UNIVERSIDADE FEDERAL DE SANTA CATARINA

CURSO DE CIÊNCIAS DA COMPUTAÇÃO

Romulo Flávio Bittencourt

ANÁLISE DE SISTEMA PARA INFORMATIZAÇÃO DOS PROCESSOS DE COMUNICAÇÃO EM ORGANIZAÇÕES

Proposta de Trabalho de Conclusão de Curso a ser submetido à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de Bacharel em Ciências da Computação.

Florianópolis-SC, Julho de 2008.

Romulo Flávio Bittencourt

ANÁLISE DE SISTEMA PARA INFORMATIZAÇÃO DOS PROCESSOS DE COMUNICAÇÃO EM ORGANIZAÇÕES

Proposta de Trabalho de Conclusão de Curso a ser submetido à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de Bacharel em Ciências da Computação.

_____

Monica Maria Jungues
Orientadora

Banca Examinadora

_____

Prof. Ricardo Pereira e Silva, Dr.
Universidade Federal de Santa Catarina
Co-Orientador

_____

Prof. José Eduardo de Lucca, Dr.
Universidade Federal de Santa Catarina

AGRADECIMENTOS

Agradeço a meus pais, pela paciência, compreensão e o ambiente que proporcionaram para minha formação, tanto pessoal como profissional. Pelo forte incentivo e apoio em meus projetos de vida, sempre me fornecendo acesso aos melhores ambientes de ensino e educação.

Agradeço a todos os avaliadores deste trabalho, especialmente pela sensatez que demonstraram mesmo nos momentos mais complicados.

Agradeço aos meus colegas de classe na universidade, especialmente ao Abel, Dalton Filho, Erich Silvestre e Pedro Lima, pelo companheirismo, compartilhamento de informação e camaradagem em diversos momentos.

Agradeço aos meus amigos, especialmente a Charlene, Cesar Augusto dos Santos, Georgia kasten, Lara Almeida, Luis Ribeiro e Mariana Knierim, pela troca de experiências, motivando-me sempre a seguir em frente ouvindo minhas emoções e a minha forte intuição.

Muito obrigado também a Janaina, minha irmã, pelo seu interesse e torcida pelo meu sucesso.

Creio que tudo aconteceu por intuição, por garra, por dedicação. Estudei muito, mas também deixei-me levar pelo coração. Porque sempre quis compartilhar meu aprendizado de vida, pessoal e profissional, com as outras pessoas. Por isso, a essência de minha experiência de vida é o amor.

Além das fundamentações científicas, a base de minha experiência é o amor que vejo em cada ser humano que me procura e a percepção do enorme potencial que carrega.

Esse é o talento que Deus me deu para ajudar as pessoas a descobrirem a extraordinária força que existe nelas, fazendo-as perceber-se mais fortes e capazes, possibilitando transformações inacreditáveis.

O que antes parecia impossível torna-se possível quando alguém se sente apto a transpor todos os obstáculos, conhecendo a sua verdadeira grandeza interior.

A conclusão deste trabalho, bem como de minha vida acadêmica no curso de ciências da computação tem a ver com o sucesso! E tem a ver também com o acreditar. Além de ser incrivelmente gratificante para minha pessoa, como você pode sentir, também o é para todos que participaram direta ou indiretamente desta minha formação pessoal e profissional. Todo o amor que houver nesta vida é pra vocês.

RESUMO

O procedimento de comunicação em algumas organizações ainda é feito, nos dias de hoje, através de documentos impressos. Ao fazer uma análise desta forma de comunicação foi possível constatar que este procedimento, muitas vezes, apresenta problemas com morosidade, pois o processo de comunicação depende da confecção, expedição e ciência do documento pelas partes envolvidas. Existem casos onde uma destas etapas pode ser morosa possibilitando a chance de comprometer todo o processo. Pode-se citar como exemplo o caso da comunicação estar destinada a lugares distantes. Somado a isto ela também pode apresentar problemas em termos de segurança, pois durante o percurso pelo qual o documento percorre, existe a possibilidade do mesmo ser extraviado. Também pode não possuir histórico em virtude de não existir um procedimento padronizado de arquivamento das comunicações a fim de compor um histórico confiável. Por fim pode-se citar os custos gerados com papel, impressão, postagem, envelopamento dentre outros ônus.

O objetivo deste trabalho é propor uma solução plausível com o intuito de alcançar a informatização deste procedimento. Para tal há necessidade de passar da fronteira da simples escolha e implantação de programas de computadores que apóiem as tarefas dos funcionários no que diz respeito à edição de textos, planilhas ou imagens, envio e recebimento de e-mails bem como outras opções que já existam soluções no mercado. Tendo em mente este aspecto, haverá a necessidade da contratação de pessoal com conhecimento técnico na criação e implantação de novas ferramentas para compor o processo de informatização. Apesar de gerar um investimento monetário inicial diante da dependência de mão de obra qualificada, este investimento poderá proporcionar um retorno, não somente financeiro bem como um ganho na eficiência, eficácia e satisfação do serviço prestado.

Para a análise da informatização das comunicações serão utilizados passos bem definidos nas áreas de Engenharia de Software e Análise de Sistemas, e através destes é que se tem a oportunidade de aplicar os conhecimentos adquiridos ao longo do curso de ciências da computação para atingir a proposta de trabalho.

Palavras-Chave: processo, metodologia, análise, requisitos, RUP, padrões, java, frameworks, organização, comunicação.

ABSTRACT

The process of communication in some organizations is still made, today, through printed documents. In doing an analysis of this form of communication was possible to see that this procedure often presents problems with slow because the communication process depends on manufacturing, shipping and science of the document by the parties involved. There are cases where one of these steps can be time consuming allowing the chance of compromising the whole process. You can cite as an example the case of the communication be to distant places. In addition to this it can also present problems in terms of security, because during the journey by which the document covers, there is the possibility of it being lost. It may not have history because there is no standardized procedure for filing of communications in order to compose a reliable history. Finally you can cite the costs generated with paper, printing, post, among others envelopamento burden.

The objective of this work is to propose a plausible solution in order to achieve the computerization of this procedure. For this we need to pass the border of simple choice and implementation of programmes for computers that support the work of officials regarding the publication of texts, images or spreadsheets, sending and receiving e-mails and other options that already exist solutions on the market. Mindful of this, there will be a need for recruitment of personnel with expertise in the creation and deployment of new tools to compose the process. Despite generating an initial investment money before the dependence of qualified workforce, this investment will provide a return, not only financial as well as a gain in efficiency, effectiveness and satisfaction of service provided.

For the analysis of computerization of communications will be used well-defined steps in the areas of Software Engineering and Systems Analysis, and through these is that if you have the opportunity to apply the knowledge acquired during the course of computer science to achieve the proposed work.

Keywords: process, methodology, analysis, requirements, RUP, patterns, java, frameworks, organization, communication.

SUMÁRIO

Lista de figuras

Lista de tabelas

1 INTRODUÇÃO

A foco das atividades é a Análise do Sistema bem como da Engenharia de Software para o planejamento, desenvolvimento e manutenção de um sistema de informatização de comunicação interna em organizações, através da aplicação dos conhecimentos adquiridos ao longo do curso de Ciências da Computação da Universidade Federal de Santa Catarina possibilitando alcançar essa meta, capacitando a novas oportunidades de empregos no mercado de trabalho.

Uma das ferramentas utilizadas será o processo de desenvolvimento conhecido pela sigla RUP que vem da abreviação de Rational Unified Process. Este processo, mundialmente utilizado, representa uma forma sistemática para desenvolvimento de software, onde as etapas são ligadas umas às outras em ciclos de desenvolvimento o qual, a cada nova iteração, pode-se fazer novos refinamentos do que está sendo desenvolvido (KRUCHTEN, 2003).

Esta nova filosofia de trabalho e desenvolvimento de sistemas informatizados para criação de programas é que está sendo tratada nesta atividade de conclusão de curso.

1.1 TEMA

O tema deste projeto de conclusão de curso é a análise do sistema e o estudo da engenharia de software, focados no processo de comunicação interna de organizações, através do estudo de ferramentas para desenvolvimento de software, como o RUP, citado anteriormente, utilizando-se do conhecimento com relação a este tipo de processo de desenvolvimento de programas no Brasil e no mundo.

Para isto, no decorrer do trabalho, são apresentadas as etapas das metodologias utilizadas e dos seus principais aspectos presentes no sistema proposto.

1.2 JUSTIFICATIVA

Em diversos setores e células das organizações além da necessidade do espaço físico, muitos dos procedimentos realizados ainda, são feitos em papel. Havendo desta forma uma necessidade crescente em se informatizar alguns destes procedimentos e de que os mesmos pudessem ser acessados de qualquer ponto com acesso a internet. Em muitos casos reais constata-se que a informatização parcial dos processos organizacionais de uma empresa já é o suficiente para a criação e manutenção da mesma.

O sistema final deste projeto será a base para a informatização de parte dos processos organizacionais mais em particular das comunicações internas gerando melhorias na sua comunicação reduzindo também os gastos gerados com a mesma.

Metaforicamente pode-se comparar que ao final deste projeto tem-se a estrutura de um prédio erguida, para a partir dela, construir sistemas completos e especializados de acordo com as necessidades de cada organização.

Assim almeja-se com o desenvolver deste projeto de conclusão de curso, obter reconhecimento, inspiração, e um recurso ao qual poderá ser utilizado como referência na busca por uma vaga no mercado de trabalho no término da formação acadêmica no curso de ciências da computação, agregando conhecimentos na área de análise de sistemas e engenharia de software, através da aplicação dos conhecimentos adquiridos ao longo do curso.

Todas as funcionalidades do sistema serão construídas seguindo padrões e, portanto todo o conhecimento adquirido no estudo para desenvolvimento deste trabalho serão de extrema valia, sendo completamente utilizados em todos os ciclos de desenvolvimento tornando-se o principal fator motivador na execução deste projeto.

1.3 OBJETIVOS

1.3.1 Objetivo Geral

Apresentar um sistema de fácil planejamento, desenvolvimento e manutenção do processo de comunicação interna, e sensibilizar a importância da informatização das comunicações nos dias de hoje através de um sistema prático, seguro e com boa performance nas funcionalidades.

Projetar um sistema de fácil utilização com o intuito de que as organizações possam continuar operando com os mesmos funcionários, não necessitando troca de sua equipe. Para tal, apenas conhecimentos básicos de informática, no que diz respeito à edição de textos e acesso a páginas na internet, serão necessários.

1.3.2 Objetivos Específicos

- Enfatizar e descrever os principais passos dos estudos e levantamento dos requisitos bem como dos casos de uso.
- Definir o escopo de atuação que esse sistema é capaz de atuar no processo de comunicação interna.
- Explanar acerca do sistema analisado.
- Utilizar padrões consagrados no mercado de trabalho no desenvolvimento de software.

3 ALÉM DA PROGRAMAÇÃO

Durante o curso de ciências da computação foram mostrados estudos e preocupações com o que estava além da tecnologia e que fazia bons projetos falharem: engenharia de software, metodologia de desenvolvimento e gerenciamento de projetos.

O objetivo desta monografia é utilizar metodologias de desenvolvimento de software que sejam mais racionais, com usabilidade e padrões consagrados no mercado de trabalho.

3.1 Metodologias de Desenvolvimento

Metodologias de desenvolvimento e estruturas de avaliação de processos podem ser comparadas sob duas dimensões: de um lado, tem o vértice pouca formalidade / muita formalidade e de outro, o método cascata / método iterativo, exemplificado através da Figura 1.

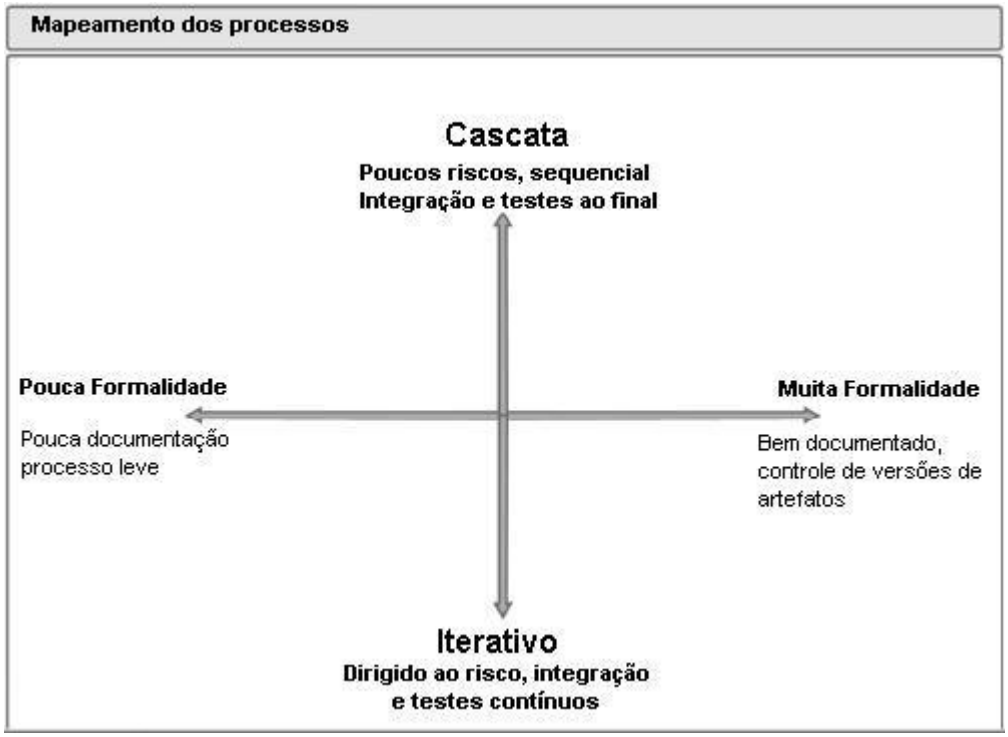


Figura 1 – Mapa de Processos (SOMMERVILLE, 2001)

Os processos com pouca formalidade produzem o mínimo de documentação possível e procedimentos de trabalho bastante informais. Os formais possuem maior documentação, mantém o histórico dos artefatos gerados e possuem gerenciamento de mudanças.

O método cascata é um procedimento linear, onde a integração e os testes são feitos no fim do ciclo de desenvolvimento. O método iterativo é guiado pelo risco, ou seja, é voltado para a eliminação e minimização de riscos.

A implementação da arquitetura, a integração e os testes são realizados desde o início do ciclo de vida do aplicativo.

3.1.1 Método Cascata

Esse método, também conhecido como seqüencial, ou linear, foi utilizado por muitos anos e ainda é utilizado. Segundo (KROLL e KRUCHTEN 2003), esse processo se baseia nos seguintes passos:

- Entender completamente o problema a ser resolvido, seus requisitos e suas restrições;
- Projetar uma solução que atenda todos os requisitos e restrições. Examinar o projeto cuidadosamente e ter certeza que todas as partes interessadas concordam que essa é a solução certa;
- Fazer a implementação do projeto, usando as melhores técnicas de engenharia;
- Verificar se a solução atende aos requisitos estabelecidos;
- Distribuir o produto.

Esse processo é similar à forma a qual pontes e edifícios são construídos. Algumas coisas devem ser feitas dessa maneira. Em um projeto com dois meses de duração, essa metodologia poderia ser usada. Mas normalmente, softwares não devem ser desenvolvidos dessa forma.

3.1.2 Método Iterativo e Incremental

O método iterativo foi criado para superar as dificuldades impostas pelo modelo cascata, já que o modelo cascata pode ser usado com sucesso em projetos pequenos, onde o domínio do problema é bem conhecido, a solução encontrada foi dividir grandes projetos em projetos menores (KROLL, 2003).

Dessa maneira, alguns requisitos e alguns riscos podem ser identificados, um projeto pode ser realizado, uma implementação pode ser construída para esse projeto, validada e testada. Esse processo se repete com

outras partes do sistema até que o sistema inteiro seja terminado. Isso é chamado de modo iterativo.

Em cada pequena parte do sistema é feita uma iteração. A iteração segue o modelo seqüencial tradicional, com identificação de necessidades, análise, projeto, implementação e testes. A cada iteração o sistema é incrementado até que o ciclo de desenvolvimento do aplicativo termine. Nesse ponto, um novo ciclo de desenvolvimento pode ser iniciado.

A maneira de desenvolver projetos através de várias iterações que vão incrementando o projeto até que se chegue a um objetivo é chamada de modo iterativo e incremental. Atualmente esse paradigma de desenvolvimento é bem aceito e vem sendo utilizado por várias metodologias de desenvolvimento de software.

4 O RATIONAL UNIFIED PROCESS

O RUP é um processo iterativo e incremental que provê uma abordagem disciplinada para o desenvolvimento de software (POLLICE). Na engenharia de software, um modelo de processo é um conjunto de passos parcialmente ordenados com a intenção de construir um produto de software de qualidade, capaz de atender às necessidades e exigências do usuário final, de acordo com planejamento e orçamento previstos (SOMMERVILLE, 2001)

Conforme (KROLL e KRUCHTEN, 2003), pode-se ter três definições para o Rational Unified Process (RUP):

- O RUP é uma maneira de desenvolvimento de software que é iterativa, centrada à arquitetura e guiada por casos de uso. É descrita em vários livros e artigos. Uma das maiores fontes de informações é o próprio produto IBM RUP, que contém guias detalhados, exemplos e modelos cobrindo todo o ciclo de vida do software;

- O RUP é um processo de engenharia de software bem definido e bem estruturado. O RUP define claramente quem é responsável pelo que, como as coisas devem ser feitas e quando fazê-las. O RUP também provê uma estrutura bem definida para o ciclo de vida de um projeto RUP, articulando claramente os marcos essenciais e pontos de decisão;
- O RUP é também um produto de processo que oferece uma estrutura de processo customizável para a engenharia de software. O produto IBM RUP suporta a customização e autoria de processos, e uma vasta variedade de processos, ou configuração de processos, podem ser montadas nele. Essas configurações do RUP podem ser criadas para suportar equipes grandes e pequenas, e técnicas de desenvolvimento disciplinadas ou menos formais. O produto IBM RUP contém várias configurações e visões de processos prontas que guiam analistas, desenvolvedores, testadores, gerentes de projeto, gerentes de configuração, analistas de dados, e outros membros da equipe de desenvolvimento em como desenvolver o software. Ele tem sido utilizado por muitas companhias em diferentes setores da indústria.

Segundo (BOOCH, 2000) uma das principais características do RUP é o uso da iteração que através de refinamentos sucessivos melhora o entendimento do problema. (BOOCH, 2000) afirma que o uso de uma linguagem de programação orientada a objetos e de notação UML para documentação de um projeto possibilita a utilização de modelos visuais que, também, contribuem para a qualidade final do software. Quanto ao fato do processo de desenvolvimento de software sofrer mudanças sistemáticas de requisitos, o RUP consegue identificar estas mudanças e solucionar os problemas o mais rápido possível, isso porque o RUP fornece uma abordagem disciplinada para dividir atividades e responsabilidades no desenvolvimento do software.

Cada projeto tem suas peculiaridades; por isso, é importante fazer um planejamento antes de começar a desenvolver qualquer software. Para isso, identificam-se quais são os requisitos do software e, depois, escolhe-se uma metodologia de desenvolvimento que, com certeza, o seu projeto terá uma qualidade superior aquela que o seu cliente está esperando.

Por ser flexível e configurável, o RUP pode ser utilizado em projetos de pequeno, médio e grande porte. (KROLL e KRUCHTEN, 2003) mostram como o RUP pode ser utilizado em um projeto de uma semana com uma equipe de uma pessoa.

4.1 Os princípios do RUP

Não existe uma maneira exata de aplicar o RUP, pois ele pode ser aplicado de várias formas e será diferente em cada projeto e organização. Porém, segundo (KROLL e KRUCHTEN, 2003) existem alguns princípios que podem caracterizar e diferenciar o RUP de outros métodos iterativos:

- Atacar os riscos cedo e continuamente;
- Certificar-se de entregar algo de valor ao cliente;
- Focar no programa executável;
- Acomodar mudanças cedo;
- Liberar um executável da arquitetura cedo;
- Construir o sistema com componentes;
- Trabalhar junto como um time;
- Fazer da qualidade um estilo de vida, não algo para depois.

4.2 Elementos do RUP

O RUP possui cinco elementos principais: papéis, atividades, artefatos, fluxos de trabalho e disciplinas.

Um papel (ou perfil) define o comportamento e as responsabilidades de um determinado indivíduo ou grupo de indivíduos trabalhando como uma

equipe. Papéis não são indivíduos e nem títulos de trabalho. Um indivíduo pode assumir vários papéis. São exemplos de papéis:

- Analista de sistema – O indivíduo que assume este papel coordena a obtenção dos requisitos e a modelagem dos casos de uso identificando funcionalidades do sistema e estabelecendo limites do sistema;
- Projetista – Esse indivíduo define responsabilidades, operações, atributos, relacionamentos de uma ou mais classes e determina como elas devem ser ajustadas para serem implementadas no ambiente;
- Projetista de testes – Responsável pelo planejamento, projeto, implantação e avaliação de testes, incluindo a geração de plano e modelo de teste, implementando procedimentos de testes e avaliando a abrangência dos testes, resultados e a efetividade.

Uma atividade é uma unidade de trabalho que um indivíduo executa quando está exercendo um determinado papel e produz um resultado importante para o contexto do projeto. Cada atividade pode ser dividida em passos. São exemplos de atividades:

- Planejar uma iteração: realizada pelo papel gerente de projeto;
- Encontrar casos de uso e atores: realizada pelo papel analista de sistemas;
- Rever o projeto: realizada pelo papel revisor de projeto;
- Executar um teste de performance: realizado pelo papel testador de performance.

Um artefato é um pedaço de informação que é produzido, modificado ou utilizado em um processo. Os artefatos são os produtos de um projeto. São as coisas produzidas durante o desenvolvimento do projeto. Artefatos são utilizados como entradas de atividades e são produzidos como saída. Os artefatos podem ter várias formas:

- Um modelo, como um modelo de caso de uso ou um modelo de projeto;

- Um elemento de um modelo, como uma classe, um caso de uso, um subsistema;
- Um documento, como um caso de negócio, glossário, visão;
- Código fonte;
- Executáveis.

A enumeração de atividades, papéis e artefatos não constituem um processo. É necessário saber a seqüência do desenvolvimento das atividades para que possam ser produzidos artefatos de valor para o projeto. Um fluxo de trabalho é uma seqüência de atividades que são executadas para a produção de um resultado valioso para o projeto. Fluxos de trabalho podem ser representados por diagramas de seqüência, diagramas de colaboração e diagramas de atividades da linguagem UML. O RUP utiliza três tipos de fluxos de trabalho:

- Fluxos de trabalho principais, associados com cada disciplina (vide figura dois mais abaixo);
- Fluxos de trabalho de detalhe, para detalhar cada fluxo de trabalho principal (vide figura três mais abaixo);
- Planos de iteração, que mostram como a iteração deverá ser executada.

Uma disciplina é uma coleção de atividades relacionadas que fazem parte de um contexto comum em um projeto. As disciplinas proporcionam um melhor entendimento do projeto sob o ponto de vista tradicional de um processo cascata. A separação das atividades em disciplinas torna a compreensão das atividades mais fácil, porém dificulta mais o planejamento das atividades. O RUP possui nove disciplinas, divididas em disciplinas do processo e de suporte. As disciplinas de processo são: modelagem de negócios, requisitos, análise e projeto, implementação, teste e distribuição. As de suporte são: configuração e gerenciamento de mudanças, gerenciamento de projeto, e ambiente.

5 PADRÕES DE PROJETO (DESIGN PATTERNS)

A origem dos Padrões de Projeto vem do trabalho de um arquiteto chamado Christopher Alexander, no final da década de 70. Ele escreveu dois livros, inicialmente, “A Pattern Language” e “A Timeless Way of Building”, nos quais ele exemplificava o uso e descrevia seu raciocínio para documentar os padrões (Erich, Richard, Ralph, John, 1995).

Em 1995, este grupo de quatro profissionais escreveu e lançou o livro "Design Patterns: Elements of Reusable Object-Oriented Software", um catálogo com 23 padrões de projeto. Estes autores ficaram mais conhecidos como A Gangue dos Quatro (Gang Of Four ou GoF), considerados os maiores entusiastas dos Padrões de Projeto.

5.1 O que são os Padrões de Projeto

Segundo Christopher Alexander, "Cada Pattern descreve um problema o qual ocorre repetidamente em nosso ambiente, e então descreve um conjunto de soluções para este problema, de maneira que você possa usar esta solução um milhão de vezes, sem o fazer da mesma maneira duas vezes." (ALEXANDER, 1977).

Um Padrão de Projeto descreve uma solução comprovada para um problema de projeto recorrente, dando ênfase particular no contexto e forçando a aproximação do problema, e as conseqüências e o impacto de sua solução.

Padrões são dispositivos que permitem que os programas compartilhem conhecimento sobre o seu projeto. Quando se desenvolve software, encontram-se muitos problemas que ocorrem, ocorreram e irão ocorrer novamente. Para solucionar estes problemas pode-se documentar um Padrão com o intuito de reusar e possivelmente compartilhar informação que se aprendeu sobre a melhor maneira de resolver um problema de projeto de programa.

Segundo o catálogo de padrões do Gang Of Four (BUSCHMANN, 1996), tem-se 23 padrões e está basicamente, dividido em três seções:

- Criacional (Creational).
- Estrutural (Structural)
- Comportamental (Behavioral)

Nesta monografia, são apresentados padrões que podem ser utilizados exemplificando cada um deles com diagramas UML.

5.1.1 Por que usar os Padrões de Projeto?

De acordo com Gamma (Gamma, 1995), a parte difícil sobre projeto orientado a objetos é a decomposição do sistema em objetos. Abaixo são listados três importantes aspectos:

- **Foram aprovados:** Os padrões refletem a experiência, conhecimento e soluções de desenvolvedores que obtiveram sucesso usando determinados padrões em seus trabalhos.
- **São reusáveis:** Os padrões provêem uma solução pronta que pode ser aplicada a diferentes problemas.
- **São expressíveis.** Os padrões provêem um vocabulário comum de soluções que podem expressar muitas soluções, sucintamente.

Vale ressaltar que os padrões, por si só, não garantem o sucesso do seu uso. A descrição do padrão indica quando ele pode ser aplicado, mas apenas a experiência pode proporcionar o entendimento de quando um padrão particular irá melhorar o projeto do sistema.

5.1.2 Documentando um Padrão de Projeto

A intenção e motivação de se criar um Padrão de Projeto é poder abstrair um problema recorrente e criar uma solução viável, além de poder compartilhar este conhecimento para que outras pessoas se beneficiem dele. A documentação de um Padrão de Projeto é feita numa forma bem definida. A forma geral para se documentar um padrão inclui a definição dos seguintes itens:

- A motivação ou o contexto que o padrão se aplica;
- Pré-requisitos que devem ser satisfeitos antes de se decidir aplicar um padrão;
- Uma descrição da estrutura do programa que o padrão será definido;
- Uma lista dos participantes necessários para completar um padrão;
- Conseqüências do uso do padrão, positivas e negativas;
- Exemplos;

Nos capítulos anteriores e no decorrer deste, foi apresentada uma visão geral sobre o problema proposto. Também foram abordados aspectos relevantes sobre processo de desenvolvimento de software e padrões de projeto, citando algumas referências bibliográficas como subsídio para a busca de maiores informações a respeito. A partir do próximo capítulo será dado início ao desenvolvimento do projeto, dando ênfase aos aspectos mais diretamente correlacionados ao mesmo.

6 PLANO DE DESENVOLVIMENTO DO PROGRAMA

Passada a etapa que focava a revisão bibliográfica, será apresentado neste e nos próximos capítulos os aspectos referentes ao desenvolvimento do projeto. Começará sendo apresentado o plano de desenvolvimento do programa. Este plano visa fornecer a todos os envolvidos no projeto uma visão

do ambiente de desenvolvimento, as ferramentas utilizadas, como programas de terceiros, e a estrutura de diretórios, para orientar o desenvolvimento do software.

6.1 Escopo

6.1.1 Definições, acrônimos e abreviações.

Na descrição deste plano de desenvolvimento do programa são utilizados os seguintes termos e suas respectivas definições:

- **BD** – Banco de Dados
- **SO** – Sistema Operacional
- **SGBD** –Sistema Gerenciador de Banco de Dados

6.2 Ambiente e Desenvolvimento

6.2.1 Estrutura do projeto

Figura 2 – Estrutura do Projeto

- **Documentação** – Documentação do sistema CE;
- **Artefatos** – Documentação técnica do sistema CE;
- **Pendências** – Local onde ficam armazenadas as pendências individuais. Desta forma, caso exista algum problema todos podem visualizar para tomar nota;
- **Tutoriais** – Documentação referente às ferramentas, como programas de terceiros ou APIs, utilizadas pelos participantes do projeto;

- **Ferramentas** – Ferramentas, como programas de terceiros ou APIs utilizadas pelos participantes do projeto;
- **Frameworks** – Estrutura do projeto utilizada para acoplar as ferramentas utilizadas, como por exemplo o conjunto de APIs;
- **Java** – Plataforma utilizada para a programação e sua devida implementação do sistema;
- **IDE** – Ide's utilizadas no desenvolvimento;
- **JDK** – Máquina virtual da plataforma Java;
- **LIB** – Armazenamentos das lib's que o eclipse não disponibiliza;
- **SERVIDORES** – Servidores utilizados;
- **SGBD** – Sistema Gerenciador de Banco de Dados;
- **WEB** – Servidor web;

6.3 Ferramentas e Frameworks

Abaixo são listadas as ferramentas e frameworks que todos os envolvidos no projeto poderão utilizar seguindo assim uma padronização, em que todos os envolvidos têm os mesmo ambiente de desenvolvimento.

Ferramentas e Frameworks		
Ferramenta	**Nome**	**Versão**
SO	XP/Linux	
Máquina Virtual	JDK	5.0 Update 5
IDE	Eclipse	3.1
Container JSP	TomCat	5.5.9
Plugin	Lomboz	3.1 RC 2
Framework J2EE	Spring – Framework	1.2.5
Framework J2EE	Struts	1.2.7
Cliente CVS	WinCVS	2.0.2.4
Cliente BD	PHPPGAdmin	1.2.2
Browser	Internet Explorer	5.5 ou superior
Browser	Mozilla FireFox	0.8 ou superior

Tabela 1 – Ferramentas e Frameworks

* wincvs ou gcvs somente existe a necessidade de instalar um destes aplicativos caso o eclipse não seja instalado.

7 VISÃO DE NEGÓCIO

Neste capítulo será apresentado o documento referente à visão do negócio. Esta visão de negócio visa fornecer a todos os envolvidos uma visão do negócio na qual o desenvolvimento deste sistema está inserido, apresentando subsídios para os especialistas no negócio e analistas de sistemas na orientação do desenvolvimento do programa.

7.1 Escopo

Esta seção trata especificamente do negócio o qual o programa que será construído se propõe a solucionar, não sendo aplicado a questões periféricas como treinamento dos usuários, adequação dos procedimentos ao uso do programa e processo de implantação.

7.2 Definições, acrônimos e abreviações.

Na descrição desta visão de negócio são utilizados os seguintes termos e suas respectivas definições:

- **CI** – Comunicação Interna
- **CE** – Comunicação Eletrônica
- **Emissor** – Pessoa responsável por gerar uma CE.
- **Destinatário** – Pessoa a qual uma CE é dirigida.
- **Notificado** – Pessoa que recebe uma notificação quando o prazo para execução de uma CE não for cumprido.

7.3 Posicionamento

7.3.1 Descrição do Problema

Atualmente, em muitas organizações as comunicações são feitas com documentos impressos. Quando realizado desta forma este procedimento pode apresentar os seguintes problemas:

- **Moroso**: o processo de comunicação depende da confecção, expedição e ciência do documento pelas partes envolvidas. Se uma destas etapas for morosa compromete todo o processo, como no caso da comunicação estar destinada a lugares distantes;
- **Inseguro**: durante o percurso pelo qual o documento percorre, existe a possibilidade de ser extraviado;
- **Sem acompanhamento**: o emissor em um dado tempo, não tem conhecimento da situação na qual se encontra o documento;
- **Sem histórico**: não existe um procedimento padronizado de arquivamento das comunicações a fim de compor um histórico confiável;
- **Oneroso**: as comunicações geram custos com papel, impressão, postagem, envelopamento dentre outros.

Estes aspectos podem afetar a estrutura de uma organização, tendo em vista que este processo de comunicação pode estar difundido entre diversas de suas células. Assim, sua melhoria impacta na eficiência, eficácia e satisfação na comunicação.

7.4 Alternativa

Uma solução proposta neste projeto de conclusão de curso sugere informatizar o procedimento de comunicação interna. Esta nova modalidade de comunicação será denominada como Comunicação Eletrônica (CE).

7.5 Descrição dos Envolvidos (Atores)

Segue abaixo descrição dos envolvidos.

Nome	Descrição	Responsabilidades

Usuário Comum	Qualquer funcionário efetivo, comissionado ou terceirizado que utilizará o sistema no âmbito da mesma organização.	- Emitir CE - Encaminhar CE - Acompanhar CE;

Tabela 2 – Descrição dos Envolvidos

7.6 Ambiente dos Envolvidos (Atores)

As Comunicações Internas são confeccionadas por meio eletrônico. Assim que o destinatário ficar ciente da comunicação, deve realizar as devidas providências necessárias para o cumprimento dos assuntos contidos no documento. Após estas tarefas estarem realizadas, informa ao emissor da comunicação a respeito dos seus resultados.

7.7 Principais Necessidades dos Envolvidos

Necessidade	**Prioridade**	**Solução Atual**	**Soluções Propostas**
Diminuir o tempo de encaminhamento das comunicações internas	Alta	Processo realizado pelos correios.	Possibilitar o envio eletrônico das comunicações para diminuição do tempo de emissão e ciência das mesmas.
Diminuir o custo com a comunicação	Alta	Processo realizado com documentos impressos.	Utilizar documentos eletrônicos.
Manter o histórico das comunicações, bem como dos documentos relacionados a estas.	Média	O arquivamento destes documentos não é padronizado e confiável.	Utilizar um repositório central das comunicações e documentos vinculados a estas. Observando que as CE terão prazos para ficarem arquivadas, respeitando a tabela de temporalidade vinculada ao assunto definido pela diretoria.

Identificar o destinatário da comunicação	Média	A obtenção dos dados do destinatário, como nome e setor é feita de maneira manual.	Utilizar o cadastro de usuários.
Controlar prazos para execução das tarefas descritas nas comunicações	Alta	Os prazos são controlados pelos envolvidos sem padronização ou controles evidentes.	Os prazos serão controlados pelo próprio programa que notificará os responsáveis a respeito do cumprimento dos mesmos
Acompanhar o andamento das comunicações	Média	O acompanhamento é realizado por diversos meios de comunicação como: e-mail, telefone, fax e outros	Utilizar a própria aplicação para informar a situação de cada comunicação emitida.

Tabela 3 – Principais Necessidades dos Envolvidos

7.8 Restrições

A solução tem como premissa que os usuários estejam devidamente cadastrados a fim de possibilitar a identificação e autorização do usuário(s) para acessar a aplicação.

A disponibilidade para uso da aplicação será realizada por partes, iniciando com um grupo restrito de pessoas.

7.9 Requisitos

7.9.1 Padrões Aplicáveis

A aplicação poderá se aderir aos padrões visuais utilizados pela organização.

7.9.2 Requisitos do Sistema

Fazer uma breve descrição do que são requisitos funcionais e requisitos não funcionais.

Requisito Funcional	Código
O sistema deve possuir uma interface para a emissão de uma comunicação eletrônica com os seguintes campos: • Para - destinatário da CE; • Notificado - pessoa que receberá a notificação quando o prazo de atendimento não for cumprido; • Prazo para atendimento/ prazo de publicidade; • Assunto; • Texto – texto da CE; • Arquivo(s) anexo(s); Os campos referentes à Data e Número da CE serão gerados pelo sistema.	RF01
O sistema deve gerar automaticamente o número da CE com a seguinte formatação: • SiglaDaOrganização-DDDDD/AAAA, onde Sigla da Organização é a organização a qual pertence o emissor, DDDDD é um número seqüencial gerado e que é iniciado do zero a cada ano. e AAAA é o ano vigente. Exemplo: UFSC-00001/2007	RF02
O sistema deve permitir anexar arquivos em uma comunicação eletrônica. Os arquivos anexados devem, necessariamente, possuir os formatos: pdf, doc, xls, xlx, sxw, zip, txt, ppt e odp. O tamanho máximo dos arquivos deve ser parametrizado.	RF03
Uma comunicação eletrônica do tipo solicitação será encaminhada apenas para um destinatário por vez, porém pode ser encaminhada diversas vezes. Uma CE poderá ser encaminhada em qualquer situação.	RF04
Uma CE não tem limite de destinatário e nem prazo de validade, apenas prazo para atendimento e prazo de publicidade.	RF05

Uma comunicação eletrônica possuirá as seguintes situações: A) Tipo solicitação • Encaminhada: quando um emissor origina uma comunicação, porém o destinatário não tem ciência da mesma; • Em atendimento: o destinatário teve ciência da CE e está providenciando a execução da solicitação descrita nela; • Em atraso/em atendimento: o prazo para cumprimento da solicitação contida na CE foi excedido e as tarefas ainda estão em execução; • Em atraso: o prazo para cumprimento da solicitação contida na CE foi excedido e o destinatário não tem ciência da CE; • Encerrada: o destinatário cumpriu as solicitações e encerrou a CE; • Cancelada: quando o seu emissor julgar conveniente cancelá-la, exceto quando estiver na situação Encerrada. B) Tipo informativa • Enviada: Quando o emissor origina uma CE e envia aos destinatários selecionados; • Suprimida: Depois de vencer o prazo de publicidade da CE. Esta não será mais apresentada aos usuários. • Cancelada: quando o seu emissor julgar conveniente cancelá-la, exceto quando estiver na situação Suprimida.	RF06
O sistema deve apresentar na tela inicial definida como "Hoje", uma lista das comunicações ordenadas pelas seguintes situações: • enviada; • encaminhada; Esta lista conterá os seguintes campos: • número; • assunto; • emissor;	RF07
O sistema deve possuir uma interface para visualização das CE emitidas e recebidas conforme os critérios de ordenação e campos do requisito funcional sete (RF07). O usuário receptor, ao visualizar a CE pela primeira vez, faz com que a mesma seja registrada, com a data e hora da ação, de sua leitura.	RF08
O sistema deve possuir uma interface para o encaminhamento de uma comunicação eletrônica apresentando todo seu conteúdo, emissão e encaminhamentos e os seguintes campos: • Para - destinatário da CE; • Texto; • Arquivo(s) anexo(s);	RF09
Ao expirar o prazo de atendimento da CE e a mesma, não tiver sido encerrada, faz com que o sistema notifique por e-mail a respectiva pessoa determinada na emissão para esta finalidade.	RF10

O sistema emitirá CE de dois tipos: • Solicitação: quando destina-se a realização de alguma tarefa. • Informativa: apenas dar ciência de um determinado assunto.	RF11
Quando a CE for do tipo informativa, o prazo para atendimento será substituído por prazo de publicidade. Vencida esta data, a CE sairá de circulação, porém seu emissor poderá ter acesso a esta. Para este tipo de CE será permitido selecionar como destinatário um e-mail do tipo lista ou vários destinatários.	RF12
O sistema deve realizar consultas nas CE por assunto, palavra chave no texto e período. As CE em questão serão apenas aquelas as quais o usuário teve participação	RF13
O sistema deve permitir aos usuários administradores do sistema emitir relatórios contabilizando as comunicações eletrônicas por situação, por setor e período de inicio e fim.	RF14
Uma CE poderá ser cancelada unicamente pelo emissor em todas as situações exceto "encerrada", os motivos para tal procedimento devem ser justificados.	RF15
Ao encerrar uma CE o usuário deverá expor os motivos de encerramento e definir se foi atendida ou não.	RF16
As CE poderão ser reabertas apenas pelos seus emissores, expondo motivos para tal procedimento. Um novo encaminhamento da CE original para qualquer destinatário poderá ser feito.	RF17
O sistema irá emitir uma mensagem de notificação ao emissor de uma CE sempre que esta for encaminhada ou a sua situação for alterada.	RF18
Para as CE do tipo solicitação o sistema deve apresentar uma interface para pesquisa e seleção dos assuntos com os seguintes campos: • Palavra-chave; • Lista dos assuntos contendo as palavras-chaves informadas. Para comunicações do tipo informativo, o campo assunto será livre.	RF19

Tabela 4 – Requisitos Funcionais

Requisito Não Funcional	**Código**
O sistema estará disponível na Web e acessado por navegador.	RNF01
O sistema será desenvolvido na plataforma Java.	RNF02
O sistema utilizará o cadastro de usuários da base de dados existente.	RNF03

Tabela 5 – Requisitos Não Funcionais

7.9.3 Requisitos de Desempenho

O sistema deve atender os seguintes níveis de acesso:

- Número máximo de usuários geradores de CE: 500;
- Máximo de 50 usuários/hora;
  Máximo de 10 comunicações/minuto.

8 ARQUITETURA DO PROGRAMA

Esta seção apresenta a arquitetura escolhida na fase de análise do sistema em decorrência das características do mesmo. A arquitetura descrita detalhadamente nesta seção destina-se a orientação de todos os técnicos que se envolverem com o desenvolvimento ou manutenção do programa.

8.1 Escopo

A presente seção deste documento restringe-se a informar a arquitetura do programa não abordando questões relativas às regras de negócio e construção. Entretanto, as definições aqui contidas têm influência direta ou indireta sobre estas questões. Não é intenção descrever detalhadamente ou ensinar sobre padrões bem como arquitetura de programas, e sim enumerar os padrões utilizados, suas responsabilidades e seus relacionamentos de maneira a compor toda a arquitetura.

8.2 Definições, acrônimos e abreviações.

Na descrição desta arquitetura são utilizados os seguintes termos e suas respectivas definições:

- **API** – (Application Programming Interface): definição de um conjunto de rotinas e padrões que permitem a um programa acessar um conjunto de funcionalidades especializadas;
- **CI** – Mensagens que este sistema propõe-se a gerenciar;
- **DAO** – (Data Access Object): categoria de objetos relacionados somente ao acesso a dados;
- **PADRÕES DE PROJETO** – Padrões para desenvolvimento de programas elaborados segundo boas práticas e ampla experiência;
- **DTO** – (Data Transfer Object): Categoria de objetos responsáveis somente para o transporte de dados;
- **J2EE** – (Java 2 Platform, Enterprise Edition): versão da plataforma Java para desenvolvimento de aplicações corporativas;
- **JDBC** – (Java DataBase Connectivity): API utilizada pela linguagem Java para acesso a banco de dados;
- **MVC** – (Model View Control): padrão utilizado para dividir a aplicação em camadas;
- **RDBMS** – (Relational Database Management System): servidor de banco de dados relacionais;
- **SINGLETON** – padrão de projeto para criação (instanciação) de objetos;
- **SMTP** – (Simple Mail Transfer Protocol): protocolo para envio de mensagens por correio eletrônico.

8.3 Visão Geral

As próximas seções estão organizadas da seguinte maneira:

- Representação da Arquitetura: descreve qual é a arquitetura do programa do sistema atual e como ela é representada;
- Metas e Restrições de Arquitetura: descreve os requisitos do programa e os objetivos que têm um impacto significativo na arquitetura;
- Visão de Casos de Uso: esta seção lista os casos de uso ou cenários do modelo de casos de uso se eles representam uma

funcionalidade central e significativa do sistema final ou se têm uma ampla cobertura de arquitetura;

- Visão Lógica: Esta seção descreve as partes significativas do ponto de vista da arquitetura do modelo de design, como sua divisão em subsistemas e pacotes;
- Visão de Implementação: Esta seção descreve a estrutura geral do modelo de implementação, a divisão do programa em camadas e subsistemas;
- Qualidade: Uma descrição de como a arquitetura do programa contribui para todos os recursos (exceto a funcionalidade) do sistema.

8.4 Representação da Arquitetura

A arquitetura adotada foi baseada nos seguintes Padrões de Projeto: Singleton, DAO, DTO, MVC, Iversion of Control e View Helper. Além dos padrões, empregou-se a programação orientada a aspectos. Para ilustrar como estes padrões funcionam e se interagem, serão utilizadas as seguintes visões: de Casos de Uso, Lógica e de Implementação.

Singleton é um padrão de instanciação, ou seja, criação de objetos que orienta como criar uma instância única de um objeto para toda aplicação. Por exemplo, a criação de conexões com o banco de dados deve ser controlada por apenas um objeto a fim de evitar excesso de conexões desnecessárias.

Todo acesso a dados persistidos como um RDBMS ou um sistema de arquivos dever ser feito por um conjunto de classes especializadas para este propósito. Estas classes seguem o padrão DAO. Isto se faz necessário para isolar as demais classes da aplicação de aspectos relacionados com estes recursos. Assim, na eventualidade de alteração do acesso aos dados persistidos, como por exemplo, a troca do RDBMS, implicaria num o impacto restrito somente as classes com estas responsabilidades.

As classes que implementam o DTO são classes de dados. São responsáveis em transportar os dados através das diversas camadas da aplicação.

MVC é um padrão muito utilizado em aplicações no ambiente Web. Sua finalidade é separar a aplicação em três camadas bem definidas. A camada de modelo (M) possui a implementação do modelo de domínio, ou seja, da lógica do negócio. A camada View (V) compõe a parte de apresentação do aplicativo, ou seja, as interfaces homem-máquina. Elas serão implementadas em JSP com utilização de Taglibs padrões do JSTL 1.1 e Struts. Por último, tem-se a camada de controle (C) que é responsável pelo controle do fluxo da aplicação. Todas as requisições vindas dos clientes são centralizadas em um Servlet responsável em redireciona-las para os respectivos objetos de acordo com a tarefa a ser executada. Será utilizado o Framework Struts para implementação deste padrão (CAVANESS, 2003).

Para a associação de objetos é utilizado o padrão Inversion of Control no qual define que a associação seja por agregação ou por composição é realizada pela referência da instância do objeto (FOWLER, 2004). Assim os objetos não são criados dentro da classe que recebe a dependência. Para implementação deste padrão será utilizado o Framework Spring que permite criar as associações por declaração.

O padrão View Helper é utilizado para criar classe auxiliares para camada de apresentação (View), como formatadores de texto.

A orientação a aspecto é utilizada para tratar aspectos do programa. Entendam-se como aspectos características presentes nas suas diversas funcionalidades como segurança, auditoria e controle transacional. Neste programa utiliza-se este recurso para controle de todas as transações do banco de dados. Utiliza-se o Framework Spring para implementação deste Pattern.

8.5 Metas e Restrições da Arquitetura

Toda a arquitetura descrita neste sistema está direcionada para o desenvolvimento do programa a ser acessado no ambiente Web sob a plataforma Java.

8.6 Visão de Caso de Uso

Para ilustrar as características da arquitetura será utiizado um caso de uso que abrange todas estruturas do programa, o UC01 – Emitir CI.

Esta funcionalidade esta disponível para gerar uma comunicação interna, portanto é necessário definir o destinatário da CI, qual o tempo de resposta necessário para a CI, quem será notificado na eventualidade de não ser comprido o prazo para resposta e a mensagem da CI propriamente dita. Depois de registrar estas informações, será enviado e-mail ao destinatário da CI notificando-o a respeito da mesma.

É importante estar claro que a CI não é um e-mail e sim uma mensagem que permite anexar documentos e fica armazenada em um banco de dados. O acesso a estas informações será realizado exclusivamente pelo sistema sendo o e-mail apenas uma forma de notificar o destinatário.

Outra forma possível de notificação poderia ser o envio de uma mensagem para o celular do destinatário.

8.7 Realização de Caso de Uso

Para emitir uma CI um usuário aciona o item do menu “Emitir CI”. Esta requisição é submetida ao Servlet controlador responsável em gerenciar o fluxo da aplicação fazendo com que o usuário seja direcionado para a tela de edição da nova CI.

Depois de preenchido todos os campos necessários, o usuário aciona o botão de conclusão. Esta requisição é enviada para o Servlet controlador que redireciona para o objeto que implementa esta ação. Este objeto é do tipo FormAction do Struts e é o responsável por validar os dados submetidos, criar o DTO que transportará estes dados e acionar o objeto que implementa a lógica de negócio (CAVANESS, 2003). A validação feita neste momento é

apenas de tipos de dados, isto é, data, inteiro, alfabético e outros. As validações de negócio são realizadas pelos objetos de negócio.

O objeto de negócio realiza as tarefas necessárias para emissão da CI. Entre estas tarefas esta a inclusão de dados da CI no banco que é realizada pelos DAOs. Para isto, o objeto de negócio repassa ao DAO os dados através de DTOs.

Finalizada a lógica de negócio, o fluxo da aplicação é redirecionado para a tela que lista todas as CIs emitidas pelo usuário.

Para facilitar o entendimento do procedimento descrito acima, será apresentado o mesmo, no diagrama de seqüência mostrado abaixo.

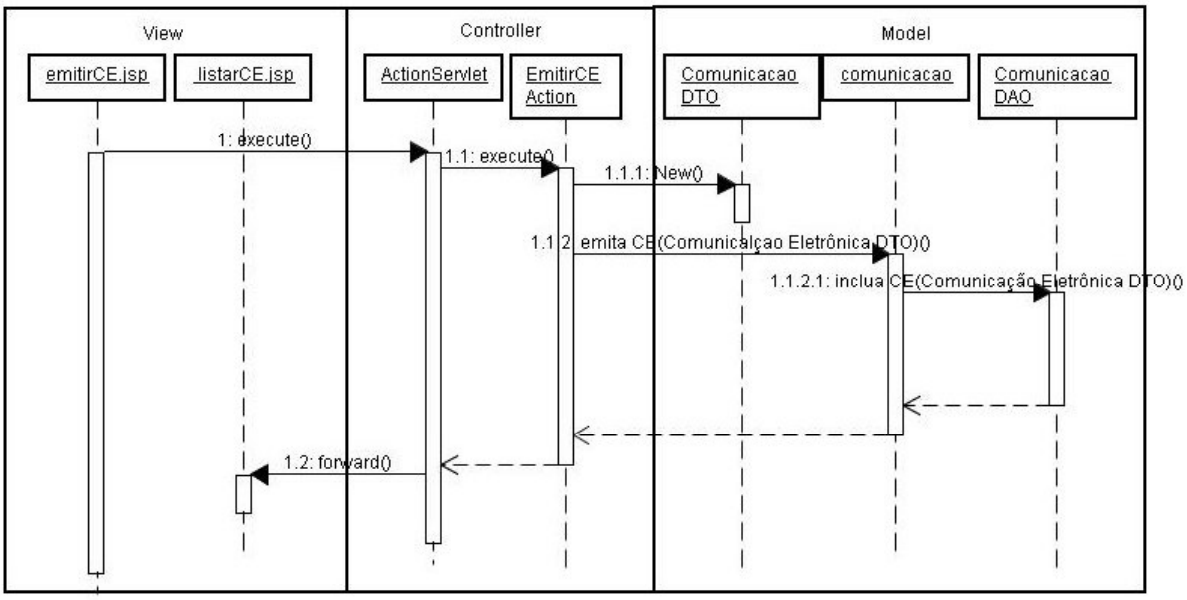


Figura 3 – Diagrama de Seqüência Modelo

8.8 Visão Lógica

Os diversos pacotes e classes da aplicação estão organizados de acordo com as camadas e padrões adotados. Alem das camadas apresentadas pelo Padrão MVC, existe a subcamada composta pelo Padrão DAO necessária para definir bem o isolamento dos demais recursos da camada Model. A região

denominada Common não constitui uma camada, mas sim recursos comuns a toda aplicação como o Log e DateUtil dentre outros.

8.9 Pacotes Significativos do Ponto de Vista da Arquitetura

A figura abaixo ilustra a organização dos pacotes da aplicação:

Figura 4 – Pacotes da Aplicação

Segue a descrição dos pacotes.

Pacote	Descrição
br.ufsc.inf.ce.common	Contém classes utilizadas por vários objetos do Sistema de Comunicação Eletrônica.
br.ufsc.inf.ce.exception	Contém classes de exceção do Sistema de Comunicação Eletrônica.
br.ufsc.inf.ce.model.business	Contém objetos do domínio do negócio. Podem ser subdivididos dependendo do número de classes geradas.
br.ufsc.inf.ce.model.dão.jdbc	Contém objetos de persistência em banco de dados.
br.ufsc.inf.ce.model.dto	Contém objetos de dados que seguem o padrão DTO.
br.ufsc.inf.ce.web.action	Composto pelas classes do tipo ActionForm.
br.ufsc.inf.ce.web.helper	Composto pelas classes que implementam o padrão View Helper
br.ufsc.inf.ce.web.servlet	Composto por classes que implementam a especificação Servlet.

br.ufsc.inf.common	Contém classes utilizadas por várias aplicações e não somente o Sistema de Comunicação Eletrônica. Pode-citar formatadores de data e texto.
br.ufsc.inf.exception	Contêm classes de exceção utilizadas por várias aplicações e não somente o Sistema de Comunicação Eletrônica.

Tabela 6 – Descrição dos Pacotes

8.10 Visão de Processos

Como esta é uma aplicação destinada ao ambiente web, exige um servidor de aplicação (Container) que suporte a especificação JSP 2.0 e Servlet 2.4 para ser implantado e executado. As tarefas relacionadas ao carregamento da aplicação serão providas pelo próprio container. Basicamente, o programa possui duas instâncias de servlets que são iniciados com a aplicação. O primeiro é o controlador (ControllerServlet) cuja função foi descrita nas seções acima. O outro é utilizado exclusivamente para gerar um processo paralelo a execução do programa para notificar por e-mail sobre o prazo de expiração da conclusão da CI. Este processo deve ser mantido enquanto a aplicação estiver disponível.

8.11 Visão de Implantação

Abaixo é apresentado o diagrama de implantação do programa. O programa será implantado como um contexto denominado “CE” (Comunicação Eletrônica). Um servidor é responsável pelo serviço de web compostos pelo Apache em conjunto com o Tomcat. Em outro servidor estão instalados os demais serviços.

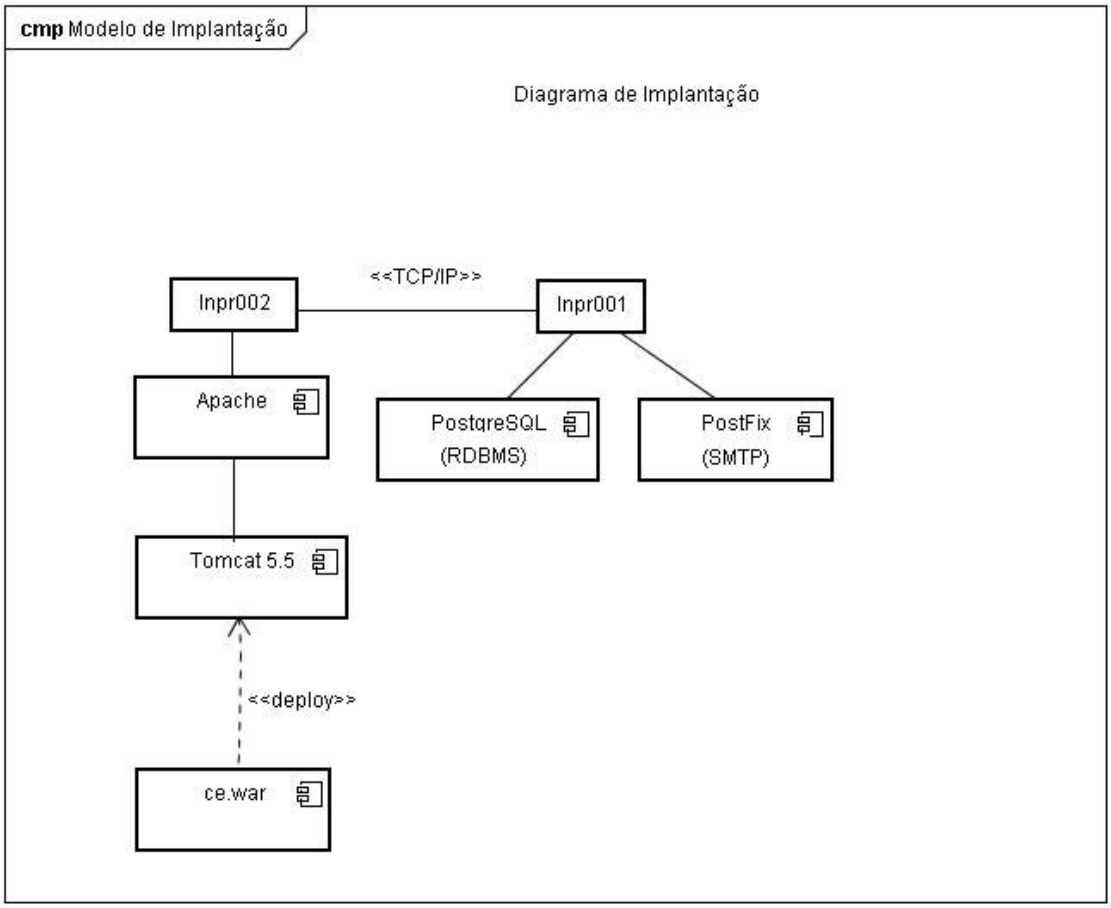


Figura 5 – Diagrama de Implantação

Na seqüência é mostrada a estrutura física do contexto “CE” que fisicamente é denominado ce.war:

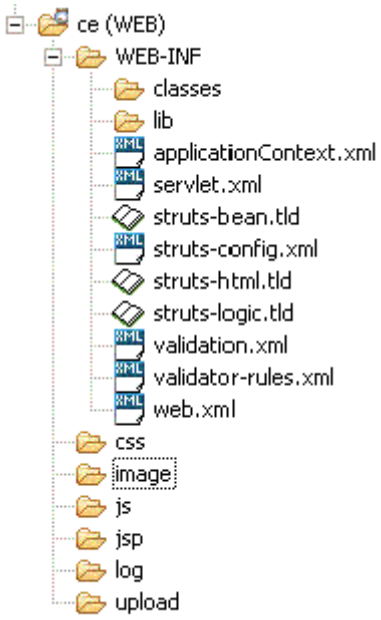


Figura 6 – Estrutura Física do Sistema

A estrutura dos diretórios é descrita na próxima tabela:

Diretório	Descrição
WEB-INF	Diretório exigido pela especificação J2EE. Contém o subdiretório classes onde são localizados os pacotes com classes da aplicação, e lib onde estão os componentes necessários para aplicação. Estão presentes também os arquivos de configuração.
Css	Diretório onde estão os arquivos de estilo.
Image	Diretório de arquivos de imagem utilizados nas páginas.
Js	Diretório das bibliotecas de JavaScript.
Jsp	Diretório onde estão todas as páginas HTML e JSP.
Log	Diretório onde se encontram os arquivos de log da aplicação.
Upload	Diretório onde estão os arquivos que serão enviados para o computador pessoal do usuário

Tabela 7 – Estrutura dos Diretórios

8.11 Visão de Implementação

A aplicação esta dividida em quatro camadas: Controller, View, Model e DAO. Estas camadas se comunicam segundo o diagrama de robustez abaixo.

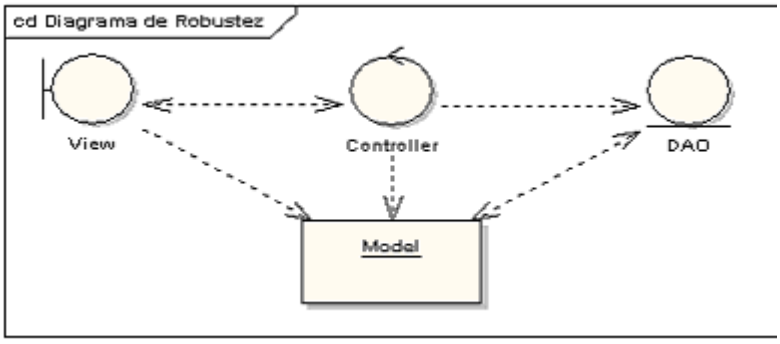


Figura 7 – Diagrama de Robustez

A camada View submete suas requisições para Controller e utiliza objetos DTO de Model para apresentar valores nas telas. Controller instancia objetos da camada Model para realização de funcionalidades, utiliza objetos DAO apenas para consulta de dados, e retorna objetos e o fluxo da aplicação para View. A camada Model utiliza objetos DAO para persistência de dados, e seus objetos DTO são utilizados por todas as camadas.É importante perceber que não é permitido à camada View comunicar-se diretamente com a camada de persistência (DAO).

No diagrama seguinte são apresentadas as relações de dependência entre diversos pacotes da aplicação, as páginas JSP e os componentes utilizados.

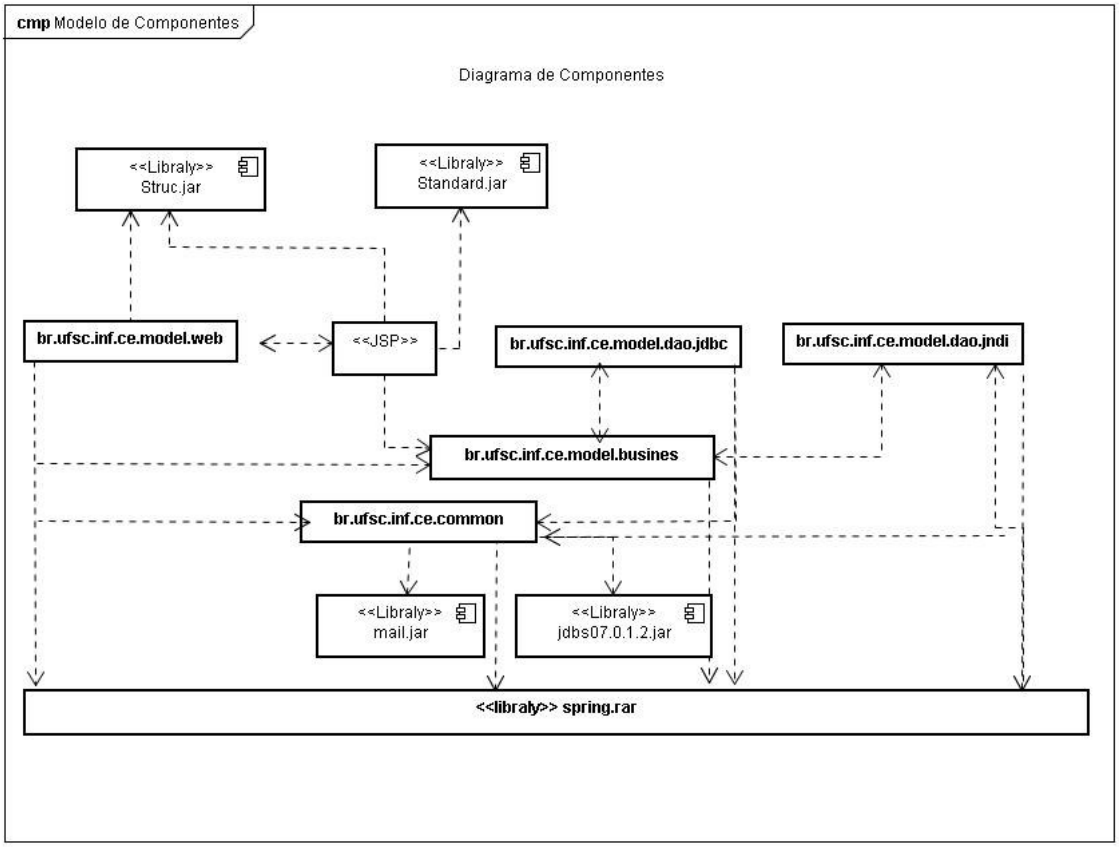


Figura 8 – Diagrama de Componentes

8.12 Camadas

Na tabela abaixo são descritas as características de cada camada:

Camada	Composição	Dependência
View	- páginas HTML e JSP; - arquivos de imagens JPG, GIF e PNG; - arquivos de estilo CSS; - arquivos JavaScript JS;	-**standard.jar**: biblioteca de tags padrão JSTL 1.1 da Jakarta; -**struts.jar**: biblioteca de tags do Struts 1.2; -**struts-bean.tld, struts** **-html.tld e struts-logic.tld**: arquivos de validação de taglibs do Struts; -objetos DTO da camada Model.

Controlller	- classes que herdam Action e ActionForm do Struts; -classes do tipo Servlet.	-**struts.jar**: framework Struts 1.2; -**struts-config.xml**: arquivo de configuração do Struts; - **validation.xml e validator-rules.xml**: arquivos de configuração de validação do Struts; -objetos da camada Model; -objetos da camada DAO.
Model	-classes de domínio do negócio.	-**applicationContext.xml e servlet.xml**: arquivos de configuração do Spring; -**mail.jar e activation.jar**: componentes para prover envio de mensagens (SMTP); -objetos da camada DAO.
DAO	-classes responsáveis pela manipulação de dados persistidos. Utilizam basicamente as APIs java.sql e org.springframework.jdbc	-**applicationContext.xml e servlet.xml**: arquivos de configuração do Spring; -**jdbc7.0-1.2.jar**: driver JDBC para o banco PostgreSQL; -objetos DTO da camada Model

Tabela 8 – Descrição das Camadas

8.13 Visão de Dados

A aplicação possui uma fontes de dado: a base de dados “cedb”. O registro de informações sobre as CEs emitidas ficarão mantidas nesta base de dados relacional e manipuladas via JDBC.

8.14 Qualidade

A arquitetura definida acima possui um elevado grau de maturidade, pois foi concebida segundo padrões de fato definido por consagrados desenvolvedores, especialistas em Orientação a Objeto, na plataforma Java e no ambiente Web. Os Frameworks utilizados estão adotados pelo mercado e apresentam facilidade de utilização se comparados aos padrões J2EE.

9 LEVANTAMENTO DE REQUISITOS - CASOS DE USO

9.1 Especificação de Caso de Uso Emitir CE

9.1.1 Breve Descrição

A emissão da Comunicação Eletrônica (CE) é o fato que dá origem a uma CE. Esta pode ter a finalidade apenas informativa ou ser uma solicitação. Neste último caso, o gerente deve informar o prazo para execução da tarefa que corresponde à solicitação através do campo Data para Atendimento. Para a CE do tipo informativa, deverá ser informado o prazo para exclusão da mesma. O número da CE é criado pelo sistema e segue o padrão “NomeDaOrganização-DDDDD/AAAA” onde as letras D e A são dígitos de zero a nove representando respectivamente um número composto de cinco dígitos e o ano composto por quatro dígitos. Vale ressaltar que também existe a possibilidade de anexar arquivos a uma CE.

Abaixo é apresentado o diagrama dos casos de uso da aplicação demonstrando as relações entre os casos de uso com este em questão.

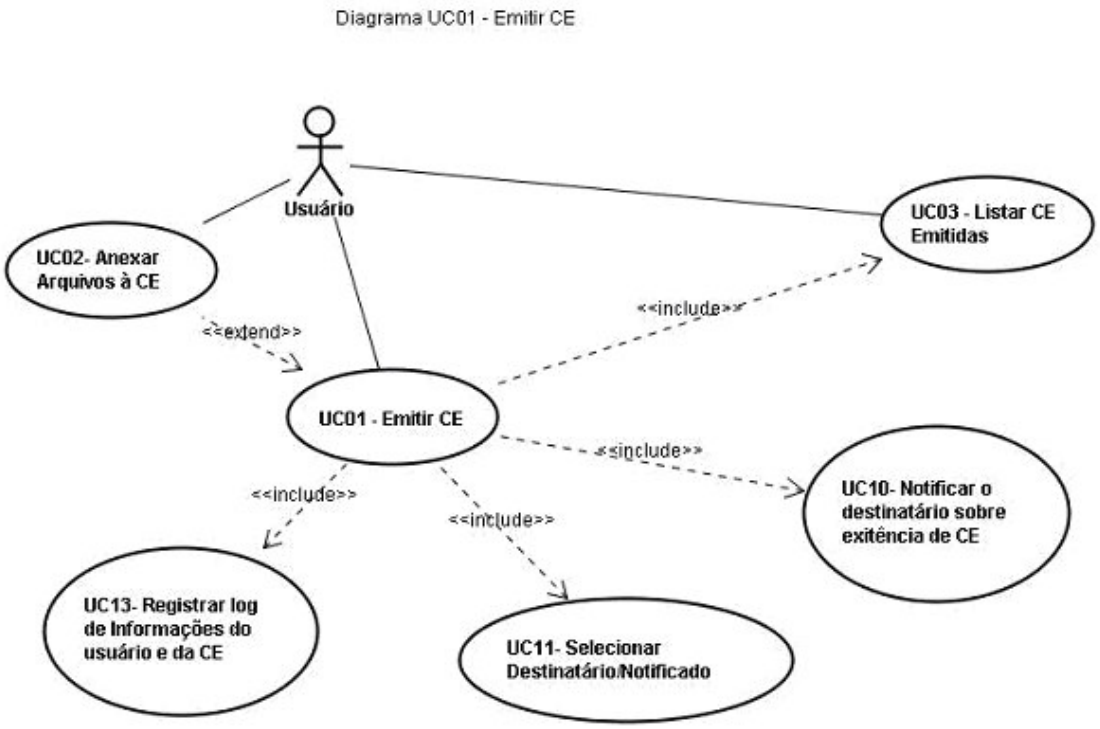


Figura 9 – Diagrama UC01 Emitir Comunicação

9.1.2 Fluxo de Eventos

9.1.2.1 Fluxo Básico

O usuário aciona o menu "Emitir CE"

Mostrar formulário com os dados necessários para emitir CE

20. O usuário seleciona o tipo de CE

30. O usuário aciona o botão "Para"

40. Executar UC11 Selecionar Destinatário/Notificado

CE tipo solicitação?

[sim]

50. O usuário aciona o botão "Notificado"

60. Executar UC11 Selecionar Destinatário/Notificado

70. O usuário informa a data e hora para atendimento da CE

80. O usuário seleciona o assunto

90. O usuário redige o texto

Anexar arquivos?

[sim]

100.Executar UC02 Anexar Arquivos à CE

Concluir?

[sim]

110.O usuário aciona o botão "Concluir"

120.Verificar todos campos obrigatórios

[ok]

130.Validar campos digitados

[ok]

140.Gerar número da CE segundo padrão "NomeDaOrganização-NNNN/AAAA"

150.Gravar dados da CE

160.Executar UC10 Notificar o destinatário sodre existência de CE

170.Executar UC03 Listar CE Emitida

9.1.2.2 Fluxos Alternativos

CE tipo solicitação?

[não]

50 - 70. O usuário informa a data e hora para exclusão da CE

Anexar arquivos?

[não]

100.Não executar UC02 Anexar Arquivos à CE

Concluir?

[não]

110.O usuário aciona o botão "Cancelar"

120 - 160. Avançar até a atividade 17

120. Verificar todos campos obrigatórios

[não]

130.Mensagem "O campo XXX deve ser preenchido"

140 -170. Retornar à atividade 2

130. Validar campos digitados

[não]

140.Mensagem "O campo XXX é inválido"

150 - 170. Retornar à atividade 2

Esta seqüência de eventos é mostrada pelo diagrama abaixo.

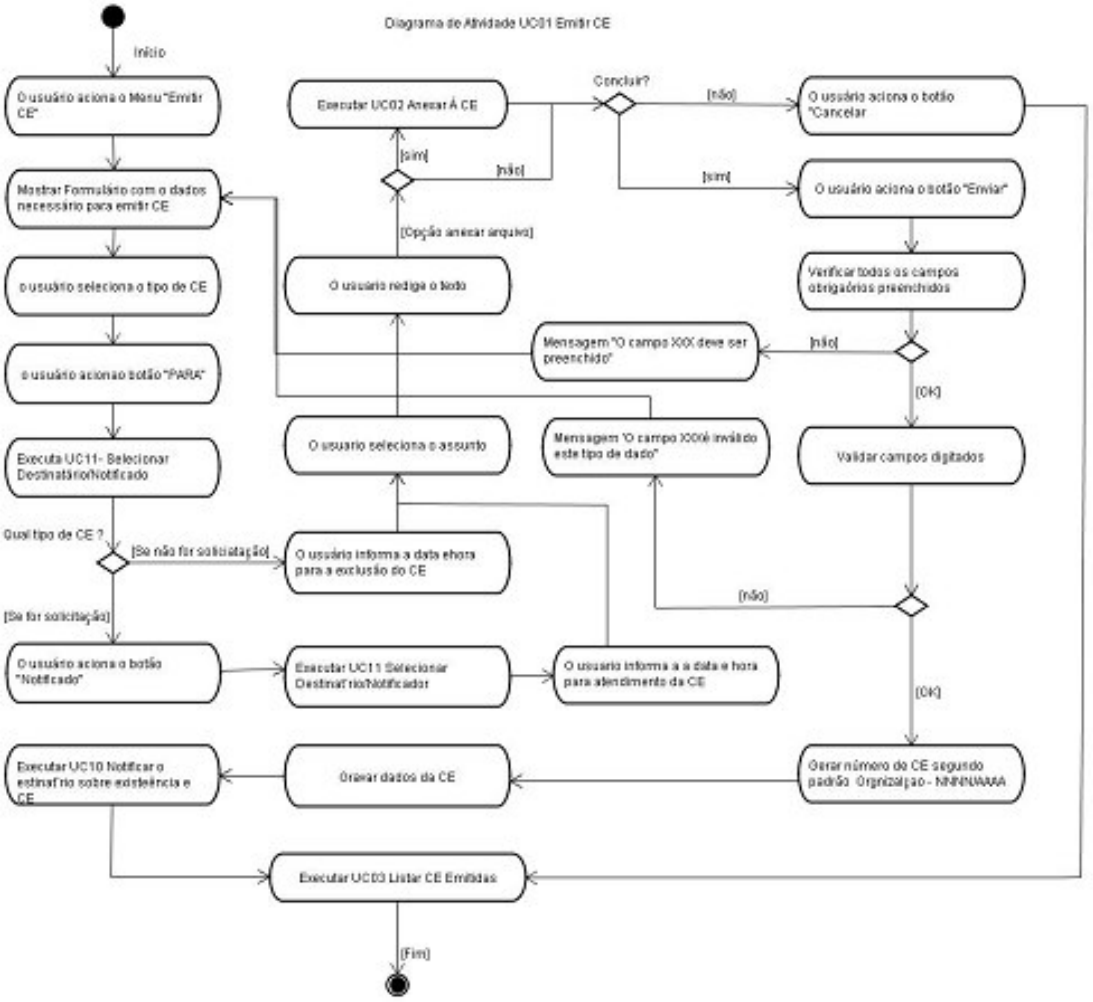


Figura 10 – Diagrama de Atividade UC01 Emitir Comunicação

9.1.3 Condições Posteriores

9.1.3.1 Registrar as Informações da CE

Depois de concluir o processo de emissão da CE, seus dados devem estar persistidos em um banco e a contagem do prazo para cumprimento da solicitação deve ser iniciada.

9.1.4 Pontos de Extensão

9.1.4.1 Anexar arquivos a CE

Ao emitir uma CE, o gerente pode optar por anexar arquivos a esta. Esta funcionalidade é realizada pelo caso de uso UC02 Anexar Arquivos à CE.

9.2 Realização de Caso de Uso Emitir CE

Esta seção demonstra como um caso de uso é realizado pelo programa, ou seja, quais os recursos e o fluxo de informações necessárias para implementação do caso de uso em questão. Entendam-se como recursos interfaces de interação homem máquina, classes, regras de negócios, regras de validação, recuperação, alteração, exclusão e inclusão de dados persistentes.

9.3 Escopo

Esta seção restringe-se a fornecer informações de alto nível para os desenvolvedores dos recursos envolvidos no caso de uso bem como seus relacionamentos. Não é intenção desta seção descrever características detalhadas para implementação do software como padrões de codificação, forma de validações, sentenças de banco de dados relacionais e outros.

9.4 Visão Geral

Esta seção apresenta uma descrição textual de como o caso de uso é realizado em termos de objetos de colaboração. Sua principal finalidade é resumir os diagramas vinculados ao caso de uso e explicar como eles estão relacionados.

9.5 Fluxo de Eventos

No diagrama de seqüência abaixo é apresentado como ocorre o fluxo de eventos do caso de uso.

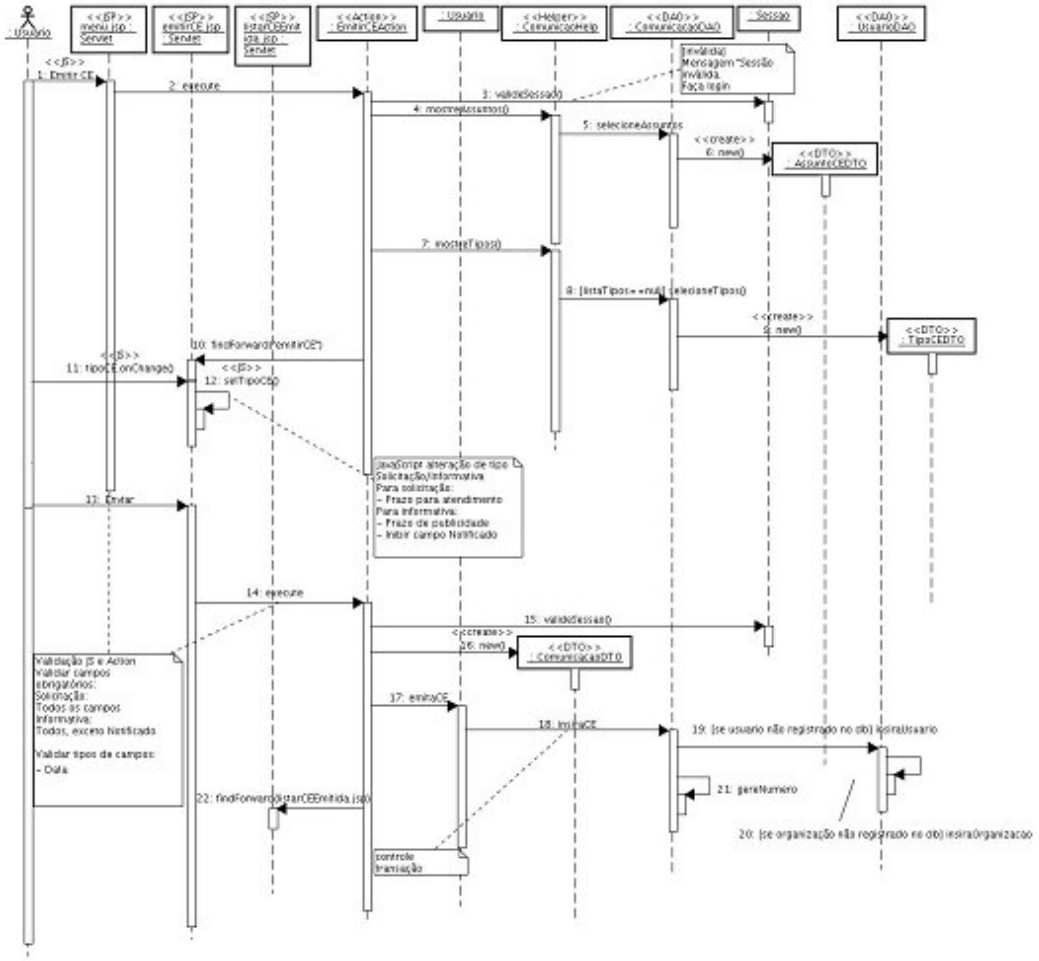


Figura 11 – Diagrama de Seqüência UC01 Emitir Comunicação

9.6 Especificação de Caso de Uso Anexar Arquivos a CE

9.6.1 Breve Descrição

Opcionalmente, durante a emissão e encaminhamento de uma CE, um usuário pode anexar arquivos a esta. Os arquivos anexados devem ter obrigatoriamente os formatos PDF, DOC, XLS, XLX, SXW, ZIP e TXT, e possuir tamanho máximo parametrizado. Será permitida a inclusão de 20 arquivos por CE. Estes arquivos serão gravados em disco e registros no banco de dados apenas farão referência a estes.

Abaixo é apresentado o diagrama dos casos de uso da aplicação demonstrando as relações entre os casos de uso com este em questão.

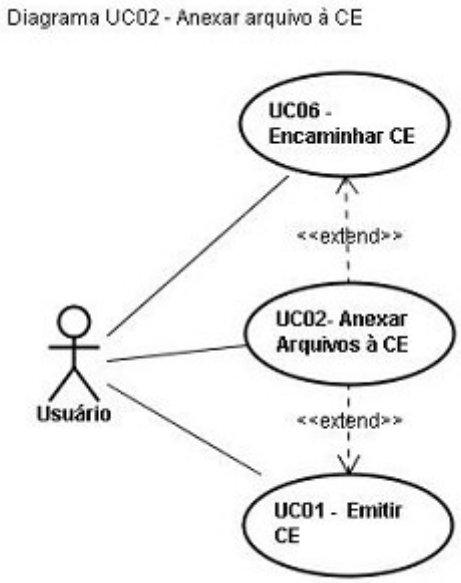


Figura 12 – Diagrama UC02 Anexar Arquivo a Comunicação

9.6.2 Fluxo de Eventos

9.6.2.1 Fluxo Básico

Mostrar formulário com os dados necessários para emitir CE

1. O gerente aciona o botão "Procurar"
2. Mostrar a tela para seleção do arquivo
3. O usuário seleciona o arquivo
4. Apresentar o caminho completo do arquivo no respectivo campo

Adicionar ou remover arquivo?

[não]

Concluir?

[sim]

O usuário aciona o botão "Enviar"

Validar tamanho e formato dos arquivos

Tamanho válido?

[sim]

Formato válido?

[sim]

Gerar nome do arquivo segundo o padrão "NomeDaOrganização-NNNN/AAAA_NNN
Gravar arquivos no diretório "files" da aplicação
Gravar dados do arquivo
Ação?
[Emitir]
Executar UC03 Listar CE Emitidas

9.6.2.2 Fluxos Alternativos

Adicionar ou remover arquivo?
[adicionar]
6. O usuário aciona do botão "+"
7. Adicionar um campo para arquivo
8 - 11 Retornar à atividade 2

Adicionar ou remover arquivo?
[remover]
6. O usuário aciona o botão "-"
7. Remover um campo para arquivo
8 - 11 Retornar à decisão “Adicionar ou remover arquivo?”

Concluir?
[não]
6. O usuário aciona o botão "Cancelar"
7 - 10. Avançar até a atividade 11

Tamanho válido?
[não]
8. Mensagem "O arquivo xxx possui xxx Kb e excede o tamanho máximo permitido (xxx Kb)."
9 - 11. Retornar à atividade 1

Formato válido?

[não]

8. Mensagem "O arquivo xxx possui formato inválido. Formatos válidos: pdf, doc, xls, xlx, sxw, zip e txt."

9-11. Retornar à atividade 1

Ação?

[Encaminhar]

11. Executar UC04 Listar CE Recebidas

Esta seqüência de eventos é mostrada pelo diagrama abaixo.

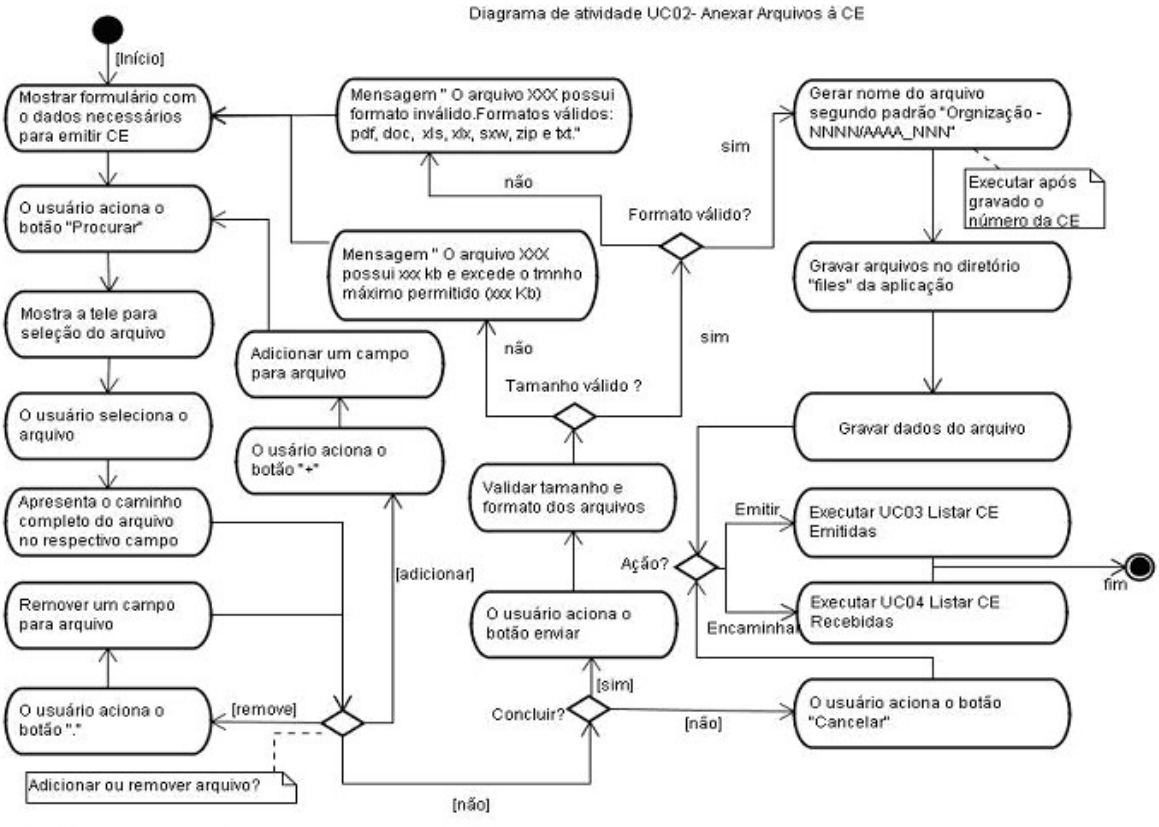


Figura 13 – Diagrama de Atividade UC02 Anexar Arquivos a Comunicação

9.6.3 Condições Posteriores

9.6.3.1 Registrar as Informações do Arquivo

Depois de concluir o processo de emissão ou encaminhamento da CE, os dados dos seus arquivos anexados devem ser persistidos no banco de dados.

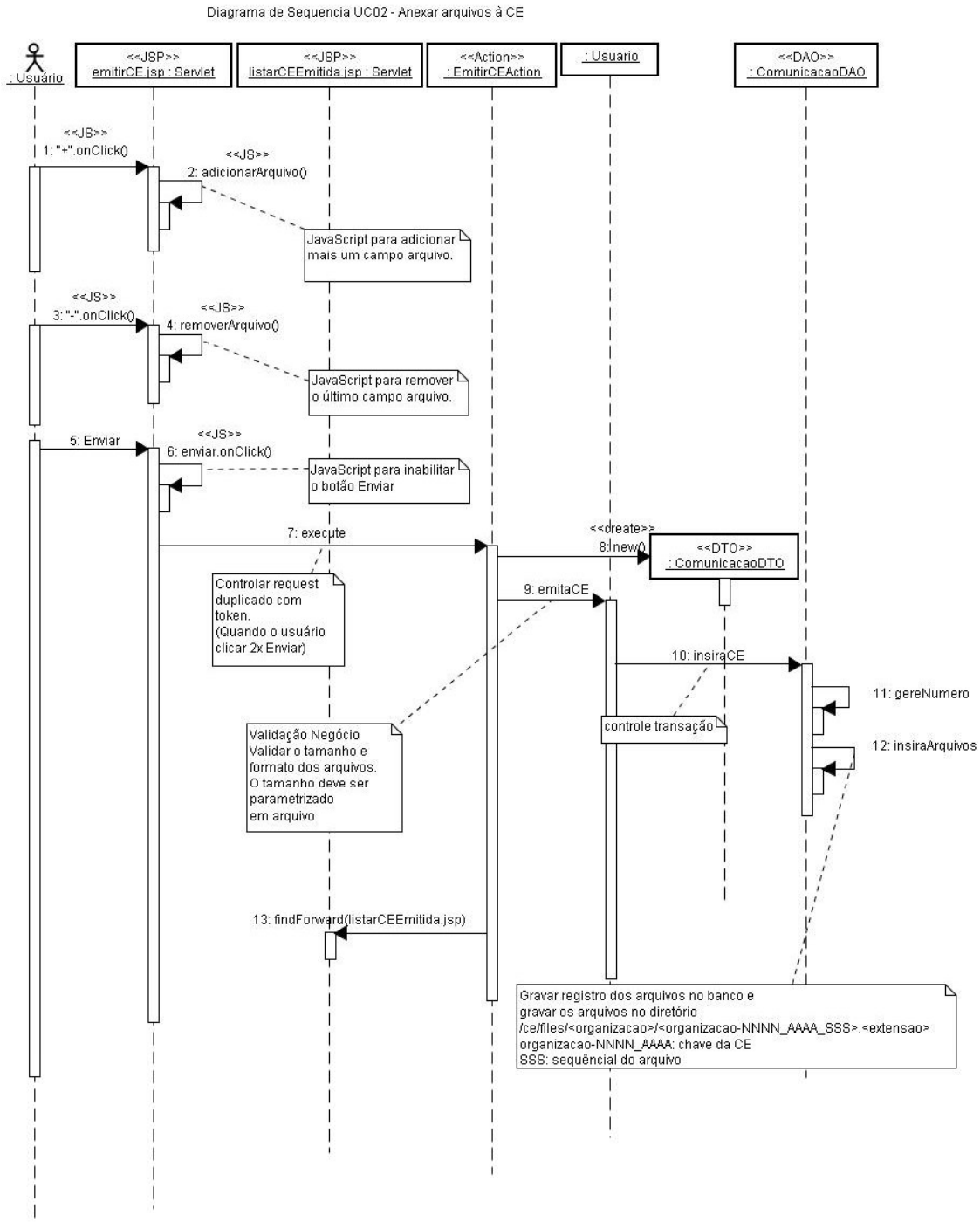
9.6.3.2 Armazenar o Arquivo

Depois de concluir o processo de emissão ou encaminhamento da CE, os seus arquivos anexados ser persistidos no sistema de arquivos e diretórios em disco.

9.7 Realização de Caso de Uso Anexar Arquivos

9.8 Fluxo de Eventos

No diagrama de seqüência abaixo é apresentado como ocorre o fluxo de eventos do caso de uso anexar arquivos tendo em visto o cenário Emitir CE.



No próximo diagrama, é apresentada a seqüência de eventos quando o cenário é Encaminhar CE.

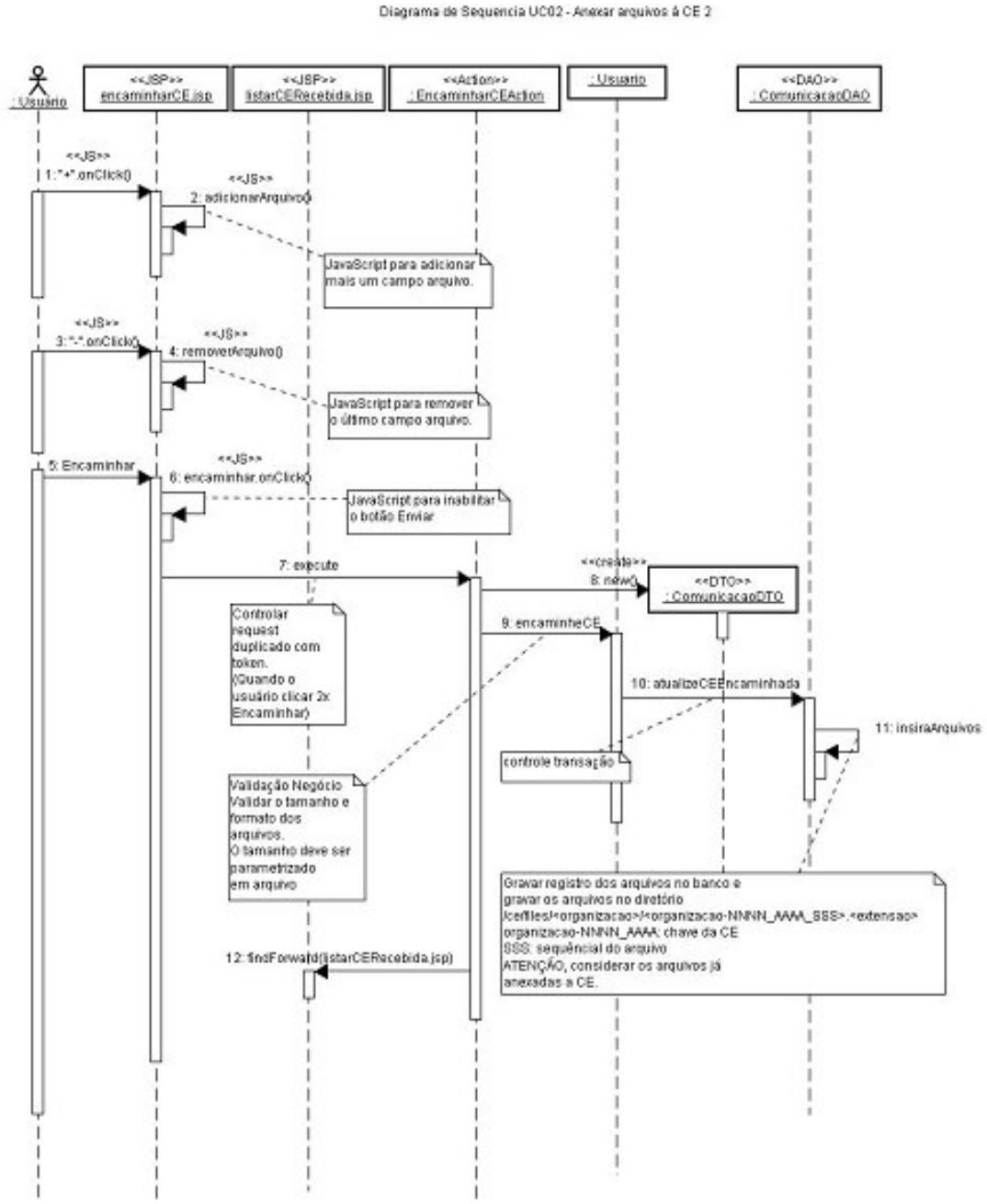


Figura 14 – Diagrama de Seqüência UC02 Anexar Arquivos a Comunicação

9.9 Especificação de Caso de Uso Anexar Arquivos a CE

9.9.1 Breve Descrição

Os usuários do sistema poderão listar todas as comunicações de sua autoria. Devido ao grande número de comunicações que podem ser originadas por um usuário emissor, esta lista será paginada, exibindo vinte comunicações

por página (este valor deve ser parametrizado) e serão separadas por ano. Esta separação se dará através de um menu e seus respectivos itens:

- “Comunicação Eletrônica”
    - **“Emitida” -**lista as comunicações emitidas pelo usuário no ano corrente;
    - 2004, lista as CE emitidas pelo usuário em 2004;
    - 2003, lista as CE emitidas pelo usuário em 2003

Os seguintes dados serão apresentados na lista:

- Número: no formato <sigla-organização>-<NNNNN>/<AAAA>;
- Situação: texto com a descrição da situação;
- Assunto: se possuir mais do que 35 caracteres, apresenta-los seguido de “...”;
- Destinatário: se possuir mais de um apresentar o primeiro seguido de “...”;
- Emitida em: data e horário de emissão.

A lista será apresentada pela seguinte ordem de situação:

- Em atraso;
- Em atraso/Em Atendimento;
- Enviada;
- Encaminhada;
- Em atendimento;
- Encerrada;
- Suprimida.

O usuário poderá navegar pelas páginas da lista acionando os itens “anterior”, “próximo” ou clicando sobre o número da página.

Abaixo é apresentado o diagrama dos casos de uso da aplicação demonstrando as relações entre os casos de uso com este em questão.

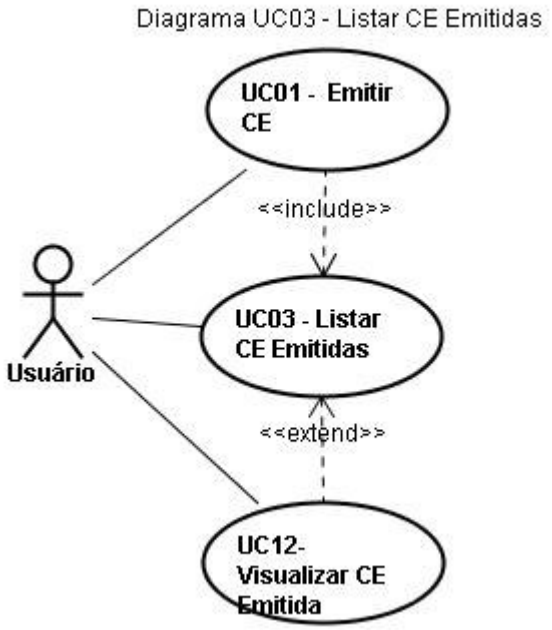


Figura 15 – Diagrama UC03 Listar Comunicações Emitidas

9.9.2 Fluxo de Eventos

9.9.2.1 Fluxo Básico

A seqüência de eventos é ilustrada pelo diagrama abaixo:

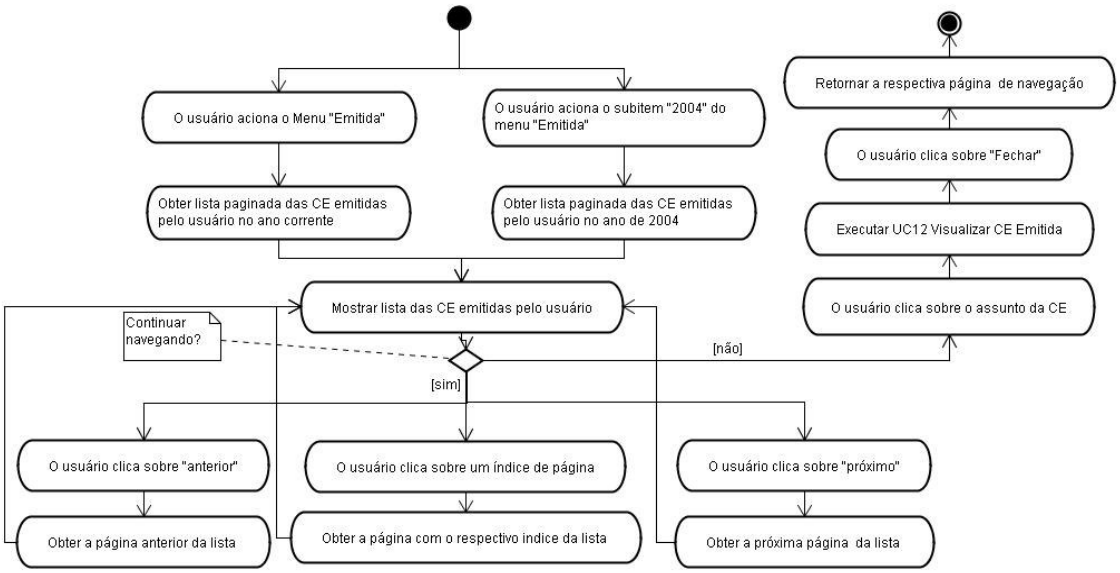


Figura 16 – Diagrama do Fluxo de Eventos

9.9.3 Condições Anteriores

9.9.3.1 O usuário deve possuir comunicações emitidas

Para listar as comunicações, o usuário deve necessariamente ter sido emissor delas.

9.9.3 Condições Posteriores

9.9.3.1 Persistir na Sessão do Usuário as Informações da Navegação

Ao visualizar uma comunicação da lista e retornar a esta, todos atributos necessários a navegação, devem ser recuperados.

9.9.4 Pontos de Extensão

9.9.4.1 Visualizar CE

No momento em que estiver listando as comunicações emitidas, o usuário pode optar por visualiza-las. Esta funcionalidade é realizada pelo caso de uso UC12 Visualizar CE Emitida.

9.10 Realização de Caso de Uso Listar Comunicações emitidas

9.11 Fluxo de Eventos

No diagrama de seqüência abaixo é apresentado como ocorre o fluxo de eventos do caso de uso listar comunicações emitidas.

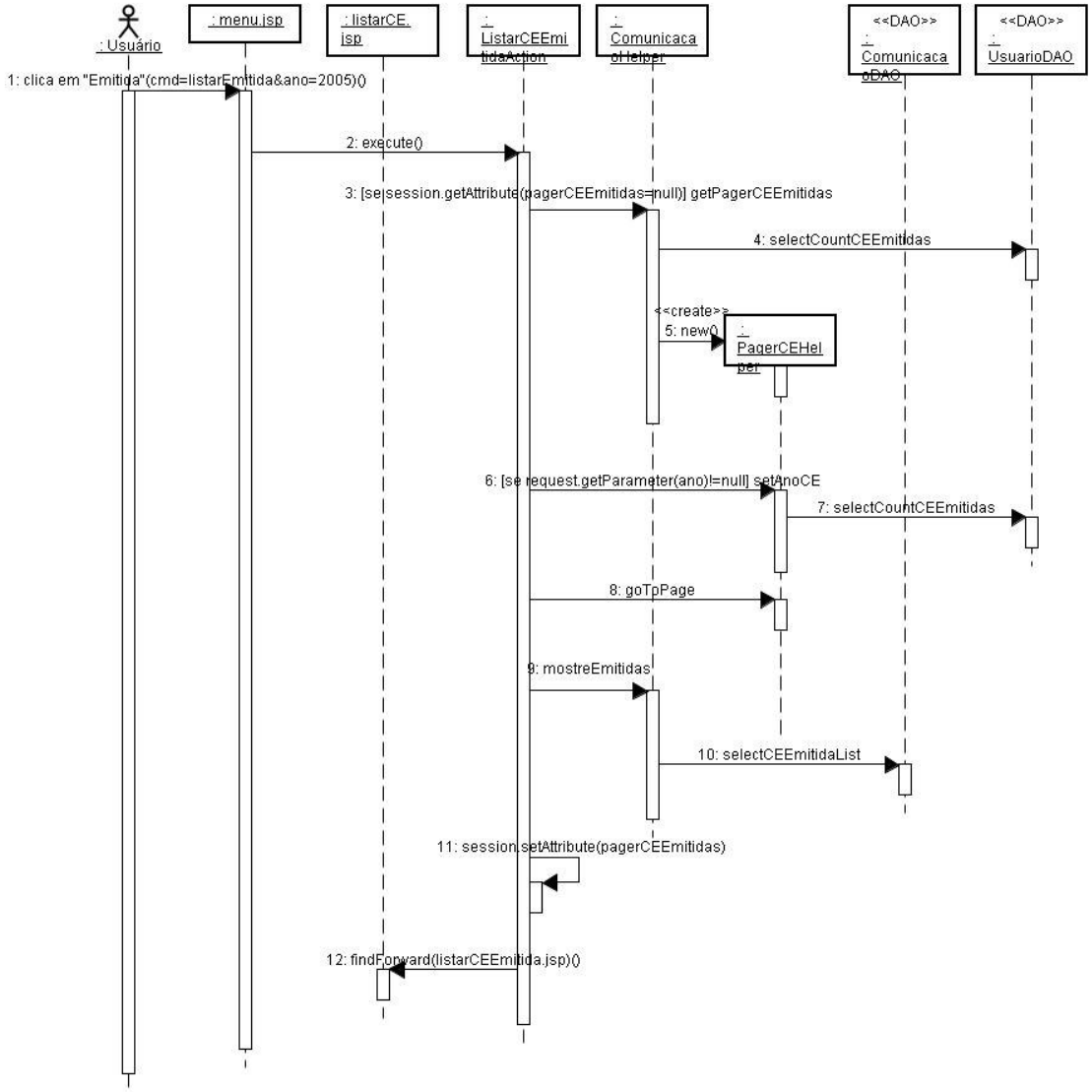


Figura 17 – Diagrama de Seqüência UC03 Listar Comunicações Emitidas

10 CONCLUSÃO

Diante dos objetivos propostos, pode-se perceber que esta monografia foi bem sucedida. Ao longo deste excelente trabalho foram descritos os principais passos no estudo dos levantamentos dos requisitos do sistema proposto.

Também houve uma boa definição do escopo de atuação que o sistema é capaz de atuar no processo de comunicação interna, explanando de forma clara e concisa acerca do sistema analisado.

Foram atendidos os objetivos que se referiam ao uso de padrões de desenvolvimento de software, consagrados no mercado de trabalho, dando uma visão geral na forma com que cada um foi utilizado no sistema proposto.

Neste projeto tem-se uma clara visão da intersecção de conhecimentos no que diz respeito às áreas de gerência de desenvolvimento de software, análise de sistemas e engenharia de software, conhecimentos estes adquiridos ao longo do curso de ciências da computação.

Outro importante aspecto desta monografia é o aprendizado adquirido com relação aos custos no processo de desenvolvimento de programas bem como na manutenção dos mesmos, aproveitando o ambiente web para que eles possam ser utilizados através de qualquer computador com acesso a internet. Outro fator que reduz também o custo da concepção e uso do sistema sugerido diz respeito à utilização de ferramentas gratuitas no desenvolvimento deste projeto bem como o posterior uso e manutenção do mesmo.

Desta forma, uma sugestão para possíveis trabalhos futuros seria fazer a análise de funcionalidades que não foram estudadas nesta monografia com o intuito de fornecer mais recursos ao sistema de comunicação proposto.

Referências Bibliográficas

KROLL, P.; KRUCHTEN, P. **Rational Unified Process Made Easy: A Practitioner's Guide to the RUP**. Boston: Addison Wesley, 2003.

KRUCHTEN, P.; **Rational Unified Process, The: An Introduction. 3. ED**. Boston: Pearson Education, 2003.

CAVANESS, C. **Programming Jakarta Struts**. O'Reilly, 2003. ISBN: 0-596-00328-5.

FOWLER, M. **Inversion of Control Containers and the Dependency Injection Pattern**, 23 jan. 2004. Disponível em http://martinfowler.com/articles/injection.html Acesso em: 08 out. 2007.

ALEXANDER C., S. et al. **A pattern language**: towns buildings construction. Oxford University Press, 1977.

BUSCHMANN, Frank et al. **Pattern - oriented software architecture**: a system of patterns. New York : J. Wiley, 1996.

Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. **Design Patterns: Elements of Reusable Object-Oriented Software**. 1.ed. Estados Unidos da América: Addison-Wesley, 1995.

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. **UML, guia do usuário**. Rio de Janeiro: Campus, 2000.

POLLICE, Gary. **Using the Rational Unified Process for Small Projects: Expanding Upon extreme Programming. Rational Software** White Paper.

SOMMERVILLE, Ian. **Software Engineering, 6. ed**. Addison-Wesley Publishers Ltd., 2001. ISBN 0-201-39815-X.

Anexo 1 - Artigo

Análise de Sistema para Informatização dos Processos de Comunicação em Organizações

Romulo Flávio Bittencourt[1]

[1]Departamento de Informática e Estatística – Universidade Federal de Santa Catarina (UFSC)
Caixa Postal 476 – Florianópolis –SC – Brazil

romulo@inf.ufrgs.br, jovemempreendedor@gmail.com

Abstract. *The process of communication in some organizations is still made, today, through printed documents. In doing an analysis of this form of communication was possible to see that this procedure often presents problems with slow because the communication process depends on manufacturing, shipping and science of the document by the parties involved. There are cases where one of these steps can be time consuming allowing the chance of compromising the whole process. You can cite as an example the case of the communication be to distant places. In addition to this it can also present problems in terms of security, because during the journey by which the document covers, there is the possibility of it being lost. It may not have history because there is no standardized procedure for filing of communications in order to compose a reliable history. Finally you can cite the costs generated with paper, printing, post, among others.*

Resumo. *O procedimento de comunicação em algumas organizações ainda é feito, nos dias de hoje, através de documentos impressos. Ao fazer uma análise desta forma de comunicação foi possível constatar que este procedimento, muitas vezes, apresenta problemas com morosidade, pois o processo de comunicação depende da confecção, expedição e ciência do documento pelas partes envolvidas. Existem casos onde uma destas etapas pode ser morosa possibilitando a chance de comprometer todo o processo. Pode-se citar como exemplo o caso da comunicação estar destinada a lugares distantes. Somado a isto ela também pode apresentar problemas em termos de segurança, pois durante o percurso pelo qual o documento percorre, existe a possibilidade do mesmo ser extraviado. Também pode não possuir histórico em virtude de não existir um procedimento padronizado de arquivamento das comunicações a fim de compor um histórico confiável. Por fim pode-se citar os custos gerados com papel, impressão, postagem, envelopamento dentre outros ônus.*

1. Introdução

A Análise do domínio de comunicações em organizações foi realizada através de metodologias de desenvolvimento de programas. Para a análise efetivada foram utilizados conhecimentos da Engenharia de Software para o planejamento, desenvolvimento e manutenção de um sistema de informatização de comunicação em organizações. Outra ferramenta utilizada foi o processo de desenvolvimento conhecido pela sigla RUP que vem da abreviação de Rational Unified Process. Este processo, mundialmente utilizado, representa uma forma sistemática para desenvolvimento de

software, onde as etapas são ligadas umas às outras em ciclos de desenvolvimento o qual, a cada nova iteração, pode-se fazer novos refinamentos do que está sendo desenvolvido (KRUCHTEN, 2003). Estas áreas para desenvolvimento de sistemas com o intuito de informatizar o domínio das comunicações em organizações é que está sendo tratado neste trabalho.

2. Descrição do Domínio do Problema para Análise

Atualmente, em muitas organizações as comunicações são feitas com documentos impressos. Quando realizado desta forma este procedimento pode apresentar os seguintes problemas:

- **Moroso**: o processo de comunicação depende da confecção, expedição e ciência do documento pelas partes envolvidas. Se uma destas etapas for morosa compromete todo o processo, como no caso da comunicação estar destinada a lugares distantes;
- **Inseguro**: durante o percurso pelo qual o documento percorre, existe a possibilidade de ser extraviado;
- **Sem acompanhamento**: o emissor em um dado tempo, não tem conhecimento da situação na qual se encontra o documento;
- **Sem histórico**: não existe um procedimento padronizado de arquivamento das comunicações a fim de compor um histórico confiável;
- **Oneroso**: as comunicações geram custos com papel, impressão, postagem, envelopamento dentre outros.

3. O Rational Unified Process

Foi escolhido o RUP como processo de desenvolvimento para o desenvolvimento do sistema proposto. O RUP é um processo iterativo e incremental que provê uma abordagem disciplinada para o desenvolvimento de software (POLLICE). Na engenharia de software, um modelo de processo é um conjunto de passos parcialmente ordenados com a intenção de construir um produto de software de qualidade, capaz de atender às necessidades e exigências do usuário final, de acordo com planejamento e orçamento previstos (SOMMERVILLE, 2001).

4. Padrões de Projeto

O trabalho foi desenvolvido com padrões de projeto. A origem dos Padrões de Projeto vem do trabalho de um arquiteto chamado Christopher Alexander, no final da década de 70. Ele escreveu dois livros, inicialmente, “A Pattern Language” e “A Timeless Way of Building”, nos quais ele exemplificava o uso e descrevia seu raciocínio para documentar os padrões (Erich, Richard, Ralph, John, 1995). Em 1995, este grupo de quatro profissionais escreveu e lançou o livro "Design Patterns: Elements of Reusable Object-Oriented Software", um catálogo com 23 padrões de projeto. Estes autores ficaram mais conhecidos como A Gangue dos Quatro (Gang Of Four ou GoF), considerados os maiores entusiastas dos Padrões de Projeto.

5. Ferramentas e Frameworks Escolhidos

Durante o processo de análise foi decidido que todos os envolvidos no desenvolvimento do projeto podem utilizar o mesmo ambiente de desenvolvimento, utilizando-se das mesmas ferramentas e frameworks (veja tabela 1).

Tabela 1. Ferramentas e Frameworks

Ferramentas e Frameworks		
Ferramenta	**Nome**	**Versão**
SO	XP/Linux	
Máquina Virtual	JDK	5.0 Update 5
IDE	Eclipse	3.1
Container JSP	TomCat	5.5.9
Plugin	Lomboz	3.1 RC 2
Framework J2EE	Spring – Framework	1.2.5
Framework J2EE	Struts	1.2.7
Cliente CVS	WinCVS	2.0.2.4
Cliente BD	PHPPGAdmin	1.2.2
Browser	Internet Explorer	5.5 ou superior
Browser	Mozilla FireFox	0.8 ou superior

6. Definição da Arquitetura

A arquitetura adotada foi baseada nos seguintes Padrões de Projeto: Singleton, DAO, DTO, MVC, Iversion of Control e View Helper. Além dos padrões, empregou-se a programação orientada a aspectos. A arquitetura escolhida está direcionada para o desenvolvimento do programa a ser acessado no ambiente Web sob a plataforma Java.

7. Conclusões

Ao longo do trabalho foram descritos os principais passos no estudo dos levantamentos dos requisitos do sistema proposto utilizando-se um processo de desenvolvimento de software e conceitos da engenharia de software.

Também houve uma boa definição do escopo de atuação que o sistema é capaz de atuar no processo de comunicação interna, explanando de forma clara e concisa acerca do sistema analisado.

Foram atendidos os objetivos que se referiam ao uso de padrões de desenvolvimento de software, consagrados no mercado de trabalho, dando uma visão geral na forma com que cada um foi utilizado no sistema proposto.

Neste projeto tem-se uma clara visão da intersecção de conhecimentos no que diz respeito às áreas de gerência de desenvolvimento de software, análise de sistemas e engenharia de software, conhecimentos estes adquiridos ao longo do curso de ciências da computação.

Outro importante aspecto diz respeito aos custos gerados no processo de desenvolvimento de programas bem como na manutenção dos mesmos, aproveitando o ambiente web para que eles possam ser utilizados através de qualquer computador com acesso a internet. Outro fator que reduz também o custo da concepção e uso do sistema sugerido diz respeito à utilização de ferramentas gratuitas no desenvolvimento deste projeto bem como o posterior uso e manutenção do mesmo.

Referências Bibliográficas

Kroll P.; Kruchten P. Rational Unified Process Made Easy: A Practitioner's Guide to the RUP. Boston: Addison Wesley, 2003.

Cavaness, C. Programming Jakarta Struts. O'Reilly, 2003. ISBN: 0-596-00328-5.

Alexander C., S. et al. A pattern language: towns buildings construction. Oxford University Press, 1977

Buschmann, Frank et al. Pattern - oriented software architecture: a system of patterns. New York : J. Wiley, 1996

Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. Design Patterns: Elements of Reusable Object-Oriented Software. 1.ed. Estados Unidos da América: Addison-Wesley, 1995