void setTextopauta(String textopauta) {
        this.textopauta = textopauta;
    }

    public String getTitulo() {
        return this.titulo;
    }

    public void setTitulo(String titulo) {
        this.titulo = titulo;
    }

    public br.ufsc.cate.hibernate.EspecieAtividade getEspecieatividade() {
        return this.especieatividade;
    }

    public        void        setEspecieatividade(br.ufsc.cate.hibernate.EspecieAtividade
especieatividade) {
        this.especieatividade = especieatividade;
    }

    public br.ufsc.cate.hibernate.Apoiador getApoiador() {
        return this.apoiador;
    }

    public void setApoiador(br.ufsc.cate.hibernate.Apoiador apoiador) {
        this.apoiador = apoiador;
    }

    public br.ufsc.cate.hibernate.Responsavel getResponsavel() {
        return this.responsavel;
    }

    public void setResponsavel(br.ufsc.cate.hibernate.Responsavel responsavel) {
        this.responsavel = responsavel;
    }

    public br.ufsc.cate.hibernate.ProjetoFecam getProjetofecam() {
        return this.projetofecam;
    }

    public void setProjetofecam(br.ufsc.cate.hibernate.ProjetoFecam projetofecam) {
        this.projetofecam = projetofecam;
    }

    public br.ufsc.cate.hibernate.ConselhoRepresentacao getConselhorepresentacao() {
        return this.conselhorepresentacao;
    }

    public   void   setConselhorepresentacao(br.ufsc.cate.hibernate.ConselhoRepresentacao
conselhorepresentacao) {
        this.conselhorepresentacao = conselhorepresentacao;
    }

    public String toString() {
        return new ToStringBuilder(this)
            .append("id", getId())
            .toString();
    }

}
```

12.1.8. Autor.java

```
package br.ufsc.cate.hibernate;

import java.io.Serializable;
import java.util.Set;
import org.apache.commons.lang.builder.ToStringBuilder;


/** @author Hibernate CodeGenerator */
public class Autor implements Serializable {

    /** identifier field */
    private Integer id;

    /** persistent field */
    private String nome;

    /** persistent field */
    private Set documentos;

    /** full constructor */
    public Autor(Integer id, String nome, Set documentos) {
        this.id = id;
        this.nome = nome;
        this.documentos = documentos;
    }

    /** default constructor */
    public Autor() {
    }

    public Integer getId() {
        return this.id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public String getNome() {
        return this.nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }

    public Set getDocumentos() {
        return this.documentos;
    }

    public void setDocumentos(Set documentos) {
        this.documentos = documentos;
    }

    public String toString() {
        return new ToStringBuilder(this)
            .append("id", getId())
            .toString();
    }

}
```

12.1.9. ConselhoRepresentacao.java

```
package br.ufsc.cate.hibernate;

import java.io.Serializable;
import java.util.Set;
import org.apache.commons.lang.builder.ToStringBuilder;


/** @author Hibernate CodeGenerator */
public class ConselhoRepresentacao implements Serializable {

    /** identifier field */
    private Integer id;

    /** persistent field */
    private String nome;

    /** persistent field */
    private Set atividades;

    /** persistent field */
    private Set gestaos;

    /** full constructor */
    public ConselhoRepresentacao(Integer id, String nome, Set atividades, Set gestaos) {
        this.id = id;
        this.nome = nome;
        this.atividades = atividades;
        this.gestaos = gestaos;
    }

    /** default constructor */
    public ConselhoRepresentacao() {
    }

    public Integer getId() {
        return this.id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public String getNome() {
        return this.nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }

    public Set getAtividades() {
        return this.atividades;
    }

    public void setAtividades(Set atividades) {
        this.atividades = atividades;
    }

    public Set getGestaos() {
        return this.gestaos;
```

```
    }

    public void setGestaos(Set gestaos) {
        this.gestaos = gestaos;
    }

    public String toString() {
        return new ToStringBuilder(this)
            .append("id", getId())
            .toString();
    }

}
```

12.1.10. Destinarario.java

```
package br.ufsc.cate.hibernate;

import java.io.Serializable;
import java.util.Set;
import org.apache.commons.lang.builder.ToStringBuilder;


/** @author Hibernate CodeGenerator */
public class Destinatario implements Serializable {

    /** identifier field */
    private Integer id;

    /** persistent field */
    private String nome;

    /** persistent field */
    private int tipo;

    /** persistent field */
    private Set documentos;

    /** full constructor */
    public Destinatario(Integer id, String nome, int tipo, Set destinatariodocumentos) {
        this.id = id;
        this.nome = nome;
        this.tipo = tipo;
        this.documentos = destinatariodocumentos;
    }

    /** default constructor */
    public Destinatario() {
    }

    public Integer getId() {
        return this.id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public String getNome() {
        return this.nome;
    }
```

```
    public void setNome(String nome) {
        this.nome = nome;
    }

    public int getTipo() {
        return this.tipo;
    }

    public void setTipo(int tipo) {
        this.tipo = tipo;
    }

    public Set getDocumentos() {
        return this.documentos;
    }

    public void setDocumentos(Set destinatariodocumentos) {
        this.documentos = destinatariodocumentos;
    }

    public String toString() {
        return new ToStringBuilder(this)
            .append("id", getId())
            .toString();
    }

}
```

12.1.11. Documento.java

```
package br.ufsc.cate.hibernate;

import java.io.Serializable;
import java.util.Set;
import org.apache.commons.lang.builder.ToStringBuilder;


/** @author Hibernate CodeGenerator */
public class Documento implements Serializable {

    /** identifier field */
    private Integer id;

    /** persistent field */
    private String caminhoRelativo;

    /** persistent field */
    private long dataElaboracao;

    /** nullable persistent field */
    private Long dataEnvio;

    /** persistent field */
    private String identificador;

    /** persistent field */
    private String nomeArquivo;

    /** nullable persistent field */
    private br.ufsc.cate.hibernate.Modalidade modalidade;
```

```
    /** nullable persistent field */
    private br.ufsc.cate.hibernate.Autor autor;

    /** nullable persistent field */
    private br.ufsc.cate.hibernate.Assinante assinante;

    /** nullable persistent field */
    private br.ufsc.cate.hibernate.FormaEnvio formaenvio;

    /** persistent field */
    private Set destinatarios;

    /** full constructor */
    public Documento(Integer id, String caminhoRelativo, long dataElaboracao, Long
dataEnvio, String identificador, String nomeArquivo, br.ufsc.cate.hibernate.Modalidade
modalidade,    br.ufsc.cate.hibernate.Autor    autor,    br.ufsc.cate.hibernate.Assinante
assinante, br.ufsc.cate.hibernate.FormaEnvio formaenvio, Set destinatariodocumentos) {
        this.id = id;
        this.caminhoRelativo = caminhoRelativo;
        this.dataElaboracao = dataElaboracao;
        this.dataEnvio = dataEnvio;
        this.identificador = identificador;
        this.nomeArquivo = nomeArquivo;
        this.modalidade = modalidade;
        this.autor = autor;
        this.assinante = assinante;
        this.formaenvio = formaenvio;
        this.destinatarios = destinatariodocumentos;
    }

    /** default constructor */
    public Documento() {
    }

    /** minimal constructor */
    public Documento(Integer id, String caminhoRelativo, long dataElaboracao, String
identificador, String nomeArquivo, Set destinatariodocumentos) {
        this.id = id;
        this.caminhoRelativo = caminhoRelativo;
        this.dataElaboracao = dataElaboracao;
        this.identificador = identificador;
        this.nomeArquivo = nomeArquivo;
        this.destinatarios = destinatariodocumentos;
    }

    public Integer getId() {
        return this.id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public String getCaminhoRelativo() {
        return this.caminhoRelativo;
    }

    public void setCaminhoRelativo(String caminhoRelativo) {
        this.caminhoRelativo = caminhoRelativo;
    }

    public long getDataElaboracao() {
        return this.dataElaboracao;
    }
```

```
public void setDataElaboracao(long dataElaboracao) {
    this.dataElaboracao = dataElaboracao;
}

public Long getDataEnvio() {
    return this.dataEnvio;
}

public void setDataEnvio(Long dataEnvio) {
    this.dataEnvio = dataEnvio;
}

public String getIdentificador() {
    return this.identificador;
}

public void setIdentificador(String identificador) {
    this.identificador = identificador;
}

public String getNomeArquivo() {
    return this.nomeArquivo;
}

public void setNomeArquivo(String nomeArquivo) {
    this.nomeArquivo = nomeArquivo;
}

public br.ufsc.cate.hibernate.Modalidade getModalidade() {
    return this.modalidade;
}

public void setModalidade(br.ufsc.cate.hibernate.Modalidade modalidade) {
    this.modalidade = modalidade;
}

public br.ufsc.cate.hibernate.Autor getAutor() {
    return this.autor;
}

public void setAutor(br.ufsc.cate.hibernate.Autor autor) {
    this.autor = autor;
}

public br.ufsc.cate.hibernate.Assinante getAssinante() {
    return this.assinante;
}

public void setAssinante(br.ufsc.cate.hibernate.Assinante assinante) {
    this.assinante = assinante;
}

public br.ufsc.cate.hibernate.FormaEnvio getFormaenvio() {
    return this.formaenvio;
}

public void setFormaenvio(br.ufsc.cate.hibernate.FormaEnvio formaenvio) {
    this.formaenvio = formaenvio;
}

public Set getDestinatarios() {
    return this.destinatarios;
}
```

```
    public void setDestinatarios(Set destinatariodocumentos) {
        this.destinatarios = destinatariodocumentos;
    }

    public String toString() {
        return new ToStringBuilder(this)
            .append("id", getId())
            .toString();
    }

}
```

12.1.12. EspecieAtividade.java

```
package br.ufsc.cate.hibernate;

import java.io.Serializable;
import java.util.Set;
import org.apache.commons.lang.builder.ToStringBuilder;


/** @author Hibernate CodeGenerator */
public class EspecieAtividade implements Serializable {

    /** identifier field */
    private Integer id;

    /** persistent field */
    private String descricao;

    /** persistent field */
    private int sequencial;

    /** persistent field */
    private String sigla;

    /** persistent field */
    private Set atividades;

    /** full constructor */
    public EspecieAtividade(Integer id, String descricao, int sequencial, String sigla,
Set atividades) {
        this.id = id;
        this.descricao = descricao;
        this.sequencial = sequencial;
        this.sigla = sigla;
        this.atividades = atividades;
    }

    /** default constructor */
    public EspecieAtividade() {
    }

    public Integer getId() {
        return this.id;
    }

    public void setId(Integer id) {
        this.id = id;
    }
```

```
    public String getDescricao() {
        return this.descricao;
    }

    public void setDescricao(String descricao) {
        this.descricao = descricao;
    }

    public int getSequencial() {
        return this.sequencial;
    }

    public void setSequencial(int sequencial) {
        this.sequencial = sequencial;
    }

    public String getSigla() {
        return this.sigla;
    }

    public void setSigla(String sigla) {
        this.sigla = sigla;
    }

    public Set getAtividades() {
        return this.atividades;
    }

    public void setAtividades(Set atividades) {
        this.atividades = atividades;
    }

    public String toString() {
        return new ToStringBuilder(this)
            .append("id", getId())
            .toString();
    }

}
```

12.1.13. FormaEnvio.java

```
package br.ufsc.cate.hibernate;

import java.io.Serializable;
import java.util.Set;
import org.apache.commons.lang.builder.ToStringBuilder;


/** @author Hibernate CodeGenerator */
public class FormaEnvio implements Serializable {

    /** identifier field */
    private Integer id;

    /** persistent field */
    private String descricao;

    /** persistent field */
    private Set documentos;
```

```
    /** full constructor */
    public FormaEnvio(Integer id, String descricao, Set documentos) {
        this.id = id;
        this.descricao = descricao;
        this.documentos = documentos;
    }

    /** default constructor */
    public FormaEnvio() {
    }

    public Integer getId() {
        return this.id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public String getDescricao() {
        return this.descricao;
    }

    public void setDescricao(String descricao) {
        this.descricao = descricao;
    }

    public Set getDocumentos() {
        return this.documentos;
    }

    public void setDocumentos(Set documentos) {
        this.documentos = documentos;
    }

    public String toString() {
        return new ToStringBuilder(this)
            .append("id", getId())
            .toString();
    }

}
```

12.1.14. Gestao.java

```
package br.ufsc.cate.hibernate;

import java.io.Serializable;
import org.apache.commons.lang.builder.ToStringBuilder;


/** @author Hibernate CodeGenerator */
public class Gestao implements Serializable {

    /** identifier field */
    private Integer id;

    /** nullable persistent field */
    private Integer dataFim;

    /** nullable persistent field */
```

```
    private Integer datainicio;

    /** persistent field */
    private String suplente;

    /** persistent field */
    private String titular;

    /** nullable persistent field */
    private br.ufsc.cate.hibernate.ConselhoRepresentacao conselhorepresentacao;

    /** full constructor */
    public Gestao(Integer id, Integer dataFim, Integer datainicio, String suplente,
String titular, br.ufsc.cate.hibernate.ConselhoRepresentacao conselhorepresentacao) {
        this.id = id;
        this.dataFim = dataFim;
        this.datainicio = datainicio;
        this.suplente = suplente;
        this.titular = titular;
        this.conselhorepresentacao = conselhorepresentacao;
    }

    /** default constructor */
    public Gestao() {
    }

    /** minimal constructor */
    public Gestao(Integer id, String suplente, String titular) {
        this.id = id;
        this.suplente = suplente;
        this.titular = titular;
    }

    public Integer getId() {
        return this.id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public Integer getDataFim() {
        return this.dataFim;
    }

    public void setDataFim(Integer dataFim) {
        this.dataFim = dataFim;
    }

    public Integer getDatainicio() {
        return this.datainicio;
    }

    public void setDatainicio(Integer datainicio) {
        this.datainicio = datainicio;
    }

    public String getSuplente() {
        return this.suplente;
    }

    public void setSuplente(String suplente) {
        this.suplente = suplente;
    }
```

```
    public String getTitular() {
        return this.titular;
    }

    public void setTitular(String titular) {
        this.titular = titular;
    }

    public br.ufsc.cate.hibernate.ConselhoRepresentacao getConselhorepresentacao() {
        return this.conselhorepresentacao;
    }

    public   void   setConselhorepresentacao(br.ufsc.cate.hibernate.ConselhoRepresentacao
conselhorepresentacao) {
        this.conselhorepresentacao = conselhorepresentacao;
    }

    public String toString() {
        return new ToStringBuilder(this)
            .append("id", getId())
            .toString();
    }

}
```

12.1.15. Modalidade.java

```
package br.ufsc.cate.hibernate;

import java.io.Serializable;
import java.util.Set;
import org.apache.commons.lang.builder.ToStringBuilder;


/** @author Hibernate CodeGenerator */
public class Modalidade implements Serializable {

    /** identifier field */
    private Integer id;

    /** persistent field */
    private String descricao;

    /** persistent field */
    private int sequencial;

    /** persistent field */
    private String sigla;

    /** persistent field */
    private Set documentos;

    /** full constructor */
    public Modalidade(Integer id, String descricao, int sequencial, String sigla, Set
documentos) {
        this.id = id;
        this.descricao = descricao;
        this.sequencial = sequencial;
        this.sigla = sigla;
        this.documentos = documentos;
    }
```

```
    /** default constructor */
    public Modalidade() {
    }

    public Integer getId() {
        return this.id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public String getDescricao() {
        return this.descricao;
    }

    public void setDescricao(String descricao) {
        this.descricao = descricao;
    }

    public int getSequencial() {
        return this.sequencial;
    }

    public void setSequencial(int sequencial) {
        this.sequencial = sequencial;
    }

    public String getSigla() {
        return this.sigla;
    }

    public void setSigla(String sigla) {
        this.sigla = sigla;
    }

    public Set getDocumentos() {
        return this.documentos;
    }

    public void setDocumentos(Set documentos) {
        this.documentos = documentos;
    }

    public String toString() {
        return new ToStringBuilder(this)
            .append("id", getId())
            .toString();
    }

}
```

12.1.16. ProjetoFecan.java

```
package br.ufsc.cate.hibernate;

import java.io.Serializable;
import java.util.Set;
import org.apache.commons.lang.builder.ToStringBuilder;
```

```
/** @author Hibernate CodeGenerator */
public class ProjetoFecam implements Serializable {

    /** identifier field */
    private Integer id;

    /** persistent field */
    private String nome;

    /** persistent field */
    private Set atividades;

    /** full constructor */
    public ProjetoFecam(Integer id, String nome, Set atividades) {
        this.id = id;
        this.nome = nome;
        this.atividades = atividades;
    }

    /** default constructor */
    public ProjetoFecam() {
    }

    public Integer getId() {
        return this.id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public String getNome() {
        return this.nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }

    public Set getAtividades() {
        return this.atividades;
    }

    public void setAtividades(Set atividades) {
        this.atividades = atividades;
    }

    public String toString() {
        return new ToStringBuilder(this)
            .append("id", getId())
            .toString();
    }

}
```

12.1.17. Responsavel.java

```
package br.ufsc.cate.hibernate;
```

```
import java.io.Serializable;
import java.util.Set;
import org.apache.commons.lang.builder.ToStringBuilder;


/** @author Hibernate CodeGenerator */
public class Responsavel implements Serializable {

    /** identifier field */
    private Integer id;

    /** persistent field */
    private String nome;

    /** persistent field */
    private Set atividades;

    /** full constructor */
    public Responsavel(Integer id, String nome, Set atividades) {
        this.id = id;
        this.nome = nome;
        this.atividades = atividades;
    }

    /** default constructor */
    public Responsavel() {
    }

    public Integer getId() {
        return this.id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public String getNome() {
        return this.nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }

    public Set getAtividades() {
        return this.atividades;
    }

    public void setAtividades(Set atividades) {
        this.atividades = atividades;
    }

    public String toString() {
        return new ToStringBuilder(this)
            .append("id", getId())
            .toString();
    }
}
```

12.1.18. Apoiador.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 2.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-2.0.dtd">
```

```
<hibernate-mapping>
    <class name="br.ufsc.cate.hibernate.Apoiador" table="apoiador">
        <id name="id" column="id_Apoiador" type="int">
            <generator class="assigned" />
        </id>
        <property name="nome" column="nome" type="string" not-null="true" />
        <set name="atividades" cascade="none" lazy="false">
            <key column="id_Apoiador" />
            <one-to-many class="br.ufsc.cate.hibernate.Atividade" />
        </set>
    </class>
</hibernate-mapping>
```

12.1.19. Assinante.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 2.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-2.0.dtd">
<hibernate-mapping>
    <class name="br.ufsc.cate.hibernate.Assinante" table="assinante">
        <id name="id" column="id_Assinante" type="int">
            <generator class="assigned" />
        </id>
        <property name="nome" column="nome" type="string" not-null="true" />
        <set name="documentos" cascade="none" lazy="false">
            <key column="id_Assinante" />
            <one-to-many class="br.ufsc.cate.hibernate.Documento" />
        </set>
    </class>
</hibernate-mapping>
```

12.1.20. Atividade.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 2.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-2.0.dtd">
<hibernate-mapping>
    <class name="br.ufsc.cate.hibernate.Atividade" table="atividade">
        <id name="id" column="id_Atividade" type="int">
            <generator class="assigned" />
        </id>
        <property name="caminhorRelativoPauta" column="caminhor_Relativo_Pauta"
type="string" not-null="true" />
        <property name="caminhorRelativoResumo" column="caminhor_Relativo_Resumo"
type="string" />
        <property name="data" column="data" type="long" not-null="true" />
        <property name="duracao" column="duracao" type="string" not-null="true" />
        <property name="horario" column="horario" type="string" not-null="true" />
        <property name="local" column="local" type="string" not-null="true" />
        <property name="nomeArquivoPauta" column="nome_Arquivo_Pauta" type="string" not-
null="true" />
        <property name="nomeArquivoResumo" column="nome_Arquivo_Resumo" type="string" />
        <property name="textoResumo" column="texto_Resumo" type="string" />
        <property name="textopauta" column="textoPauta" type="string" not-null="true" />
        <property name="titulo" column="titulo" type="string" not-null="true" />
        <many-to-one name="especieatividade" column="id_Especie_Atividade"
class="br.ufsc.cate.hibernate.EspecieAtividade" cascade="none" />
        <many-to-one name="apoiador" column="id_Apoiador"
class="br.ufsc.cate.hibernate.Apoiador" cascade="none" />
        <many-to-one name="responsavel" column="id_Responsavel"
class="br.ufsc.cate.hibernate.Responsavel" cascade="none" />
```

```
        <many-to-one               name="projetofecam"               column="id_Projeto"
class="br.ufsc.cate.hibernate.ProjetoFecam" cascade="none" />
        <many-to-one          name="conselhorepresentacao"          column="id_Conselho"
class="br.ufsc.cate.hibernate.ConselhoRepresentacao" cascade="none" />
    </class>
</hibernate-mapping>
```

12.1.21. Autor.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE   hibernate-mapping   PUBLIC   "-//Hibernate/Hibernate   Mapping   DTD   2.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-2.0.dtd">
<hibernate-mapping>
    <class name="br.ufsc.cate.hibernate.Autor" table="autor">
        <id name="id" column="id_Autor" type="int">
            <generator class="assigned" />
        </id>
        <property name="nome" column="nome" type="string" not-null="true" />
        <set name="documentos" cascade="none" lazy="false">
            <key column="id_Autor" />
            <one-to-many class="br.ufsc.cate.hibernate.Documento" />
        </set>
    </class>
</hibernate-mapping>
```

12.1.22. ConselhoRepresentacao.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE   hibernate-mapping   PUBLIC   "-//Hibernate/Hibernate   Mapping   DTD   2.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-2.0.dtd">
<hibernate-mapping>
    <class                          name="br.ufsc.cate.hibernate.ConselhoRepresentacao"
table="conselho_representacao">
        <id name="id" column="id_Conselho" type="int">
            <generator class="assigned" />
        </id>
        <property name="nome" column="nome" type="string" not-null="true" />
        <set name="atividades" cascade="none" lazy="false">
            <key column="id_Conselho" />
            <one-to-many class="br.ufsc.cate.hibernate.Atividade" />
        </set>
        <set name="gestaos" cascade="none" lazy="false">
            <key column="id_Conselho" />
            <one-to-many class="br.ufsc.cate.hibernate.Gestao" />
        </set>
    </class>
</hibernate-mapping>
```

12.1.23. Destinatário.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE   hibernate-mapping   PUBLIC   "-//Hibernate/Hibernate   Mapping   DTD   2.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-2.0.dtd">
```

```
<hibernate-mapping>
    <class name="br.ufsc.cate.hibernate.Destinatario" table="destinatario">
        <id name="id" column="id_Destinatario" type="int">
            <generator class="assigned" />
        </id>
        <property name="nome" column="nome" type="string" not-null="true" />
        <property name="tipo" column="tipo" type="int" not-null="true" />
        <set     name="documentos"     table="destinatario_documento"     cascade="none"
lazy="false">
            <key column="id_Documento" />
            <many-to-many class="br.ufsc.cate.hibernate.Documento" />
        </set>
    </class>
</hibernate-mapping>
```

12.1.24. Documento.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE   hibernate-mapping   PUBLIC   "-//Hibernate/Hibernate   Mapping   DTD   2.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-2.0.dtd">
<hibernate-mapping>
    <class name="br.ufsc.cate.hibernate.Documento" table="documento">
        <id name="id" column="id_Documento" type="int">
            <generator class="assigned" />
        </id>
        <property  name="caminhoRelativo"  column="caminho_Relativo"  type="string"  not-
null="true" />
        <property   name="dataElaboracao"   column="data_Elaboracao"   type="long"   not-
null="true" />
        <property name="dataEnvio" column="data_Envio" type="long" />
        <property   name="identificador"   column="identificador"   type="string"   not-
null="true" />
        <property name="nomeArquivo" column="nome_Arquivo" type="string" not-null="true"
/>
        <many-to-one               name="modalidade"               column="id_Modalidade"
class="br.ufsc.cate.hibernate.Modalidade" cascade="none" />
        <many-to-one  name="autor"  column="id_Autor"  class="br.ufsc.cate.hibernate.Autor"
cascade="none" />
        <many-to-one                name="assinante"                column="id_Assinante"
class="br.ufsc.cate.hibernate.Assinante" cascade="none" />
        <many-to-one               name="formaenvio"               column="id_Forma_Envio"
class="br.ufsc.cate.hibernate.FormaEnvio" cascade="none" />
        <set    name="destinatarios"    table="destinatario_documento"    cascade="none"
lazy="false">
            <key column="id_Destinatario" />
            <many-to-many class="br.ufsc.cate.hibernate.Destinatario" />
        </set>
    </class>
</hibernate-mapping>
```

12.1.25. EspecieAtividade.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE   hibernate-mapping   PUBLIC   "-//Hibernate/Hibernate   Mapping   DTD   2.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-2.0.dtd">
```

```
<hibernate-mapping>
    <class name="br.ufsc.cate.hibernate.EspecieAtividade" table="especie_atividade">
        <id name="id" column="id_Especie_Atividade" type="int">
            <generator class="assigned" />
        </id>
        <property name="descricao" column="descricao" type="string" not-null="true" />
        <property name="sequencial" column="sequencial" type="int" not-null="true" />
        <property name="sigla" column="sigla" type="string" not-null="true" />
        <set name="atividades" cascade="none" lazy="false">
            <key column="id_Especie_Atividade" />
            <one-to-many class="br.ufsc.cate.hibernate.Atividade" />
        </set>
    </class>
</hibernate-mapping>
```

12.1.26. FormaEnvio.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 2.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-2.0.dtd">
<hibernate-mapping>
    <class name="br.ufsc.cate.hibernate.FormaEnvio" table="forma_envio">
        <id name="id" column="id_Forma_Envio" type="int">
            <generator class="assigned" />
        </id>
        <property name="descricao" column="descricao" type="string" not-null="true" />
        <set name="documentos" cascade="none" lazy="false">
            <key column="id_Forma_Envio" />
            <one-to-many class="br.ufsc.cate.hibernate.Documento" />
        </set>
    </class>
</hibernate-mapping>
```

12.1.27. Gestão.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 2.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-2.0.dtd">
<hibernate-mapping>
    <class name="br.ufsc.cate.hibernate.Gestao" table="gestao">
        <id name="id" column="id_Gestao" type="int">
            <generator class="assigned" />
        </id>
        <property name="dataFim" column="data_Fim" type="int" />
        <property name="datainicio" column="dataInicio" type="int" />
        <property name="suplente" column="suplente" type="string" not-null="true" />
        <property name="titular" column="titular" type="string" not-null="true" />
        <many-to-one name="conselhorepresentacao" column="id_Conselho"
class="br.ufsc.cate.hibernate.ConselhoRepresentacao" cascade="none" />
    </class>
</hibernate-mapping>
```

12.1.28. Modalidade.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 2.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-2.0.dtd">
```

```
<hibernate-mapping>
    <class name="br.ufsc.cate.hibernate.Modalidade" table="modalidade">
        <id name="id" column="id_Modalidade" type="int">
            <generator class="assigned" />
        </id>
        <property name="descricao" column="descricao" type="string" not-null="true" />
        <property name="sequencial" column="sequencial" type="int" not-null="true" />
        <property name="sigla" column="sigla" type="string" not-null="true" />
        <set name="documentos" cascade="none" lazy="false">
            <key column="id_Modalidade" />
            <one-to-many class="br.ufsc.cate.hibernate.Documento" />
        </set>
    </class>
</hibernate-mapping>
```

12.1.29. ProjetoFecam.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE   hibernate-mapping   PUBLIC   "-//Hibernate/Hibernate   Mapping   DTD   2.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-2.0.dtd">
<hibernate-mapping>
    <class name="br.ufsc.cate.hibernate.ProjetoFecam" table="projeto_fecam">
        <id name="id" column="id_Projeto" type="int">
            <generator class="assigned" />
        </id>
        <property name="nome" column="nome" type="string" not-null="true" />
        <set name="atividades" cascade="none" lazy="false">
            <key column="id_Projeto" />
            <one-to-many class="br.ufsc.cate.hibernate.Atividade" />
        </set>
    </class>
</hibernate-mapping>
```

12.1.30. Responsável.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE   hibernate-mapping   PUBLIC   "-//Hibernate/Hibernate   Mapping   DTD   2.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-2.0.dtd">
<hibernate-mapping>
    <class name="br.ufsc.cate.hibernate.Responsavel" table="responsavel">
        <id name="id" column="id_Responsavel" type="int">
            <generator class="assigned" />
        </id>
        <property name="nome" column="nome" type="string" not-null="true" />
        <set name="atividades" cascade="none" lazy="false">
            <key column="id_Responsavel" />
            <one-to-many class="br.ufsc.cate.hibernate.Atividade" />
        </set>
    </class>
</hibernate-mapping>
```

12.1.31. DateDecorator.java

```
package br.ufsc.cate.util;

import java.text.DateFormat;
```

```
import java.util.Date;
import java.util.Locale;

import org.displaytag.decorator.ColumnDecorator;
import org.displaytag.exception.DecoratorException;

/**
 * @author smendonca
 */
public class DateDecorator implements ColumnDecorator {

    /**
     * Metodo Sobrescrito
     */
    public String decorate(Object arg0) throws DecoratorException {
           if (arg0 instanceof Long) {
                  DateFormat df = DateFormat.getDateInstance(DateFormat.SHORT,
                                new Locale("pt_BR"));
                  return df.format(new Date(((Long) arg0).longValue()));
           } else {
                  throw new DecoratorException(this.getClass(),
                                "Não foi possível tranformar o valor '" + arg0 + "'");
           }
    }

}
```

12.1.32. HibernateUtil.java

```
package br.ufsc.cate.util;

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

import net.sf.hibernate.HibernateException;
import net.sf.hibernate.Session;
import net.sf.hibernate.SessionFactory;

import org.apache.log4j.Logger;

/**
 * @author Sergio
 *
 * TODO To change the template for this generated type comment go to Window -
 * Preferences - Java - Code Style - Code Templates
 */
public class HibernateUtil {
    private static String SESSION_JNDI_NAME = "java:/hibernate-cate-sessionfactory";

    private SessionFactory sessionFactory;

    private final ThreadLocal session = new ThreadLocal();

    private static HashMap sessions = new HashMap();

    public static List list(String query) throws Exception {
           Session session = null;
           List lista = new ArrayList();
           Exception ex = null;
           try {
                  session = currentSession();
```

```
                lista = session.find(query);
        } catch (Exception e) {
                ex = e;
        } finally {
                closeSession();
                if (ex != null) {
                        throw ex;
                }
        }
        return lista;
    }

    static {
        try {
                Logger.getLogger(HibernateUtil.class).debug(
                                "Inicialiando configuração do hibernate");

                HibernateUtil util = new HibernateUtil();
                util.sessionFactory = (SessionFactory) ServiceLocator.getInstance()
                                .getService(SESSION_JNDI_NAME);
                sessions.put(SESSION_JNDI_NAME, util);
        } catch (HibernateException e) {
                Logger.getLogger(HibernateUtil.class).error(e);
                throw new RuntimeException(
                                "Erro   ao   construir   sessão   do   Hibernate!   "   +
e.getMessage(),
                                e);
        } catch (Exception e) {
                Logger.getLogger(HibernateUtil.class).error(e);
                throw new RuntimeException(
                                "Erro   ao   construir   sessão   do   Hibernate!   "   +
e.getMessage(),
                                e);
        }
    }

    public static SessionFactory getSessionFactory() {
        return ((HibernateUtil) sessions.get(SESSION_JNDI_NAME)).sessionFactory;
    }

    /**
     * Busca a sessão aberta na ThreadLocal atual do processamento do DAO.
     */
    public static Session currentSession() throws HibernateException {
        Logger.getLogger(HibernateUtil.class).debug(
                        "Buscanco sessão hibernate na thread local.");
        try {
                return currentSession(SESSION_JNDI_NAME);
        } catch (HibernateException e) {
                throw e;
        } catch (Exception e) {
                throw new HibernateException(e);
        }
    }

    /**
     * Busca a sessão aberta na ThreadLocal atual do processamento do DAO. Esta
     * sessão é aperta apartir da Factory encontrada no JNDI passado como
     * parâmetro.
     *
     * @throws HibernateException,
     *              Exception
     */
    public static Session currentSession(String factoryJndiName)
                throws HibernateException {
```

```
        try {
                Logger.getLogger(HibernateUtil.class).debug(
                                "Buscanco sessão hibernate na thread local via JNDI.");

                HibernateUtil util = ((HibernateUtil) sessions.get(factoryJndiName));
                if (util == null) {
                        util = new HibernateUtil();
                        util.sessionFactory = (SessionFactory) ServiceLocator
                                        .getInstance().getService(factoryJndiName);
                        sessions.put(factoryJndiName, util);
                }

                Session s = (Session) util.session.get();
                if (s == null) {
                        s = util.sessionFactory.openSession();
                        util.session.set(s);
                }

                if (!s.isConnected()) {
                        s.reconnect();
                }
                return s;
        } catch (HibernateException e) {
                Logger.getLogger(HibernateUtil.class).error(e);
                throw e;
        } catch (Exception e) {
                Logger.getLogger(HibernateUtil.class).error(e);
                throw new HibernateException(e);
        }
}

/**
 * Fecha a sessão da ThreadLocal atual.
 */
public static void closeSession(String factoryJndiName)
                throws HibernateException {
        Logger.getLogger(HibernateUtil.class).debug(
                        "Fechando conexão sessão da thread local");

        HibernateUtil util = ((HibernateUtil) sessions.get(factoryJndiName));
        if (util == null) {
                return;
        }

        Session s = (Session) util.session.get();
        util.session.set(null);
        if (s != null) {
                try {
                        if (!s.connection().isClosed()) {
                                s.connection().close();
                        }
                } catch (Exception e) {
                        Logger.getLogger(HibernateUtil.class).error(e);
                }
                s.close();
        }
}

/**
 * Fecha a sessão da ThreadLocal atual.
 */
public static void closeSession() throws HibernateException {
        Logger.getLogger(HibernateUtil.class).debug(
                        "Fechando conexão sessão da thread local");
```

```
            try {
                    closeSession(SESSION_JNDI_NAME);
            } catch (HibernateException e) {
                    Logger.getLogger(HibernateUtil.class).error(e);
                    throw e;
            }
    }
}
```

12.1.33. ServiceLocator.java

```
package br.ufsc.cate.util;

import java.io.IOException;
import java.util.Collections;
import java.util.HashMap;
import java.util.Map;
import java.util.Properties;

import javax.ejb.EJBHome;
import javax.ejb.EJBLocalHome;
import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.rmi.PortableRemoteObject;
import javax.sql.DataSource;

/**
 * @author Sergio Ferreira de Mendonça
 * @since 16 Mar 2005
 * @version 0.1
 */
public class ServiceLocator {

    private InitialContext ic;

    private Map cache;

    private static ServiceLocator me;

    private static Properties jndiProperties = new Properties();

    private   final   static   String   SERVICE_LOCATOR_PROPERTIES   =   "service-locator-
jndi.properties";

    static {

            try {

                    me = new ServiceLocator();

            } catch (Exception se) {
                    System.err.println(se);
                    se.printStackTrace(System.err);
            }

    }

    private static InitialContext makeContext() throws NamingException {

            try {
                    jndiProperties.load(ServiceLocator.class
                                    .getResourceAsStream(SERVICE_LOCATOR_PROPERTIES));
```

```
        } catch (IOException e) {
                System.err.println(e);
                e.printStackTrace(System.err);
        }

        return new InitialContext(jndiProperties);

}

private ServiceLocator() throws Exception {

        try {

                ic = makeContext();
                cache = Collections.synchronizedMap(new HashMap());

        } catch (NamingException ne) {
                throw new Exception(ne);
        }

}

public static ServiceLocator getInstance() {
        return me;

}

public static void init() throws Exception {
        me = new ServiceLocator();
}

public EJBHome getRemoteHome(String jndiName, Class className)
                throws Exception {
        EJBHome home = null;

        try {

                if (cache.containsKey(jndiName)) {
                        home = (EJBHome) cache.get(jndiName);
                } else {
                        Object obj = ic.lookup(jndiName);
                        home = (EJBHome) PortableRemoteObject.narrow(obj, className);
                        cache.put(jndiName, home);
                }

        } catch (NamingException ne) {
                throw new Exception(ne);
        } catch (Exception e) {
                throw new Exception(e);
        }

        return home;

}

public EJBLocalHome getLocalHome(String jndiName, Class className)
                throws Exception {
        EJBLocalHome home = null;

        try {

                if (cache.containsKey(jndiName)) {
                        home = (EJBLocalHome) cache.get(jndiName);
                } else {
                        Object obj = ic.lookup(jndiName);
```

```
                        home = (EJBLocalHome) PortableRemoteObject.narrow(obj,
                                        className);
                        cache.put(jndiName, home);
                }

        } catch (NamingException ne) {
                throw new Exception(ne);
        } catch (Exception e) {
                throw new Exception(e);
        }

        return home;

}

public EJBLocalHome getLocalHome(String jndiName) throws Exception {

        EJBLocalHome home = null;

        try {

                if (cache.containsKey(jndiName)) {
                        home = (EJBLocalHome) cache.get(jndiName);
                } else {
                        home = (EJBLocalHome) ic.lookup(jndiName);
                        cache.put(jndiName, home);
                }

        } catch (NamingException ne) {
                throw new Exception(ne);
        } catch (Exception e) {
                throw new Exception(e);
        }

        return home;

}

public static Properties getJndiProperties() {
        return jndiProperties;
}

public DataSource getDataSource(String jndiName) throws Exception {
        DataSource ds = null;

        try {

                Context confCtx = (Context) ic.lookup("java:");

                ds = (DataSource) confCtx.lookup(jndiName);
        } catch (NamingException e) {
                throw new Exception(e);
        }

        return ds;
}

public Object getService(String jndiName) throws Exception {
        Object home = null;
        try {
                home = ic.lookup(jndiName);
        } catch (NamingException ne) {
                throw new Exception(ne);
        } catch (Exception e) {
                throw new Exception(e);
```

```
            }
            return home;
    }

}
```

12.1.34. StringArrayDecorator.java

```
package br.ufsc.cate.util;

import org.displaytag.decorator.ColumnDecorator;
import org.displaytag.exception.DecoratorException;

/**
 * @author smendonca
 */
public class StringArrayDecorator implements ColumnDecorator
{

   /**
    * Metodo Sobrescrito
    */
   public String decorate(Object obj) throws DecoratorException
   {
      if(obj instanceof String[])
      {
         String[] array = (String[])obj;
         String retorno = "";
         if(array.length > 0)
         {
            retorno = array[0];
            for(int i = 1; i < array.length; i++){
                   retorno = retorno.concat(", ").concat(array[i]);
            }
         }
         return retorno;
      }
      else
      {
         throw new DecoratorException(this.getClass(), "Não foi possível tranformar o
valor '" + obj + "'");
      }
   }

}
```

12.1.35. service-locator-jndi.properties

```
java.naming.factory.initial=org.jnp.interfaces.NamingContextFactory
java.naming.provider.url=localhost:1099
java.naming.factory.url.pkgs=org.jboss.naming
data.source=MySql
```

12.1.36. CateUploadFileAction.java

```
package br.ufsc.cate.web.action;

import java.io.File;
```

```
import java.io.FileOutputStream;
import java.io.InputStream;
import java.util.ArrayList;

import org.apache.struts.action.Action;
import org.apache.struts.upload.FormFile;

/**
 *
 * @author Sergio
 *
 * TODO To change the template for this generated type comment go to Window -
 * Preferences - Java - Code Style - Code Templates
 */
public class CateUploadFileAction extends Action {

    protected static final String TEMPORARY_DIR = "/temp";

    protected File[] getFiles(FormFile[] formFiles, String tempDir)
                    throws Exception {
            ArrayList files = new ArrayList();
            for (int i = 0; i < formFiles.length; i++) {
                    files.add(this.getFile(formFiles[i], tempDir));
            }
            return (File[]) files.toArray(new File[] {});
    }

    protected File getFile(FormFile formFile, String tempDir) throws Exception {
            InputStream input = formFile.getInputStream();

            System.out.println("uploadeando arquivo: " + formFile.getFileName());
            if (formFile.getFileName().equals("")) {
                    throw new Exception("Arquivo inválido.");
            }

            if (tempDir == null) {
                    tempDir = TEMPORARY_DIR;
            }
            File temporaryDir = new File(tempDir);
            if (!temporaryDir.exists()) {
                    temporaryDir.mkdirs();
            }
            File file = new File(temporaryDir, formFile.getFileName());
            FileOutputStream output = new FileOutputStream(file);

            int b;
            while ((b = input.read()) >= 0) {
                    output.write(b);
            }
            input.close();
            output.close();
            System.out.println("arquivo uploadeado com sucesso para: "
                            + file.getPath());
            return file;
    }
}
```

12.1.37. CadastrarAtividadeExternaAction.java

```
package br.ufsc.cate.web.action;

import java.io.File;
import java.util.HashMap;
```

```
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.action.DynaActionForm;
import org.apache.struts.upload.FormFile;

import br.ufsc.cate.delegate.BusinessDelegate;

/**
 * Action para cadastrar um atividade externa
 *
 * @author Sergio
 *
 * TODO To change the template for this generated type comment go to Window -
 * Preferences - Java - Code Style - Code Templates
 */
public class CadastrarAtividadeExternaAction extends CateUploadFileAction {
    protected static final String TEMPORARY_DIR = "/temp/pautas";

    public ActionForward execute(ActionMapping mapping, ActionForm form,
                HttpServletRequest request, HttpServletResponse response)
                throws Exception {
            DynaActionForm dynaForm = (DynaActionForm) form;
            HashMap parametros = new HashMap(dynaForm.getMap());

            File file = this.getFile((FormFile) dynaForm.get("arquivoPauta"),
                        TEMPORARY_DIR);
            parametros.put("arquivoPauta", file);

            BusinessDelegate buss = BusinessDelegate.getInstance();
            buss.cadastrarDocumento(parametros);

            ActionMessages messages = new ActionMessages();
            messages
                        .add("sucesso",                                                 new
ActionMessage("cadastro.documento.sucesso"));
            addMessages(request, messages);
            return mapping.findForward("sucesso");
    }
}
```

12.1.38. CadastrarDocumentoAction.java

```
package br.ufsc.cate.web.action;

import java.io.File;
import java.util.HashMap;
import java.util.Iterator;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
```

```
import org.apache.struts.action.ActionMessages;
import org.apache.struts.action.DynaActionForm;
import org.apache.struts.upload.FormFile;

import br.ufsc.cate.delegate.BusinessDelegate;

/**
 * Action para cadastrar um documento
 *
 * @author Sergio
 *
 * TODO To change the template for this generated type comment go to Window -
 * Preferences - Java - Code Style - Code Templates
 */
public class CadastrarDocumentoAction extends CateUploadFileAction {
    private static final String CADASTRAR_ACTION = "cadastrar";

    protected static final String TEMPORARY_DIR = "/temp/docs";

    public ActionForward execute(ActionMapping mapping, ActionForm form,
                HttpServletRequest request, HttpServletResponse response)
                throws Exception {
            DynaActionForm dynaForm = (DynaActionForm) form;

            ActionForward forward = null;
            if (CADASTRAR_ACTION.equals(dynaForm.get("action"))) {
                    forward = doCadastrarAction(mapping, form, request, response);
            } else {
                    forward = mapping.findForward("default");
            }

            dynaForm.reset(mapping, request);

            this.setarCollections(request);

            return forward;
    }

    /**
     * @param request
     */
    private void setarCollections(HttpServletRequest request) {
            BusinessDelegate del = BusinessDelegate.getInstance();
            request.setAttribute("modalidades", del.getModalidades());
            request.setAttribute("destinatarios", del.getDestinatarios());
            request.setAttribute("autores", del.getAutores());
            request.setAttribute("assinantes", del.getAssinantes());
            request.setAttribute("formasEnvio", del.getFormasEnvio());
    }

    private ActionForward doCadastrarAction(ActionMapping mapping,
                ActionForm form, HttpServletRequest request,
                HttpServletResponse response) throws Exception {
            try {
                DynaActionForm dynaForm = (DynaActionForm) form;
                HashMap parametros = new HashMap(dynaForm.getMap());

                File file = super.getFile((FormFile) dynaForm.get("arquivo"),
                            TEMPORARY_DIR);

                parametros.put("arquivo", file);

                BusinessDelegate.getInstance().cadastrarDocumento(parametros);

                this.zerarForm(dynaForm);
```

```
                ActionMessage m = new ActionMessage("cadastro.documento.sucesso");
                ActionMessages mes = new ActionMessages();
                mes.add("sucesso", m);
                saveErrors(request, mes);
                return mapping.findForward("sucesso");
        } catch (Exception e) {
                e.printStackTrace();
                ActionMessage m = new ActionMessage("cadastro.documento.erro", e
                                .getLocalizedMessage());
                ActionMessages mes = new ActionMessages();
                mes.add("erro", m);
                saveErrors(request, mes);
                return mapping.findForward("erro");
        }

    }

    private void zerarForm(DynaActionForm form) {
        Iterator keys = form.getMap().keySet().iterator();
        while (keys.hasNext()) {
                String key = (String) keys.next();
                Object prop = form.get(key);
                if (prop.getClass().equals(String.class)) {
                        form.set(key, "");
                } else if (prop.getClass().equals(String[].class)) {
                        form.set(key, new String[] {});
                }
        }
    }
}
```

12.1.39. DownloadDocumentoAction.java

```
/*
 * Created on 18/04/2005
 *
 * TODO To change the template for this generated file go to
 * Window - Preferences - Java - Code Style - Code Templates
 */
package br.ufsc.cate.web.action;

import java.io.File;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.actions.DownloadAction;

import br.ufsc.cate.business.documento.DocumentoUtil;

/**
 * @author Sergio
 *
 * TODO To change the template for this generated type comment go to Window -
 * Preferences - Java - Code Style - Code Templates
 */
public class DownloadDocumentoAction extends DownloadAction {
```

```
    protected StreamInfo getStreamInfo(ActionMapping mapping, ActionForm form,
                    HttpServletRequest request, HttpServletResponse response)
                    throws Exception {

            String nomeArq = request.getParameter("nomeArq");
            String contentType = "application/msword";
            DocumentoUtil du = DocumentoUtil.getInstance();
            File file = new File(du.getCaminhoCompletoArquivo(nomeArq));

            return new FileStreamInfo(contentType, file);
    }

    public ActionForward execute(ActionMapping mapping, ActionForm form,
                    HttpServletRequest request, HttpServletResponse response)
                    throws Exception {
            String[] nome = request.getParameter("nomeArq").split("/");
            response.setHeader("Content-disposition", "attachment; filename="
                            + nome[nome.length - 1]);

            return super.execute(mapping, form, request, response);

    }
}
```

12.1.40. PesquisarDocumentosAction.java

```
package br.ufsc.cate.web.action;

import java.util.HashMap;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.action.DynaActionForm;

import br.ufsc.cate.delegate.BusinessDelegate;

/**
 * Action para pesquisar um documento
 *
 * @author Sergio
 *
 * TODO To change the template for this generated type comment go to Window -
 * Preferences - Java - Code Style - Code Templates
 */
public class PesquisarDocumentosAction extends Action {
    private static final String PESQUISAR_ACTION = "pesquisar";

    public ActionForward execute(ActionMapping mapping, ActionForm form,
                    HttpServletRequest request, HttpServletResponse response)
                    throws Exception {
            DynaActionForm dynaForm = (DynaActionForm) form;

            ActionForward forward = null;
            if (PESQUISAR_ACTION.equals(dynaForm.get("action"))) {
                    forward = this.doPesquisarAction(mapping, form, request, response);
```

```
            } else {
                  forward = mapping.findForward("default");
            }
            this.setarCollections(request);

            return forward;
      }

      private void setarCollections(HttpServletRequest request) {
            BusinessDelegate del = BusinessDelegate.getInstance();
            request.setAttribute("modalidades", del.getModalidades());
      }

      private ActionForward doPesquisarAction(ActionMapping mapping,
                  ActionForm form, HttpServletRequest request,
                  HttpServletResponse response) throws Exception {
            try {
                  DynaActionForm dynaForm = (DynaActionForm) form;
                  HashMap parametros = new HashMap(dynaForm.getMap());
                  request.setAttribute("documentos", BusinessDelegate.getInstance()
                              .filtrarDocumentos(parametros));
                  return mapping.findForward("sucesso");
            } catch (Exception e) {
                  e.printStackTrace();
                  ActionMessage m = new ActionMessage("pesquisa.documento.erro", e
                              .getLocalizedMessage());
                  ActionMessages mes = new ActionMessages();
                  mes.add("erro", m);
                  saveErrors(request, mes);
                  return mapping.findForward("erro");
            }
      }
}
```

12.1.41. Mensagens.properties

```
cadastro.documento.sucesso=<tr><td         height="1pt"         style="{background-color:
#244F83}"></td></tr><tr><td    height="10"    class="msg"    ></tr><tr><td    class="msg"
align="center">Documento  cadastrado  com  sucesso.       </td></tr><tr><td  height="10"
class="msg" ></tr>
cadastro.documento.erro=            <tr><td    height="1pt"    style="{background-color:
#244F83}"></td></tr><tr><td    height="10"    class="erro"></tr><tr><td    class="erro"
align="center">Erro  ao  cadastrar  o  documento:  {0}    </td></tr><tr><td  height="10"
class="erro"></tr>
pesquisa.documento.erro=            <tr><td    height="1pt"    style="{background-color:
#244F83}"></td></tr><tr><td    height="10"    class="erro"></tr><tr><td    class="erro"
align="center">Erro  ao  pesquisar  os  documentos:  {0}</td></tr><tr><td  height="10"
class="erro"></tr>

cadastrarDocumentoForm.assunto=Assunto
cadastrarDocumentoForm.dataElaboracao=Data de Elaboração
cadastrarDocumentoForm.dataEnvio=Data de Envio
cadastrarDocumentoForm.arquivo=Arquivo
cadastrarDocumentoForm.destinatario=Destinatarios

cadastrarAtividadeExternaForm.responsavel=Responsável
cadastrarAtividadeExternaForm.apoiador=Apoiadores
cadastrarAtividadeExternaForm.titulo=Título
cadastrarAtividadeExternaForm.modalidade=Modalidade
cadastrarAtividadeExternaForm.local=Local
cadastrarAtividadeExternaForm.data=Data
cadastrarAtividadeExternaForm.horario=Horário
```

```
cadastrarAtividadeExternaForm.duracao=Duração
cadastrarAtividadeExternaForm.projeto=Projeto
cadastrarAtividadeExternaForm.pauta=Pauta
cadastrarAtividadeExternaForm.arquivoPauta=Arquivo de Pauta

cadastrarAtividadeInternaForm.assunto=Assunto
cadastrarAtividadeInternaForm.dataElaboracao=Data de Elaboração
cadastrarAtividadeInternaForm.dataEnvio=Data de Envio
cadastrarAtividadeInternaForm.arquivo=Arquivo
cadastrarAtividadeInternaForm.destinatario=Destinatarios

pesquisarDocumentosForm.modalidades=Modalidades


###### mensagens pro validator ############
errors.required=O campo {0} \u00E9 requerido.
errors.minlength=O campo {0} n\u00E3o pode ter menos de {1} caracteres.
errors.maxlength=O campo {0} n\u00E3o pode ter mais de {1} caracteres.
errors.invalid=O campo {0} \u00E9 inv\u00E1lido.
errors.byte=O campo {0} deve ser um byte.
errors.short=O campo {0} deve ser um short.
errors.integer=O campo {0} deve ser um n\u00FAmero inteiro.
errors.long=O campo {0} deve ser um n\u00FAmero inteiro.
errors.float=O campo {0} deve ser um float.
errors.double=O campo {0} deve ser um double.
errors.date=O campo {0} n\u00E3o \u00E9 uma data v\u00E1lida. A data deve ser informada
no formato DD/MM/AAAA.
errors.range=O campo {0} is not in the range {1} through {2}.
errors.creditcard=O campo {0} is an invalid credit card number.
errors.email=O campo {0} n\u00E3o \u00E9 um endere\u00E7o de e-mail v\u00E1lido.
```

12.1.42. ejb-jar.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ejb-jar PUBLIC "-//Sun Microsystems, Inc.//DTD Enterprise JavaBeans 2.0//EN"
"http://java.sun.com/dtd/ejb-jar_2_0.dtd">

<ejb-jar >

   <description><![CDATA[No Description.]]></description>
   <display-name>Generated by XDoclet</display-name>

   <enterprise-beans>

      <!-- Session Beans -->
      <session >
         <description><![CDATA[]]></description>
         <display-name>DocumentosBusiness</display-name>

         <ejb-name>cate/DocumentosBusiness</ejb-name>

         <home>br.ufsc.cate.business.documento.DocumentosBusinessHome</home>
         <remote>br.ufsc.cate.business.documento.DocumentosBusiness</remote>
         <local-home>br.ufsc.cate.business.documento.DocumentosBusinessLocalHome</local-
home>
         <local>br.ufsc.cate.business.documento.DocumentosBusinessLocal</local>
         <ejb-class>br.ufsc.cate.business.documento.DocumentosBusinessBean</ejb-class>
         <session-type>Stateless</session-type>
         <transaction-type>Container</transaction-type>

      </session>
```

```
     <!--
       To add session beans that you have deployment descriptor info for, add
       a file to your XDoclet merge directory called session-beans.xml that contains
       the <session></session> markup for those beans.
     -->

      <!-- Entity Beans -->
     <!--
       To add entity beans that you have deployment descriptor info for, add
       a file to your XDoclet merge directory called entity-beans.xml that contains
       the <entity></entity> markup for those beans.
     -->

      <!-- Message Driven Beans -->
     <!--
       To add message driven beans that you have deployment descriptor info for, add
       a file to your XDoclet merge directory called message-driven-beans.xml that
contains
       the <message-driven></message-driven> markup for those beans.
     -->

   </enterprise-beans>

   <!-- Relationships -->

   <!-- Assembly Descriptor -->
   <assembly-descriptor >
     <!--
       To add additional assembly descriptor info here, add a file to your
       XDoclet merge directory called assembly-descriptor.xml that contains
       the <assembly-descriptor></assembly-descriptor> markup.
     -->

   <!-- finder permissions -->

   <!-- transactions -->
   <container-transaction >
      <method >
         <ejb-name>cate/DocumentosBusiness</ejb-name>
          <method-name>*</method-name>
       </method>
       <trans-attribute>Supports</trans-attribute>
    </container-transaction>

   <!-- finder transactions -->
   </assembly-descriptor>

</ejb-jar>
```

12.1.43. jboss.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE        jboss        PUBLIC        "-//JBoss//DTD        JBOSS        3.0//EN"
"http://www.jboss.org/j2ee/dtd/jboss_3_0.dtd">

<jboss>
```

```
   <enterprise-beans>

     <!--
       To add beans that you have deployment descriptor info for, add
       a file to your XDoclet merge directory called jboss-beans.xml that contains
       the <session></session>, <entity></entity> and <message-driven></message-driven>
       markup for those beans.
     -->

      <session>
         <ejb-name>cate/DocumentosBusiness</ejb-name>
         <jndi-name>cate/DocumentosBusiness</jndi-name>
         <local-jndi-name>cate/DocumentosBusinessLocal</local-jndi-name>

      </session>

   </enterprise-beans>

   <resource-managers>
   </resource-managers>

</jboss>
```

12.1.44. jboss-service.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- $Id: jboss-service.xml,v 1.3 2005/06/01 07:13:54 sfermend Exp $ -->

<server>
     <!-- Config da Factory do CRUD -->
     <mbean                                  code="net.sf.hibernate.jmx.HibernateService"
name="jboss.jca:service=HibernateFactory, name=HibernateFactory">
       <depends>jboss.jca:service=RARDeployer</depends>
       <depends>jboss.jca:service=DataSourceBinding,name=xalala</depends>
            <attribute                                                  name="MapResources">
br/ufsc/cate/hibernate/Apoiador.hbm.xml,br/ufsc/cate/hibernate/Assinante.hbm.xml,br/ufsc/
cate/hibernate/Atividade.hbm.xml,br/ufsc/cate/hibernate/Autor.hbm.xml,br/ufsc/cate/hibern
ate/ConselhoRepresentacao.hbm.xml,br/ufsc/cate/hibernate/Destinatario.hbm.xml,br/ufsc/cat
e/hibernate/Documento.hbm.xml,br/ufsc/cate/hibernate/EspecieAtividade.hbm.xml,br/ufsc/cat
e/hibernate/FormaEnvio.hbm.xml,br/ufsc/cate/hibernate/Gestao.hbm.xml,br/ufsc/cate/hiberna
te/Modalidade.hbm.xml,br/ufsc/cate/hibernate/ProjetoFecam.hbm.xml,br/ufsc/cate/hibernate/
Responsavel.hbm.xml</attribute>
            <attribute name="JndiName">java:/hibernate-cate-sessionfactory</attribute>
            <attribute name="Datasource">java:xalala</attribute>
            <attribute name="Dialect">net.sf.hibernate.dialect.MySQLDialect</attribute>
            <attribute name="ShowSql">true</attribute>
     </mbean>
</server>
```

12.1.45. struts-config.xml

```
<!DOCTYPE struts-config PUBLIC "-//Apache Software Foundation//DTD Struts Configuration
1.2//EN" "http://jakarta.apache.org/struts/dtds/struts-config_1_2.dtd">
<struts-config>
     <display-name>Cate</display-name>
     <description>Cate</description>
     <data-sources />
```

```
<!-- ============================================ Form Bean Definitions -->

    <form-beans>
        <form-bean                                          name="cadastrarDocumentoForm"
type="org.apache.struts.validator.DynaValidatorActionForm">
            <form-property name="action" type="java.lang.String" />
            <form-property name="modalidade" type="java.lang.String" />
            <form-property name="autor" type="java.lang.String" />
            <form-property name="assinante" type="java.lang.String" />
            <form-property name="destinatario" type="java.lang.String[]" />
            <form-property name="dataElaboracao" type="java.lang.String" />
            <form-property name="dataEnvio" type="java.lang.String" />
            <form-property name="assunto" type="java.lang.String" />
            <form-property name="formaEnvio" type="java.lang.String" />
            <form-property name="arquivo" type="org.apache.struts.upload.FormFile" />
            </form-bean>
        <form-bean                                      name="cadastrarAtividadeExternaForm"
type="org.apache.struts.validator.DynaValidatorActionForm">
            <form-property name="responsavel" type="java.lang.String" />
            <form-property name="apoiador" type="java.lang.String[]" />
            <form-property name="titulo" type="java.lang.String" />
            <form-property name="modalidade" type="java.lang.String" />
            <form-property name="local" type="java.lang.String" />
            <form-property name="data" type="java.lang.String" />
            <form-property name="horario" type="java.lang.String" />
            <form-property name="duracao" type="java.lang.String" />
            <form-property name="pauta" type="java.lang.String" />
            <form-property  name="arquivoPauta"  type="org.apache.struts.upload.FormFile"
/>
            <form-property name="projeto" type="java.lang.String" />
            <form-property name="conselho" type="java.lang.String" />
            </form-bean>
        <form-bean                                      name="cadastrarAtividadeInternaForm"
type="org.apache.struts.validator.DynaValidatorActionForm">
            <form-property name="responsavel" type="java.lang.String" />
            <form-property name="apoiador" type="java.lang.String[]" />
            <form-property name="titulo" type="java.lang.String" />
            <form-property name="modalidade" type="java.lang.String" />
            <form-property name="local" type="java.lang.String" />
            <form-property name="data" type="java.lang.String" />
            <form-property name="horario" type="java.lang.String" />
            <form-property name="duracao" type="java.lang.String" />
            <form-property name="resumo" type="java.lang.String" />
            <form-property                                          name="arquivoResumo_0"
type="org.apache.struts.upload.FormFile" />
            <form-property                                          name="arquivoResumo_1"
type="org.apache.struts.upload.FormFile" />
            <form-property                                          name="arquivoResumo_2"
type="org.apache.struts.upload.FormFile" />
            <form-property                                          name="arquivoResumo_3"
type="org.apache.struts.upload.FormFile" />
            <form-property                                          name="arquivoResumo_4"
type="org.apache.struts.upload.FormFile" />
            <form-property                                          name="arquivoResumo_5"
type="org.apache.struts.upload.FormFile" />
            <form-property                                          name="arquivoResumo_6"
type="org.apache.struts.upload.FormFile" />
            <form-property                                          name="arquivoResumo_7"
type="org.apache.struts.upload.FormFile" />
            <form-property                                          name="arquivoResumo_8"
type="org.apache.struts.upload.FormFile" />
            <form-property                                          name="arquivoResumo_9"
type="org.apache.struts.upload.FormFile" />
            </form-bean>
```

```
            <form-bean                                          name="pesquisarDocumentosForm"
type="org.apache.struts.validator.DynaValidatorActionForm">
            <form-property name="action" type="java.lang.String" />
             <form-property name="modalidades" type="java.lang.String[]" />
            </form-bean>
    </form-beans>


<!-- ========================================== Global Exception Definitions -->

    <global-exceptions>
    </global-exceptions>


<!-- ============================================ Global Forward Definitions -->
     <global-forwards>
            <forward name="error" path="/error.do" redirect="true" contextRelative="true"
/>
            <forward       name="cadastro.documento"       path="/cadastrarDocumento.do"
redirect="true" contextRelative="true" />
            <forward      name="pesquisa.documentos"      path="/pesquisarDocumentos.do"
redirect="true" contextRelative="true" />
            <forward                                        name="cadastro.atividade.externa"
path="/cadastroAtividadeExterna.do" redirect="true" contextRelative="true" />
            <forward                                        name="cadastro.atividade.interna"
path="/cadastroAtividadeInterna.do" redirect="true" contextRelative="true" />
     </global-forwards>
<!-- ============================================ Action Mapping Definitions -->

    <action-mappings>
    <!-- forwards para as difinições do tiles -->
    <action         path="/index"         type="org.apache.struts.actions.ForwardAction"
parameter="tiles.cadastro.documento" />
    <action         path="/error"         type="org.apache.struts.actions.ForwardAction"
parameter="tiles.error" />
        <action                                          path="/cadastroAtividadeExterna"
type="org.apache.struts.actions.ForwardAction"
parameter="tiles.cadastro.atividade.externa" />
        <action                                          path="/cadastroAtividadeInterna"
type="org.apache.struts.actions.ForwardAction"
parameter="tiles.cadastro.atividade.interna" />
        <action                                                 path="/downloadDocumento"
type="br.ufsc.cate.web.action.DownloadDocumentoAction"/>

        <!-- actions normais -->
         <action         path="/cadastrarDocumento"         name="cadastrarDocumentoForm"
attribute="cadastrarDocumentoForm"                          input="/cadastroDocumento.do"
type="br.ufsc.cate.web.action.CadastrarDocumentoAction" scope="request" validate="true">
            <forward      name="sucesso"      redirect="true"      contextRelative="true"
path="tiles.cadastro.documento" />
            <forward      name="default"      redirect="true"      contextRelative="true"
path="tiles.cadastro.documento" />
            <forward       name="erro"       redirect="true"       contextRelative="true"
path="tiles.cadastro.documento" />
        </action>
        <action  path="/cadastrarAtividadeExterna"  name="cadastrarAtividadeExternaForm"
attribute="cadastrarAtividadeExternaForm"            input="/cadastroAtividadeExterna.do"
type="br.ufsc.cate.web.action.CadastrarAtividadeExternaAction"            scope="request"
validate="true">
            <forward      name="sucesso"      redirect="true"      contextRelative="true"
path="/cadastroAtividadeExterna.do?sucesso=true" />
        </action>
        <action  path="/cadastrarAtividadeInterna"  name="cadastrarAtividadeInternaForm"
attribute="cadastrarAtividadeInternaForm"            input="/cadastroAtividadeInterna.do"
```

```
type="br.ufsc.cate.web.action.CadastrarAtividadeInternaAction"            scope="request"
validate="true">
            <forward      name="sucesso"      redirect="true"      contextRelative="true"
path="/cadastroAtividadeInterna.do?sucesso=true" />
        </action>
        <action        path="/pesquisarDocumentos"        name="pesquisarDocumentosForm"
attribute="pesquisarDocumentosForm"                          input="/pesquisaDocumentos.do"
type="br.ufsc.cate.web.action.PesquisarDocumentosAction" scope="request" validate="true">
            <forward      name="sucesso"      redirect="true"      contextRelative="true"
path="tiles.pesquisa.documentos" />
            <forward      name="default"      redirect="true"      contextRelative="true"
path="tiles.pesquisa.documentos" />
            <forward       name="erro"       redirect="true"       contextRelative="true"
path="tiles.pesquisa.documentos" />
        </action>
    </action-mappings>


<!-- ========================================== Controller Configuration -->

    <controller processorClass="org.apache.struts.tiles.TilesRequestProcessor" />


<!-- ======================================== Message Resources Definitions -->

    <message-resources parameter="br.ufsc.cate.web..properties.Mensagens" />


<!-- ============================================== Plug Ins Configuration -->

  <!-- ======================================================= Tiles plugin -->

    <plug-in className="org.apache.struts.tiles.TilesPlugin">

      <!-- Path to XML definition file -->
      <set-property property="definitions-config" value="/WEB-INF/conf/tiles-defs.xml" />
      <!-- Set Module-awareness to true -->
      <set-property property="moduleAware" value="true" />
    </plug-in>


  <!-- ======================================================= Validator plugin -->

  <plug-in className="org.apache.struts.validator.ValidatorPlugIn">
    <set-property   property="pathnames"   value="/WEB-INF/conf/validator-rules.xml,/WEB-
INF/conf/validation.xml" />
  </plug-in>


</struts-config>
```

12.1.46. tiles-defs.xml

```
<!DOCTYPE   tiles-definitions   PUBLIC   "-//Apache   Software   Foundation//DTD   Tiles
Configuration 1.1//EN" "http://jakarta.apache.org/struts/dtds/tiles-config_1_1.dtd">
<tiles-definitions>
    <definition name="tiles.principal" path="/layouts/principal.jsp">
        <put name="css" type="string" value="css/estilo.css" />
        <put name="titulo" type="string" value="..:: Cate - Fecam ::.." />
        <put name="cabecalho" value="/pages/cabecalho.jsp" />
        <put name="menu" value="/pages/menu.jsp" />
```

```
        <put name="titulo.pagina" type="string" value="${titulo.pagina}" />
        <put name="corpo" value="${corpo}" />
        <put name="rodape" value="/pages/rodape.jsp" />
    </definition>
    <definition extends="tiles.principal" name="tiles.cadastro.documento">
        <put name="corpo" value="/pages/documento/cadastroDocumento.jsp" />
        <put name="titulo.pagina" type="string" value="Cadastro de Documento" />
    </definition>
    <definition extends="tiles.principal" name="tiles.pesquisa.documentos">
        <put name="corpo" value="/pages/documento/pesquisaDocumentos.jsp" />
        <put name="titulo.pagina" type="string" value="Pesquisa de Documentos" />
    </definition>
    <definition name="tiles.error" extends="tiles.principal">
           <put name="corpo" value="/pages/erro.jsp" />
           <put name="titulo.pagina" value="Página de Erro" type="string" />
    </definition>
    <definition extends="tiles.principal" name="tiles.cadastro.atividade.externa">
        <put  name="corpo"  value="/pages/atividadeExterna/cadastroAtividadeExterna.jsp"
/>
        <put name="titulo.pagina" type="string" value="Cadastro de Atividade Externa" />
    </definition>
    <definition extends="tiles.principal" name="tiles.cadastro.atividade.interna">
        <put  name="corpo"  value="/pages/atividadeInterna/cadastroAtividadeInterna.jsp"
/>
        <put name="titulo.pagina" type="string" value="Cadastro de Atividade Interna" />
    </definition>
</tiles-definitions>
```

12.1.47. validation.xml

```
<?xml version="1.0" encoding="ISO-8859-1" ?>

<!DOCTYPE form-validation PUBLIC
          "-//Apache  Software  Foundation//DTD  Commons  Validator  Rules  Configuration
1.1.3//EN"
          "http://jakarta.apache.org/commons/dtds/validator_1_1_3.dtd">

<form-validation>

<!--
     This is a minimal Validator form file with a couple of examples.
-->

    <global>
    </global>

    <formset>
        <form name="cadastrarDocumentoForm">
            <field property="assunto" depends="required">
                   <arg0 key="cadastrarDocumentoForm.assunto"/>
            </field>
            <field property="destinatario" depends="required">
                   <arg0 key="cadastrarDocumentoForm.destinatario"/>
            </field>
            <field property="dataElaboracao" depends="required,date">
                    <arg0 key="cadastrarDocumentoForm.dataElaboracao"/>
                    <var><var-name>datePattern</var-name><var-value>dd/MM/yyyy</var-
value></var>
                </field>

                <field property="dataEnvio" depends="required,date">
```

```
                    <arg0 key="cadastrarDocumentoForm.dataEnvio"/>
                    <var><var-name>datePattern</var-name><var-value>dd/MM/yyyy</var-
value></var>
                </field>

                <field property="arquivo" depends="required">
                    <arg0 key="cadastrarDocumentoForm.arquivo"/>
            </field>
        </form>
        <!-- ############################## -->
        <form name="pesquisarDocumentosForm">
            <field property="modalidades" depends="required">
                    <arg0 key="pesquisarDocumentosForm.modalidades"/>
            </field>
        </form>
        <!-- ############################## -->
            <form name="cadastrarAtividadeExternaForm">
            <field property="responsavel" depends="required">
                    <arg0 key="cadastrarAtividadeExternaForm.responsavel"/>
            </field>
            <field property="apoiador" depends="required">
                    <arg0 key="cadastrarAtividadeExternaForm.apoiador"/>
            </field>
            <field property="titulo" depends="required">
                    <arg0 key="cadastrarAtividadeExternaForm.titulo"/>
            </field>
            <field property="modalidade" depends="required">
                    <arg0 key="cadastrarAtividadeExternaForm.modalidade"/>
                </field>
                <field property="local" depends="required">
                    <arg0 key="cadastrarAtividadeExternaForm.local"/>
            </field>
                <field property="data" depends="required,date">
                    <arg0 key="cadastrarAtividadeExternaForm.data"/>
                    <var><var-name>datePattern</var-name><var-value>dd/MM/yyyy</var-
value></var>
                </field>
                <field property="horario" depends="required,date">
                    <arg0 key="cadastrarAtividadeExternaForm.horario"/>
                    <var><var-name>datePattern</var-name><var-value>hh:mm</var-
value></var>
                </field>
                <field property="duracao" depends="required">
                    <arg0 key="cadastrarAtividadeExternaForm.duracao"/>
            </field>
                    <field property="projeto" depends="required">
                    <arg0 key="cadastrarAtividadeExternaForm.projeto"/>
            </field>
            <field property="pauta" depends="required">
                    <arg0 key="cadastrarAtividadeExternaForm.pauta"/>
            </field>
                <field property="arquivoPauta" depends="required">
                    <arg0 key="cadastrarAtividadeExternaForm.arquivoPauta"/>
            </field>
        </form>
        <!-- ############################## -->
        <form name="cadastrarAtividadeInternaForm">
            <field property="assunto" depends="required">
                    <arg0 key="cadastrarAtividadeInternaForm.assunto"/>
            </field>
            <field property="destinatario" depends="required">
                    <arg0 key="cadastrarAtividadeInternaForm.destinatario"/>
            </field>
            <field property="dataElaboracao" depends="required,date">
                    <arg0 key="cadastrarAtividadeInternaForm.dataElaboracao"/>
```

```
                    <var><var-name>datePattern</var-name><var-value>dd/MM/yyyy</var-
value></var>
                </field>

                <field property="dataEnvio" depends="required,date">
                    <arg0 key="cadastrarAtividadeInternaForm.dataEnvio"/>
                    <var><var-name>datePattern</var-name><var-value>dd/MM/yyyy</var-
value></var>
                </field>

                <field property="arquivo" depends="required">
                   <arg0 key="cadastrarAtividadeInternaForm.arquivo"/>
            </field>
        </form>
    </formset>
</form-validation>
```

12.1.48. web.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!DOCTYPE web-app
  PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.2//EN"
  "http://java.sun.com/j2ee/dtds/web-app_2_2.dtd">

<web-app>
  <display-name>Struts Blank Application</display-name>

  <!-- Standard Action Servlet Configuration (with debugging) -->
  <servlet>
    <servlet-name>action</servlet-name>
    <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
    <init-param>
      <param-name>config</param-name>
      <param-value>/WEB-INF/conf/struts-config.xml</param-value>
    </init-param>
    <init-param>
      <param-name>debug</param-name>
      <param-value>2</param-value>
    </init-param>
    <init-param>
      <param-name>detail</param-name>
      <param-value>2</param-value>
    </init-param>
    <load-on-startup>2</load-on-startup>
  </servlet>


  <!-- Standard Action Servlet Mapping -->
  <servlet-mapping>
    <servlet-name>action</servlet-name>
    <url-pattern>*.do</url-pattern>
  </servlet-mapping>


  <!-- The Usual Welcome File List -->
  <welcome-file-list>
    <welcome-file>index.do</welcome-file>
  </welcome-file-list>
```

```
  <!-- Struts Tag Library Descriptors -->
  <taglib>
    <taglib-uri>/tags/struts-bean</taglib-uri>
    <taglib-location>/WEB-INF/tlds/struts-bean.tld</taglib-location>
  </taglib>

  <taglib>
    <taglib-uri>/tags/struts-html</taglib-uri>
    <taglib-location>/WEB-INF/tlds/struts-html.tld</taglib-location>
  </taglib>

  <taglib>
    <taglib-uri>/tags/struts-logic</taglib-uri>
    <taglib-location>/WEB-INF/tlds/struts-logic.tld</taglib-location>
  </taglib>

  <taglib>
    <taglib-uri>/tags/struts-nested</taglib-uri>
    <taglib-location>/WEB-INF/tlds/struts-nested.tld</taglib-location>
  </taglib>

  <taglib>
    <taglib-uri>/tags/struts-tiles</taglib-uri>
    <taglib-location>/WEB-INF/tlds/struts-tiles.tld</taglib-location>
  </taglib>

</web-app>
```

12.1.49. principal.jsp

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN" >
<%@taglib uri="/WEB-INF/tlds/struts-tiles.tld" prefix="tiles" %>
<html>
    <head>
           <META HTTP-EQUIV="Pragma" CONTENT="no-cache" />
           <META HTTP-EQUIV="Cache-Control" CONTENT="no-cache" />
           <META HTTP-EQUIV="Expires" CONTENT="0" />
           <META HTTP-EQUIV="CONTENT-TYPE" CONTENT="text/html; charset=iso-8859-1" />
           <META HTTP-EQUIV="Content-Style-Type" CONTENT="text/css" />
           <link     href="<tiles:getAsString     name="css"     />"     rel="stylesheet"
type="text/css" />
           <title><tiles:getAsString name="titulo" /></title>
    </head>
    <body topmargin="0" marginwidth="0" leftmargin="0" marginheight="0">
           <table cellspacing="0" cellpadding="0" width="100%" border="0">
                  <tr>
                         <td colspan="2"><tiles:insert attribute="cabecalho" /></td>
                  </tr>
                  <tr><td height="1pt" bgcolor="#244F83" colspan="2"></td></tr>
                  <tr>
                         <td width="160" valign="top" bgcolor="#244F83" align="left">
                                <tiles:insert attribute="menu" />
                         </td>
                         <td>
                                <table cellspacing="0" cellpadding="0" width="100%">
                                       <tr><td height="10" /></tr>
                                       <tr><td                          align="center"
class="titulo"><tiles:getAsString name="titulo.pagina" /></td></tr>
                                       <tr><td height="10" /></tr>
                                       <tr><td            align="center"><tiles:insert
attribute="corpo" /></td></tr>
```

```
                                        <tr><td height="10" /></tr>
                                    </table>
                            </td>
                    </tr>
                    <tr>
                            <td  colspan="2"  align="left"><tiles:insert  attribute="rodape"
/></td>
                    </tr>
            </table>
    </body>
</html>
```

12.1.50. cadastroAtividadeExterna.jsp

```
<%@ taglib uri="/WEB-INF/tlds/struts-html.tld"  prefix="html"%>
<%@ taglib uri="/WEB-INF/tlds/struts-logic.tld" prefix="logic"%>
<script language="JavaScript" src="js/popcalendar.js"></script>
<table>
    <tr><td colspan="2"> </td></tr>
    <logic:present parameter="sucesso">
            <tr>
                    <td width="30" />
                    <td class="msg" align="left">
                            Atividade cadastrada com sucesso.
                    </td>
            <tr>
    </logic:present>
    <html:messages id="error">
            <tr>
                    <td width="30" />
                    <td class="erro" align="left">
                            <bean:write name="error" />
                    </td>
            <tr>
    </html:messages>
    <tr><td colspan="2"> </td></tr>
</table>
<html:form       action="/cadastrarAtividadeExterna"       enctype="multipart/form-data"
onsubmit="return validateCadastrarAtividadeExternaForm(this);" >
    <table>
            <tr>
                    <td class="texto_padrao" valign="middle" align="right">
                            Responsável:
                    </td>
                    <td>
                            <html:select  property="responsavel"  styleClass="select_padrao"
styleId="comboGrupo">
                            <html:option value="1">Edinando Brustolin</html:option>
                            <html:option value="2">Zezinho</html:option>
                        </html:select>
                    </td>
            </tr>
            <tr>
                    <td class="texto_padrao" valign="middle" align="right">
                            Apoiadores:
                    </td>
                    <td>
                            <html:select   property="apoiador"   styleClass="select_padrao"
size="4" styleId="comboGrupo" multiple="yes">
                            <html:option value="1">João da Silva Machado</html:option>
```

```
                        <html:option           value="2">Givanildo           Arsendino
Fonseca</html:option>
                                <html:option value="3">Arselino Matias</html:option>
                                <html:option     value="4">Ivanilda     Joaquina     da
Cruz</html:option>
                    </html:select>
                </td>
            </tr>
            <tr>
                <td class="texto_padrao" valign="middle" align="right">
                    Título:
                </td>
                <td>
                    <html:text             property="titulo"             size="40"
styleClass="edit_padrao" styleId="comboGrupo" />
                </td>
            </tr>
            <tr>
                <td class="texto_padrao" valign="middle" align="right">
                    Modalidade:
                </td>
                <td>
                    <html:select  property="modalidade"  styleClass="select_padrao"
styleId="comboGrupo">
                      <html:option value="1">Reunião</html:option>
                      <html:option value="2">Evento</html:option>
                      <html:option value="3">Outro</html:option>
                    </html:select>
                </td>
            </tr>
            <tr>
                <td class="texto_padrao" valign="middle" align="right">
                    Local:
                </td>
                <td>
                    <html:text property="local" size="40" styleClass="edit_padrao"
styleId="comboGrupo" />
                </td>
            </tr>
            <tr>
                <td class="texto_padrao" valign="middle" align="right">
                    Data:
                </td>
                <td>
                    <html:text  property="data"  size="10"  styleClass="edit_padrao"
styleId="comboGrupo" />
                    <img  src="images/popcalendar/calendario.png"  alt="Selecione  a
data  de  elaboração"  onclick="popUpCalendar(this,  cadastrarAtividadeExternaForm.data,
'dd/mm/yyyy')" />
                </td>
            </tr>
            <tr>
                <td class="texto_padrao" valign="middle" align="right">
                    Horário:
                </td>
                <td>
                    <html:text             property="horario"             size="10"
styleClass="edit_padrao" styleId="comboGrupo" />
                </td>
            </tr>
            <tr>
                <td class="texto_padrao" valign="middle" align="right">
                    Duração:
                </td>
                <td>
```

```
                        <html:text            property="duracao"            size="10"
styleClass="edit_padrao" styleId="comboGrupo" />
                </td>
            </tr>
            <tr>
                <td class="texto_padrao" valign="middle" align="right">
                    Projeto:
                </td>
                <td>
                    <html:select    property="projeto"    styleClass="big_select"
styleId="comboGrupo">
                    <html:option   value="1">Ampliação   da   Força   Política   do
Municipalismo de Santa Catarina</html:option>
                    <html:option    value="2">Consultoria    Especializada    as
Associações de Municípios</html:option>
                    <html:option  value="3">Planos  e  Projetos  de  Desenvolvimento
Regional</html:option>
                            <html:option   value="4">Gestão   da   Informação   e
Desenvolvimento Tecnológico</html:option>
                            <html:option   value="5">Programa   de   Capacitação   de
Agentes Políticos e de Servidores Públicos Municipais</html:option>
                            <html:option   value="6">Programa   Gestão   Pública   de
Qualidade</html:option>
                        </html:select>
                </td>
            </tr>
            <tr>
                <td class="texto_padrao" valign="middle" align="right">
                    Conselho de Representação:
                </td>
                <td>
                    <html:text            property="conselho"            size="40"
styleClass="edit_padrao" styleId="comboGrupo" />
                </td>
            </tr>
            <tr>
                <td class="texto_padrao" valign="middle" align="right">
                    Pauta:
                </td>
                <td>
                    <html:textarea          property="duracao"          cols="40"
styleClass="edit_padrao" styleId="comboGrupo" />
                </td>
            </tr>
            <tr>
                <td class="texto_padrao" valign="middle" align="right">
                    Arquivo da Pauta:
                </td>
                <td>
                    <input     type="file"     name="arquivoPauta"     size="40"
styleClass="edit_padrao" styleId="comboGrupo" />
                </td>
            </tr>
            <tr>
                <td colspan="2" align="center">
                    <html:submit value="Enviar" />
                    <html:reset value="Limpar" />
                </td>
            <tr>
    </table>
    <html:javascript  formName="cadastrarAtividadeExternaForm"  dynamicJavascript="true"
staticJavascript="true"/>
</html:form>
<br>
```

12.1.51. cadastroDocumento.jsp

```
<%@ taglib uri="/WEB-INF/tlds/struts-bean.tld"  prefix="bean"%>
<%@ taglib uri="/WEB-INF/tlds/struts-html.tld"  prefix="html"%>
<%@ taglib uri="/WEB-INF/tlds/struts-logic.tld" prefix="logic"%>
<script language="JavaScript" src="js/popcalendar.js"></script>
<table width="100%" cellspacing="0" cellpadding="0" border="0">
    <tr><td height="10"></td></tr>
    <html:errors />
    <tr><td height="1pt" style="{background-color: #244F83}"></td></tr>
    <tr><td height="10" style="{background-color: #EEEEEE}"></td></tr>
    <tr>
           <td align="center" style="{background-color: #EEEEEE}">
                  <html:form action="/cadastrarDocumento" enctype="multipart/form-data"
onsubmit="return validateCadastrarDocumentoForm(this);" >
                        <html:hidden property="action" value="cadastrar" />
                               <table cellspacing="0" cellpadding="0" border="0">
                               <tr>
                                      <td     class="texto_padrao"     valign="middle"
align="right">
                                             Assunto:
                                      </td>
                                      <td width="10"></td>
                                      <td>
                                             <html:text  property="assunto"  size="40"
styleClass="edit_padrao" styleId="comboGrupo" />
                                      </td>
                               </tr>
                               <tr><td height="10" colspan="3"></td></tr>
                               <tr>
                                      <td     class="texto_padrao"     valign="middle"
align="right">
                                             Modalidade:
                                      </td>
                                      <td width="10"></td>
                                      <td>
                                             <html:select       property="modalidade"
styleClass="edit_padrao" styleId="comboGrupo">
                                                    <html:optionsCollection
name="modalidades"

value="id"

label="descricao"/>
                                          </html:select>
                                      </td>
                               </tr>
                               <tr><td height="10" colspan="3"></td></tr>
                               <tr>
                                      <td     class="texto_padrao"     valign="middle"
align="right">
                                             Autor:
                                      </td>
                                      <td width="10"></td>
                                      <td>
                                             <html:select            property="autor"
styleClass="edit_padrao" styleId="comboGrupo">
                                           <html:optionsCollection name="autores"

value="id"
```

```
label="nome"/>
                                        </html:select>
                                    </td>
                                </tr>
                                <tr><td height="10" colspan="3"></td></tr>
                                <tr>
                                    <td     class="texto_padrao"     valign="middle"
align="right">
                                        Assinante:
                                    </td>
                                    <td width="10"></td>
                                    <td>
                                        <html:select        property="assinante"
styleClass="edit_padrao" styleId="comboGrupo">
                                            <html:optionsCollection
name="assinantes"

value="id"

    label="nome"/>
                                        </html:select>
                                    </td>
                                </tr>
                                <tr><td height="10" colspan="3"></td></tr>
                                <tr>
                                    <td     class="texto_padrao"     valign="middle"
align="right">
                                        Destinatarios:
                                    </td>
                                    <td width="10"></td>
                                    <td>
                                        <html:select     property="destinatario"
styleClass="edit_padrao" styleId="comboGrupo" multiple="yes">
                                            <html:optionsCollection
name="destinatarios"

value="id"

label="nome"/>
                                        </html:select>
                                    </td>
                                </tr>
                                <tr><td height="10" colspan="3"></td></tr>
                                <tr>
                                    <td     class="texto_padrao"     valign="middle"
align="right">
                                        Forma de Envio:
                                    </td>
                                    <td width="10"></td>
                                    <td>
                                        <html:select       property="formaEnvio"
styleClass="edit_padrao" styleId="comboGrupo">
                                            <html:optionsCollection
name="formasEnvio"

value="id"

label="descricao"/>
                                        </html:select>
                                    </td>
                                </tr>
                                <tr><td height="10" colspan="3"></td></tr>
                                <tr>
```

```
                                        <td     class="texto_padrao"     valign="middle"
align="right">
                                                Data Elaboração:
                                        </td>
                                        <td width="10"></td>
                                        <td>
                                                <html:text     property="dataElaboracao"
size="10" styleClass="edit_padrao" styleId="comboGrupo" />
                                                <img
src="images/popcalendar/calendario.png"    alt="Selecione    a    data    de    elaboração"
onclick="popUpCalendar(this, cadastrarDocumentoForm.dataElaboracao, 'dd/mm/yyyy')" />
                                        </td>
                                </tr>
                                <tr><td height="10" colspan="3"></td></tr>
                                <tr>
                                        <td     class="texto_padrao"     valign="middle"
align="right">
                                                Data Envio:
                                        </td>
                                        <td width="10"></td>
                                        <td>
                                                <html:text           property="dataEnvio"
size="10" styleClass="edit_padrao" styleId="comboGrupo" />
                                                <img
src="images/popcalendar/calendario.png"     alt="Selecione     a     data     de     envio"
onclick="popUpCalendar(this, cadastrarDocumentoForm.dataEnvio, 'dd/mm/yyyy')" />
                                        </td>
                                </tr>
                                <tr><td height="10" colspan="3"></td></tr>
                                <tr>
                                        <td     class="texto_padrao"     valign="middle"
align="right">
                                                Arquivo:
                                        </td>
                                        <td width="10"></td>
                                        <td>
                                                <input     type="file"     name="arquivo"
size="40" styleClass="edit_padrao" styleId="comboGrupo" />
                                        </td>
                                </tr>
                                <tr><td height="10" colspan="3"></td></tr>
                                <tr>
                                        <td colspan="3" align="center">
                                                <html:submit value="Enviar" />
                                                <html:reset value="Limpar" />
                                        </td>
                                <tr>
                        </table>
                        <html:javascript              formName="cadastrarDocumentoForm"
dynamicJavascript="true" staticJavascript="true"/>
                </html:form>
            </td>
    </tr>
    <tr><td height="10" style="{background-color: #EEEEEE}"></td></tr>
    <tr><td height="1pt" style="{background-color: #244F83}"></td></tr>
    <tr><td height="10"></td></tr>
</table>
```

12.1.52. pesquisaDocumento.jsp

```
<%@ taglib uri="/WEB-INF/tlds/struts-bean.tld"  prefix="bean"%>
```

```
<%@ taglib uri="/WEB-INF/tlds/struts-html.tld"  prefix="html"%>
<%@ taglib uri="/WEB-INF/tlds/struts-logic.tld" prefix="logic"%>
<%@ taglib uri="/WEB-INF/tlds/displaytag-12.tld" prefix="display" %>
<script language="JavaScript" src="js/popcalendar.js"></script>
<table width="100%" cellspacing="0" cellpadding="0" border="0">
    <tr><td height="10"></td></tr>
    <html:errors />
    <tr><td height="1pt" style="{background-color: #244F83}"></td></tr>
    <tr><td height="10" style="{background-color: #EEEEEE}"></td></tr>
    <tr>
           <td align="center"  style="{background-color: #EEEEEE}">
                  <html:form       action="/pesquisarDocumentos"       onsubmit="return
validatePesquisarDocumentosForm(this);" >
                        <html:hidden property="action" value="pesquisar" />
                        <table cellspacing="0" cellpadding="0" border="0">
                               <tr>
                                      <td class="texto_padrao" valign="middle">
                                             Modalidades:   
                                      </td>
                                      <td valign="middle">
                                             <html:select      property="modalidades"
styleClass="edit_padrao" styleId="comboGrupo"  multiple="yes">
                                            <html:optionsCollection
name="modalidades"
                                                                  value="id"
                                                                  label="descricao"/>
                                        </html:select>
                                      </td>
                                      <td class="texto_padrao" valign="middle">

       Status:   
                                      </td>
                                      <td valign="middle">
                                             <select               nome="modalidades"
class="edit_padrao" styleId="comboGrupo" >
                                            <option                         value="id"
checked>Todos</option/>
                                                    <option
value="id">Enviados</option/>
                                                    <option            value="id">Não
Enviados</option/>
                                        </select>
                                      </td>
                                      <td valign="middle">
                                                <html:submit
value="Pesquisar" />
                                      </td>
                               <tr>
                        </table>
                        <html:javascript            formName="pesquisarDocumentosForm"
dynamicJavascript="true" staticJavascript="true"/>
                  </html:form>
           </td>
    </tr>
    <tr><td height="10" style="{background-color: #EEEEEE}"></td></tr>
    <tr><td height="1pt" style="{background-color: #244F83}"></td></tr>
    <tr><td height="10"></td></tr>
    <tr>
           <td align="center">
                  <!-- se existir os documentos na sessao mostra aqui -->
                  <logic:present name="documentos" scope="request">
                         <display:table  name="requestScope.documentos"
                                       id="doc"
                                        pagesize="10"
                                       defaultorder="ascending"
```

```
                                        requestURI="pesquisarDocumentos.do"
                                       class="cate"
                                        cellspacing="1"
                                        cellpadding="1">
                                <display:column   property="identificador"   title="ID:"
sortable="true" headerClass="sortable" align="center"/>
                                <display:column                    property="modalidade"
title="Modalidade:" sortable="true" headerClass="sortable" align="center" />
                                <display:column     property="autor"     title="Autor:"
sortable="true" headerClass="sortable" align="center" />
                                <display:column  property="dataElaboracao"  title="Data
de Elaboração:" sortable="true" headerClass="sortable" align="center"/>
                                <display:column    property="status"    title="Status:"
sortable="true" headerClass="sortable" align="center"/>
                                <display:column   property="arquivo"   title="Nome   do
Arquivo:" sortable="true" headerClass="sortable" align="left"/>
                                <display:column sortable="false" headerClass="sortable"
autolink="true"               url="/downloadDocumento.do"               paramId="nomeArq"
paramProperty="caminhoRelativo" align="center" style="{border:0}">
                                        <font size="1">download</font>
                                </display:column>
                                <display:column sortable="false" headerClass="sortable"
autolink="true"               url="/downloadDocumento.do"               paramId="nomeArq"
paramProperty="caminhoRelativo" align="center" style="{border:0}">
                                        <font size="1">envio</font>
                                </display:column>
                        </display:table>
                </logic:present>
        </td>
    </tr>
    <tr><td height="10"></td></tr>
</table>
```

12.1.53. menu.jsp

```
<%@ taglib uri="/WEB-INF/tlds/struts-html.tld"  prefix="html"%>
<script>
function mostrarMenu(menu){
    var img = document.getElementById(menu + '_img');
    var array = img.src.split('/');
    var visible = '';
    if(array[array.length - 1] == 'mais.gif'){
        img.src = 'images/menos.gif';
    } else {
        img.src = 'images/mais.gif';
        visible = 'none';
    }

    for(i = 0; ; i++){
        var tr = document.getElementById(menu + '_sub_' + i);
        if(tr == null){
            break;
        }
        tr.style.display = visible;
    }
}
</script>
<table height="100%" cellspacing="0" cellpadding="0" >
    <tr>
        <td height="30" />
    </tr>
    <tr id="doc">
```

```
            <td align="left">
            <img id="doc_img" src="images/mais.gif" />
            <a                                      href="javascript:mostrarMenu('doc');"
class="link_normal"><b>Documentos</b></a>
        </td>
    </tr>
    <tr id="doc_sub_0"  style="{display:none}">
            <td align="left">
                       <img src="images/submenu.gif"/>
                   <html:link    forward="cadastro.documento"    styleClass="link_normal"
styleId="link1">
                   Cadastro
            </html:link>
        </td>
    </tr>
    <tr id="doc_sub_1"  style="{display:none}">
            <td align="left">
                       <img src="images/submenu.gif"/>
                   <html:link    forward="cadastro.documento"    styleClass="link_normal"
styleId="link1">
                   Confirmação de Envio
            </html:link>
        </td>
    </tr>
    <tr id="doc_sub_2"  style="{display:none}">
            <td align="left">
                       <img src="images/submenu.gif"/>
                   <html:link    forward="pesquisa.documentos"    styleClass="link_normal"
styleId="link1">
                   Pesquisa
            </html:link>
        </td>
    </tr>
    <tr id="ativ_ext">
            <td align="left">
            <img id="ativ_ext_img" src="images/mais.gif" />
            <a                                 href="javascript:mostrarMenu('ativ_ext');"
class="link_normal"><b>Atividades Externas</b></a>
        </td>
    </tr>
    <tr id="ativ_ext_sub_0"  style="{display:none}">
            <td align="left">
                       <img src="images/submenu.gif"/>
                   <html:link                       forward="cadastro.atividade.externa"
styleClass="link_normal" styleId="link1">
                   Cadastro
            </html:link>
        </td>
    </tr>
    <tr id="ativ_ext_sub_1"  style="{display:none}">
            <td align="left">
                       <img src="images/submenu.gif"/>
                   <html:link                       forward="cadastro.atividade.externa"
styleClass="link_normal" styleId="link1">
                   Conclusão
            </html:link>
        </td>
    </tr>
    <tr id="ativ_ext_sub_2"  style="{display:none}">
            <td align="left">
                       <img src="images/submenu.gif"/>
                   <html:link                       forward="cadastro.atividade.externa"
styleClass="link_normal" styleId="link1">
                   Pesquisa
            </html:link>
```

```
        </td>
    </tr>
</table>
```

12.1.54. rodape.jsp

```
<table width="100%" border="0" cellpadding="0" cellspacing="0" height="30">
    <tr>
            <td class="fundo_rodape" width="100%" align="left">
                    <div align="center" class="texto_rodape">
                     © 2005   ·   www.fecam.org.br   ·   fecam@fecam.org.br
                    </div>
            </td>
    </tr>
</table>
```

12.1.55. cabecalho.jsp

```
<table width="100%" border="0" cellpadding="0" cellspacing="0">
    <tr>
            <td class="fundo_cabecalho" width="100%" align="left">
                    <table width="100%" border="0" cellpadding="0" cellspacing="0">
                            <tr>
                                    <td align="left">
                                            <img src="images/topo_fecam.gif" />
                                    </td>
                                    <td align="right" class="titulo_cabecalho">
                                            CATE
                                    </td>
                                    <td align="right" width="20">
                                             
                                    </td>
                            </tr>
                    </table>
            </td>
    </tr>
</table>
```