CESAR AUGUSTO FELIX LEITE

ANÁLISE DE UM PROCESSO DE DESENVOLVIMENTO WEB UTILIZANDO UML

Monografia apresentada à Escola Politécnica da Universidade de São Paulo, para conclusão do Curso de MBA em Engenharia de Software.

São Paulo
2003

CESAR AUGUSTO FELIX LEITE

ANÁLISE DE UM PROCESSO DE DESENVOLVIMENTO WEB UTILIZANDO UML

Monografia apresentada à Escola Politécnica da Universidade de São Paulo, para conclusão do Curso de MBA em Engenharia de Software.

Área de Concentração:
Engenharia de Software

Orientadora:
Profª Jussara Pimenta Matos

São Paulo
2003

AGRADECIMENTOS

À minha esposa, e aos meus pais pelo apoio incansável.

À orientadora, Prof.ª Jussara Pimenta Matos, pela orientação direta e permanente incentivo.

À Profª Selma Shin Shimizu Melnikoff pelo seu profissionalismo na coordenação do curso.

Aos colegas de profissão que, colaboraram na elaboração deste trabalho.

RESUMO

O objetivo deste trabalho é apresentar os principais conceitos referentes a um sistema voltado para aplicações Web e analisar o processo de desenvolvimento, tendo como base o processo unificado e a notação UML, para esse tipo de aplicação. Além disso, são identificados os artefatos adequados a cada tipo de projeto, e apresentada à relação dos perfis das equipes envolvidas nas diferentes fases do processo de desenvolvimento.

O tempo de desenvolvimento de um sistema Web é um fator muito importante a ser considerado, visto que, o usuário tem acesso a uma variedade de produtos similares na internet. É preciso implantar um sistema com qualidade e rapidamente, aumentando as chances de se obter a fidelidade do usuário em relação ao produto disponibilizado, no mercado web, altamente competitivo. Para a obtenção de um sistema Web em um tempo curto de desenvolvimento, desde o surgimento da idéia do negócio até a sua implantação, é necessário o uso de um processo configurável e flexível, características essas encontradas no UP (Unified Process), utilizando-se a anotação UML (Unified Modeling Language).

ABSTRACT

The objective of this work it to present the main concepts referring to a system directed to Web applications and also analyze the development process, based on the unified process and UML notation for this type of application. Besides, the adequate artifacts are identified for each type of project and presented to the list of the team profiles involved in the different phases of the development process.

The development time of a Web system is a very important factor to be taken into account considered, considering that the user has access to a variety of similar products on the Net. It is necessary to introduce a system with quality and increasing quickly the chances of gaining the user's fidelity related to the available product on the Web market, which is highly competitive. To obtain a Web system in a short time of development, since the business idea came forth up to its implementation, it is necessary to use a manageable and flexible process. These characteristics can be found on UP (Unified Process), using the UML (Unified Modeling Language) annotation.

LISTA DE ABREVIATURAS E SIGLAS

ASP - Active Server Pages

B2B - Business-to-Business

B2C - Business-to-Consumer

C2C - Consumer-to-Consumer

CGI - Common Gateway Interface

EJB - Enterprise JavaBeans

HTML - HyperText Markup Language

HTTP – HyperText Transfer Protocol

HTTPS – HyperText Transfer Protocol Secure

IBM – Industry Business Machine

J2EE - Java 2 Plataform, Enterprise Edition

JSP - JavaServer Pages

MVC - Model View and Controller

RUP - Rational Unified Process

UP - Unified Process

UML - Unified Modeling Language

TCP/IP – Transmission Control Protocol/ Internet Protocol

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

1 – INTRODUÇÃO

O desenvolvimento de sistemas voltados para aplicações Web possui algumas características particulares, tais como velocidade e padronização no processo de desenvolvimento, reutilização de código, portabilidade e escalabilidade (Offutt, J., 2002). O fator tempo é crucial para os projetos Web, um produto que hoje é novidade, em menos de um mês pode já se tornar obsoleto, por isso, as empresas necessitam de um processo que seja rápido o bastante para acompanhar as mudanças constantes do mercado. Além disso, existem dificuldades para implantar um processo no dia-a-dia, de forma a não perder a agilidade no desenvolvimento dos sistemas nas empresas.

Para que um processo consiga ser inserido, é necessário que este seja configurável, tanto na seleção quanto na geração dos artefatos necessários ao projeto, ou seja, para cada tipo de projeto devem ser gerados diferentes artefatos e com variações em seus detalhes. Portanto, é também importante que a relação entre as equipes envolvidas em um desenvolvimento, não seja baseada somente nos artefatos gerados durante as fases do processo, dificultando o entendimento do projeto por todos os envolvidos.

1.1 - Considerações Iniciais

A utilização do processo unificado (UP), com a notação UML (Booch et al, 1999), fornece os artefatos necessários para se obter as notificações das características para o desenvolvimento Web, mas na prática as empresas que utilizam UP, ou algum outro processo como cascata ou espiral (Pressman, R., 2001) não conseguem obter a agilidade desejada no desenvolvimento de um sistema. Para isso, é preciso realizar adaptações no processo para atender a realidade em que a empresa está inserida (Connalen,J., 2002).

Uma característica intrínseca do processo unificado e importante no desenvolvimento de sistemas Web é a iteração, tendo como objetivo a obtenção de uma versão

executável do produto o mais breve possível, independente da complexidade do sistema em termos de requisitos ou tecnologia utilizada.

1.2 - Objetivo do Trabalho

O objetivo dessa monografia é analisar um processo de desenvolvimento de sistemas para WEB utilizando UML e fazer críticas em relação a alguns aspectos, tais como, dimensionamento das fases, das equipes e dos artefatos necessários no desenvolvimento dos diferentes tipos de sistemas, sem perder o mapeamento dos requisitos funcionais e não funcionais do sistema (Kotonya; Sommerville, 1998), nos artefatos.

1.3 - Motivação do Trabalho

A motivação para a realização desse trabalho é a experiência do autor em desenvolver sistemas voltados para aplicações Web, relacionando alguns pontos a serem melhorados de um processo para esse tipo de aplicações, incluindo a identificação dos artefatos mais adequados e a indicação de como solucionar pontos falhos na comunicação entre as equipes. Além disso, também apresenta algumas sugestões de como integrar as necessidades do projeto ao ambiente de desenvolvimento, de forma a atender de maneira mais eficiente e adequada às expectativas do usuário final.

Geralmente os artefatos são gerados a fim de facilitar a comunicação entre as equipes, mas na maioria das vezes, essa comunicação através somente dos artefatos faz com que o processo tenha pouca flexibilidade, fazendo com que o tempo de desenvolvimento do sistema, desde a idéia do negócio até a implantação do sistema se prolongue e conseqüentemente, para não ultrapassar o prazo do projeto, alguns requisitos importantes são ignorados e não implementados.

1.4 - Estrutura do Trabalho

Os capítulos desse trabalho estão divididos da seguinte forma.

O capítulo 1 apresenta o tema a ser tratado, o objetivo do trabalho, a motivação para a escolha deste tema, e a metodologia do trabalho.

O capitulo 2 tem como objetivo descrever as características dos sistemas voltados a Web e é realizada uma classificação de sistemas simples e sistemas complexos, levando em consideração o tempo de desenvolvimento os tipos de arquitetura de infra-estrutura, nível de segurança e linguagem de programação.

O capítulo 3 tem como objetivo apresentar o processo de desenvolvimento baseado no processo unificado (Kruchten P. ,1999) e descrever adaptações necessárias do processo para sistemas Web, aplicando a classificação de sistemas simples e complexos. Além disso, também apresenta a descrição de um ambiente de trabalho, as equipes envolvidas para a implantação do projeto e seus relacionamentos.

O capítulo 4 tem como objetivo apresentar quais são os artefatos que são necessários ao processo de desenvolvimento Web, levando em conta sistemas do tipo simples e complexo, quais as equipes são responsáveis pela geração dos artefatos em cada uma das fases do processo de desenvolvimento (Abmbler S. W.; Constantine, L. L. , 2000) e as ferramentas e linguagens utilizadas.

O capítulo 5 tem como objetivo identificar os pontos falhos e apresentar formas para melhorar o processo, a seleção de artefatos e o relacionamento entre as equipes.

O capítulo 6 apresenta a conclusão final, as contribuições e sugestões para trabalhos futuros.

O processo de elaboração deste trabalho é apresentado no digrama de atividades da figura 1.1.

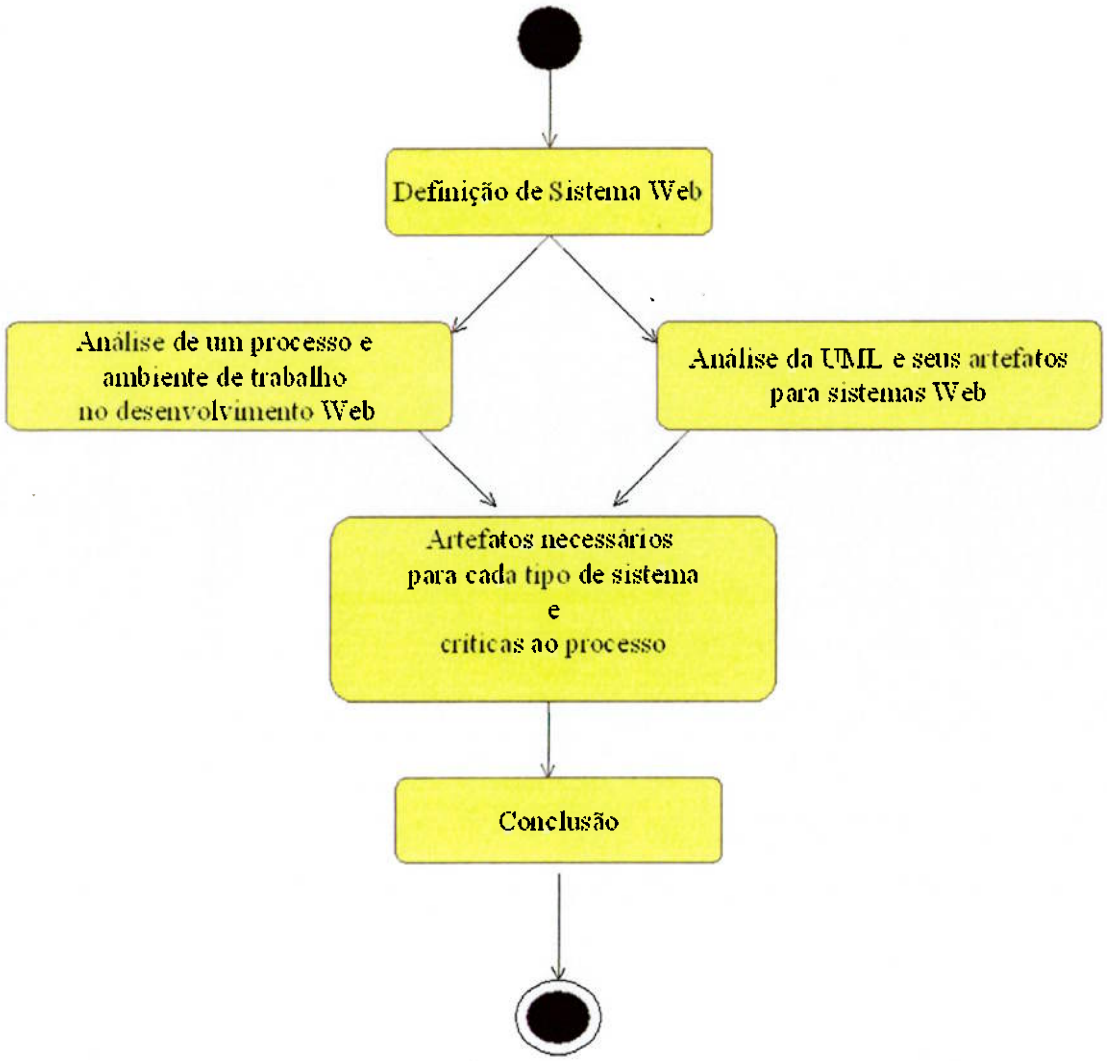

Figura 1.1 – Diagrama de atividades do processo de elaboração da monografia.

2 – DEFINIÇÃO DE SISTEMA WEB

2.1 – Apresentação

Este capítulo tem por finalidade apresentar as características de um sistema Web e de classificar os sistemas em dois tipos, simples e complexos em relação às características específicas de cada grupo.

2.2 - Características gerais de um sistema para WEB

Sistemas Web são aplicações que utilizam tecnologias de ativação, como CGI (Common Gatway Interface) (Umar, A. , 1997), permitindo um modo padrão para que os usuários Web executem aplicações no servidor, tornando-se seu conteúdo dinâmico de forma a atuar na regra do negócio dos sistemas, ou seja, interagir com o sistema.

Entretanto, um site Web são páginas estáticas(Connalen,J., 2002), que utilizam arquivos HTML (HyperText Markup Language), pré-formatados, fornecidos por um servidor, a medida que o usuário requisita a página, não havendo nenhuma forma do usuário, via navegador, alterar a regra de negócio do sistema.

Os sistemas Web, segundo Offutt, J., (2002), possuem algumas características particulares, independente da natureza da aplicação, ou seja, B2B (Business-to-Business), B2C (Business-to-Consumer), C2C (Consumer-to-Consumer) (Turban et al, 2000) e essas características são:

- **Usabilidade**: Quanto mais simples o uso de um sistema Web, maiores são as chances de obter a fidelidade do usuário, em relação ao sistema. Sistemas com uma navegação complexa e com um complicado entendimento, geralmente são abandonados pelo usuário.
- **Segurança**: Um fator importante para o usuário Web, geralmente os sistemas Web armazenam informações pessoais dos usuários, como sistemas bancários, por isso a segurança é um requisito fundamental em sistemas desse tipo.

- **Disponibilidade**: Um sistema Web precisa estar disponível vinte quatro horas por dia, sete dias na semana. Um sistema seguro e com alta disponibilidade pode ser considerado confiável.
- **Escalabilidade**: O sistema deve ser implementado para ter o mesmo comportamento, caso possua poucos ou muitos usuários, um problema na escalabilidade pode afetar a segurança, confiabilidade e a disponibilidade.
- **Fácil Manutenção**: Um sistema Web deve ser implementado de tal forma que seja fácil corrigir, adicionar ou remover uma funcionalidade.
- **Tempo de mercado**: O fator tempo é crucial dentro do mercado de internet, um sistema que hoje é novidade, em menos de um mês, pode se tornar obsoleto. A necessidade de implantação de um sistema em um tempo mais curto, pode impactar no processo e no gerenciamento dos projetos.

Dentre os fatores citados acima, o tempo de desenvolvimento é um dos mais importantes para as empresas que produzem sistemas Web, são raros os sistemas que possuem um tempo superior a um ano de desenvolvimento. Esses tipos de sistemas, geralmente são sistemas B2B (Turban et al, 2000), ou sistemas que utilizam alguma tecnologia nova para a empresa, onde existe a necessidade de ser testada e homologada essa tecnologia. A maioria dos sistemas, C2C e B2C (Turban et al, 2000), possuem um período curto no seu processo de desenvolvimento pelo próprio dinamismo do mercado internet.

2.3 – Tecnologias envolvidas em um sistema Web

Há poucos anos atrás sítios Web, somente eram formados por páginas HTML estáticas, JavaScript (Java Sun, 2003), onde alguns sistemas Web possuíam programas CGI para recuperar ou armazenar dados dos usuários (Offutt, J., 2002).

A figura 2.1 representa um modelo da arquitetura do tipo cliente-servidor dos primeiros sítios Web, onde possuíam um limite de tráfego, não eram escaláveis e possuíam páginas simples.

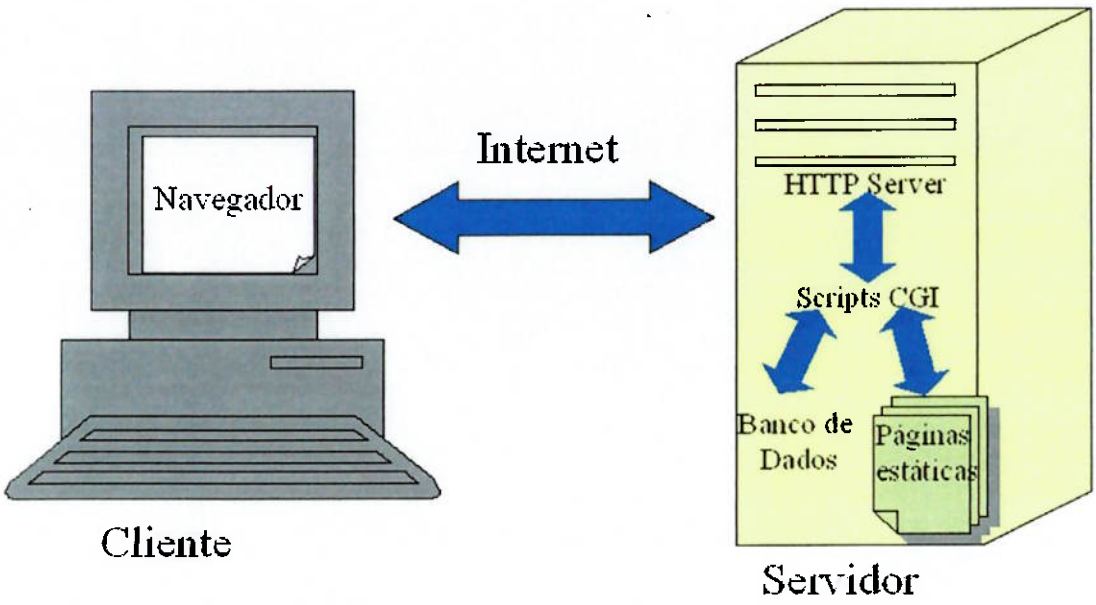

Figura 2.1- Primeira Geração de uma Arquitetura Cliente-Servidor para sítios Web (Offutt, J., 2002)

Atualmente os sistemas Web não fornecem apenas páginas estáticas, há uma maior interação do usuário com as regras de negócio do sistema, novas tecnologias de ativação, que permite o usuário interagir com o sistema Web, tais como Servlets Java, Enterprise JavaBeans, Applets, JavaServer Pages (Java Sun, 2003). Essas tecnologias são usadas a fim de tornar os sistemas mais rápidos, tanto em relação ao tempo de resposta do pedido de uma página, quanto permitir uma maior escalabilidade.

Segundo Conallen, J. (2002), os dois principais enfoques para as tecnologias de ativação das aplicações Web são:

- **As soluções de módulos compilados:** são binários carregáveis e compilados executados, pelo servidor no padrão CGI e podem ser escritos em qualquer linguagem. Embora o HTML (Umar, A., 1997) seja a saída mais comum dos módulos CGI, eles podem retornar qualquer tipo de saída para o usuário áudio, vídeo um texto ASCII.
- **As soluções baseadas em scripts interpretados:** são correspondentes a arquivos interpretados pelo servidor Web. Esses arquivos interagem com objetos no servidor e produzem uma saída em HTML. Os arquivos possuem identificadores especiais interpretados, pelo servidor de aplicação, como por exemplo arquivos JSP (JavaServer Pages), ASP (Active Server Pages).

O padrão de desenvolvimento dos componentes e a arquitetura de infra-estrutura, são também pontos em comum entre os sistemas Web (Offutt, J., 2002). Em relação ao padrão de desenvolvimento geralmente utiliza-se o modelo de três camadas, MVC (Model View and Controller) (JAVA SUN, 2003). Na arquitetura de infra-estrutura procura-se utilizar os mesmos componentes: um servidor, uma conexão de rede, navegadores clientes, servidor de banco de dados, quando houver necessidade, e um ou mais servidores de aplicação, são esses servidores que contêm as regras do negócio.

A figura 2.2 apresenta o modelo de uma arquitetura de cliente com múltiplos servidores, atualmente mais comum, utilizando a tecnologia Java de ativação.

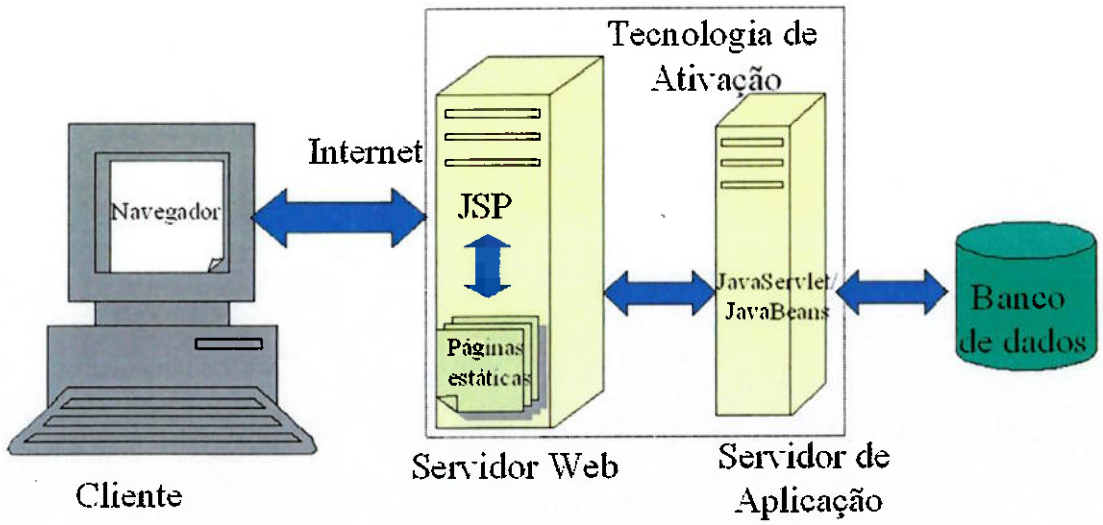

Figura 2.2- Multi-servidores e tecnologias de ativação (Offutt, J., 2002)

Devido às mudanças das tecnologias para ambientes Web e com o aumento da interatividade do usuário com os sistemas, surgiu a preocupação em gerenciar o estado do cliente nos servidores pois, à medida que o usuário navega pelo sistema, devido à natureza do protocolo usado na comunicação entre cliente servidor (TCP/IP) (Amjad Umar, 1997), onde é estabelecida uma conexão para troca de informações e em seguida a conexão é fechada, é difícil realizar o rastreamento do estado do usuário no sistema.

Para controlar esse estado nos sistemas Web existem duas formas:

- **Por meio de cookies:** esse recurso consiste de um conjunto de dados, onde um servidor Web pode solicitar para o navegador Web para manter e retornar cada vez que for realizada uma solicitação subseqüente de um recurso HTTP para esse servidor, esse conjunto de dados é criado pelos programas em padrão CGI ou pelos scripts interpretados.

  Contudo há limites de tamanho desse conjunto de informações e de tempo de expirações, ou seja depois de um certo intervalo de tempo o

servidor não reconhece mais essas informações sendo necessário criá-las novamente.

Um exemplo de uso de cookie na Web, são sistemas que necessitam de uma autenticação por parte do usuário. Quando o usuário faz a sua autenticação em um sistema, é gerado um cookie com as suas informações, nas quais são utilizadas em todo o sistema a medida que o usuário navega pelas funcionalidades.

- **Por meio de sessões**: esse recurso, ao contrário dos cookies consiste em o próprio servidor de aplicação ser capaz de armazenar algum estado sobre o cliente durante a navegação do usuário pelo sistema. A maneira mais comum do servidor armazenar essa informação é mantendo uma chave única para cada navegador aberto pelo cliente, tendo assim um dicionário ou um mapa de qualquer tipo de objeto que, pode ser armazenado na sessão do usuário. Essa sessão é finalizada quando o usuário fechar o navegador.

As Diferentes características de um sistema Web podem classifica-los como sistema simples ou em complexo. Connallen, J., (2002) cita a tecnologia de ativação, o uso ou não de páginas simples no navegador do cliente para fazer essa classificação dos sistemas Web. Russ, M.; McGregor, J., (2000), referem-se à complexidade dos requisitos do projeto e à maturidade da empresa em relação à tecnologia empregada, para diferenciar os tipos de sistemas.

Considerando as características mencionadas acima, considerando os tipos de linguagem de programação, assim com a arquitetura de infra-estrutura e a maturidade das equipes em relação ao processo de desenvolvimento utilizado na empresa, pode-se agrupar os sistemas em simples e complexos, em relação as aplicações para Web, com a finalidade de melhorar e adaptar o processo de desenvolvimento aos sistemas.

2.4 - Sistemas Simples

Os sistemas simples para Web são os mais freqüentes no ambiente internet, são sistemas que possuem as seguintes características (Conallen, J., 2002):

Período curto de desenvolvimento, até seis meses.

Aplicações voltadas a ter uma base de dados grande de clientes com funcionalidades que satisfaçam o usuário, ao contrário de ter uma aplicação tecnologicamente sofisticada, e uma base de dados mínima de clientes. Para obter um número grande de usuários nos sistemas há uma preocupação em relação às páginas nos navegadores, à parte cliente do sistema, onde possa ser solicitado e apresentado páginas HTML, com a capacidade de enviar e aceitar tipos simples de dados como seqüências de caracteres, seleções e booleanos durante as solicitações das páginas.

Geralmente as páginas são leves, não tendo muito código que execute no navegador do usuário como applet java e activex (Amjad Umar, 1997), o desenvolvimento dessas páginas se preocupa com todos os tipos de acessos que um usuário pode ter ao sistema desde linha discada, através de modem, até o acesso através de banda larga.

São aplicações onde à segurança do sistema não é o principal requisito, devido as suas características e a fim de alcançar uma simplicidade no desenvolvimento, obtendo-se assim, um decréscimo no tempo da construção do sistema.

O desenvolvimento ocorre utilizando linguagem de programação, padrões e tecnologia que possuem uma certa maturidade dentro da empresa, pois isso afeta diretamente ao tempo e a complexidade do desenvolvimento do sistema. Um sistema simples gera poucos artefatos pois o comportamento do desenvolvimento e de seus componentes, é de conhecimento das equipes que desenvolvem o sistema.

Com uma arquitetura de infra-estrutura composta de um servidor Web responsável pelas publicações das páginas estáticas e de toda as entradas das requisições do sistema, um servidor de aplicação que contém a regra do negócio do sistema, e um servidor de banco de dados onde são armazenadas as informações geradas e modificadas, pelo servidor de aplicação, Figura 2.3.

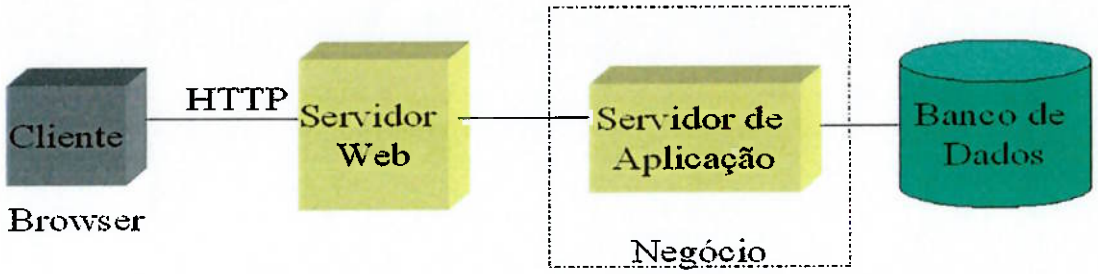

Figura 2.3- Exemplo da arquitetura de infra-estrutura de um sistema simples Web, baseado em Connallen, J. *(2002)*.

Para facilitar o desenvolvimento do sistema simples, geralmente procura-se não utilizar, servidores de componentes como EJB (Enterprise Java Beans), a fim de se ter uma complexidade menor no sistema e com isso reduzir o tempo de desenvolvimento. Muitos sistemas considerados simples se tornam complexos pelas soluções adotadas pelos analistas, soluções essas baseadas em tecnologias novas, as famosas "balas de prata"(Ambler, S., W; Constantine, L., L., 2000), fazendo com que o tempo de desenvolvimento seja maior do que o esperado.

Os Componentes da arquitetura de sistemas simples são:

1. **Navegador do Cliente**: é o dispositivo de interface do cliente com o sistema, utilizado para requisitar e exibir as páginas em HTML. O uso

de páginas simples, onde o tempo de requisição e resposta seja muito rápido pois, como já mencionado, isso é vital para o objetivo de obter uma grande base de clientes. As validações de dados são realizadas tanto no cliente, com o uso de java script, quanto no servidor através do código do sistema, de forma a ter uma segurança mínima nas transações dos dados.

2. **Servidor Web**: é o responsável pela entrada das requisições do usuário no sistema e pela delegação dos módulos a serem chamados, de acordo com o tipo de requisição. Além disso, também fornece as páginas estáticas do sistema. A comunicação entre cliente e servidor se dá através do protocolo HTTP. O servidor Web é escalável, ou seja, pode se ter mais de um servidor a fim de atender as requisições dos usuários, conforme o número previsto de acessos de usuários na aplicação.
3. **Servidor de Aplicação**: é o responsável pela lógica de negócio do sistema, pela execução do código e retorno das páginas dinâmicas, geradas conforme solicitação do usuário. Esse servidor, pode estar ou não na mesma máquina que o servidor Web, incluindo o compartilhamento dos mesmos recursos de processamento. Todos os objetos do negócio estão contidos nesse servidor a fim de facilitar o desenvolvimento da aplicação.
4. **Servidor de Banco de Dados**: é responsável por manter a persistência dos objetos de negócio da aplicação. Geralmente, o servidor é representado por um sistema de gerenciamento de banco de dados e encontra-se em uma máquina diferente das máquinas do servidor Web e do servidor de aplicação, a fim de se ter um nível maior na segurança (Conallen, J., 2002).

As linguagens de programação utilizadas no desenvolvimento de sistemas Web, são linguagens que utilizam o padrão CGI tais como perl, php, asp, jsp e java (JAVA SUN, 2003).

A opção de utilização de uma linguagem é determinada pela estratégia da empresa e depende das características do projeto. Sistemas simples requerem a utilização de uma linguagem de programação em que o desenvolvimento seja rápido, tal como perl (Larry, W.,2001) e php (Larry, U., 2001) mas, isso não é uma regra, pois o uso de linguagem orientadas objetos tais como a linguagem java, que não possui um desenvolvimento rápido como php ou perl, traz um ganho no reuso de código, ou seja, no desenvolvimento de sistemas futuros pode-se utilizar o mesmo padrão e conseqüentemente o mesmo código já implementado.

Os sistemas Web, classificados como simples, atendem a principal característica das aplicações para internet, que é o tempo curto de desenvolvimento, geralmente essas aplicações também possuem um tempo curto de durabilidade, devido ao surgimento muito rápido na internet de novos sistemas para os usuários e novas tecnologias (Offutt, J., 2002).

Geralmente, quando uma tecnologia nova passa a ser usada por uma empresa, conseqüentemente é necessário realizar alterações nos sistemas legados. Portanto adoção de uma nova tecnologia implica em adaptações nos sistemas legados.

Essa prática de reciclagem em relação aos sistemas legados é comum nas empresas Web, e principalmente nos sistemas considerados simples pois as mudanças para a nova tecnologia tendem a ter um período curto de desenvolvimento. Contudo se as mudanças exigirem um período maior, as transformações são divididas em iterações .

2.5 – Sistemas Complexos

Os sistemas complexos, assim como os sistemas simples, possuem características que fazem com que as aplicações se enquadrem dentro dessa divisão, são elas:

Com um período desenvolvimento maior de seis meses.

O tempo de desenvolvimento dos sistemas complexos é maior que seis meses, na maioria das vezes, esse tempo de desenvolvimento é aceitável dentro das

empresas, ainda que o tempo seja crucial no desenvolvimento de um sistema Web, pois os requisitos de segurança, escalabilidade e ou o uso de uma tecnologia nova passam a ser mais importante do que se ter um produto em um tempo curto de desenvolvimento. Sistemas complexos costumam ter um número de iterações maiores que os sistemas simples a fim de se ter algum produto, nem que seja apenas para de teste de unidade, executável.

Aplicações voltadas a ter uma base de dados fixa, seja ela grande ou não de usuários e as grandes variações de usuários na base de dados se dão em longo espaço de tempo. Possuem um acesso maior de usuários do que as aplicações simples.

São aplicações onde à segurança do sistema é um dos principais requisitos do sistema. Possuem requisitos mais complexos em termos de segurança e por serem aplicações que são mais acessadas do que os sistemas simples por exemplo, precisam ser confiáveis com uma arquitetura distribuída a fim de ter o desempenho e a segurança das aplicações como um dos principais requisitos dos sistemas.

O desenvolvimento ocorre utilizando padrões e tecnologias novas dentro da empresa.

O uso de uma arquitetura de infra-estrutura mais distribuída em relação aos sistemas simples é outra em característica dos sistemas complexos pois há necessidade de ter nas camadas de controle e de dados uma distribuição dos objetos responsáveis por essas tarefas a fim de se atingir os requisitos de desempenho escalabilidade e segurança, conforme descrito na figura 2.4.

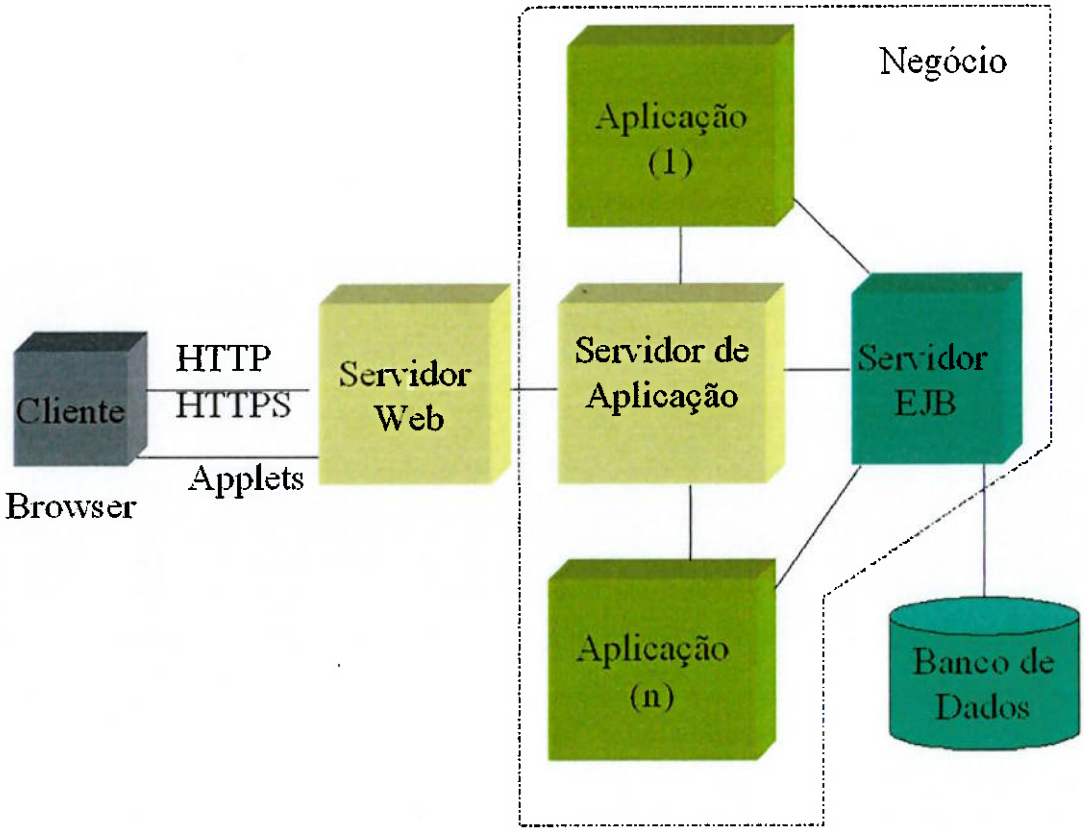

Figura 2.4- Exemplo da arquitetura de um sistema complexo Web, baseado em Connallen, J., 2002.

Os componentes da arquitetura de sistemas complexos são:

1. **Navegador do Cliente**: possui as mesmas funcionalidades mencionadas nos sistemas simples, tais como, controle das entradas de dados e requisições nos sistemas, mas podendo ter parte da lógica do negócio da aplicação na máquina do cliente, através de applets e activex tendo um critério de validação mais rigoroso em relação aos dados, visando à segurança e o desempenho dos servidores.
2. **Servidor Web**: possui as mesmas funções mencionadas nos sistemas simples, podendo também ser um servidor de applets java. A comunicação entre o cliente e o servidor é feita geralmente através dos protocolos HTTP ou HTTPS.
3. **Servidor de Aplicação**: possui as mesmas funcionalidades de um sistema simples, a diferença é na utilização de mais de um servidor de

aplicação, com uma arquitetura distribuída. O uso da arquitetura distribuída nos sistemas complexo é devido ao grande número de acesso de usuário, a fim de garantir o desempenho dos sistemas. No caso de se usar EJB, componentes distribuídos, o sistema pode possuir um ou mais servidores para a persistência dos dados esses servidores são capazes de manter a consistência entres os objetos persistidos entre os servidores.

4. **Servidor de Banco de Dados**: possui as mesmas funcionalidades dos sistemas simples, em casos de sistemas onde a segurança é um dos requisitos principais, é comum a utilização de mais de um banco de dados com a duplicação de dados.

As linguagens de programação utilizadas para o desenvolvimento das aplicações complexas são as mesmas linguagens já mencionadas nas aplicações consideradas simples, sendo enfatizada as soluções de uma arquitetura distribuída, como por exemplo utilização da linguagem Java e a arquitetura EJB(Enterprise JavaBeans) na Plataforma J2EE (JAVA SUN, 2003). Em relação às páginas executadas no navegador do cliente, o uso de controles tais como, activex e applets Java também são comuns nessas aplicações. Dessa forma é possível validar os campos de um formulário, antes do envio dos dados ao servidor, havendo um controle na segurança dos dados e no desempenho do sistema, sendo ignorado o envio de dados inválidos.

Um exemplo de sistemas que se enquadram nas características já mencionadas, são os sistemas bancários, pois são sistemas de alta complexidade, seguros, confiáveis e a tecnologia é constantemente atualizada, com a finalidade de melhorar esses requisitos. Outros exemplos de sistemas complexos, dentro do desenvolvimento Web, são as aplicações ou componentes que dão suporte ao desenvolvimento de outros sistemas da empresa, tais como um servidor de sessão ou até mesmo um "middleware" que possa conter toda à parte de controle das conexões de banco de dados, que possuem as características tais como, alto desempenho e uso de tecnologia de ponta.

2.6 – Tabela Comparativa entre sistemas simples e complexos

A tabela 2.1, apresenta um quadro comparativo entre as características dos sistemas simples e complexos.

Característica	**Sistemas Simples**	**Sistemas Complexos**
Tempo de desenvolvimento	Até seis meses	Maior que seis meses
Uso de Componentes distribuídos	Raramente	Freqüentemente
Preocupação com a segurança	Baixa	Alta
Quantidade de acesso por usuários	Média	Alta
Tecnologia envolvida no desenvolvimento	Geralmente o padrão da empresa	Geralmente uso e homologação de tecnologias novas no mercado
Páginas executadas no navegador do usuário	Páginas leves, sem uso de muitos recursos.	Uso de freqüente de tecnologias como "applets" "activex."

Tabela 2.1 - Quadro comparativo entre os sistemas simples e sistemas complexos

A divisão entre sistemas complexos e simples não é um conjunto de regras pré-estabelecidas e sim, uma forma de identificação das características de um sistema e quais são os artefatos que devem ser produzidos, para cada uma das fases do desenvolvimento. As características entre os sistemas Web são muitos semelhantes, a proposta de classificação em sistemas simples e complexos, é definir um conjunto de referências mínimas, para identificar o perfil das equipes, os tipos de testes necessários, as necessidades de números e de tipos de servidores. Enfim, obter o conjunto necessário de informações que não está explicitamente apresentado nos requisitos iniciais do sistema. Dessa forma, é possível planejar os prazos e as atividades das fases do processo de desenvolvimento.

3 – PROCESSO E AMBIENTE DE TRABALHO NO DESENVOLVIMENTO WEB

3.1 – Apresentação

O objetivo desse capítulo é apresentar um processo para o desenvolvimento de sistemas Web, baseado no processo unificado, além disso, são considerados o ambiente de desenvolvimento, os termos e os conceitos aplicados no processo, incluindo a classificação dos sistemas simples e complexos.

3.2 – Processo de desenvolvimento para sistemas Web

O conhecimento cada vez mais aprimorado da tecnologia que envolve a internet, tornam os sistemas Web mais diversificados e as equipes de desenvolvimento maiores e especializadas.

Para gerenciar as equipes cada vez maiores e ter ainda um desenvolvimento rápido e com qualidade, atributos imprescindíveis na Web, é necessário um processo robusto, configurável e flexível para que seja bem compreendido pelos diferentes integrantes das equipes envolvidas.

A necessidade de adoção de um processo para Haley, T., J., (1996), é para garantir uma produtividade e um alto padrão de qualidade. Para Conallen, J., (2002), um processo de desenvolvimento de software tem quatro funções dentro de uma empresa.

1. Fornecer uma diretriz em relação à seqüência de atividades de uma equipe.
2. Especificar os artefatos que devem ser desenvolvidos.
3. Direcionar as tarefas dos desenvolvedores de acordo com o perfil necessário e da equipe como um todo.
4. Oferecer critérios para monitorar e comparar os produtos e as atividades do projeto.

Para que os quatro itens acima sejam atendidos é necessário definir:

- **Os Fluxos de Trabalho**: é um conjunto de atividades (requisitos, análise, projeto, implementação, testes e implantação) que ao final de cada uma das atividades, produza um resultado tangível e observável. Cada fluxo de trabalho requer vários operadores, atividades e artefatos para ser concluído.
- **Os operadores**: são os papéis executados pelos recursos humanos para atender ao processo.
- **Atividades**: são os procedimentos que os operadores executam para produzir os artefatos de saída do fluxo de trabalho.
- **Artefatos**: é qualquer parte tangível da informação produzida pelos operadores durante o processo, consistindo dos modelos, dos elementos dos modelos, do código-fonte e dos documentos. Um artefato está sujeito ao controle de versão e configuração (Ambler, S., W; Constantine, L., L., 2000). Portanto é preciso manter um histórico da evolução dos artefatos que se faz presente em todo o desenvolvimento. Esse histórico de artefatos permite que os membros das equipes rastreiem as diferentes tarefas para cada equipe, permitindo a estimativa do projeto em termos de custo, de uma maneira mais realista e consistente com as atividades do processo.

O processo unificado é recomendado para os projetos Web, devido as suas características intrínsecas: um processo centrado nos casos de uso, iterativo e incremental, significando que a cada fluxo de trabalho do processo possa ser repetido e redefinido, até que os requisitos do sistema e seja atendidos e implantados.

A iteratividade facilita a descoberta de problemas em cada uma das fases do projeto, e de quais foram às circunstâncias que proporcionaram esses problemas, sendo possível questionar as decisões tomadas nas fases anteriores. Sendo assim, essas alterações também fazem parte do processo de desenvolvimento, podendo ser reconhecidas e planejadas (Kruchten, *P., 1999).*

Para que a iteração tenha efeito no processo de desenvolvimento alguns itens são destacados:

1. **Rever e Refinar o Plano de Iteração**: antes de uma iteração, a equipe de desenvolvimento examina o plano e o gerente de projeto faz alocação de recursos, de acordo com a necessidade da iteração, lembrando que as iterações são planejadas de acordo com as prioridades da empresa, principalmente em relação ao tempo de mercado.

2. **Rever o Progresso com as Partes Interessadas**: antes de finalizar uma iteração é importante mostrar o resultado às partes interessadas no sistema, com o objetivo de avaliar o trabalho já desenvolvido, incluindo a visão das diferentes equipes envolvidas.

3. **Avaliar a Iteração e Ajustar o Próximo Plano de Iteração**: é realizada uma avaliação por parte da equipe a cada iteração, para saber se o tem o tempo de desenvolvimento está de acordo com o cronograma, se todos os componentes da equipe estão de acordo com o que foi desenvolvido até o momento, sendo possível rever os casos de uso da próxima iteração, a fim de remover qualquer dúvida que exista nas equipes para a próxima iteração.

Um processo que possui as características acima mencionadas é o processo Rational Unified Process (RUP) baseado no trabalho de Krutchen, P., 1999, onde o autor descreve o que é o processo RUP e seus conceitos, as melhores práticas dentro do processo e como esse pode auxiliar como guia no desenvolvimento de um projeto.

A figura 3.1 ilustra o processo iterativo, de acordo com o RUP

Figura 3.1.- Processo Iterativo (Kruchten, P., 1999)

Para uma empresa adotar um processo de desenvolvimento, não é possível apenas optar por um processo e incorporar esse processo na empresa. É preciso que o processo sofra uma adaptação, pois cada corporação possui características e fatores humanos diferentes, e é nesse aspecto que muitas empresas Web, que utilizam algum processo para o desenvolvimento de suas aplicações, não conseguem obter o resultado de qualidade e agilidade, esperado em seus projetos.

De acordo com Conallen, J., 2002, um processo deve se adaptar levando em conta os seguintes aspectos:

- **Composição da Empresa e da Organização**
  Grandes empresas com uma grande diversidade de talentos especializados e com divisões de equipes bem estabelecidas e definidas, podem utilizar um processo que possua uma grande quantidade de artefatos, devido a essa divisão e o número grande de pessoas nas equipes a comunicação entre as equipes responsáveis por cada atividade se da no nível da documentação dos artefatos, pois é muito difícil todos os membros das equipes estarem sempre participando das reuniões e mudanças sobre a

aplicação, é claro que esse tipo de processo possui um tempo maior no desenvolvimento.

Empresas onde as equipes são menores e onde o tempo é um fator importante no desenvolvimento possuem um processo com menos artefatos, havendo uma dinâmica em relação às comunicações entre as equipes pode se eliminar alguns pontos do processo como revisões de código e reuniões formais sobre as mudanças do projeto, ainda assim as equipes seguem um processo, mas com menos burocracia.

- **Natureza da Aplicação**

  A funcionalidade de uma aplicação pode afetar a estrutura do processo do seu desenvolvimento. Se um sistema Web for complexo em termos de funcionalidade as fases de requisitos e de análise vão gerar mais artefatos, se o sistema for complexo no que se refere à segurança há implicações arquitetônicas e de segurança que devem ser monitoradas durante o processo com revisões adicionais e protótipos exploratórios, agora se o objetivo da aplicação é trabalhar com uma nova tecnologia a definição de requisitos estritos não é tão importante quanto as possíveis descobertas das equipes em relação à nova tecnologia. Ao trabalhar com um sistema Web considerado simples as equipes tem em mente um conjunto mínimo de artefatos a serem desenvolvidos, a fim de se obter uma maior agilidade dentro do processo.

- **Nível de Habilidade da Equipe de Desenvolvimento**

  Para equipes que não estão treinadas com o processo de desenvolvimento da empresa mesmo que a aplicação seja considera simples é necessário que utilizem o processo mais definido em todas as suas fases, como um aprendizado. À medida que os operadores se familiarizem com o processo da empresa certos artefatos das atividades de um fluxo de trabalho, que não são estritamente necessários, podem não ser utilizado mais por essa equipe em determinados projetos.

- **A Prioridade Relativa do Conjunto dos Recursos**

  O que for importante no sistema final, tempo de desenvolvimento, total de defeitos aceitáveis e assim por diante, é o que vai determinar quais elementos do processo são importantes, por exemplo, no caso de sistemas Web lançar o produto no mercado primeiro, muitas vezes é o fator mais importante do produto, o processo tem que se adaptar para que se tenha uma produção rápida com isso certas inspeções e revisões nos artefatos podem ser ignoradas e a iteratividade tem que ser aplicada a fim de se lançar um produto com os requisitos mínimos de funcionamento e garantir a fatia do mercado e a base de dados de usuários, e depois com a estabilidade do produto produzir novas versões a fim de se atingir todos os requisitos iniciais do sistema.

3.2.1 – Conjunto de Atividades

O conjunto de atividades de um processo é responsável pela geração dos artefatos, que permite avaliar o estágio em que se encontra o projeto.

Em cada uma das fases e para cada fluxo de trabalho há um conjunto de atividades e de geração dos artefatos.

As fases do processo unificado são: Concepção, Elaboração Construção e Transição, conforme apresentado por Booch et al, (1999):

Os Fluxos de Trabalho consistem da modelagem de negócio, requisitos, análise e projeto, implementação, teste, implantação. Além disso, devem ser considerados os Fluxos de trabalho de Apoio, que são o gerenciamento de configuração e alteração, o gerenciamento de projeto, a definição do ambiente de desenvolvimento.

As atividades para cada uma das fases em relação ao fluxo de trabalho estão apresentadas na figura 3.2.

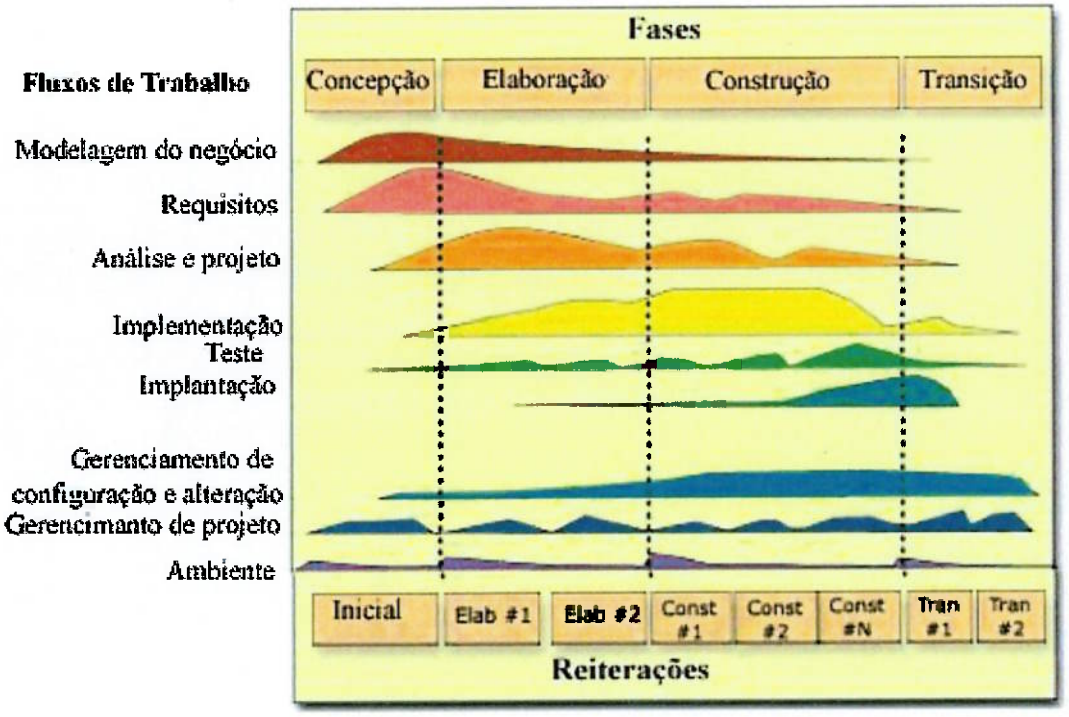

Figura 3.2.- Fases e fluxo de trabalho do Processo Unificado (Booch et al, 1999)

3.3 – Ambiente de desenvolvimento

Para a implantação de um processo é preciso identificar os operadores para realização das tarefas dos fluxos de trabalho, em geral esses operadores são selecionados e classificados por equipes, de acordo com o perfil necessário para atender as necessidades de um projeto.

As equipes que participam de um desenvolvimento Web são:

- **Produtos**: é equipe responsável pela definição do produto, que representa o usuário do sistema, ou seja, é essa equipe que interage com as outras equipes para resolver os conflitos de requisitos, principalmente em relação à identificação da funcionalidade a ser implementada. Essa equipe

é composta por editores, que são responsáveis pelo conteúdo das páginas do sistema e colaboram diretamente com os webmasters, que são responsáveis pela elaboração artísticas das páginas Web.

- **Gerentes e Diretores de Projeto**: são responsáveis pela validação inicial do produto proposto pela equipe de produtos, são as pessoas que possuem a visão tecnológica, do sistema guiando as outras equipes em termos de tecnologia, metodologia e linguagens usadas dentro da empresa e para cada projeto.

  Um produto dentro de uma empresa de desenvolvimento Web, pode também ser definido pelos gerentes e diretores de projeto, se houver necessidade. Nesse caso não há o contato diretamente com a equipe de produtos, apenas com os webmasters, se houver necessidade.
- **Equipe de Desenvolvimento**: é a equipe responsável pela análise e implementação do produto e pela geração da maioria dos artefatos que compõem o sistema. A interação com a equipe de produtos e com o gerente de projeto é intensa, na fase de implementação é preciso ter cuidado apurado para aceitação de mudanças propostas pela equipe de produtos pois, essas mudanças devem ser validas e documentadas antes de qualquer alteração, pelo gerente do desenvolvimento, responsável pelas alterações do projeto.
- **Equipe de Administradores de Banco de Dados:** é a equipe responsável pela modelagem dos dados, esses dados são geralmente entidades persistidas no banco de dados. A participação dos membros dessa equipe deve ser mantida ao longo de todo o desenvolvimento de um projeto, ou seja, um operador dessa equipe deve ser escolhido para ser responsável pelo projeto dentro da equipe de banco de dados, participando das reuniões e das mudanças do projeto. A comunicação entre as equipes de banco de dados e de desenvolvimento é crucial para garantir a qualidade do projeto e para tornar o processo de desenvolvimento ágil.
- **Equipe de Sistemas ou Engenharia**: é a equipe responsável pela seleção e instalação do ambiente de testes e de produção. Assim como, para a equipe de administração de banco de dados, é vital ter um operador da

equipe de engenharia responsável pelo projeto dentro da equipe. A comunicação entre equipe de desenvolvimento, administradores de banco de dados e engenharia deve ser feita de forma flexível, ou seja, não pode ser uma comunicação formal somente através dos artefatos.

- **Equipe de Qualidade**: é a equipe responsável pelos testes do sistema e pela sua validação, para que esse possa ser colocado em produção.

  Existe um papel que a equipe de qualidade deve assumir e que em muitas empresas não exercem, que é o papel de validação do processo de desenvolvimento, que visa sempre questionar as outras equipes sobre o que está errado e o que está certo, em relação ao processo, e aplicar mudanças junto às equipes para que o processo melhore. Para essa validação ocorrer é necessário que a equipe de qualidade participe do projeto desde o seu início até o final da implantação, finalizando a sua participação com a validação do projeto no ambiente de produção.

  Em relação à elaboração e realização dos planos de teste, atividade da equipe de qualidade, não basta apenas o caso de uso do sistema, pois na maioria das vezes, as informações importantes são implementadas e não descritas nos casos de uso, fazendo com que requisitos importantes não sejam testados e validados, é necessária a participação da equipe de qualidade em todas as fases do projeto a fim de se ter um entendimento de todo o sistema para que as informações não descritas no caso de uso sejam testadas no plano de testes.

A figura 3.3 mostra o fluxo de relacionamento entre as equipes que participam do processo de desenvolvimento de um projeto Web.

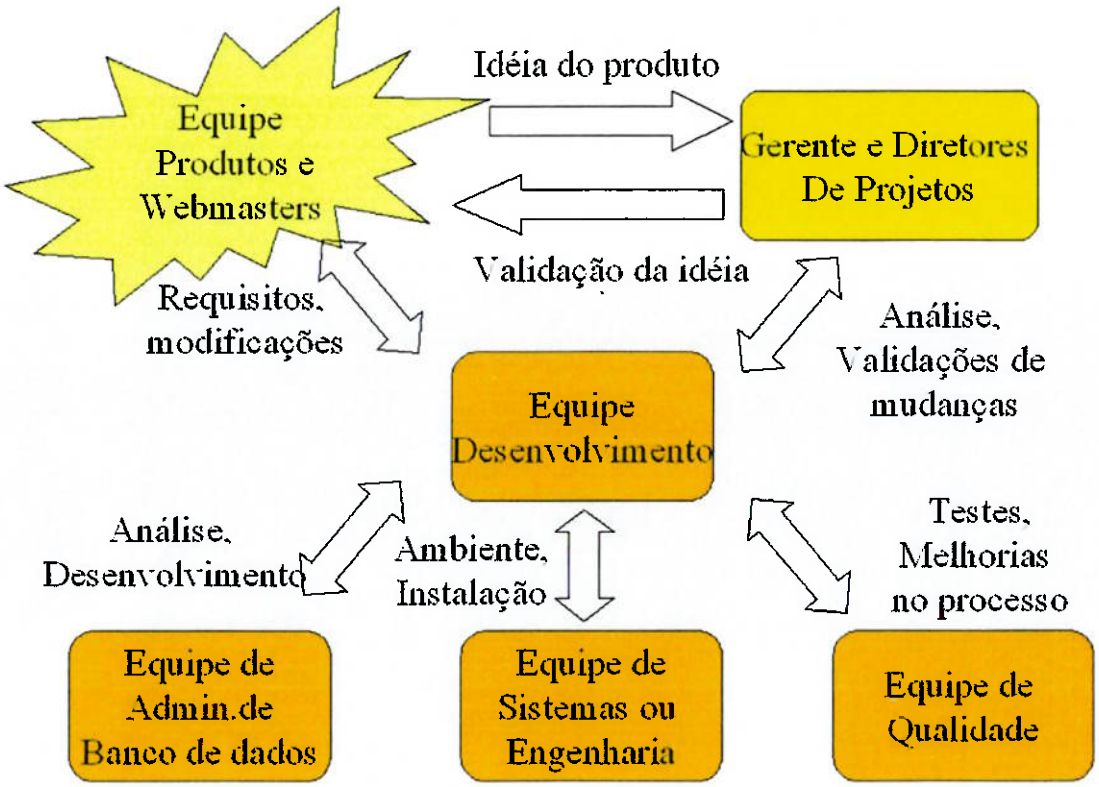

Figura 3.3.- Ambiente de Desenvolvimento e o relacionamento entre as equipes

A tabela 3.1, apresenta a relação entre equipes e atividade.

Equipes	**Atividades**
Produtos	Definição do Produto e criação do conteúdo das páginas do produto
Gerente e Diretores de Projeto	Validação da idéia do produto, gerenciamento do processo de desenvolvimento e definição do produto.
Equipe de desenvolvimento	Codificação do produto e responsável pela iteração entre as equipes participantes do projeto
Equipe de Administradores de Banco de Dados	Responsável pela modelagem de dados do negócio e administração do banco de dados do sistema
Equipe de Sistemas ou Engenharia	Responsável pela seleção e instalação do ambiente de testes e de produção.
Equipe de Qualidade	Testes e validação do processo

Tabela 3.1 - Quadro da relação entre equipes e atividades.

Um processo para funcionar dentro de uma empresa deve principalmente ser aceito pelas equipes, e para que isso aconteça, o mesmo tem que levar em consideração as necessidades dos seus usuários, sofrendo as adaptações necessárias, pois um processo só é considerado bom se for usado.

4 – ANÁLISE DA NOTAÇÃO UML E OS ARTEFATOS PARA SISTEMAS WEB

4.1 – Apresentação

O objetivo desse capítulo é apresentar os artefatos necessários do processo utilizando a notação UML, e mostrar quais são os artefatos que devem ser considerados para os sistemas simples e complexos, as ferramentas utilizadas em cada fase do processo.

4.2 – UML e os artefatos dentro da Web

O fluxo de desenvolvimento de um sistema Web é o mesmo de qualquer outro sistema que utilize o processo unificado. A partir de uma idéia inicial do sistema, tem a evolução e a transformação dos artefatos, que tem origem no documento de visão do software até o produto final codificado testado e implementado.

Para obter um sistema com qualidade da maneira mais rápida possível, já que o tempo é um requisito critico dentro da Web (Offutt, J., 2002), além de se ter um processo bem definido, é necessário definir um conjunto de artefatos a ser produzido em cada fase do desenvolvimento do sistema (Conallen, J., 2002).

A UML é uma anotação utilizada para a documentar os artefatos a serem gerados no decorrer do processo de desenvolvimento. Essa notação facilita a geração dos artefatos (UML OMG, 2003).

As ferramentas que utilizam a anotação UML procuram atender a todos os artefatos, de forma a facilitar o rastreamento dos requisitos.

Além de ser uma forma de documentar os sistema, a geração dos artefatos em cada fase do processo é uma forma de se ter:

- **Rastreamento**: a fim de localizar outros requisitos afetados pela alteração, inserção ou remoção de um outro requisito.

- **Transformação**: o desenvolvimento rápido dos sistemas Web depende diretamente das transformações claras e implícitas de um artefato para o outro, não ter o caminho definido claramente para a transformação dos requisitos em um sistema utilizável resulta na exploração de muitos caminhos de desenvolvimento desnecessários, ou seja, um processo mal definido, onde o tempo de desenvolvimento com certeza será maior do que o planejado.
- **Comunicação entre as equipes**: os artefatos são usados para ajudar nas comunicações entre as equipes envolvidas no desenvolvimento, um artefato gerado em uma fase do desenvolvimento, é usado como documento de entrada em uma próxima fase que muitas vezes é realizada por outra equipe.

  A linguagem usada para definir os artefatos em cada fase tem que ser de conhecimento dos operadores de cada equipe, pois por exemplo um documento de visão tem que ser entendido pela equipe que elaborou e pela equipe de desenvolvimento.
- **Gerenciamento das complexidades**: sendo uma forma de se localizar e prevenir os requisitos mais complexos do sistema, antes que se tornem obstáculos para o cronograma do projeto durante o processo.

  Para gerenciar as complexidades de um sistema utiliza todos os, artefatos, gerados nas fases de desenvolvimento, esse conjunto de artefatos chama-se modelo de sistema, que é a representação abstrata do sistema desenvolvido, esse modelo é utilizado e modificado por todas as equipes que participam do desenvolvimento do sistema. Através dessa representação que o gerente de projeto consegue visualizar os riscos e complexidades dentro do projeto, por isso a necessidade de sempre ter os artefatos o mais próximo da realidade do sistema, se algum artefato necessário não for criado ou atualizado a visualização das complexidades do sistema ficam afetadas.

Outra função de se ter um conjunto de artefatos gerado, e que na maioria das empresas não é utilizada, é de ajudar a definir melhor o processo do desenvolvimento. Essa atividade de melhoria no processo deve ser acompanhada da equipe de qualidade junto com os operadores das outras equipes, no decorrer do desenvolvimento de um sistema, e o objetivo é validar ou descartar o uso de um

determinado artefato em projetos futuros, levando em consideração a importância do mesmo para as equipes envolvidas e as características da empresa e dos sistemas que ela produz.

4.2.1 – Artefatos gerenciais

O gerente de projeto para controlar um projeto utiliza um conjunto de artefatos que são basicamente os planos projeto e iteração, e gerenciamento de alteração para a gestão de versão e configuração.

Os planos de projeto e de iterações são representados pelo cronograma de desenvolvimento e procedimentos para a seleção da equipe, as áreas de responsabilidades dentro do projeto e eventos principais que acontecem no decorrer do cronograma. O gerente de projeto trabalha junto com a equipe de desenvolvimento para levantar todas as iterações do sistema. Esses documentos sofrem alterações no decorrer do desenvolvimento, principalmente o plano de iterações, por isso a responsabilidade de descrever como, por quem e qual impacto desses artefatos é de responsabilidade do plano de gerenciamento de alteração.

O plano de gerenciamento de alteração é um artefato que está presente em todo ciclo de vida do desenvolvimento, é a forma como as mudanças são realizadas dentro do projeto. A existência do gerenciamento de alteração é essencial dentro do processo pois uma mudança em uma iteração por exemplo pode afetar os artefatos futuros e os já desenvolvidos, contudo esse plano, dentro do desenvolvimento Web, não pode ser dominante e complexo, assim como todo o processo, tem que simplesmente trabalhar para a equipe.

4.2.2 – Artefatos recomendados para sistemas Web

Apesar da UML oferecer um conjunto de artefatos, é necessário estabelecer critérios de escolha dos artefatos, quais são indispensáveis, pois o uso inadequado desses pode

comprometer a produtividade das equipes envolvidas. Porém a falta de artefatos adequados pode resultar em perda de tempo e retrocesso no desenvolvimento.

De acordo com Conallen, J., *,2002*, os modelos recomendados no desenvolvimento de sistemas Web, são:

1. **Modelo de Domínio**

   O modelo de domínio do sistema, o glossário e a visão do sistema são os artefatos principais do conjunto de domínio. O modelo de domínio captura a essência do contexto do negócio e do domínio na qual o sistema residirá e possivelmente estará trabalhando. A extensão de modelagem do negócio é usada para capturar e expressar os objetos do domínio em termos de operadores e entidades.

   O glossário captura definições de termos-chaves e conceitos que são usados nas discussões do negócio e, nas discussões das operações do sistema proposto.

   Apesar do documento de visão conter requisitos do sistema, itens do modelo de requisitos, em que o modelo de caso de uso é o principal artefato, ele pode ser considerado do modelo de domínio já que contém o contexto do negócio do sistema além da descrição do publico alvo, os requisitos do sistema, um protótipo, em anexo, da navegação das páginas do produto, a finalidade do produto, a situação do mercado e dos concorrentes em relação a produtos com as mesmas características, destaque aos pontos fortes do sistema em relação aos concorrentes e possíveis pontos fracos do software.

   A visão do sistema representa o inicio do ciclo de vida do projeto e é iniciado antes mesmo de um inicio sério do projeto, os requisitos funcionais e não funcionais do documento são alterados ao longo do ciclo de vida do desenvolvimento , e principalmente na fase da definição dos casos de uso.

Outro item importante iniciado pelo documento da visão é o conjunto de gerenciamento do projeto que vai estar presente em todo ciclo de vida, a visão do sistema é completada com a definição de uma equipe preliminar e as possíveis responsabilidades, isso porque só a partir do término da elaboração dos casos de uso será finalizada o levantamento da equipe final e seus participantes.

Tanto o documento de visão quanto o glossário devem estar acessíveis a todos que participam do desenvolvimento do sistema, estabelecer um Web site interno para a equipe de projeto é uma boa prática para se ter acesso à informação de uma forma rápida .

2. Modelo de Casos de Uso

O modelo de casos de uso tem a responsabilidade de levantar os limites e as funcionalidades do sistema. Um dos artefatos mais importante dentro do desenvolvimento das aplicações.

No levantamento dos casos de uso são separados os requisitos funcionais dos não-funcionais. Os requisitos funcionais são facilmente mapeados pelos casos de uso, em uma primeira instância é feito um mapeamento utilizando alguma ferramenta gráfica a fim de se ter uma visualização dos casos de uso, dos atores e suas interações, essa representação gráfica é necessária para que a equipe de desenvolvimento tenha uma idéia do tamanho e da quantidade de casos de uso do sistema a fim de saber a mão de obra a ser utilizada no projeto, as possíveis tecnologias e arquitetura do sistema envolvida e um cronograma das fases restantes do projeto. É nessa fase onde se confirma se o sistema a ser desenvolvido vai ser um sistema simples ou complexo para Web.

Os requisitos não-funcionais do sistema podem ser categorizados em requisitos do negócio ou arquitetônicos. Os requisitos do negócio são derivados do processo do negócio, como padrão de páginas de um sistema

Web por exemplo. Os requisitos arquitetônicos enfocam os limites das tecnologias envolvidas na solução do sistema, como tempo de resposta em certos processamentos do sistema e versões de navegadores aceitáveis pelo sistema.

A descrição de cada caso de uso acontece depois da representação gráfica da maior parte dos casos, a descrição de um caso de uso para sistemas Web podem possuir uma particularidade que é descrever junto com a funcionalidade a possível página que se refere o caso de uso, tentando descrever a maioria dos campos utilizados em determinada tela e a obrigatoriedade do preenchimento de alguns campos, mas somente com a definição do modelo de dados do negócio, pela equipe de administradores de banco de dados, que se tem todos campos, essa particularidade faz com que o programador consiga rastrear o caso de uso com a tela do mapa de navegação sem muito esforço economizando tempo na fase de codificação.

O caso de uso precisa estar descrito de uma tal forma que as equipes, que vão utilizá-los nas fases posteriores, consigam identificar o fluxo principal e os seus fluxos alternativos do caso de uso, na maioria das vezes não há necessidade de se ter um detalhamento muito grande, na primeira descrição, pois o processo de descrição dos casos de uso é interativo de forma que o mesmo vai pode ser completado ou refeito durante as fases seguintes do desenvolvimento. Junto com o levantamento dos casos de uso o arquiteto começa a visualizar a divisão de componentes do sistema ou seja a divisão dos subsistemas dentro do sistema.

O documento final dessa fase é ter a maioria dos casos de uso funcionais representados graficamente e as suas descrições, o plano de gerenciamento do projeto é atualizado de acordo com a equipe de desenvolvimento já definida, e com as iterações pré-estabelecidas.

3. Modelo de Análise/Projeto

O artefato principal do modelo de análise é o documento de arquitetura de software, com isso o arquiteto pode fechar a divisão, se houver, do sistema em subsistemas e apenas quando esses subsistemas estiverem divididos e as áreas de responsabilidades formadas é que pode haver o desenvolvimento assíncrono do sistema pelas equipes.

O documento de arquitetura do software, a divisão do sistema em subsistemas, é refinado com a experiência a partir de um protótipo de arquitetura. A criação de um protótipo da arquitetura é importante dentro do desenvolvimento Web pois com o rápido surgimento de novas tecnologias, os arquitetos muitas vezes adotam tecnologias que ainda não foram bem testadas e os riscos são maiores nesses casos.

Outros artefatos dentro da fase de análise são os diagramas de seqüência e os diagramas de estado, que são os diagramas da UML criados a partir da análise e realização dos casos de uso e do relacionamento entre eles e os limites do sistema (Booch et al, 1999). Esses artefatos podem alterar a representação e a descrição dos casos de uso, por isso há uma grande conectividade entre as classes de análise surgidas nesses artefatos com os casos de uso.

O modelo de projeto tem o objetivo de aplicar arquitetura estabelecida nos outros artefatos da fase de análise, é definir em quais subsistemas, ou camadas apresentação, aplicação e dados as classes de análise residem, é uma forma de tornar o modelo de análise realizável em software.

O modelo de projeto tem a participação do cliente ou de um usuário especializado do domínio, geralmente representado pelos membros da equipe que solicita o desenvolvimento do sistema para o departamento de tecnologia da empresa denominada equipe de produtos, essa participação tem a finalidade de validar a divisão dos subsistemas, essa validação se da no

refinamento da navegação das páginas do sistema junto como o modelo de projeto.

No final da fase de análise e do projeto tem se a documentação de grande parte das classes utilizadas no sistema, a divisão do mesmo em subsistemas, com a finalidade de cada classe definida através dos diagramas de classe, de seqüência e de estado e as definições das páginas em HTML. É nessa fase que se inicia o preenchimento das páginas com o conteúdo estático feitos pelos webmasters, por isso é necessário iniciar a próxima fase, implementação, com um ambiente de desenvolvimento já configurado, esse ambiente é configurado pela equipe de sistemas junto com os analistas responsáveis pela implementação, consiste em máquinas e softwares similares ao que se vai usar em produção, onde qualquer operador tem acesso as configurações desse ambiente.

4. Modelo de Implementação

A fase de implementação de um sistema é a fase de maior concentração de trabalho e não consiste somente no desenvolvimento dos códigos dos programas.

Nessa fase há uma preocupação em realizar os testes de unidades da aplicação esses testes são geralmente realizados pelos programadores e analistas e não pela equipe de testes, tem por finalidade validar as classes desenvolvidas e suas funcionalidades. Há também um teste de integração do sistema desenvolvido pelos programadores com as páginas desenvolvidas pelos webmasters.

É nessa fase onde há um maior número de levantamento de conflito entre os requisitos, pois a equipe de produtos possui um acesso direto ao que se esta sendo desenvolvido, através do ambiente de desenvolvimento, com isso há um aumento das mudanças de navegação e conseqüentemente de requisitos por parte da equipe de produtos, tanto o gerente de projeto e os programadores precisam estar atentos a essas mudanças a fim de documentar as alterações através do plano de gerenciamento de alteração, pois se isso não

acontecer há o surgimento de requisitos não documentados e não validados que podem ter um impacto do dentro do sistema e não serem comunicados para todas as equipes.

O resultado desse modelo é ter o sistema em um software executável em um ambiente de desenvolvimento já com a navegação das páginas todas desenvolvidas e aprovadas pelo gerente do projeto junto com o gerente da equipe de produto.

5. Modelo de Teste

O plano de teste é o artefato principal do modelo de teste, o plano começa a ser elaborado a partir da finalização dos casos de uso, mas o fechamento do plano de teste se da quando a fase de implementação já está quase no final isso porque na fase de codificação ainda a captura de algum caso de uso que não foram descritos nas fases anteriores.

A equipe de garantia qualidade é responsável pela elaboração e realização do plano de teste, uma vez que, se os testes forem realizados pelas mesmas pessoas que desenvolveram o sistema a qualidade dos mesmos tendem a ser inferiores devido ao vício dos programadores e analistas seguirem sempre o mesmo fluxo de teste.

Dentro do plano de testes que será feito e executado pela equipe de garantia da qualidade existem quatro tipos de testes importantes para os sistemas Web:

- Testes de desempenho: é a capacidade de o sistema funcionar corretamente sobre intensa demanda de usuários.
- Testes de demanda: estabelece a curva de desempenho do sistema sob demanda.
- Testes funcionais: são testes onde são validadas as funções especificadas nos requisitos do sistema, e verifica se o que os programadores e analistas implementaram é o mesmo que a equipe de produtos tinha especificado. Os testes funcionais na maioria das vezes são testes automatizados com o auxilio de alguma ferramenta (descrito no capítulo 4.4.), pois são testes que

sempre serão executados a cada nova iteração ou versão do sistema.

- Teste de Segurança: são realizados testes a fim de localizar alguma falha de segurança do sistema, o teste geralmente é realizado junto com a equipe de engenharia com a presença de um profissional especializado em segurança aplicações Web.

Como os projetos Web possuem na maioria das vezes várias iterações, é necessário muitas vezes realizar testes de regressão, onde esses consistem em testar novamente o sistema, em busca de algum tipo de erro gerado devido a nova iteração do sistema. Além disso a equipe de qualidade precisa testar o sistema em todas as versões de navegadores e plataformas possíveis já que o sistema Web é acessível por qualquer usuário que tenha acesso a internet.

Essa fase produz além do plano de teste, um documento descrevendo todos os testes realizados e os resultados, de acordo com esse documento o sistema pode sofrer alguma alteração, voltando até mesmo a se refazer algum caso de uso.

6. Modelo de Implantação

O modelo de implantação consiste de um documento que descreve todos os pontos importantes de um sistema para a sua instalação, ou seja , quais são as máquinas, versões de sistemas operacionais, servidores Web e de servidores de aplicação, quais são as portas dessas máquinas que devem estar abertas ou fechadas ao mundo exterior, os logs que a aplicação gera, a descrição dos arquivos de configuração da aplicação, onde estão localizados os binários da aplicação, qual a versão a ser instalada o motivo da atualização de versão, se houver, e uma descrição passo a passo de como instalar o sistema e os possíveis erros que podem surgir na instalação ou no funcionamento do sistema e as suas resoluções.

O documento de implantação geralmente é validado pela equipe de engenharia junto com o gerente do projeto, antes da instalação do mesmo, essa validação consiste na aprovação do documento e em instalação do sistema em um ambiente diferente do ambiente de desenvolvimento, simulando a instalação do produto em produção, esse ambiente é chamado de pré-produção e geralmente os testes da equipe de qualidade se realizam nesse ambiente e não no ambiente de desenvolvimento.

A figura 4.1 representa a dependência e rastreamento em conjuntos de artefatos

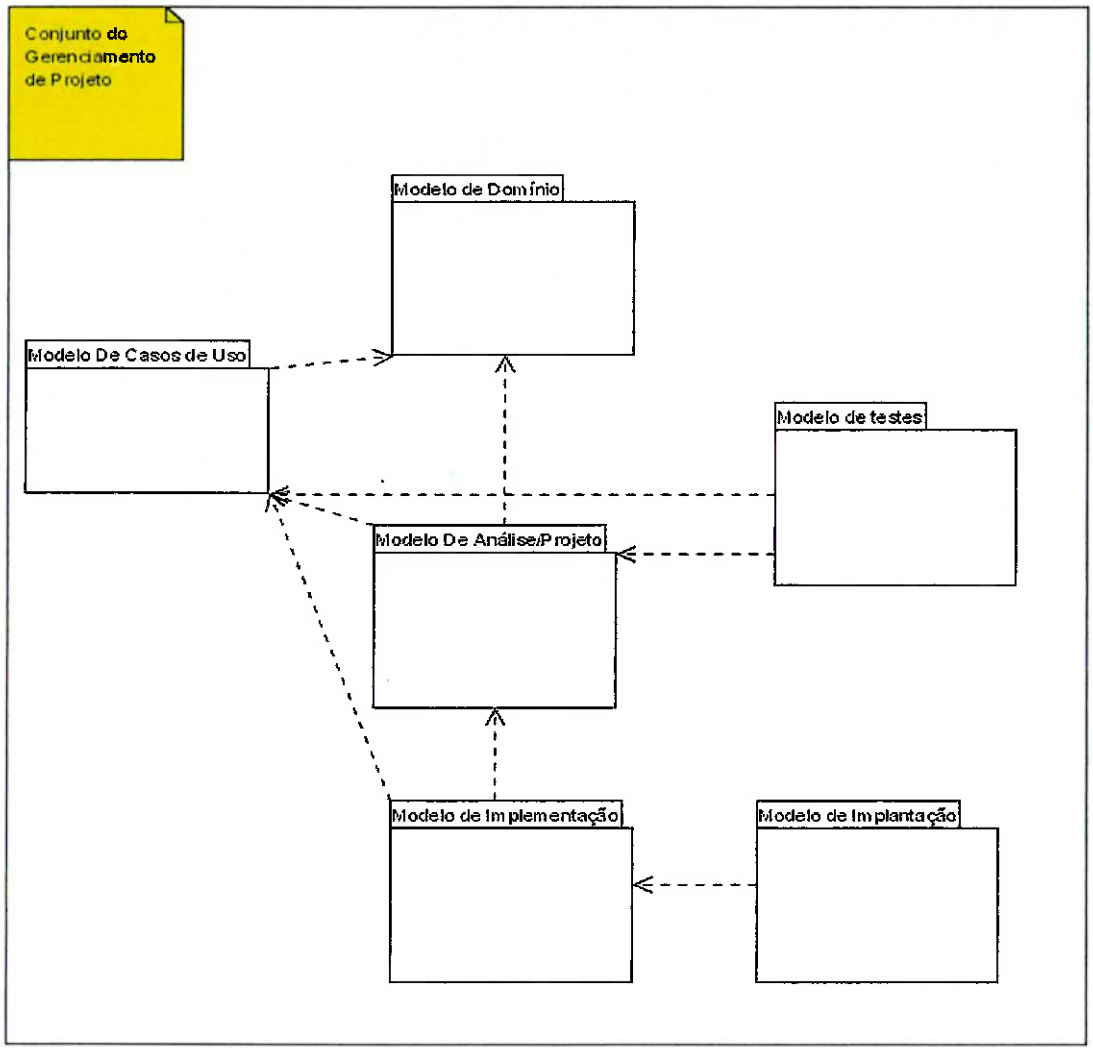

Figura 4.1.- Dependência dos Conjuntos de Artefatos (Conallen, J., 2002)

4.3 – Modelos e Documentos para Sistemas Simples e Complexos

De acordo com as características do sistema Web, existem necessidades diferentes em relação aos artefatos para cada tipo de projeto. Os projetos classificados como sistemas simples, necessitam dos artefatos simplificados, uma vez que o tempo é crucial, a arquitetura e a tecnologia empregadas são conhecidas e testadas em sistemas anteriores. Nos projetos classificados como sistemas complexos são necessários modelos com mais artefatos e com um nível maior no detalhamento desses, pois há uma necessidade natural pela complexidade do sistema de possuir mais informações sobre o negócio, arquitetura e tecnologia.

4.3.1 – Modelos Sistemas Simples

No desenvolvimento dos sistemas simples Web, alguns dos artefatos já mencionados podem ser resumidos ou supridos, geralmente o grau que alguns artefatos são resumidos ou supridos é em relação ao conhecimento do negócio, as tecnologias envolvidas, complexidade do sistema e o conhecimento do processo pelas equipes que participam do desenvolvimento do sistema, desde a equipe de produtos, responsável pela idéia do produto até a equipe de engenharia responsável pela sua implantação.

É de responsabilidade do gerente do projeto, junto com as equipes, definir quais os artefatos dentro de cada modelo necessários ao sistema, essa definição se dá logo após definir de uma forma gráfica todos os casos de uso do sistema através da visão do sistema. A fase de definição dos artefatos é importante pois se houver uma redução desnecessária dos artefatos, a fim de focalizar a fase de implementação, o gerenciamento das alterações de requisitos, que ocorrem no decorrer do projeto, pode ser prejudicada e conseqüentemente a qualidade do produto será inferior e o tempo de desenvolvimento será maior do que o previsto, por um outro lado se houver uma quantidade grande de artefatos, nos sistema simples, o tempo de desenvolvimento também pode se prolongar desnecessariamente.

Os artefatos, dentro dos modelos, necessários dentro de um sistema Web simples variam de caso a caso, mas os mais comuns são:

1. **Modelo de Domínio**

Ha definição do modelo de domínio e do glossário , o início de uma visão do produto, que será menos detalhada em relação a requisitos não funcionais do tipo segurança, arquitetura e tecnologia já conhecida.

2. **Modelo de Caso de Uso**

Sendo um dos modelos mais importantes dentro do desenvolvimento é necessário ter a representação gráfica de todos os casos de uso e os limites do sistema, pois mesmo a equipe já tendo a idéia dos casos de uso de um sistema simples é nessa fase junto com a visão do sistema que se comprova que se trata de um sistema simples.

A descrição dos fluxos dos casos de uso também é necessária pois todos os outros modelos necessitam dessas informações, analise implementação e testes .

3. **Modelo de Análise/Projeto**

Nessa fase tem a eliminação do protótipo de arquitetura, já que arquitetura estabelecida já é conhecida na sua maioria das vezes uma arquitetura no padrão MVC. Procura seguir um padrão na divisão dos subsistemas, se houver, geralmente divide o sistema em um subsistema necessário para administração do sistema, e outro subsistema aberto ao público da internet. Se houver a necessidade de outro tipo de sistema como, envio de e-mail e busca procura-se utilizar os sistemas já desenvolvidos dentro da empresa e apenas referenciá-los dentro do modelo de arquitetura.

Os diagramas de seqüências e de estado na maioria das vezes não são requeridos, pois as descrições dos casos de uso já são suficientes para o entendimento dos fluxos do sistema. Os diagramas de classes são necessários, pois a definição do que cada classe faz e com quem se relaciona é encontrado nesse diagrama.

Um padrão para divisão dos pacotes gerados na análise também é usado (modelos, dados,e apresentação), um protótipo básico da navegação UML é gerado pela equipe de produtos só para complemento das definições dos casos de uso junto com os subsistemas estabelecidos.

4. Modelo de Implementação

A codificação é necessária para qualquer tipo de desenvolvimento, mas os testes de unidade são feitos juntos com o desenvolvimento do sistema, é uma prática comum nos sistemas simples a fim de reduzir tempo no desenvolvimento.

5. Modelo de Teste

O foco do plano de testes no caso dos sistemas simples consiste nos testes funcionais e de demanda e desempenho, os testes de segurança são realizados mas são menos detalhados.

6. Modelo de Implantação

O documento de implementação é necessário, com todos os passos para a instalação do sistema, a diferença é que a implantação se da de uma forma simples pois todos os softwares usados na maioria das vezes são conhecidos pela equipe de engenharia, e com isso já se sabe o que se precisa de hardware para o sistema em questão

4.3.2 – Modelos Sistemas Complexos

Diferente dos sistemas simples os sistemas complexos utilizam a maioria dos artefatos dos modelos da UML para a Web, a fim de conseguir uma melhor documentação de um sistema que possui características particulares de seus requisitos.

As principais diferenças entre os modelos de um sistema simples e complexo estão nas fases de análise/projeto, de implementação e de testes.

1. **Modelo de Domínio**

   Todos os artefatos propostos em um sistema Web são utilizados, com ênfase na descrição dos requisitos não funcionais como a segurança, pelo documento da visão, glossário mais detalhado que um sistema simples.

2. **Modelo Casos de Uso**

   Artefatos completos com ênfase no levantamento de alguns casos de uso onde haverá a necessidade de realizar um protótipo executável para aprovação dos mesmos dentro do projeto, geralmente são alguns casos de uso que envolve alguma tecnologia diferente do padrão da empresa.

3. **Modelo de Análise/Projeto**

   Todos os diagramas, arquitetura, diagrama de classes, de análise, estado, de projeto, geralmente são utilizados e para cada caso de uso diagrama de seqüência e de estado. Nos sistemas complexos essa fase junto com o modelo de caso de uso é a principal em termos de horas de trabalho pois essa fase pode originar vários protótipos para aprovação de uma arquitetura, tecnologia ou subsistemas novos que deverão ser desenvolvidos para dar suporte ao sistema principal, por isso também a validação dos subsistemas por parte da equipe de produtos e pelo gerente de projeto é mais rígida através de um protótipo de navegação que é bem próximo das páginas do produto, sofrendo algumas alterações na fase de codificação do sistema.

4. **Modelo de Implementação**

   Além do código executável, teste de unidade, principalmente nas classes que representam as entidades que serão persistidas, em banco de dados ou arquivos.

De acordo com a necessidade do projeto pode se ter implementação dos protótipos definidos na fase de análise e projeto, antes de realizar qualquer a implementação do código referente ao projeto em si.

5. **Modelo de Teste**

Todos os testes são necessários dando ênfase nos testes de segurança e desempenho pela equipe de qualidade.

6. **Modelo de Implantação**

O documento de instalação do sistema é um documento dividido em etapas, devido à característica de se usar vários subsistemas distribuídos, por isso a implementação do sistema na maioria das vezes se da por etapas com validação e testes pela equipe de qualidade em cada etapa.

4.4 - Ferramentas e Linguagens utilizadas no processo de desenvolvimento WEB

Nas fases de concepção, elaboração, construção e transição são utilizadas ferramentas para a produção dos artefatos, e há uma infinidade de ferramentas, técnicas e linguagens.

Abaixo alguns exemplos de algumas das ferramentas e linguagens para cada conjunto de artefatos das fases de um processo de desenvolvimento Web.

1. **Artefatos do Modelo de Domínio**

Tanto o modelo de domínio, a visão e o glossário são documentos que podem ser gerados em qualquer editor de texto, o ideal é que a equipe de qualidade defina o modelo desses documentos, junto com os usuários, a fim de sempre ter um padrão na linguagem dos documentos, como já citado anteriormente o uso de uma intranet como forma de repositório dos documentos é totalmente válida no desenvolvimento das aplicações, essa prática só visa aumentar a velocidade com que as informações do projeto estejam disponíveis para as equipes.

2. Artefatos do Modelo de Casos de Uso, Análise e Projeto

O artefato de caso de uso, devido a sua importância, deve ser gerado em uma ferramenta que permita fazer o rastreamento do caso de uso com os demais artefatos gerados a partir dessa fase, tendo assim os artefatos de análise e projeto dentro da mesma ferramenta. Há no mercado várias ferramentas onde utilizam a linguagem UML para suas atividades, exemplos:

- Rational Rose – Rose Enterprise Edition - da IBM <http://www-3.ibm.com/software/rational/> Acesso em: 01 Nov. 2003.

- Together da Borland <http://www.borland.com.br/together/index.html>Acesso em: 01 Nov. 2003.

- Poseidon for Uml < http://www.gentleware.com> Acesso em: 01 Nov. 2003.

3. Artefatos do Modelo Implementação

Para a implementação do código do sistema as ferramentas dependem da linguagem que está sendo usada no projeto. exemplos:

Para a linguagem orientada a objetos Java temos:

- JBuilder - da Borland <http://www.borland.com.br/jbuilder/index.html > Acesso em: 01 Nov. 2003.

- Intellij Idea – da Intellij <http://www.intellij.com/idea/> Acesso em: 01 Nov. 2003.

- Eclipse < http://www.eclipse.org/> Acesso em: 01 Nov. 2003

Para a Linguagem Asp.Net e C# temos:

- .Net – Ambiente da Microsoft <http://www.microsoft.com/brasil/dotnet/default.asp> Acesso em: 01 Nov. 2003

Existem outras linguagens usadas no desenvolvimento Web como php, perl e coldFusion que são bastante usadas em sistemas Web.

4. Artefatos do Modelo de Testes:

O plano de teste e o resultado obtido da realização dos testes em cima da aplicação, podem ser realizados em qualquer editor de texto como forma de documentação, mas os testes dos sistemas Web, podem ser feitos com alguma ferramenta para automatização, pois como a iteração é uma característica predominante nos sistemas Web, a repetição dos testes durante o desenvolvimento de novas iterações ou alguma atualização é inevitável, exemplo:

- TestDirector - da Mercury <http://www.mercuryinteractive.com/products/testdirector/index.html> Acesso em: 01 Nov. 2003

5. Artefatos do Modelo Implantação

Além de utilizar uma ferramenta que utilize a UML para validar os componentes de instalação de uma aplicação a Ficha técnica do Sistema, que explica passo a passo a instalação do sistema pode ser desenvolvida em qualquer editor de texto. Ao instalar um sistema é interessante usar uma ferramenta que controle as versões dos componentes a serem instalados, e acessando um único servidor que contenha todas as versões dos componentes, a serem usados nas aplicações inclusive o código da própria aplicação, exemplo:

- Rational Clear Case – da IBM <http://www-3.ibm.com/software/awdtools/clearcase/index.html>. 2003.

A utilização de determinada ferramenta ou linguagem para o desenvolvimento dos artefatos se baseia nas tecnologias e padrões que a empresa emprega no desenvolvimento das aplicações.

Os artefatos necessários para o desenvolvimento de uma aplicação Web devem ser necessários para que se entenda e se consiga rastrear os requisitos do sistema, quanto mais complexo o sistema, naturalmente mais artefatos, e com mais detalhes, são necessários para a sua documentação.

5 – CRÍTICA NA ADOÇÃO DO PROCESSO UNIFICADO EM SISTEMAS WEB.

Os itens analisados nesse trabalho mostram que um processo, no caso o unificado, não pode apenas ser empregado sem passar por um período adaptativo. Em uma empresa voltada ao desenvolvimento Web, devido a sua dinâmica, é necessário antes ter um estudo dos tipos de sistema que a empresa produz e do perfil das equipes envolvidas nessa produção, antes de estabelecer um processo de desenvolvimento.

Para que um processo funcione dentro de uma empresa é preciso que o mesmo tenha o foco no produto final e não que a empresa tenha foco no processo, para isso ocorrer à comunicação entre as equipes deve ser feita de uma forma flexível, ou seja, evitar que a comunicação seja baseada nos artefatos gerados pelo processo. Com uma comunicação onde se tem um maior envolvimento dos operadores das equipes do projeto o processo ganha em qualidade e em velocidade de desenvolvimento.

Além de se ter uma comunicação flexível é preciso que o processo seja configurável, ou seja, o conjunto de artefatos deve variar para cada tipo de sistema e prioridade do projeto.

A decisão de quais são os artefatos são necessários em um projeto deve partir dos gerentes de projeto e não em relação ao processo ou de acordo com a necessidade de cada equipe. O processo apenas fornece um guia de quais artefatos são esperados de acordo com as características do sistema e quais artefatos cada equipe necessita, mas nada deve impedir que o gerente possa remover ou acrescentar determinado artefato no decorrer do ciclo, desde que não comprometa o entendimento dos requisitos pelos membros das outras equipes.

Para que o processo tenha a flexibilidade de gerar os artefatos necessários em cada projeto, é preciso que se tenha uma avaliação do processo, ou seja se o uso de determinados artefatos são suficientes para o projeto com determinada característica. A responsabilidade de fazer essa avaliação continua do processo é da equipe de

qualidade, por isso essa equipe deve estar comprometida com os projetos desde o início, levantamento e descrição dos casos de uso, até a homologação dos sistemas em produção.

Se não houver uma participação da equipe de qualidade desde do inicio do projeto, além da empresa não conseguir avaliar o processo em todas as suas fases, há um aumento do tempo da realização da fase de testes e com uma qualidade inferior da esperada, pois nesse caso pode ocorrer da equipe de qualidade não avaliar e testar requisitos não descritos nos casos de uso.

6 – CONCLUSÃO

No desenvolvimento de sistemas Web, é necessário o uso de um processo para garantir que o sistema seja desenvolvido com qualidade e no tempo adequado para atender as expectativas do mercado. Um processo flexível e configurável, são características necessárias para acompanhar a dinâmica do desenvolvimento de sistemas Web. Portanto um processo, como o processo unificado, com as devidas adaptações, considerando principalmente as características dos sistemas e os diferentes perfis dos operadores das equipes de uma empresa.

Um processo para conseguir aceitação das diferentes equipes e uma estabilidade dentro de uma empresa, precisa facilitar o trabalho dos membros dessas equipes, e não fazer as equipes trabalhar para o processo, para que isso ocorra o processo deve ter uma melhoria continua, com participação de todas equipes através de sugestões nos pontos falhos do processo, a cada término de projeto.

Portanto, para atingir as metas definidas pela empresa, as expectativas dos integrantes das equipes, ou seja, da gerência, do desenvolvimento, de testes, é preciso que as diferentes atividades que compõem o desenvolvimento de um sistema estejam integradas. O processo unificado permite a seleção dos artefatos necessários para cada tipo de sistema, simples ou complexo, além de permitir a inclusão ou exclusão de fases, assim como definição do número de iterações.

Para trabalhos futuros é importante analisar de uma forma mais detalhada a integração dos diferentes perfis das equipes, quais seriam os mecanismos necessários e desejáveis para melhorar a comunicação entre as equipes. Além disso, deve ser destacado o papel do grupo de garantia de qualidade ao longo de todo o processo de desenvolvimento.

7 – BIBLIOGRAFIA

AMJAD UMAR **Object – Oriented Client/Server internet Enviroment .** Ed. Prentice Hall PTR, 1997

BOOCH, G; RUMBAUCH, J; JACOBSON, I. **The Unified Modelling Language User Guide**. Ed. Addilson Wesley, 1999

BOOCH, G; RUMBAUCH, J; JACOBSON, I. **The Unified Software Development Process**. Ed. Addilson Wesley, 1999

EFRAIM TURBAN; JAE LEE; DAVID KING; H. MICHAEL CHUNG. **Electronic Commerce A Managerial Perspective,** Prentice Hall 2000.

GERALD KOTONYA; IAN SOMMERVILLE. **Requirements Engineering**, Willey 1998.

JAVA; **Essential Java Classes** .Disponível em: http://java.sun.com/docs/books/tutorial/essential/index.html. Acesso 23, jun 2003
http://java.sun.com/blueprints/patterns/MVC-detailed.html Acesso 23, jun 2003
http://java.sun.com/ejb/index.htm Acesso 23, jun 2003.

JEFF OFFUTT, **Quality Attributes of Web Software Applications.** IEEE 2002

JIM CONALLEN, **Building Web Applications with Uml** . Ed. Addilson Wesley, 2002

LARRY ULLMAN, **Php para a World Wide Web,** Ed. Campus, 2001

LARRY WALL, **Programação Perl,** Ed. Campus 2001

MELISSA L. RUSS; JOHN D. MCGEGOR, **A software Development Process for Small Projects.** IEEE 2000

PHILIPPE KRUTCHEN, **The Rational Unified Process. Ed.** Ed. Addilson Wesley, 1999

ROGER S. PRESSMAN, **Software Engineering A Practitioners´s Approach** Ed. McGraw-Hill, 2001

SCOTT W. AMBLER; LARRY L. CONSTANTINE, **The Unified Process Inception Phase** CPM Books 2000

THOMAS J. HALLEY, **Software Process Improvement at Raytheon,** IEEE 1996

WEBSERVICES; **Web Services Activity**. Disponível em: http://www.w3.org/2002/ws/. Acesso em 10, nov. 2002.

OMG; **UML**. Disponível em http://www.omg.org/uml/. Acesso em 01 Dez. 2003

RATIONAL ROSE; **Rose Enterprise Edition - da IBM.** Disponível em: http://www-3.ibm.com/software/rational/. Acesso em: 01 Nov. 2003.

TESTDIRECTOR; TestDirector da Mercury. Disponível em: http://www.mercuryinteractive.com/products/testdirector/index.html . Acesso em 01 Nov.2003

.NET; **Ambiente da Microsoft.** Disponível em: http://www.microsoft.com/brasil/dotnet/default.asp . Acesso em 01 Nov. 2003

JBUILDER; **Jbuilder da Borland.** Disponível em: http://www.borland.com.br/jbuilder/index.html . Acesso em: 01 Nov. 2003.

INTELLIJ IDEA; **Intellij Idea – da Intellij** . Disponível em: http://www.intellij.com/idea/ .Acesso em 01 Nov 2003.

ECLIPSE; **Eclipse** . Disponível em: http://www.eclipse.org/ . Acesso em 01 Nov.2003

TOGETHER; **Together da Borland.** Disponível em: http://www.borland.com.br/together/index.htmlAcesso em 01 Nov. 2003.

POSEIDON; **Poseidon for Uml.** Disponível em: http://www.gentleware.com Acesso em 01 Nov. 2003.