Curso de Sistemas de Informação
Disciplina: Gestão de Sistemas Digitais na WEB
Professor: José Maurício S. Pinheiro

AULA 3 – Modelagem, Construção e Implantação de Webapp

Com o crescimento da utilização da Internet as empresas começaram a publicar serviços e trocar informações entre aplicações na Web. A necessidade de uma padronização motivou o surgimento de várias tecnologias, entre elas, os serviços Web, que provêm uma forma mais estruturada de comunicação entre os pontos finais de comunicação, conforme mostra a Figura 1.

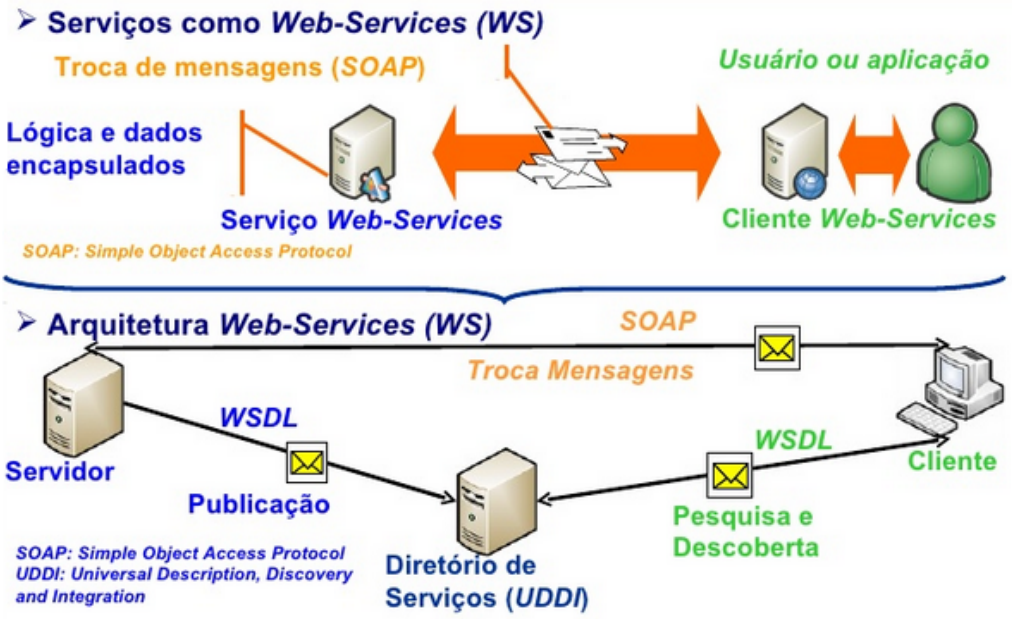

Figura 1 – Arquitetura de serviços Web

As aplicações acessam os serviços Web descritos na linguagem WSDL (*Web Services Description Language*), utilizando um serviço de diretório (repositório de informações sobre os serviços Web) UDDI (*Universal Description, Discovery and Integration*) ou diretamente quando possui o conhecimento da sua URL (*Uniform Resource Location*). Os dados são empacotados usando o protocolo de troca de mensagens SOAP (*Simple Object Access Protocol*), possuindo as características de formatação da linguagem XML (*Extended Markup Language*) para representação dos dados. O protocolo SOAP surgiu como uma alternativa aos problemas causados pelo aumento do número de portas de comunicação usadas nas aplicações distribuídas e nos serviços dos sistemas operacionais, devido a necessidade de se estabelecer uma comunicação entre os extremos usando um protocolo da camada de transporte.
Com a utilização do protocolo HTTP para o tráfego de informações, o SOAP facilitou o uso e a disseminação dos serviços Web, pois as portas de comunicação do protocolo HTTP já estão configuradas para o acesso ao conteúdo de sites. Os dados então são enviados dos servidores Web, onde residem os serviços, usando

o protocolo HTTP para os clientes que os requisitam através de browsers, aplicações que utilizam middlewares específicos ou outros serviços Web.

1. Características e Categorias das Webapp

Uma aplicação Web é um software de computador no sentido que é uma coleção de instruções executáveis e dados que oferecem informações e funcionalidades para usuários finais e suas características são diferentes das características de um software convencional. Por exemplo, a aplicação Web é uma aplicação que permite a um navegador (cliente) acessar páginas de hipertextos armazenadas em um servidor Web (Figura 2).

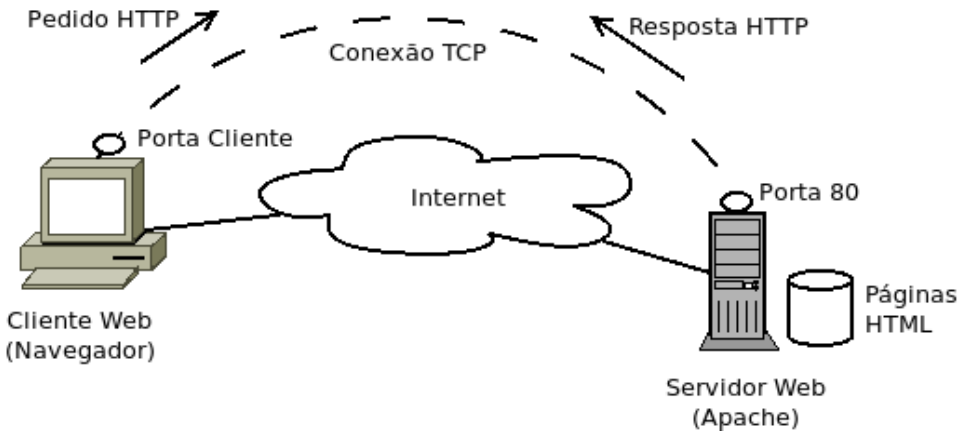

Figura 2 - Exemplo de aplicação Web

As características e categorias encontradas na grande maioria das aplicações Web são enumeradas por (PRESSMAN & LOWE, 2009):

- **Concentração na rede** - Cada aplicação Web está disponível numa rede de computadores e deve servir às necessidades de uma comunidade diversificada de clientes. A rede pode ser: Internet, Intranet ou Extranet;
- **Concorrência -** Um grande número de usuários poderá acessar a aplicação Web ao mesmo tempo e os padrões de acesso podem variar. Em alguns casos, as ações de um usuário, ou um conjunto destes, podem ter um impacto sobre as ações ou informações apresentadas a outros usuários;
- **Carga imprevisível** - O número de usuários da aplicação Web poderá variar por ordens de grandeza de um dia para outro;
- **Desempenho** - Se um usuário da aplicação tiver que esperar muito tempo para acesso, para processamento no lado do servidor, para formatação e exibição no lado do cliente, ele poderá desistir de utilizar a aplicação;
- **Disponibilidade** - Embora uma disponibilidade de 100% não seja realista, os usuários de aplicações Web normalmente exigem acesso 24 horas por dia, 7 dias por semana, 365 dias por ano;
- **Orientada a dados** - A função principal das aplicações Web é usar hipermídia para apresentar conteúdo de texto, gráficos, áudio e vídeo ao usuário final. Além disso, as aplicações Web normalmente são usadas para acessar informações que existem em bancos de dados legados (por exemplo, aplicações de comércio eletrônico ou financeiras);

- **Sensível ao conteúdo** - A qualidade e a natureza estética do conteúdo são importantes para a qualidade de uma aplicação Web;
- **Evolução contínua** - Diferente do software de aplicação convencional, que evolui por uma série de versões planejadas, cronologicamente espaçadas, aplicações Web evoluem continuamente. Não é raro que algumas aplicações Web (especificamente, seu conteúdo) sejam atualizadas minuto a minuto ou por conteúdo a ser dinamicamente produzido a cada solicitação;
- **Urgência** - Embora a urgência, a necessidade imprescindível de colocar o software no mercado rapidamente, seja uma característica de muitos domínios de aplicação, as aplicações Web normalmente exibem um tempo para o mercado que pode ser uma questão de alguns dias ou semanas. Os desenvolvedores Web devem usar métodos para planejamento, análise, projeto, implantação e testes que têm sido adaptados a cronogramas reduzidos, exigidos para o desenvolvimento da aplicação Web;
- **Segurança** - Como as aplicações Web estão disponíveis na rede, é difícil limitar o número de usuários finais que podem acessar a aplicação. Para proteger conteúdo confidencial e oferecer modos seguros de transmissão de dados, medidas de segurança fortes precisam ser implementadas por meio da infraestrutura que apoia a aplicação Web e na aplicação propriamente dita;
- **Estética** – Um ponto forte de uma aplicação Web é sua aparência. Quando uma aplicação tiver sido criada para comercializar ou vender produtos ou ideias, ou fornecer serviços que geram receita, a estética pode estar relacionada tanto ao sucesso quanto ao projeto técnico.

Quanto aos problemas que aplicações Web tratam, as seguintes categorias de aplicação são as mais encontradas (PRESSMAN, 2006):

- **Aplicações informacionais** – conteúdo somente de leitura é fornecido com navegação e ligações (links) simples;
- **Aplicações para download** – o usuário faz download de um servidor;
- **Aplicações personalizáveis** – o usuário adapta o conteúdo a necessidades específicas;
- **Aplicações de interação** – a comunicação entre usuários ocorre por intermédio de salas de bate-papo, quadros de aviso ou mensagens instantâneas;
- **Aplicações de entrada do usuário** – entrada baseada em formulários é o principal mecanismo para comunicar o propósito do usuário final;
- **Aplicações orientadas a transação** – o usuário faz uma solicitação (por exemplo, coloca um pedido) que é atendida pela aplicação Web;
- **Aplicações orientadas a serviços** – a aplicação fornece um serviço ao usuário (por exemplo, ajuda o usuário a calcular o valor da parcela do pagamento de um empréstimo);
- **Portais** – a aplicação direciona o usuário para outros conteúdos ou serviços da Web fora do domínio de aplicação do portal;
- **Acesso a banco de dados** – o usuário consulta uma grande base de dados e extrai a informação desejada;
- **Armazém de dados (*Data Warehouse*)** – o usuário consulta uma coleção de grandes bancos de dados e extrai a informação.

Uma aplicação Web pode se enquadrar em mais de uma categoria, bem como trocar de categoria ao longo de seu tempo de vida. A Figura 3, resume as principais características e categorias de aplicações Web.

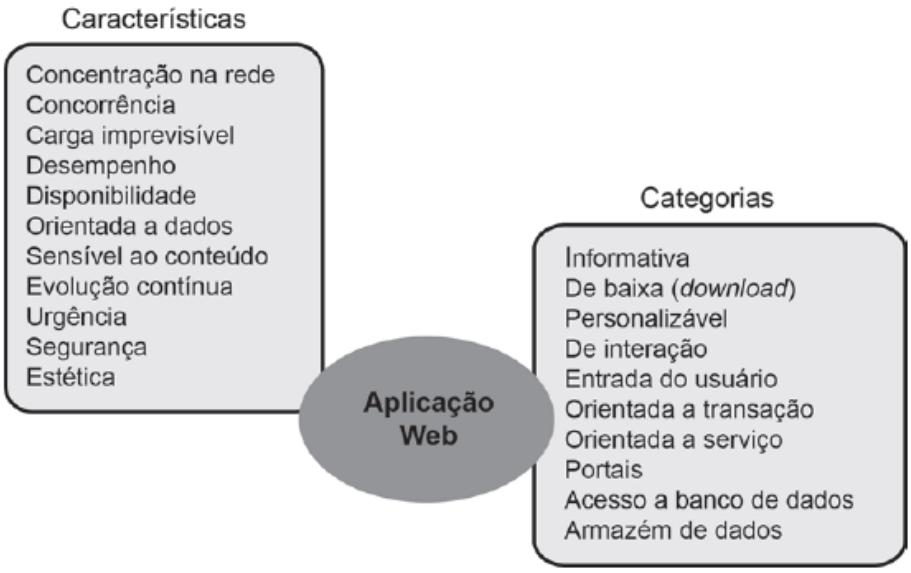

Figura 3 - Características e categorias de aplicações Web

1.1. Formulação e Análise

Formulação e Análise de sistemas e aplicativos para a Web representam uma sequência de atividades de engenharia que começam com a identificação das metas e objetivos do aplicativo e terminam com o desenvolvimento de um modelo de análise ou especificação de requisitos para o sistema.
A Formulação permite que o cliente e o desenvolvedor estabeleçam um conjunto comum de metas e objetivos para a construção do aplicativo. Ela também ajuda a identificar o escopo do trabalho de desenvolvimento e fornece meios de determinar o sucesso do projeto.
Análise é uma atividade técnica que identifica dados, funcionalidades e requisitos comportamentais de um aplicativo.

1.1.1. Formulação

As seguintes questões devem ocorrer no primeiro passo da etapa de formulação:

- Qual o principal motivo para desenvolver o aplicativo;
- Porque o aplicativo é necessário;
- Quem será o usuário do aplicativo.

A resposta para cada uma dessas questões deve ser determinada de maneira bem sucinta e objetiva. Através delas são identificadas as metas. Há basicamente duas categorias de metas:

- **Metas de informação** - Indicam a intenção de fornecer conteúdo específico e/ou informação para o usuário;
- **Metas de aplicativo** - Indicam a habilidade de executar tarefas do aplicativo.

Quando todas as metas de ambos os tipos forem identificadas, um perfil de usuário é desenvolvido. Este perfil captura características dos usuários potenciais incluindo suas experiências, conhecimentos, preferências, etc.
Quando todas as metas e perfis de usuários estiverem desenvolvidos, a atividade de formulação irá focar a declaração de escopo do aplicativo para a Web. Em muitos casos, as metas desenvolvidas estão integradas com esta declaração de escopo. Também é importante, neste estágio, indicar os graus de integração esperados e restrições de conectividade.

1.1.2. Análise

Durante esta etapa, quatro diferentes tipos de análises são conduzidos:

- **Análise de Conteúdo** - todo o conteúdo a ser fornecido pelo aplicativo é identificado. Inclui textos, gráficos e imagens, dados de áudio e vídeo;
- **Análise de Interação** - a maneira pela qual o usuário interage com o aplicativo é descrita em detalhes;
- **Análise Funcional** - os cenários de uso criados na análise de interação irão definir operações que irão ser utilizadas no aplicativo, que implicam outras funções de processamento. Todas as operações e funções são descritas em detalhes;
- **Análise de Configuração** - o ambiente e a infraestrutura na qual o aplicativo reside são descritos em detalhes. O aplicativo pode estar na Internet, Intranet ou Extranet.

2. Agilidade e Arcabouço de Processo

A Engenharia Web pode ser definida como a utilização de princípios, conceitos e métodos da Engenharia de Software, de forma que se adaptem às características inerentes de aplicações Web, tais como desempenho, disponibilidade, evolução contínua, urgência e segurança. Dois conceitos-chave são importantes: Agilidade e Arcabouço de Processo.
Um arcabouço estabelece a base para um processo completo de Engenharia Web, se resume em um conjunto de tarefas que produz um produto, cada ação é preenchida com tarefas individuais, que executam alguma parte do trabalho conforme pode ser observado na Figura 4.
Segundo Pressman (2006), as atividades a seguir fazem parte de um arcabouço genérico e ocorrem na maioria dos projetos Web:

- **Comunicação** - engloba a comunicação com o cliente e outros interessados abrangendo o levantamento de requisitos e atividades relacionadas;
- **Planejamento** - estabelece as ações da Engenharia Web que ocorrerão, as tarefas e técnicas a serem utilizadas, os riscos prováveis, os recursos que serão exigidos, os produtos de trabalho a serem produzidos e um cronograma a ser seguido;

- **Modelagem** - envolve a criação de modelos que auxiliam o desenvolvedor e cliente a entenderem melhor os requisitos da WebApp e o projeto que satisfará esse requisito;
- **Construção** - envolve a combinação de HTML, XML, PHP, ou código semelhante, juntamente com o teste necessário para revelar erros no código;
- **Implantação** - entrega um incremento da WebApp ao cliente, que avalia e oferece com base na avaliação;

Essas atividades podem ser utilizadas durante todo o processo de desenvolvimento de uma aplicação Web de qualquer tamanho e complexidade. Os detalhes específicos de cada projeto podem ser diferentes, mas o "esqueleto" principal é o mesmo.

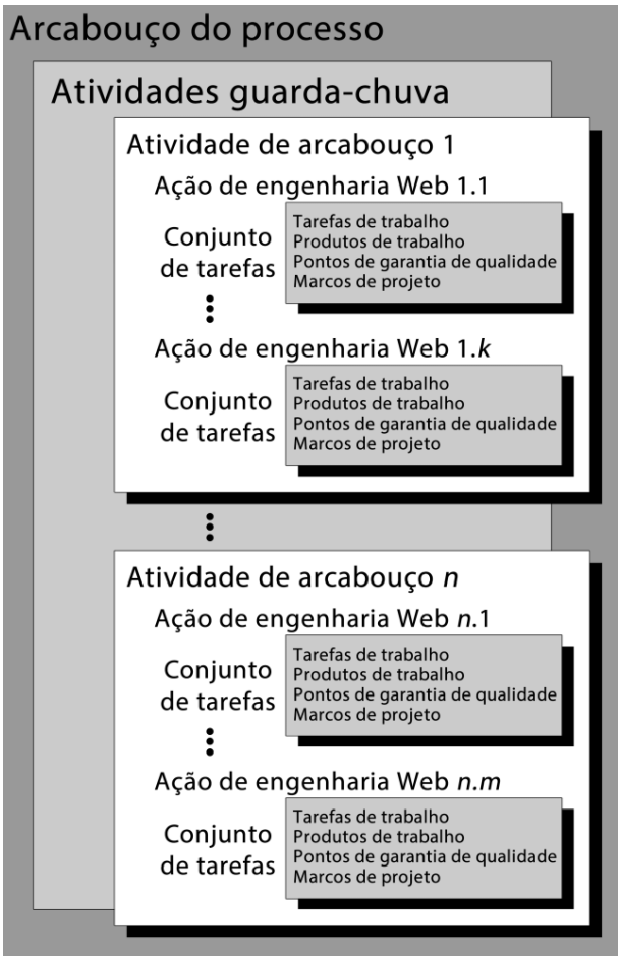

Figura 4 - Processo de engenharia Web

2.1. Agilidade de processo

Agilidade de processo implica um enfoque de Engenharia de Software enxuto que incorpora ciclos de desenvolvimentos rápidos. Cada ciclo resulta na implantação de um incremento da aplicação Web. Isto é, o desenvolvimento da aplicação Web é realizado por meio da entrega de uma série de versões chamadas de incrementos,

que fornecem progressivamente mais funcionalidades para os clientes à medida que cada incremento é entregue.
Metodologias ágeis como *Extreme Programming* (XP) e Scrum são exemplos de modelos de processo que aplicam o conceito de agilidade de processo no desenvolvimento de sistemas de software.

2.2. Arcabouço de processo

O arcabouço de processo estabelece a base para um processo de desenvolvimento completo, identificando um pequeno número de atividades que se aplicam a todos os sistemas, independentemente de tamanho e complexidade.
Alguns aspectos presentes na maioria dos projetos de desenvolvimento de aplicações Web que foram considerados na concepção do arcabouço proposto por (PRESSMAN & LOWE, 2009) são os seguintes:

- **Requisitos evoluem com o tempo** - Quando se inicia um projeto de desenvolvimento Web, pode haver incerteza sobre alguns elementos da estratégia de negócios, do conteúdo e da funcionalidade a ser entregue, questões de interoperabilidade e outros aspectos do problema;
- **As mudanças ocorrem com frequência** - A incerteza é parte inerente da maioria dos projetos de desenvolvimento Web e as mudanças nos requisitos são comuns. Além disso, o feedback do usuário (baseado na avaliação dos incrementos entregues) e alterações nas condições de negócios podem ocasionar mudanças;
- **Linhas de tempo são curtas** - Isso alivia a criação de documentação de engenharia volumosa, mas não impede a necessidade de documentar minimamente: a análise do problema, o projeto e o teste.

Considerando esses aspectos, o arcabouço proposto sugere o emprego de uma abordagem incremental, ou seja, atividades de arcabouço ocorrerão repetidamente à medida que cada incremento de aplicação Web for desenvolvido e entregue.
O arcabouço proposto contém cinco atividades principais (Figura 5):

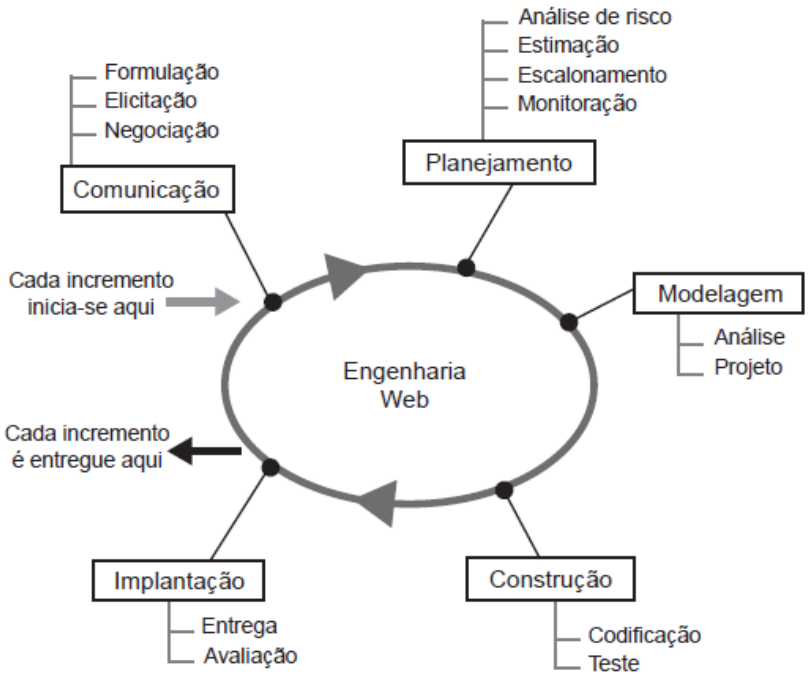

Figura 5 - Arcabouço de Processo da Engenharia Web (PRESSMAN & LOWE, 2009)

- **Comunicação** - envolve interação e colaboração intensas com o cliente (e outros interessados) e abrange o levantamento de requisitos e outras atividades relacionadas;
- **Planejamento** - estabelece um plano incremental para o trabalho de desenvolvimento;
- **Modelagem** - abrange a criação de modelos (por exemplo, diagramas de classes) que auxiliam o processo de desenvolvimento;
- **Construção** - combina a geração de código e o teste necessário para revelar erros no código;
- **Implantação** - consiste na entrega de um incremento ao cliente, que o avalia e oferece feedback com base nessa avaliação.

2.2.1. Comunicação

Entre os princípios mais fundamentais da Engenharia de Software está: “entenda o problema antes de começar a resolvê-lo e certifique-se de que a solução que você concebe é aquela que as pessoas realmente desejam” (PRESSMAN, 2006).
A atividade de comunicação oferece à equipe de Engenharia Web um modo organizado de levantar requisitos dos interessados e é caracterizada por três ações: formulação, elicitação e negociação (PRESSMAN & LOWE, 2009):

- **Formulação** - define o contexto organizacional e de negócios para a aplicação Web. Além disso, os interessados são identificados e mudanças ou requisitos em potencial no ambiente da organização são previstos. A integração entre a aplicação Web e outras aplicações de negócios, bancos de dados e funcionalidades é definida;
- **Elicitação** - atividade de coleta de requisitos que inclui todos os interessados. A intenção é descrever o problema que a aplicação Web

deverá resolver (junto com requisitos básicos para a aplicação Web) usando a melhor informação disponível. Além disso, tenta-se identificar áreas de incerteza e lugares em que ocorrerão mudanças em potencial;
- **Negociação -** normalmente é exigida para conciliar diferenças entre os diversos interessados no projeto. Durante o processo de negociação, pode-se pedir que um interessado equilibre funcionalidade, desempenho e outras características do produto ou sistema contra custo e tempo para entrega.

2.2.2. Planejamento

Outro princípio fundamental da Engenharia de Software é: “planeje o trabalho antes de começar a realizá-lo” (PRESSMAN, 2006, p. 389). A atividade de planejamento oferece à equipe de Engenharia Web um modo organizado de planejar o desenvolvimento dos incrementos da aplicação Web e é caracterizado por quatro ações: análise de risco, estimação, escalonamento e monitoração.
O planejamento é realizado por todos os membros da equipe de Engenharia Web e é coordenado pelo líder de produto. A auto-organização implica em três características:

1. A equipe de Engenharia Web se organiza para o trabalho a ser feito;
2. A equipe organiza o Arcabouço de Processo para acomodar melhor seu ambiente local;
3. a equipe organiza o cronograma de trabalho para conseguir entregar o incremento da aplicação Web.

2.2.3. Modelagem

A atividade de modelagem cria uma ou mais representações conceituais de algum aspecto da aplicação Web a ser construída. Uma representação conceitual abrange um ou mais dos seguintes artefatos: documentos escritos, esboços, diagramas esquemáticos, modelos gráficos, cenários escritos ou protótipos em papel ou executáveis (PRESSMAN & LOWE, 2009). A intenção da atividade de modelagem é desenvolver modelos de análise e projeto que definam requisitos e, concomitantemente, representem a aplicação Web que os implementem.
Duas ações de Engenharia Web ocorrem durante a modelagem: análise e projeto. Ambas as ações têm como objetivo a produção de modelos (PRESSMAN & LOWE, 2009):

- **Modelagem de análise** - ajuda a entender a natureza do problema a ser resolvido e a moldura da aplicação Web que lhe permitirá resolver esse problema;
- **Modelagem de projeto** - trata de compreender a estrutura interna da aplicação Web sendo desenvolvida e como isso cria a moldura da aplicação Web que foi identificada pelo modelo de análise.

2.2.3.1. Modelagem de análise

A modelagem de análise tem como objetivo fornecer um mecanismo para representar e avaliar o conteúdo e a funcionalidade da aplicação Web, os modos de interação que os usuários encontrarão e o ambiente e a infraestrutura em que a aplicação Web está hospedada.

Embora os modelos específicos dependam da natureza da aplicação Web, podemos identificar quatro classes principais de modelos de análise:

- **Modelo de interação** - descreve a maneira como os usuários interagem com a aplicação Web. composto de três elementos: casos de uso, diagrama de sequência e diagramas de estado;
- **Modelo de conteúdo** - identifica o espectro completo do conteúdo a ser fornecido pela aplicação Web. O conteúdo inclui dados de texto, gráficos, imagens e áudio/vídeo;
- **Modelo funcional** - define as operações que serão aplicadas ao conteúdo da aplicação Web e descreve outras funções de processamento que são independentes do conteúdo, mas necessárias para o usuário final. Segundo Pressman (2006), o modelo funcional atende a dois elementos do processamento da aplicação Web, cada um representando um nível diferente de abstração procedimental: funcionalidade que é disponibilizada pela aplicação Web aos usuários finais, que engloba qualquer funcionalidade iniciada pelo usuário final e as operações contidas nas classes de análise que implementam comportamentos (manipulando atributos) associados à classe;
- **Modelo de configuração** - descreve o ambiente e a infraestrutura em que a aplicação Web está hospedada. Segundo Pressman (2006), é uma lista de atributos no lado do servidor e no lado do cliente. Para aplicações Web mais sofisticadas, complexidades de configuração (por exemplo, distribuição de carga entre múltiplos servidores, banco de dados remotos) podem ter impacto sobre a análise e projeto. Nesses casos, diagramas de implantação UML podem ser utilizados para representar situações em que arquiteturas de configuração complexas são necessárias.

Cada um desses modelos pode ser desenvolvido usando um esquema de representação que permite que sua intenção e estrutura sejam comunicadas e avaliadas com facilidade entre os membros da equipe de engenharia Web e outros interessados.

2.2.3.2. Modelagem de projeto

A modelagem de projeto engloba atividades técnicas e não técnicas. A aparência do conteúdo é desenvolvida como parte do projeto gráfico, o layout de estética da interface com o usuário é criado como parte do projeto da interface, e a estrutura técnica da aplicação Web é modelada como parte do projeto arquitetural e navegacional (PRESSMAN, 2006), conforme mostra a Figura 6.

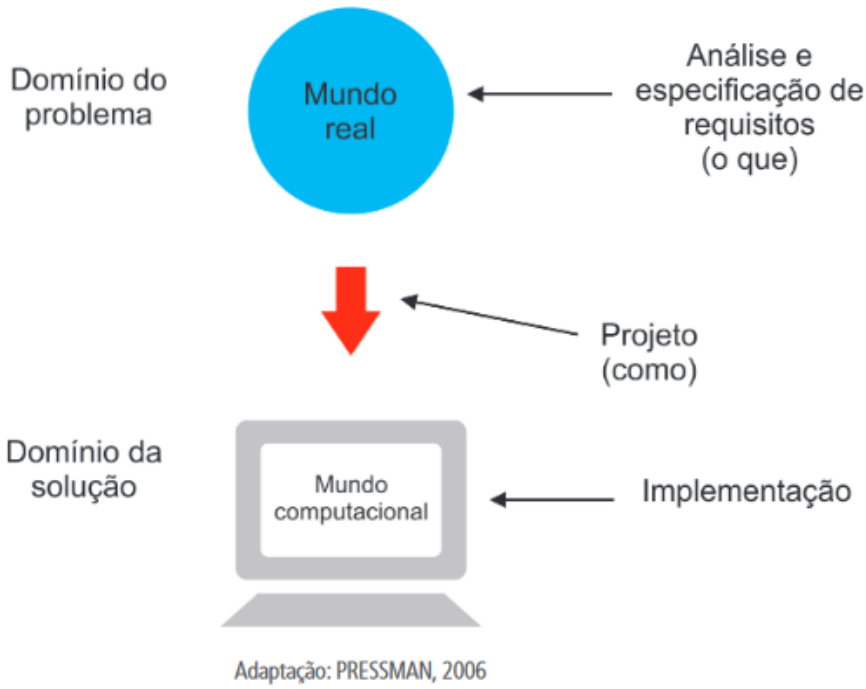

Figura 6 - Fase de projeto

2.2.3.3. Modelagem de Requisitos

A modelagem de requisitos é a representação do que se pode abstrair do mundo real e que permite descrever ou mesmo de alguma forma prever comportamentos específicos de um sistema por meio da observação de certas características aparentes e importantes. Uma visão de modelagem de requisitos considera os dados e os processos que os transformam como entidades distintas. Os objetos de dados são modelados de maneira a definir seus atributos e relacionamentos.
A análise de requisitos de WebApps engloba três tarefas principais: formulação, coleta de requisitos e modelagem de análise. Durante a formulação, a motivação básica (metas) e os objetivos para a WebApp são identificados e as categorias de usuários definidas (PRESSMAN, 2011).
O ANEXO I apresenta graficamente um resumo contendo a modelagem de requisitos: Fluxo, Comportamento, Padrões e Aplicações na Web.
Para Pressman (2011), a modelagem de análise leva à obtenção de cada um dos elementos. Entretanto, o conteúdo específico de cada elemento (os diagramas usados para construir o elemento e o modelo) pode diferir de projeto para projeto conforme a Figura 7.

Fonte: adaptado de Pressman (2011)

Figura 7 - O conteúdo de cada elemento pode diferir de projeto para projeto

- Elementos baseados em cenários representam como o usuário interage com o sistema e a sequência específica de atividades que ocorre à medida que o software é utilizado;
- Elementos baseados em classes modelam os objetos que o sistema irá manipular, as operações que serão aplicadas aos objetos para efetuar a manipulação, os relacionamentos entre os objetos e as colaborações que ocorrem entre as classes definidas;
- Elementos comportamentais representam como eventos externos mudam o estado ou as classes que neles residem;
- Elementos orientados a fluxos representam o sistema como uma transformação de informações, indicando como os objetos de dados são transformados à medida que fluem pelas várias funções do sistema.

Pressman & Lowe (2009) apresentam uma pirâmide de projeto mostrada na Figura 8, que contém as ações de projeto (níveis) que ocorrem quando um modelo de projeto completo é criado.

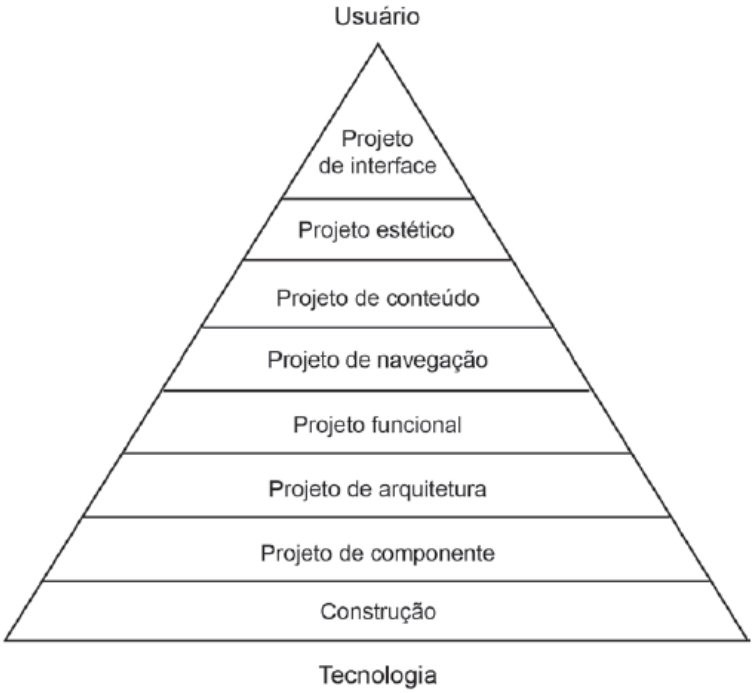

Figura 8 - Pirâmide de projeto de Aplicação Web

Constam dessa pirâmide:

- **Projeto de interface** – descreve a estrutura e a organização da interface com o usuário. Ele inclui uma representação do layout da tela, definição dos modos de interação e descrição dos mecanismos de navegação. toda interface de usuário deverá apresentar as seguintes características: ser fácil de usar, ser fácil de aprender, ser fácil de navegar, ser intuitiva, ser consistente, eficaz, livre de erros e funcional. Ela deve oferecer ao usuário final uma experiência satisfatória e recompensadora (PRESSMAN & LOWE, 2009);
- **Projeto estético** – também chamado de projeto gráfico, descreve a aparência e o estilo da aplicação Web e envolve duas atividades: projeto do layout e projeto gráfico. Ele inclui esquemas de cor, layout geométrico, tamanho de texto, fonte e posicionamento, o uso de figuras gráficas e decisões estéticas relacionadas. Importante salientar que toda página Web tem uma quantidade limitada de espaço que pode ser usado para apoiar recursos de navegação, conteúdo informativo e funcionalidade controlada pelos usuários e o planejamento desse espaço é realizado durante o projeto estético;
- **Projeto de conteúdo** – define o layout, a estrutura e o esboço para todo o conteúdo que é apresentado como parte da aplicação Web. Ele estabelece os relacionamentos entre os objetos de conteúdo. O relacionamento entre os objetos de conteúdo definidos como parte do modelo de análise e os objetos de projeto que representam conteúdo é análogo ao relacionamento de classes de análise e classes de projeto do paradigma orientado a objetos. Um objeto de conteúdo tem atributos que incluem informação específica de conteúdo (normalmente definida durante a modelagem de análise) e

atributos específicos de implementação, que são especificados como parte da modelagem de projeto;

- **Projeto de navegação** – representa o fluxo de navegação entre os objetos de conteúdo e para todas as funcionalidades da aplicação Web. O projeto de navegação se inicia com o exame da hierarquia de usuários e dos casos de uso relacionados que foram desenvolvidos para cada categoria de usuário (ator). Cada ator pode ter necessidades de navegação sutilmente diferentes. Em adição, os casos de uso, desenvolvidos para cada ator, vão definir um conjunto de classes que englobam um ou mais objetos de conteúdo ou de funcionalidades da aplicação Web;
- **Projeto funcional** – identifica o comportamento geral e a funcionalidade que é provida pela aplicação Web. As funcionalidades seguem padrões bem estabelecidos, tais como: carrinhos de compras e comércio eletrônico. Nesses casos, o projeto funcional não é necessário. Por outro, quando o projeto foca explicitamente funcionalidades altamente especializadas, o projeto funcional é fundamental. As funcionalidades disponibilizadas por uma aplicação Web podem ser divididas em dois conjuntos: funcionalidades em nível de usuário, que expressam as funcionalidades que apoiam os usuários na obtenção de seus objetivos e funcionalidades em nível de aplicação, que representam um projeto em nível mais baixo da funcionalidade interna, que pode não ser diretamente visível aos usuários;
- **Projeto de arquitetura** – está relacionado aos objetivos estabelecidos para uma aplicação Web, o conteúdo a ser apresentado, os usuários visitantes, e a filosofia de navegação estabelecida. Identifica a estrutura geral para a aplicação Web;
- **Projeto de componente** – Aplicações Web disponibilizam uma série de funcionalidades, tais como: processamento para proporcionar que conteúdos e navegações sejam dinâmicos, processamento de dados apropriados ao domínio da aplicação e consulta e acesso a base de dados sofisticados.
- **Projeto de Construção** - Ferramentas e tecnologias WebApp são aplicadas para a modelagem. A equipe de desenvolvimento deve projetar e construir componentes de programas análogos aos componentes de software convencional e desenvolver a lógica de processamento detalhada exigida para implementar componentes os funcionais.

2.2.4. Construção

Uma das vantagens do paradigma de orientação a objetos é o suporte à reusabilidade, que é a prática de incorporar componentes e ou módulos de software já existentes em sistemas de software para os quais eles não foram originalmente desenvolvidos. Entretanto, a reutilização de código-fonte é uma forma muito limitada de reusabilidade e a reusabilidade pode ser aplicada às fases de análise e projeto do desenvolvimento de software.

Um padrão de projeto documenta uma alternativa de solução para um problema específico, recorrente em diversas aplicações. Ele tem uma estrutura e formato particular, e descreve um problema que ocorre em um domínio em particular e como resolver esse problema. Geralmente a escolha de um padrão de projeto é influenciada pelos estilos e/ou padrões arquiteturais escolhidos anteriormente.

Padrões de projeto refinam as decisões de projeto definidas na fase do projeto de arquitetura. O principal benefício decorrente do uso de padrões de projeto é facilitar a comunicação entre desenvolvedores de software, de uma mesma equipe ou

independentes, ao permitir o emprego de estruturas de um nível de abstração maior do que linguagens de programação, porém com o mesmo grau de formalismo destas, contribuindo para a reutilização de software.
Grande parte do projeto, construção e implantação da aplicação Web está relacionada com as tecnologias, as ferramentas e as linguagens em particular que são usadas durante a construção.
A atividade de construção compreende seleção, codificação, autoria de páginas, integração, refatoração e ações de teste. Quando completadas, essas ações levam a uma aplicação Web operacional que está pronta para entrega aos usuários finais (PRESSMAN & LOWE, 2009):

- **Seleção** - envolve a identificação de componentes preexistentes, que podem ser reutilizados dentro do projeto proposto;
- **Codificação** - envolve a adaptação de componentes existentes ou a criação de novos componentes e pode envolver a criação de código HTML ou código-fonte em uma linguagem de programação;
- **Autoria de páginas** - envolve a integração do conteúdo com o projeto gráfico e o mapeamento do conteúdo nas páginas. Também pode incluir a implantação de folhas de estilo;
- **Integração** - compreende o vínculo do código, conteúdo e apresentação nos componentes finais liberados para uso;
- **Refatoração** - é o ato de alterar o código sem afetar a funcionalidade que ele implementa. É utilizado para melhorar sua estrutura, esclarecer e remover o código redundante;
- **Teste** - é a investigação do aplicativo a fim de extrair informações sobre sua qualidade em relação à função que ele deve operar e conseguir detectar os possíveis defeitos. Envolve a verificação da correção dos diversos componentes.

2.2.4.1. Testes de sistemas Web

É bem mais complicado testar interfaces Web do que classes de objetos e aplicações Web não permitem verificar facilmente se o conteúdo presente é o esperado. É fundamental e necessário que sejam testados os requisitos funcionais e não funcionais das aplicações Web.

2.2.4.1.1. Teste de requisitos funcionais

As técnicas comumente utilizadas no teste de requisitos funcionais de aplicações Web são testes caixas brancas e caixas-pretas.

- **Testes caixas brancas (white box)** - relacionam-se ao exame rigoroso do detalhe procedimental. Caminhos lógicos internos ao software e colaborações entre componentes são testados, definindo-se casos de testes que exercitam conjuntos específicos de condições e/ou ciclos;
- **Testes caixas-pretas (black box)** - referem-se a testes que são conduzidos na interface do software. Um teste caixa-preta examina algum aspecto fundamental do sistema, pouco se preocupando com a estrutura interna do software.

Os testes caixas brancas podem ser ocorrer em qualquer nível (teste unitário, integração ou de sistema), mas geralmente são utilizados para os testes unitários. Por outro lado, testes caixas-pretas são normalmente utilizados para testes de sistema e integração.
Um teste unitário foca um componente individual de software exercitando a interface, as estruturas de dados e a funcionalidade do componente, em um esforço para descobrir erros que são locais ao componente. Já o teste de integração foca no projeto e na integração dos componentes do software. O objetivo é assegurar que a estrutura do software, quando suas unidades estão integradas, está adequada e as interações entre essas unidades funcionam corretamente.
Durante o teste de sistema, procura-se garantir que todos os requisitos especificados para o software estão corretamente implementados. Isso inclui requisitos funcionais e não funcionais (desempenho, segurança etc.).

2.2.4.1.2. Teste de camadas

A arquitetura típica de uma aplicação Web pode ser dividida em camadas: apresentação, negócios e acesso aos dados. Devido à divisão em camadas, os testes podem ser concentrados em cada uma das camadas. Durante o teste de uma camada, os objetos da camada inferior são substituídos por dublês de teste. Os dublês de teste podem ser de vários tipos:

- **Objetos burros (*dummy*) -** utilizados somente para preencher uma lista de parâmetros e nunca são usados;
- **Objetos falsos (*fake*) -** possuem uma implementação simplificada, por exemplo, com valores fixos;
- **Objetos substitutos (*stub*) -** são também objetos falsos com implementação simplificada, mas que possuem funcionalidade adicional, como registrar as chamadas de seus métodos;
- **Objetos imitadores (*mock*) -** são os dublês mais complexos, pois podem ser programados para receber chamadas específicas, tratar chamadas inesperadas e também verificar se todas as chamadas previstas foram recebidas.

2.2.4.1.3. Teste do acesso aos dados

A camada de acesso aos dados geralmente é construída utilizando DAOs (*Data Access Objects*). Existem duas estratégias para testar DAOs:

- A primeira utiliza a base de dados real como se fosse a própria aplicação. A base de dados pode ser a base de dados local do desenvolvedor ou uma base de dados dedicada aos testes.
- A segunda é baseada em objetos imitadores. A vantagem é que não precisa utilizar uma base de dados real. Entretanto, é necessária a implementação de objetos imitadores, o que implica em um custo adicional de programação.

2.2.4.1.4. Teste da lógica de negócios

Nesse teste são utilizados objetos imitadores que fazem o papel da camada de acesso aos dados, ou seja, são criados imitadores DAO. No entanto, é possível

também utilizar DAOs reais que acessam a camada de dados. Nesse caso, é realizado o teste de integração (das camadas de negócio e de acesso aos dados), bem como o teste das regras de negócio.

2.2.4.1.5. Teste da camada de apresentação

A camada de apresentação geralmente é realizada utilizando a linguagem HTML. Páginas nesse formato são enviadas ao cliente por meio do protocolo HTTP. Navegadores apresentam as páginas enviadas ao usuário final que interage por meio delas. Ferramentas de testes podem aproveitar o conhecimento da linguagem HTML para simular automaticamente as ações do usuário como, por exemplo, preencher um campo ou apertar um botão. Dessa maneira é possível desenvolver programas ou scripts que executem e validem os casos de testes da camada de apresentação.

2.2.4.2. Teste de requisitos não funcionais

Testes que verificam requisitos não funcionais como o teste de desempenho, de carga e de estresse são fundamentais em aplicações Web:

- **Teste de desempenho** - tem como objetivo produzir dados que permitam prever o desempenho da aplicação quando diferentes níveis de carga (por exemplo, número de requisições) são submetidos. Espera-se identificar os níveis de carga que provocam a exaustão dos recursos do sistema;
- **Teste de carga** - semelhante ao teste de desempenho, porém visa verificar como a aplicação se comporta quando submetida a um grande volume de dados, cálculos e processamento intensivo;
- **Teste de estresse** - força a aplicação a operar em condições de recursos reduzidos. O objetivo é verificar o comportamento da aplicação em situações acima dos limites estabelecidos para garantir que a aplicação funciona adequadamente ou que falhas sejam tratadas adequadamente sem perda ou corrupção dos dados.

Estudiosos apontam que a técnica de teste mais indicada para os testes de desempenho, carga e estresse é a técnica de teste caixa cinza. Semelhante ao teste caixa-preta, o foco desse teste é verificar se a saída obtida corresponde à saída esperada. O diferencial é que o testador utiliza seu conhecimento sobre as estruturas de dados, os algoritmos utilizados, o ambiente de execução e arquitetura da aplicação para desenvolver os testes. No entanto, é importante salientar que o testador não precisa conhecer o código-fonte da aplicação Web.
Outros requisitos não funcionais como segurança, usabilidade e compatibilidade não podem ser negligenciados no desenvolvimento de aplicações Web. No entanto, para o teste desses requisitos não existem técnicas de teste em particular. Há estratégias como inspeção do código para detecção de falhas de segurança, avaliação do uso da aplicação para identificação de falhas de usabilidade e ergonomia e avaliação da compatibilidade da aplicação com diferentes navegadores.

2.2.5. Implantação

O principal objetivo da atividade de implantação é configurar a aplicação Web em seu ambiente operacional e disponibilizá-la aos usuários finais para que possa iniciar o período de avaliação. O feedback da avaliação é apresentado à equipe de Engenharia Web e o incremento é modificado conforme a necessidade.
A atividade de implantação compreende duas ações: entrega (empacotamento + liberação) e avaliação. Como o desenvolvimento da aplicação Web é incremental, a implantação acontece diversas vezes enquanto a aplicação Web se encaminha para o término de seu desenvolvimento (PRESSMAN & LOWE, 2009).
Cada ciclo de liberação de pacote oferece aos usuários finais um incremento operacional da aplicação Web que fornece funcionalidades e recursos úteis. Cada ciclo de avaliação oferece à equipe de Engenharia Web orientações importantes que resultam em modificações no conteúdo, funcionalidades, características e enfoque usados nesse ou no próximo incremento.

2.2.5.1. Ambientes de desenvolvimento

Pressman & Lowe (2009) argumentam que em aplicações Web muito simples e que não são de missão crítica, pode ser aceitável ter um único desenvolvedor criando conteúdo local. O conteúdo é então publicado no servidor de produção para acesso imediato pelos usuários.
Para aplicações Web mais complexas, onde a qualidade da aplicação é fundamental, um ambiente de desenvolvimento mais complexo torna-se apropriado. Pressman & Lowe (2009) sugerem um ambiente de desenvolvimento contendo quatro tipos de servidores distintos:

1. **Servidor de desenvolvimento** - onde desenvolvedores e autores utilizam esses servidores para realizar toda a autoria de páginas e testes unitários. Esses servidores são caixas onde os desenvolvedores podem criar, manipular e testar seus próprios componentes da aplicação Web;
2. **Servidor de teste** - quando os desenvolvedores completam o teste unitário dos componentes, eles podem integrá-los para verificação dentro do ambiente completo da aplicação Web;
3. **Servidor de homologação** - tem como finalidade oferecer um espelho completo do ambiente de produção, de modo que o teste do usuário da aplicação Web possa ocorrer sem ter que liberar a aplicação ao servidor de produção e, daí, à população de usuários completa;
4. **Servidor de produção** - quando a aplicação Web está pronta para ser liberada para uso por todos os usuários, ela é levada para esse servidor. É importante que as mudanças não sejam liberadas nesse servidor até que sejam totalmente testadas no servidor de homologação.

O ambiente real a ser adotado para um projeto de aplicação Web em particular dependerá de alguns fatores: escala do projeto, número de desenvolvedores e interessados e estado crítico de negócios. Em muitos casos, os quatro diferentes servidores poderiam ser reduzidos para três servidores (desenvolvimento, homologação e produção) ou ainda dois servidores (desenvolvimento e produção).

2.3. Atividades guarda-chuva

Enquanto as atividades de arcabouço são aplicadas a cada incremento da aplicação Web, uma coleção de atividades “guarda-chuva”, importantes para o sucesso do projeto, ocorre em segundo plano. Observar que essas atividades “guarda-chuva” são independentes de qualquer atividade, ocorrendo ao longo de todo o processo de desenvolvimento do software.
Dentre as muitas atividades guarda-chuva que podem ser definidas, quatro atividades são fundamentais para um projeto bem-sucedido de Engenharia Web (PRESSMAN & LOWE, 2009):

- **Gerência de riscos** - considera os riscos de projeto e técnicos à medida que um incremento é desenvolvido;
- **Gerência de projeto** - acompanha e monitora o progresso à medida que um incremento é desenvolvido;
- **Gerência de mudança** - gerencia o efeito da mudança à medida que cada incremento é desenvolvido, integrando ferramentas que auxiliam na gerência de todo o conteúdo da aplicação Web;
- **Garantia da qualidade** - define e conduz aquelas tarefas que ajudam a garantir que cada produto de trabalho e o incremento implantado apresentam qualidade.

3. Exercícios

1. Defina “arcabouço” no contexto da Engenharia Web.

2. Requisitos funcionais e não funcionais podem definir características e funcionalidades presentes na interface a ser desenvolvida para um sistema. Em relação aos requisitos não funcionais, avalie as afirmações a seguir.

I. São levantados e elicitados após os requisitos funcionais, uma vez que os requisitos funcionais determinarão as funcionalidades da interface;
II. Sempre serão definidos de forma mais concreta, através de requisitos funcionais, uma vez que o usuário manipula na interface somente as funcionalidades levantadas;
III. Podem complementar os requisitos funcionais.

É correto o que se afirma em:

a) ( ) II, apenas.
b) ( ) III, apenas.
c) ( ) I e II, apenas.
d) ( ) I e III, apenas.
e) ( ) I, II e III.

3. A verificação e a validação de uma interface de usuário ocorrem em três pontos distintos: análise, projeto e teste. Considerando um cenário de uma aplicação web, tal verificação pode ser realizada através de testes de interface, testes de usabilidade e testes de compatibilidade. Nesse contexto, avalie as afirmações a seguir.

I. O teste de interface experimenta mecanismos de interação e valida aspectos estéticos da interface do usuário, apontando erros específicos de interface e erros na maneira como interface implementa as semânticas de navegação, funcionalidade ou exibição de conteúdo;
II. O teste de usabilidade avalia o grau com o qual os usuários podem interagir efetivamente com a aplicação e o grau em que a aplicação dirige as ações do usuário;
III. O primeiro passo no teste de compatibilidade é definir uma série de configurações típicas encontradas do lado cliente e suas respectivas variantes, identificando características como plataforma, sistema operacional e navegador.

É correto o que se afirma em:

a) ( ) I, apenas;
b) ( ) III, apenas;
c) ( ) I e II, apenas;
d) ( ) II e III, apenas;
e) ( ) I, II e III.

4. SOA é um estilo arquitetônico de software usado para construir soluções empresariais baseadas em serviços Web. São características dos serviços desse estilo:

a) ( ) Encapsulamento e recursividade;
b) ( ) Modularidade e transparência;
c) ( ) Reuso e descoberta dinâmica;
d) ( ) Autonomia e dependência de protocolos Web;
e) ( ) Isolamento de responsabilidades e paralelismo de dados.

ANEXO – I: Modelagem de Requisitos: Fluxo, Comportamento, Padrões e Aplicações na Web