

13

PROJETO DE WEBAPPS

CONCEITOS-CHAVE

arquitetura	
de conteúdo ...	347
de objetos	345
arquitetura MVC.	349
arquitetura da WebApp.....	349
OOHDM.....	352
projeto	
arquitetural ...	346
de componentes .	352
de conteúdo ...	345
de metas	341
de navegação ..	350
estético	344
gráfico	344
pirâmide.....	342
qualidade	339

Em seu respeitado livro sobre projeto para Web, Jakob Nielsen [Nie00] afirma: "Existem, essencialmente, duas abordagens básicas para projeto: o ideal artístico de se expressar e o ideal de engenharia de resolver um problema para um cliente". Durante a primeira década de desenvolvimento da Web, o ideal artístico foi a abordagem que muitos desenvolvedores escolheram. O projeto ocorreu em uma maneira *ad hoc* e foi usualmente conduzido à medida que o HTML era gerado. O projeto desenvolveu-se a partir de uma visão artística que evoluiu enquanto ocorria a construção de WebApps.

Mesmo hoje, muitos desenvolvedores para Web usam WebApps como ícone para indicar "projeto limitado". Eles argumentam que o imediatismo e a volatilidade das WebApps aliviam o processo de projeto formal; que o projeto evolui à medida que uma aplicação é construída (codificada) e que relativamente pouco tempo deve ser gasto na criação de um modelo de projeto detalhado. Esse argumento tem seus méritos, mas apenas para WebApps relativamente simples. Quando conteúdo e função são complexos; quando o tamanho da WebApp engloba centenas ou milhares de objetos de conteúdo, funções e classes de análise; e quando o sucesso da WebApp terá um impacto direto no sucesso do negócio, o projeto não pode e não deve ser tratado de maneira superficial.

PANORAMA

O que é? O projeto para WebApps abrange atividades técnicas e não técnicas entre as quais: estabelecer a percepção e a aparência da WebApp, criar o layout estético da interface do usuário, definir a estrutura geral da arquitetura, desenvolver o conteúdo e a funcionalidade que residem na arquitetura e planejar a navegação que ocorre na WebApp.

Quem realiza? Engenheiros de aplicações para a Web, designers gráficos, desenvolvedores de conteúdo e outros interessados participam na criação de um modelo de projeto de uma WebApp.

Por que é importante? O projeto nos permite criar um modelo que pode ser avaliado em termos de qualidade e aperfeiçoado antes de serem gerados código e conteúdo; são realizados testes e usuários finais se envolvem em grande número. Projeto é onde se estabelece a qualidade de uma WebApp.

Quais são as etapas envolvidas? O projeto de WebApps abrange seis etapas principais orientadas por informações obtidas durante a modelagem de requisitos. O projeto de conteúdo usa o modelo de conteúdo (desenvolvido durante a análise) como base para estabelecer o projeto

de objetos de conteúdo. O projeto estético (também chamado design gráfico) estabelece o layout que o usuário final verá. O projeto da arquitetura se concentra na estrutura geral de hipermídia de todos os objetos de conteúdo e funções. O projeto da interface estabelece mecanismos de layout e interação que definem a interface do usuário. O projeto de navegação define como o usuário final navega pela estrutura de hipermídia, e o projeto de componentes representa a estrutura interna detalhada dos elementos funcionais da WebApp.

Qual é o artefato? Um modelo de projeto abrangendo questões de conteúdo, estética, arquitetura, interface, navegação e de projeto de componentes é o artefato primário gerado durante o projeto de uma WebApp.

Como garantir que o trabalho foi realizado corretamente? Cada elemento do modelo de projeto é revisado na tentativa de descobrir erros, inconsistências ou omissões. Além disso, são consideradas soluções alternativas, e o grau com que o modelo de projeto atual irá levar a uma implementação efetiva também é avaliado.

Essa realidade nos leva à segunda abordagem de Nielsen — “o ideal da engenharia de resolver um problema para um cliente”. A engenharia para Web¹ adota essa filosofia, e uma abordagem mais rigorosa para projeto de WebApps permite aos desenvolvedores alcançarem tal objetivo.

13.1 QUALIDADE DE PROJETO EM WEBAPPS

Projeto é a atividade da engenharia que conduz a um produto de alta qualidade. Isso nos leva a uma pergunta recorrente encontrada em todas as disciplinas de engenharia: O que é qualidade? Nesta seção examinaremos a resposta no contexto de desenvolvimento de WebApps.

Todo mundo que já navegou na Web ou usou uma Intranet corporativa tem opinião formada sobre o que faz uma WebApp ser “boa”. Os pontos de vista individuais variam muito. Alguns usuários adoram imagens chamativas; outros querem apenas texto. Alguns exigem informações detalhadas; outros desejam uma apresentação resumida. Alguns preferem ferramentas analíticas sofisticadas ou acesso a bancos de dados; outros preferem a simplicidade. Na realidade, a percepção do usuário de “excelência” (e a resultante aceitação ou rejeição da WebApp como consequência) talvez seja mais importante do que qualquer discussão técnica sobre a qualidade de WebApps.

Mas como a qualidade de uma WebApp é percebida? Quais atributos devem ser apresentados para atingir a excelência segundo a visão dos usuários finais e, ao mesmo tempo, apresentar as características técnicas de qualidade que lhe permitirá corrigir, adaptar, melhorar e suportar a aplicação no longo prazo?

Na realidade, todas as características técnicas da qualidade de projetos discutidas no Capítulo 8 e os atributos de qualidade gerais apresentados no Capítulo 14 se aplicam a WebApps. Entretanto, os atributos de qualidade gerais mais relevantes — usabilidade, funcionalidade, confiabilidade, eficiência e facilidade de manutenção — fornecem uma base útil para se avaliar a qualidade de sistemas baseados na Web.

Olsina e seus colegas [Ols99] prepararam uma “árvore de requisitos de qualidade” que identifica um conjunto de atributos técnicos — usabilidade, funcionalidade, confiabilidade, eficiência e facilidade de manutenção — que levam a WebApps de alta qualidade.² A Figura 13.1 sintetiza o trabalho desses pesquisadores. Os critérios citados na figura são de particular interesse caso você tenha de projetar, construir e manter WebApps no longo prazo.

Offutt [Off02] estende os cinco principais atributos de qualidade citados na Figura 13.1 acrescentando os seguintes atributos:

Segurança. As WebApps se tornaram altamente integradas a bancos de dados corporativos e governamentais críticos. Aplicações de comércio eletrônico extraem e depois armazenam informações confidenciais de clientes. Por essas e muitas outras razões, a segurança da WebApp é primordial em várias situações. A principal medida de segurança é a habilidade da WebApp e seu ambiente de servidor rechaçar acesso desautorizado e/ou frustrar um ataque mal-intencionado. Uma discussão detalhada sobre a segurança de WebApps está fora do escopo deste livro. Caso tenha maior interesse, veja [Vac06], [Kiz05] ou [Kal03].

Disponibilidade. Até mesmo a melhor WebApp não atenderá às necessidades dos usuários caso esteja indisponível. Em um sentido técnico, disponibilidade é a medida da porcentagem de tempo que uma WebApp está disponível para uso. O típico usuário final espera que as WebApps estejam disponíveis 24 horas por dia/7 dias por semana/365 dias por ano. Qualquer coisa abaixo

“Se os produtos são projetados para melhor atender às tendências naturais do comportamento humano, então as pessoas se sentirão mais satisfeitas, mais realizadas e mais produtivas.”

Susan Weinschenk

? Quais os principais atributos de qualidade para as WebApps?

¹ *Web engineering* [Pre08] é uma versão adaptada da abordagem de engenharia de software apresentada ao longo deste livro. Ela propõe uma estrutura ágil, embora disciplinada, para construção de sistemas e aplicações baseados na Web com alta qualidade.

² Esses atributos de qualidade são bem similares aos apresentados nos Capítulos 8 e 14. A implicação: características de qualidade são universais para todo software.

FIGURA 13.1

Árvore de requisitos de qualidade.

Fonte: (Ols99)



disso é considerada inaceitável.³ Porém, *estar no ar* não é o único indicador de disponibilidade. Offutt [Off02] sugere que “as características de uso disponíveis em apenas um navegador ou uma plataforma” torna a WebApp indisponível para aqueles usam um navegador/plataforma diferente. O usuário invariavelmente irá procurar outra alternativa.

Escalabilidade. A WebApp e seus servidores podem ser escalados para atender 100, 1.000, 10.000 ou 100.000 usuários? A WebApp e os sistemas com os quais está interfaceando conseguem lidar com variação significativa de volume ou sua capacidade de resposta cairá significativamente (ou cessará de vez)? Não basta construir uma WebApp que seja bem-sucedida. É igualmente importante construir uma WebApp capaz de acomodar as responsabilidades inerentes ao sucesso (um número significativamente maior de usuários finais) e se tornar ainda mais bem-sucedida.

Tempo para colocação no mercado (*time-to-market*). Embora o tempo para colocação de um produto no mercado não seja um verdadeiro atributo de qualidade no sentido técnico, é uma medida de qualidade do ponto de vista comercial. A primeira WebApp a atender determina o segmento de mercado em geral captura um número desproporcional de usuários finais.



Projeto de WebApps — check-list de qualidade

O check-list a seguir, adaptado das informações apresentadas em **Webreference.com**, fornece um conjunto de perguntas que ajudam tanto os projetistas de aplicações para Web como os usuários finais a avaliar a qualidade geral de uma WebApp:

- Opções de conteúdo e/ou função e/ou navegação podem ser ajustadas às preferências dos usuários?
- O conteúdo e/ou funcionalidade podem ser personalizados para a largura de banda em que o usuário se comunica?
- Imagens e outras mídias não textuais foram usadas adequadamente? Os tamanhos de arquivos gráficos são otimizados para eficiência de exibição?
- As tabelas são organizadas e dimensionadas para torná-las compreensíveis e exibidas de forma eficiente?
- O HTML é otimizado para eliminar ineficiências?
- O projeto geral de páginas é fácil de ler e navegar?
- Todos os links fornecem informações de interesse dos usuários?
- É provável que a maioria dos links tenha persistência na Web?
- A WebApp é equipada com recursos de administração de sites que incluem ferramentas para acompanhamento de uso, testes de links, busca de locais e segurança?

INFORMAÇÕES

³ Essa expectativa é, obviamente, irreal. As principais WebApps têm de programar *downtime* para correções e atualizações.

Bilhões de páginas Web se encontram disponíveis para aqueles em busca de informação. Mesmo as buscas na Web bem direcionadas resultam em uma avalanche de conteúdo. Com tantas fontes de informação para escolha, como o usuário pode avaliar a qualidade (por exemplo, veracidade, precisão, completude, oportunidade) do conteúdo apresentado em uma WebApp? Tillman [Til00] sugere um útil conjunto de critérios para avaliar a qualidade de conteúdos:

? O que devemos considerar ao avaliarmos a qualidade do conteúdo?

- O escopo e a profundidade do conteúdo podem ser facilmente determinados para garantir que atenda às necessidades dos usuários?
- A experiência e a autoridade dos autores do conteúdo podem ser facilmente identificadas?
- É possível determinar a atualidade do conteúdo, a última atualização e o que foi atualizado?
- O conteúdo e sua localização são estáveis (isto é, permanecerão na URL referida)?

Além dessas perguntas relacionadas a conteúdo, poderíamos adicionar o seguinte:

- O conteúdo é confiável?
- O conteúdo é exclusivo? A WebApp fornece algum benefício para aqueles que o usam?
- O conteúdo tem valor para a comunidade de usuários desejada?
- O conteúdo é bem organizado? Indexado? Facilmente acessível?

Os check-lists citados nesta seção representam apenas uma pequena amostra das questões que devem ser tratadas à medida que o projeto de uma WebApp evolui.

13.2 OBJETIVOS DE PROJETO

Em sua coluna regular sobre projeto para a Web, Jean Kaiser [Kai02] sugere um detalhado conjunto de objetivos de projeto que podem ser aplicados a praticamente qualquer WebApp independentemente do domínio de aplicação, do tamanho ou da complexidade:

Simplicidade. Embora possa parecer ultrapassado, o aforismo “tudo com moderação” se aplica às WebApps. Há uma tendência entre alguns projetistas de fornecer “em excesso” ao usuário final — conteúdo exaustivo, aspectos visuais excessivos, animação intrusiva, páginas Web enormes, a lista é longa. É melhor se esforçar pela moderação e simplicidade.

O conteúdo deve ser informativo, mas sucinto e deve usar um meio de entrega (por exemplo, texto, imagens, vídeo, áudio) apropriado para as informações que estão sendo entregues. A estética deve ser agradável, mas não opressiva (por exemplo, cores em demasia tendem a distrair o usuário em vez de melhorar a interação). A arquitetura deve atingir os objetivos da WebApp da maneira mais simples possível. A navegação deve ser simples e seus mecanismos intuitivamente óbvios para o usuário final. As funções devem ser fáceis de serem usadas e compreendidas.

Consistência. Esse objetivo de projeto se aplica a praticamente qualquer elemento do modelo de projeto. O conteúdo deve ser construído consistentemente (por exemplo, a formatação de texto e os estilos de fonte devem ser os mesmos ao longo de todos os documentos de texto; a arte gráfica deve ter aspecto, combinação de cores e estilo consistentes). O design gráfico (estética) deve apresentar um aspecto consistente ao longo de todas as partes da WebApp. O projeto da arquitetura deve estabelecer templates que levem a uma estrutura de hipermídia consistente. O projeto da interface deve definir modos de interação, navegação e exibição de conteúdo consistentes. Os mecanismos de navegação devem ser usados consistentemente por todos os elementos da WebApp. Como Kaiser [Kai02] observa: “Lembre-se de que para um visitante, um site é um lugar físico. Torna-se confuso se as páginas não forem consistentes no projeto”.

Identidade. A estética, a interface e o projeto de navegação de uma WebApp devem ser consistentes com o domínio de aplicação para o qual ela será construída. Um site para um grupo de *hip-hop* indubitavelmente terá uma percepção e aparência diferentes de uma WebApp desenhada para uma companhia financeira. A arquitetura da WebApp será completamente diferente, as

“Só porque você pode não significa que deva.”

Jean Kaiser

“Para alguns, o projeto para Web enfoca o aspecto visual... Para outros, projeto para Web significa estruturar informações e navegação pelo espaço do documento. Outros até poderiam considerar que projeto para Web significa a tecnologia... Na realidade, o projeto inclui todas essas coisas e talvez até mais.”

Thomas Powell

interfaces serão construídas para levar em conta diferentes categorias de usuários; a navegação será organizada para cumprir diferentes objetivos. Devemos (assim como outros colaboradores do projeto) trabalhar para estabelecer uma identidade para a WebApp por todo o projeto.

Robustez. Baseado na identidade estabelecida, em geral uma WebApp faz uma “promessa” implícita ao usuário, que espera funções e conteúdo robustos que são relevantes para as suas necessidades. Se esses elementos estiverem faltando ou insuficientes, é provável que a WebApp irá falhar.

Navegabilidade. Já citamos que a navegação deve ser simples e consistente. Ela também deve ser projetada de maneira intuitiva e previsível. Ou seja, o usuário deve entender como navegar pela WebApp sem ter de buscar links ou instruções para navegação. Por exemplo, se um campo de imagens ou ícones contiver ícones ou imagens selecionadas que serão usadas como mecanismos de navegação, estes devem ser identificados visualmente. Nada é mais frustrante que tentar encontrar o link ativo apropriado entre muitas imagens.

Também é necessário posicionar links para as funções e conteúdos mais importantes da WebApp em uma posição previsível em todas as páginas Web. Se for necessário rolar a página (e, normalmente, este é o caso), os links na parte superior e inferior da página tornam mais fácil as tarefas de navegação do usuário.

Apelo Visual. De todas as categorias de software, as aplicações para Web são, inquestionavelmente, as mais visuais, as mais dinâmicas e as mais estéticas. A beleza (apelo visual) é um conceito que varia segundo a ótica de quem a vê, porém, muitas características de projeto (por exemplo, a percepção e aspecto do conteúdo; o layout da interface; coordenação das cores; o equilíbrio entre texto, imagens e outras mídias; mecanismos de navegação) contribuem efetivamente para o apelo visual.

Compatibilidade. Uma WebApp será usada em uma variedade de ambientes (por exemplo, hardware diferente, tipos de conexão de Internet, sistemas operacionais, navegadores) e deve ser projetada para ser compatível com cada um deles.

13.3 UMA PIRÂMIDE DE PROJETO PARA WEBAPPS

“Se um site é perfeitamente utilizável mas falta um estilo de projeto elegante e adequado, ele falhará.”

Curt Cloninger

O que é projeto para WebApps? Essa simples questão é mais difícil de responder do que se pode imaginar. Em nosso livro [Pre08] sobre engenharia para Web, David Lowe e eu discutimos isso ao escrevermos:

A criação de um projeto eficaz exigirá, tipicamente, um conjunto de habilidades diversas. Às vezes, para pequenos projetos, um único desenvolvedor precisaria ter vários talentos. Para projetos maiores, seria aconselhável e/ou viável fazer uso da *expertise* de especialistas: engenheiros para aplicações para Web, designers gráficos, desenvolvedores de conteúdo, programadores, especialistas em bancos de dados, arquitetos da informação, engenheiros de rede, especialistas em segurança e aqueles que realizam testes. Fazer uso dessas diversas capacidades permite a criação de um modelo que pode ser avaliado em termos de qualidade e aperfeiçoado *antes* de o conteúdo e código serem gerados, os testes serem realizados e os usuários finais envolverem-se em grande número. Se análise é o momento em que se estabelece a qualidade de uma WebApp, então projeto é o momento em que a qualidade é realmente incorporada.

O mix apropriado de habilidades de projeto irá variar dependendo da natureza da WebApp. A Figura 13.2 apresenta uma pirâmide de projeto para WebApps. Cada nível da pirâmide representa uma ação de projeto descrita nas seções a seguir.

13.4 PROJETO DE INTERFACES PARA WEBAPPS

Quando um usuário interage com um sistema computacional, aplica-se um conjunto de princípios fundamentais e diretrizes de projeto primordiais. Estes foram discutidos no Capí-

FIGURA 13.2
Uma pirâmide de
projeto para WebApps



tulo 11.⁴ Embora as WebApps apresentem alguns desafios especiais no projeto da interface do usuário, as diretrizes e princípios básicos se aplicam.

Um dos desafios no projeto da interface para WebApps é a natureza indeterminada do ponto de entrada do usuário. Ou seja, o usuário poderia entrar na WebApp pela localização “home” (por exemplo, a *homepage*) ou, por meio de um link, entrar em algum nível mais baixo da arquitetura da WebApp. Em alguns casos, a WebApp pode ser desenhada para redirecionar o usuário a uma localização home, mas se isso for indesejável, o projeto da WebApp deve fornecer recursos de navegação de interface que acompanhem todos os de conteúdo e estejam disponíveis independentemente de como o usuário entre no sistema.

Os objetivos de uma interface para WebApp são: (1) estabelecer uma janela consistente para o conteúdo e a funcionalidade fornecidos pela interface, (2) guiar o usuário através de uma série de interações com a WebApp e (3) organizar as opções de navegação e conteúdo disponíveis para o usuário. Para obtermos uma interface consistente, devemos primeiro usar a estética do projeto (Seção 13.5) para estabelecer um “aspecto” coerente. Isso abrange várias características, mas deve enfatizar o layout e a forma dos mecanismos de navegação. Para orientarmos a interação com o usuário, poderíamos fazer uso de uma metáfora⁵ apropriada que permita ao usuário ter um entendimento intuitivo da interface.

Para implementarmos opções de navegação, podemos selecioná-las de uma série de mecanismos de interação:

Quais mecanismos de interação estão disponíveis para os projetistas de WebApps?

- *Menus de navegação* — menus com palavras-chave (organizados vertical ou horizontalmente) que listam o conteúdo e/ou funcionalidade principais. Esses menus poderiam ser implementados para que o usuário possa escolher de uma hierarquia de subtópicos exibidos quando a opção de menu principal for escolhida.
- *Ícones* — botões, chaves e imagens similares que permitem ao usuário selecionar alguma propriedade ou especificar uma decisão.
- *Imagens* — alguma representação gráfica que é selecionável pelo usuário e implementa um link para um objeto de conteúdo ou funcionalidade da WebApp.

É importante notar que um ou mais desses mecanismos de controle devem ser fornecidos em todos os níveis da hierarquia de conteúdo.

⁴ A Seção 11.5 é dedicada ao projeto da interface de WebApps. Caso ainda não tenha feito, leia-a agora.

⁵ Nesse contexto, *metáfora* é uma representação (extraída da experiência real do usuário) que pode ser modelada no contexto da interface. Um exemplo simples poderia ser um controle deslizante usado para controlar o volume do áudio de um arquivo .mpg.

13.5 PROJETO ESTÉTICO

Nem todo engenheiro de aplicações Web (ou engenheiro de software) tem talento artístico (estético). Caso se enquadre nessa categoria, contrate um designer gráfico experiente para realizar a tarefa de projeto estético.

“Constatamos que as pessoas avaliam rapidamente um site apenas pelo projeto visual.”

Stanford
Diretrizes para
Credibilidade na
Web



Os usuários tendem a tolerar rolagem vertical mais facilmente do que a horizontal. Evite formatos de página largos.

O projeto estético, também chamado *design gráfico*, é o esforço artístico que complementa os aspectos técnicos do projeto de WebApps. Sem ele, uma WebApp poderia ser funcional, mas não atraente. Com ele, uma WebApp atrai seus usuários para um mundo que os envolve em um nível físico, bem como intelectual.

Mas o que é estética? Há um velho ditado que diz: “A beleza existe segundo a ótica daquele que a vê”. Isso é particularmente apropriado quando se considera o projeto estético para WebApps. Para realizar um projeto estético efetivo, retorne à hierarquia de usuários desenvolvida como parte do modelo de requisitos (Capítulo 5) e pergunte: *Quem são os usuários da WebApp e que “visual” eles desejam?*

13.5.1 Problemas de layout

Toda página Web tem uma quantidade limitada de “terreno” que pode ser usado para dar suporte à estética não funcional, recursos de navegação, conteúdo de informação e funcionalidade dirigida ao usuário. O desenvolvimento desse terreno é planejado durante o projeto estético.

Assim como todas as questões estéticas, não há regras absolutas quando se desenvolve o layout da tela. Entretanto, vale a pena considerarmos uma série de diretrizes gerais para layout:

Não tenha medo de espaços em branco. É desaconselhável preencher cada centímetro de uma página Web com informação. O congestionamento visual resultante dificulta ao usuário identificar as informações ou os recursos necessários e cria um caos visual desagradável.

Enfatize o conteúdo. Afinal de contas, essa é razão para o usuário estar lá. Nielsen [Nie00] sugere que uma página Web típica deve ter 80% de conteúdo e o espaço restante dedicado à navegação e outros recursos.

Organize os elementos de layout de cima para baixo, da esquerda para a direita.

A grande maioria dos usuários varrerá uma página Web de uma forma muito similar ao que faz ao ler a página de um livro — de cima para baixo, da esquerda para a direita.⁶ Se os elementos de layout tiverem prioridades específicas, os elementos de alta prioridade devem ser colocados na parte superior esquerda do espaço da página.

Agrupe a navegação, o conteúdo e as funções geograficamente dentro da página. Os seres humanos buscam padrões em quase tudo. Se não existirem padrões discerníveis em uma página Web, a frustração do usuário provavelmente aumentará (devido a buscas infrutíferas por informação necessária).

Não estenda seu espaço com a barra de rolagem. Embora muitas vezes a rolagem seja necessária, a maioria dos estudos indica que os usuários preferem não ficar rolando a página. É melhor reduzir o conteúdo da página Web ou apresentar o conteúdo necessário em várias páginas.

Considere a resolução e o tamanho da janela do navegador ao elaborar seu layout.

Em vez de definir tamanhos fixos em um layout, o projeto deve especificar todos os itens de layout como uma porcentagem do espaço disponível [Nie00].

13.5.2 Questões de design gráfico

O design gráfico considera todos os aspectos visuais de uma WebApp. O processo de design gráfico começa com o layout (Seção 13.5.1) e prossegue com a consideração de combinações de cores gerais; tipos, tamanhos e estilos de texto; o uso de mídia complementar (por exemplo, áudio, vídeo, animação); e todos os demais elementos estéticos de uma aplicação.

Uma discussão completa sobre questões relativas ao design gráfico em WebApps está fora do escopo deste livro. Você pode obter dicas e diretrizes em muitos sites dedicados ao tema (por exemplo, www.graphic-design.com, www.grantasticdesigns.com, www.wpdfd.com) ou em um ou mais dos recursos impressos (por exemplo, [Roc06] e [Gor02]).

⁶ Existem exceções que se baseiam em questões culturais e do idioma usado, mas tal regra não se aplica à maioria dos usuários.



Sites bem projetados

Algumas vezes, a melhor maneira de entender um bom projeto de uma WebApp é ver alguns exemplos. Em seu artigo, "The Top Twenty Web Design Tips", Marcelle Toor (www.graphic-design.com/Web/feature/tips.html) sugere os seguintes sites como exemplos de design gráfico adequado:

www.creativepro.com/designresource/home/787.html — empresa de design dirigida por Primo Angeli

www.workbook.com — este site exibe trabalhos feitos por ilustradores e designers

www.pbs.org/riverofsong — série para TV pública e rádio sobre música americana

www.RKDINC.com — empresa de design com portfólio online e excelentes dicas sobre design

www.creativehotlist.com/index.html — excelente fonte de sites bem desenhados por agências de propaganda, empresas de artes gráficas e outros especialistas da comunicação

www.btdnyc.com — empresa de design dirigida por Beth Toudreau

INFORMAÇÕES

13.6 PROJETO DE CONTEÚDO

"Bons designers são capazes de criar normalidade no caos; eles conseguem transmitir ideias claramente por meio da organização e manipulação de palavras e figuras."

Jeffery Veen

O projeto de conteúdo aborda duas tarefas de projeto diferentes, cada uma delas tratadas por indivíduos com conjuntos de habilidades diferentes. Primeiro, são desenvolvidos uma representação de projeto para objetos de conteúdo e os mecanismos necessários para estabelecer seus relacionamentos. Além disso, são criadas as informações em um objeto de conteúdo específico. Esta última tarefa poderia ser realizada por redatores publicitários, designers gráficos e outros que geram o conteúdo a ser utilizado em uma WebApp.

13.6.1 Objetos de conteúdo

O relacionamento entre objetos de conteúdo definido como parte de um modelo de requisitos para a WebApp e os objetos de projeto representando o conteúdo é análogo ao relacionamento entre classes de análise e componentes de projeto descritos em capítulos anteriores. No contexto de projeto para WebApps, um objeto de conteúdo está mais alinhado com um objeto de dados para software tradicional. Um objeto de conteúdo possui atributos que incluem informações específicas de conteúdo (normalmente definidas durante a modelagem de requisitos da WebApp) e atributos de implementação exclusivos, especificados como parte do projeto.

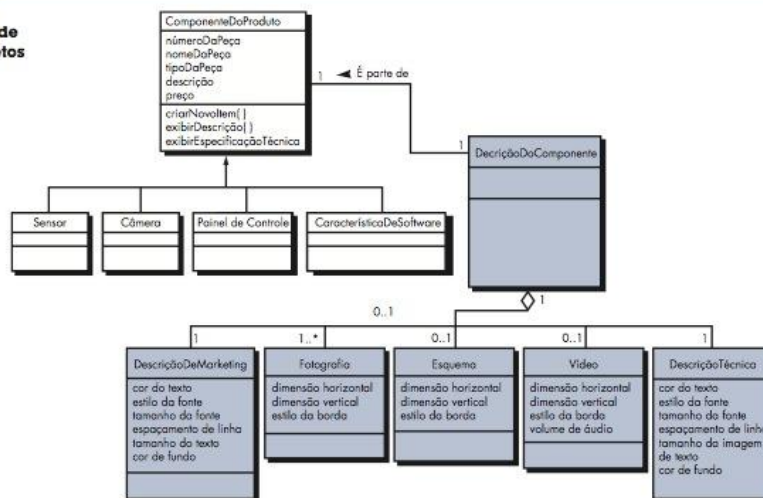
Consideremos, por exemplo, uma classe de análise, **ComponenteDoProduto**, desenvolvida para o sistema de comércio eletrônico *CasaSegura*. O atributo de classe de análise, **descrição**, é representado com uma classe chamada **DescriçãoDeComponente** composta por cinco objetos de conteúdo, **DescriçãoDeMarketing**, **Fotografia**, **DescriçãoTécnica**, **Esquema** e **Vídeo**, indicados como objetos sombreados na Figura 13.3. As informações contidas no objeto de conteúdo são indicadas na forma de atributos. Por exemplo, **Fotografia** (uma imagem .jpg) possui os atributos **dimensão horizontal**, **dimensão vertical** e **estilo da borda**.

Associação e agregação⁷ UML podem ser usadas para representar relacionamentos entre os objetos de conteúdo. Por exemplo, a associação UML da Figura 13.3 indica que é usada uma classe **DescriçãoDeComponente** para cada instância da classe **ComponenteDoProduto**. **DescriçãoDeComponente** é composta pelos cinco objetos de conteúdo mostrados. Entretanto, a notação de multiplicidade indica que **Esquema** e **Vídeo** são opcionais (é possível que não haja nenhuma ocorrência), uma **DescriçãoDeMarketing** e uma **DescriçãoTécnica** são necessárias e usadas uma ou mais instâncias de **Fotografia**.

13.6.2 Questões de projeto de conteúdo

Assim que todos os objetos de conteúdo forem modelados, as informações que cada objeto deve fornecer devem passar por um processo de autoria e formatação para melhor atender às

⁷ Essas duas representações são discutidas no Apêndice 1.

FIGURA 13.3**Representação de projeto dos objetos de conteúdo**

necessidades do cliente. A autoria de conteúdo é tarefa de especialistas que projetam o objeto de conteúdo fornecendo um resumo das informações a ser entregues e uma indicação dos tipos de objetos de conteúdo genéricos (por exemplo, texto descritivo, imagens, fotografias) usados para transmitir as informações. O projeto estético (Seção 13.5) também poderia ser aplicado para representar o aspecto visual para o conteúdo.

À medida que os objetos de conteúdo são projetados, eles são “agrupados” [Pow02] para formarem páginas Web. O número de objetos de conteúdo incorporados em uma única página é função das necessidades do usuário, das restrições impostas pela velocidade de *download* da conexão de Internet disponível e de restrições impostas pelo nível de rolagem que o usuário irá tolerar.

13.7 PROJETO ARQUITETURAL

“A estrutura arquitetural de um site bem desenhado nem sempre é aparente para o usuário — e nem deveria ser.”

Thomas Powell

O projeto arquitetural está ligado aos objetivos estabelecidos para uma WebApp, ao conteúdo a ser apresentado, aos usuários que visitarão a página e à filosofia de navegação que foi estabelecida. Como projetistas da arquitetura, temos de identificar a arquitetura de conteúdo e a arquitetura da WebApp. A *arquitetura de conteúdo*⁸ focaliza a maneira pela qual objetos de conteúdo (ou objetos compostos como páginas Web) são estruturados para apresentação e navegação. A *arquitetura de WebApps* lida com a maneira pela qual a aplicação é estruturada para administrar a interação com o usuário, tratar tarefas de processamento interno, navegação efetiva e apresentação de conteúdo.

Na maioria dos casos, o projeto arquitetural é conduzido em paralelo com os projetos da interface, estético e de conteúdo. Como a arquitetura da WebApp pode ter uma forte influência sobre a navegação, as decisões tomadas durante as etapas de projeto influenciarão o trabalho conduzido durante o projeto da navegação.

⁸ O termo *arquitetura da informação* também é usado para conotar estruturas que levam a uma melhor organização, atribuição de nomes, navegação e busca de objetos de conteúdo.

13.7.1 Arquitetura de conteúdo

O projeto da arquitetura de conteúdo concentra-se na definição da estrutura geral de hipermídia da WebApp. Embora algumas vezes sejam criadas arquiteturas personalizadas, sempre temos a opção de escolher uma de quatro estruturas de conteúdo diferentes [Pow00]:

Quais os tipos de arquitetura de conteúdo mais comuns?

Estruturas lineares (Figura 13.4) são encontradas quando uma sequência de interações previsível (com certa variação ou desvios) é comum. Um exemplo clássico poderia ser uma apresentação de um tutorial em que as páginas de informação juntamente com as imagens, vídeos de curta duração ou áudio relacionados são apresentados apenas após as informações de pré-requisito terem sido apresentadas. A sequência de apresentação de conteúdo é predefinida e, em geral, linear. Outro exemplo poderia ser a sequência de preenchimento do pedido de um produto em que informações específicas devem ser especificadas em determinada ordem. Em tais casos, as estruturas da Figura 13.4 são apropriadas. À medida que o conteúdo e o processamento forem se tornando mais complexos, o fluxo puramente linear na parte esquerda da figura dá lugar a estruturas lineares mais sofisticadas, em que conteúdo alternativo poderia ser solicitado ou a ocorrência de uma mudança de direção para obter conteúdo complementar (a estrutura do lado direito da Figura 13.4).

Estruturas em grade (Figura 13.5) é uma opção de arquitetura que podemos aplicar quando conteúdo de WebApp pode ser organizado em categorias de duas (ou mais) dimensões. Consideremos, por exemplo, uma situação em que um site de comércio eletrônico vende tacos de golfe.

FIGURA 13.4
Estruturas lineares

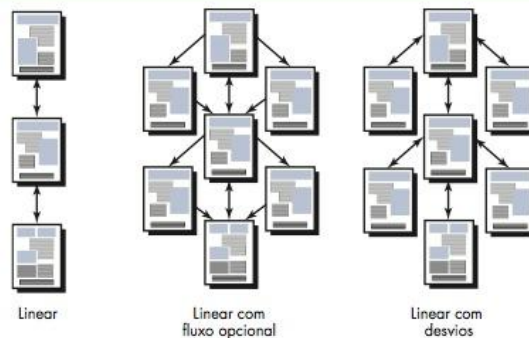
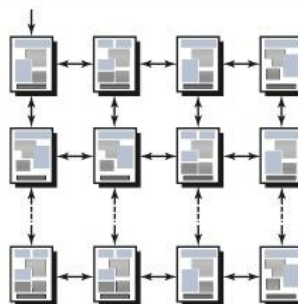


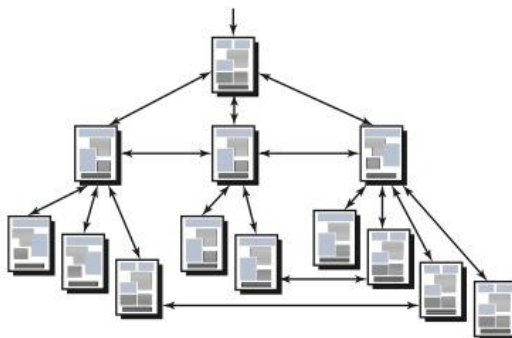
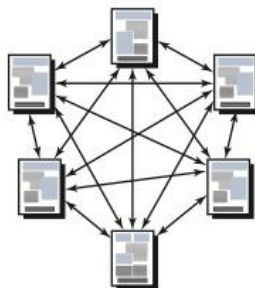
FIGURA 13.5
Estrutura em grade



A dimensão horizontal da grade representaria o tipo de taco a ser vendido (por exemplo, de madeira, ferro, embocador, curto). A dimensão vertical representa os produtos fornecidos pelos diversos fabricantes de tacos de golfe. Portanto, um usuário poderia navegar horizontalmente pela grade para encontrar a coluna tacos curtos e então verticalmente para examinar os produtos fornecidos por aqueles fabricantes que vendem tacos curtos. Essa arquitetura de WebApp é útil apenas quando é encontrado conteúdo altamente regular [Pow00].

As *estruturas hierárquicas* (Figura 13.6) são, sem dúvida nenhuma, a arquitetura para WebApp mais comum. Ao contrário das hierarquias de software particionadas discutidas no Capítulo 9, que encorajam o fluxo de controle apenas ao longo das ramificações verticais da hierarquia, uma estrutura hierárquica para WebApp pode ser projetada para possibilitar (via ramificação de hipertexto) o fluxo de controle horizontal ao longo de ramificações verticais da estrutura. Portanto, o conteúdo apresentado no ramo mais à esquerda da hierarquia pode ter links de hipertexto que levam diretamente a conteúdo existente no meio ou no ramo mais à direita da estrutura. Deve-se notar, entretanto, que embora tal ramificação possibilite rápida navegação pelo conteúdo de uma WebApp, ela pode gerar confusão para o usuário.

Uma *estrutura em rede* ou “*pura teia*” (Figura 13.7) é similar em muitos aspectos à arquitetura que evoluiu para sistemas orientados a objetos. Os componentes da arquitetura (nesse caso, páginas Web) são projetados de modo que possam passar o controle (via links de hipertexto) para praticamente qualquer outro componente do sistema. Essa abordagem possibilita uma flexibilidade de navegação considerável, mas, ao mesmo tempo, pode ser confusa para o usuário.

FIGURA 13.6**Estrutura hierárquica****FIGURA 13.7****Estrutura em rede**

As estruturas da arquitetura discutidas nos parágrafos anteriores podem ser combinadas para formar *estruturas compostas*. A arquitetura geral de uma WebApp pode ser hierárquica, porém, parte de sua estrutura poderia apresentar características lineares, enquanto outra poderia ser em rede. Nosso objetivo como projetista de arquiteturas é combinar a estrutura da WebApp com o conteúdo a ser apresentado e o processamento a ser realizado.

13.7.2 Arquitetura de uma WebApp

A arquitetura de uma WebApp descreve a infraestrutura que permite a uma aplicação ou sistema baseado na Web atingir seus objetivos de aplicação. Jacyntho e seus colegas [Jac02b] descrevem as características básicas dessa infraestrutura da seguinte maneira:

As aplicações devem ser construídas usando-se camadas em que diferentes preocupações são levadas em conta; em particular, os dados da aplicação devem ser separados do conteúdo da página (nós de navegação) e, por sua vez, os conteúdos devem estar claramente separados dos aspectos da interface (páginas).

Os autores sugerem uma arquitetura de projeto em três camadas que desassocia a interface da navegação e do comportamento da aplicação. Eles argumentam que manter a interface, a aplicação e a navegação separadas simplifica a implementação e aumenta a reutilização.

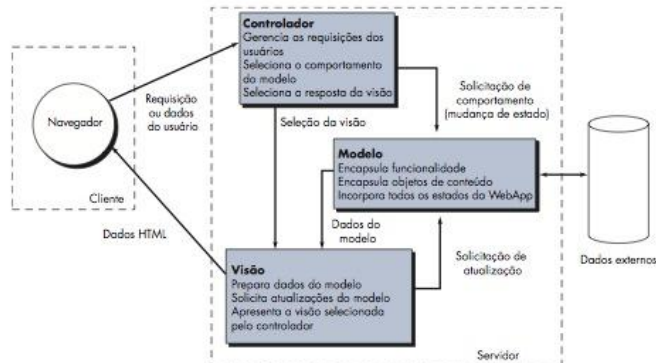
A arquitetura Modelo-Visão-Controlador (*Model-View-Controller MVC*) [Kra88]⁹ é uma de vários modelos de infraestrutura sugeridos para WebApps que desassocia a interface do usuário da funcionalidade e do conteúdo de informações de uma WebApp. O *modelo* (algumas vezes conhecido como “objeto-modelo”) contém todo o conteúdo e a lógica de processamento específicos à aplicação, inclusive todos os objetos de conteúdo, acesso a fontes de dados/informações externas e toda a funcionalidade de processamento específica para a aplicação. A *visão* contém todas as funções específicas à interface e possibilita a apresentação do conteúdo e lógica de processamento, inclusive todos os objetos de conteúdo, acesso a fontes de dados/informações externas e toda a funcionalidade de processamento requerida pelo usuário final. O *controlador* gerencia o acesso ao modelo e à visão e coordena o fluxo de dados entre eles. Em uma WebApp, “a visão é atualizada pelo controlador com dados do modelo baseados nas informações fornecidas pelos usuários” [WMT02]. Uma representação esquemática da arquitetura MVC está na Figura 13.8.

PONTO-CHAVE

A arquitetura MVC desassocia a interface do usuário da funcionalidade e do conteúdo de informação da WebApp.

FIGURA 13.8

A arquitetura MVC
Fonte: Adaptado de (Jac02)



⁹ Deve-se observar que a MVC é, na verdade, um padrão de projeto da arquitetura desenvolvido para o ambiente Smalltalk (veja www.cetus-links.org/oo_smalltalk.html) e que pode ser usado para qualquer aplicação interativa.

Referindo-se à figura, os dados ou solicitações de usuários são tratados pelo controlador, o qual também seleciona um objeto de visão aplicável baseado na solicitação do usuário. Uma vez determinado o tipo de solicitação, é transmitida uma solicitação de comportamento ao modelo, que implementa a funcionalidade ou recupera o conteúdo necessário para atender à solicitação. O objeto-modelo pode acessar dados armazenados em um banco de dados corporativo, como parte de um repositório de dados local ou de um conjunto de arquivos independentes. Os dados desenvolvidos pelo modelo devem ser formatados e organizados pelo objeto de visão apropriado e transmitidos do servidor de aplicações de volta para o navegador instalado no cliente para exibição na máquina do usuário.

Em muitos casos, a arquitetura da WebApp é definida no contexto do ambiente de desenvolvimento em que a aplicação será implementada. Caso tenha maior interesse, veja [Fow03] para uma discussão dos ambientes de desenvolvimento e seus papéis no projeto de arquiteturas para aplicações para Web.

13.8 PROJETO DA NAVEGAÇÃO

“Apenas espere, Mario, até que a lua surja e então iremos ver os migalhos de pão que espalhei pelo chão; eles nos indicarão o caminho de volta para casa.”

João e Maria

Assim que a arquitetura da WebApp tiver sido estabelecida e os componentes (páginas, scripts, applets e outras funções de processamento) da arquitetura identificados, temos de definir os percursos de navegação que permitirão aos usuários acessarem o conteúdo e as funções da WebApp. Para tanto, devemos: (1) identificar a semântica de navegação para diferentes usuários do site e (2) definir a mecânica (sintaxe) para atingir a navegação.

13.8.1 Semântica de navegação

Assim como muitas atividades de projeto para WebApps, o projeto da navegação começa considerando-se a hierarquia dos usuários e casos de uso relativos (Capítulo 5) desenvolvidos para cada categoria de usuário (ator). Cada ator deve usar a WebApp de forma ligeiramente distinta e, portanto, apresentar diferentes necessidades de navegação. Além disso, os casos de uso desenvolvidos para cada ator irão definir um conjunto de classes que engloba um ou mais objetos de conteúdo ou funções de WebApp. À medida que cada usuário interage com a WebApp, ele encontra uma série de *unidades semânticas de navegação* (*navigation semantic units, NSUs*) — “um conjunto de informações e estruturas de navegação relacionadas que colaboram no cumprimento de um subconjunto de requisitos de usuário relacionados” [Ca02].

Uma NSU é composta por um conjunto de elementos de navegação denominado *modos de navegação* (*ways of navigating, WoN*) [Gna99]. Um WoN representa o melhor percurso de navegação para atingir uma meta de navegação para um tipo de usuário específico. Cada WoN é organizado como um conjunto de *nós de navegação* (*navigational nodes, NN*) interligados por links de navegação. Em alguns casos, um link de navegação poderia ser uma outra NSU. Consequentemente, a estrutura geral de navegação para uma WebApp poderia ser organizada como uma hierarquia de NSUs.

Para ilustrarmos o desenvolvimento de uma NSU, consideremos o caso de uso **Selecionar Componentes do CasaSegura**:

Caso de uso: Selecionar Componentes do CasaSegura

A WebApp irá recomendar componentes de produto (por exemplo, painéis de controle, sensores, câmeras) e outras características (por exemplo, funcionalidade baseada em PC implementada por software) para cada cômodo e para a entrada externa. Se eu solicitar alternativas, a WebApp irá fornecê-las, caso elas existam. Serei capaz de obter informações descritivas e de preços para cada componente do produto. A WebApp criará e exibirá uma lista de materiais à medida que for selecionando vários componentes. Serei capaz de dar um nome à lista de materiais e salvá-la para referência futura (veja o caso de uso **Salvar Configuração**).

Os itens sublinhados na descrição do caso de uso representam classes e objetos de conteúdo que serão incorporados em uma ou mais NSUs que possibilitarão a um novo cliente representar o cenário descrito no caso de uso **Selecionar Componentes do CasaSegura**.

PONTO-CHAVE

Uma NSU descreve os requisitos de navegação para cada caso de uso. Em essência, a NSU mostra como um ator se movimenta pelos objetos de conteúdo ou funções da WebApp.

"O problema da navegação em um site é conceitual, técnico, espacial, filosófico e logístico. Consequentemente, as soluções tendem a exigir combinações improvisadas e complexas de arte, ciências e psicologia organizacional."¹⁰

Tim Morgan

A Figura 13.9 representa uma análise semântica parcial da navegação implícita no caso de uso **Selecionar Componentes do CasaSegura**. Usando a terminologia introduzida anteriormente, a figura também representa uma forma de navegação (WoN) para a WebApp **CasaSeguraGarantida.com**. São mostradas importantes classes de domínio do problema com objetos de conteúdo selecionados (nesse caso, o pacote de objetos de conteúdo chamado **DescriçãoDeComponente**, um atributo da classe **ComponenteDeProduto**). Esses itens são nós de navegação. Cada uma das setas representa um link de navegação¹⁰ e é rotulado com a ação iniciada pelo usuário que faz com que o link ocorra.

Podemos criar uma NSU para cada caso de uso associado ao papel de cada usuário. Por exemplo, um **novo cliente** do **CasaSeguraGarantida.com** poderia ter casos de uso diferentes, todos resultando no acesso a diferentes informações e funções de WebApp. É criada uma NSU para cada objetivo.

Durante os estágios iniciais do projeto de navegação, a arquitetura de conteúdo da WebApp é avaliada para determinar um ou mais WoNs para caso de uso. Conforme citado, um WoN identifica nós de navegação (por exemplo, conteúdo) e links que possibilitam a navegação entre eles. Os WoNs são organizados em NSUs.

13.8.2 Sintaxe de navegação

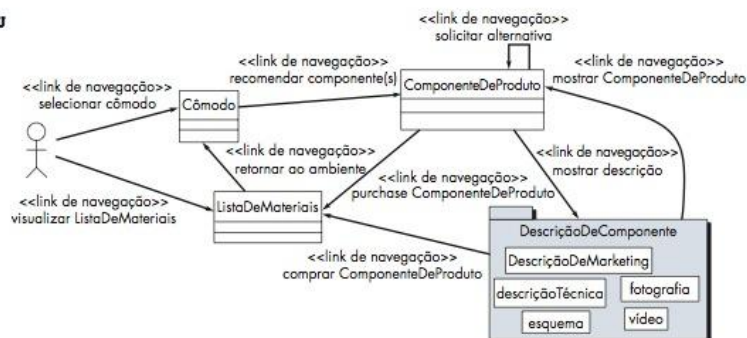
À medida que o projeto prossegue, sua tarefa é definir a mecânica de navegação. Estão disponíveis várias abordagens para implementação de cada NSU:

- **Link de navegação individual** — inclui links de texto, ícones, botões e chaves e metáforas gráficas. Temos de escolher links de navegação apropriados para o conteúdo e consistentes com a heurística que levam a uma interface de projeto de alta qualidade.
- **Barra de navegação horizontal** — lista as principais categorias funcionais ou de conteúdo em uma barra contendo links apropriados. Em geral, são listadas entre quatro e sete categorias.
- **Coluna de navegação vertical** — (1) lista as principais categorias funcionais ou de conteúdo ou (2) lista praticamente todos os principais objetos de conteúdo de uma WebApp. Caso escolha a segunda opção, tais colunas de navegação podem "expandir-se" para apresentar objetos de conteúdo como parte da hierarquia (selecionar uma entrada na coluna original fará com que uma expansão liste uma segunda camada de objetos de conteúdo relacionados).

AVISO

Na maioria das situações, opte por mecanismos de navegação horizontais ou verticais, mas não ambos.

FIGURA 13.9
Criação de uma NSU



¹⁰ Esses são, algumas vezes, conhecidos como *links de semântica de navegação* (*navigational semantic links, NSL*) [Cac02].



O mapa de um site deve ser acessível de qualquer página. O próprio mapa deve ser organizado de modo que a estrutura de informações da WebApp estejam prontamente visíveis.

- *Guias* — uma metáfora que nada mais é que uma variação de uma barra ou coluna de navegação, representando categorias funcionais ou de conteúdo, como páginas de guias selecionadas quando um link for necessário.
- *Mapas de sites* — fornecem um sumário completo para navegação a todos os objetos de conteúdo e funcionalidade contidos na WebApp.

Além de escolhermos a mecânica de navegação, também podemos estabelecer convenções e ferramentas de ajuda de navegação adequadas. Por exemplo, ícones e links gráficos devem ter um aspecto “cliqueável” por meio de elevação das arestas para conferir à imagem um aspecto tridimensional. Deve-se implementar feedback sonoro ou visual para dar ao usuário uma indicação de que a opção de navegação foi escolhida. Para navegação baseada em elementos textuais, devem-se usar cores para indicar links de navegação e fornecer uma indicação dos links já navegados. Essas são apenas algumas das dezenas de convenções de projeto que tornam a navegação mais amigável.

13.9 PROJETO DOS COMPONENTES

As WebApps modernas oferecem funções de processamento cada vez mais sofisticadas que: (1) executam processamento localizado para gerar recursos de navegação e conteúdo de forma dinâmica, (2) fornecem recursos de cálculo ou processamento de dados apropriados para o campo de aplicação da WebApp, (3) fornecem sofisticadas consultas e acesso a bancos de dados e (4) estabelecem interfaces de dados com sistemas corporativos externos. Para alcançarmos essas (e muitas outras) capacidades, temos de projetar e construir componentes de programa que sejam idênticos em sua forma aos componentes para software tradicional.

Os métodos de projeto discutidos no Capítulo 10 se aplicam aos componentes para WebApp com poucas, se realmente alguma, modificações. O ambiente de implementação, as linguagens de programação e os padrões de projeto, estruturas e software podem variar um pouco, porém, a abordagem de projeto geral permanece a mesma.

13.10 MÉTODO DE PROJETO DE HIPERMÍDIA ORIENTADO A OBJETOS (OBJECT-ORIENTED HYPERMEDIA DESIGN METHOD – OOHDM)

Foram propostos vários métodos de projeto para aplicações para Web ao longo da última década. Até hoje, nenhum método exclusivo atingiu uma predominância.¹¹ Nesta seção apresentamos uma visão breve de um dos métodos de projeto para WebApps mais amplamente discutidos — o OOHDM.

Daniel Schwabe e seus colegas [Sch95, Sch98b] propuseram originalmente o *método para projeto de hipermídia orientado a objetos* (Object-Oriented Hypermedia Design Method, OOHDM), que é composto por quatro atividades de projeto diversas: atividade conceitual, projeto da navegação, projeto da interface abstrata e implementação. Uma síntese dessas atividades de projeto é mostrada na Figura 13.10 e discutida brevemente nas seções a seguir.

13.10.1 Projeto conceitual para o OOHDM





O *projeto conceitual* para o OOHDM cria uma representação dos subsistemas, classes e relações que definem o domínio de aplicação para a WebApp. Poderíamos usar a UML¹² para criar diagramas de classes apropriados, agregações e representações de classes compostas, diagramas de colaboração e outras informações que descrevem o domínio de aplicação.

¹¹ Na realidade, relativamente poucos desenvolvedores de aplicações para Web usam um método específico ao projetar uma WebApp. Espera-se que essa abordagem de projeto *ad hoc* mude com o passar do tempo.

¹² O OOHDM não prescreve uma notação específica; entretanto, o uso da UML é comum quando se aplica esse método.

FIGURA 13.10

Resumo do método OOHDM
Fonte: Adaptado de (Sch95)

	 Projeto conceitual	 Projeto da navegação	 Projeto da interface abstrata	 Implementação
Artefatos	Classes, subsistemas, relacionamentos, atributos	Links de nós, estruturas de acesso, contextos de navegação, transformações de navegação	Objetos de interface abstrata, respostas a eventos externos, transformações	WebApp executável
Mecanismos de projeto	Classificação, composição, agregação, generalização, especialização	Mapeamento entre objetos conceituais e de navegação	Mapeamento entre objetos de navegação e perceptíveis	Recurso fornecido pelo Ambiente-alvo
Interesses de projeto	Modelagem semântica do domínio de aplicação	Leva em conta o perfil e as tarefas do usuário. Ênfase em aspectos cognitivos.	Modelagem de objetos perceptíveis, implementando as metáforas escolhidas. Descreve a interface para os objetos de navegação.	Correção; desempenho da aplicação; completude

Como um exemplo simples do projeto conceitual para o OOHDM, consideremos a aplicação de comércio eletrônico **CasaSeguraGarantida.com**. Um “esquema conceitual” parcial está na Figura 13.11. Os diagramas de classes, as agregações e informações relativas desenvolvidas como parte da análise de WebApps são reutilizados durante o projeto conceitual para representar relações entre as classes.

13.10.2 Projeto da navegação para o OOHDM

O *projeto da navegação* identifica um conjunto de “objetos” obtidos das classes definidas no projeto conceitual. São definidas “classes de navegação” ou “nós” para encapsular esses objetos. Poderíamos usar a UML para criar casos de uso, tabelas de estados e diagramas de sequências apropriados — todas as representações que nos ajudam a compreender melhor os requisitos de navegação. Além desses, podem ser usados os padrões de projeto para o projeto de navegação à medida que o projeto é desenvolvido. O OOHDM usa um conjunto de classes de navegação predefinido — nós, links, âncoras e estruturas de acesso [Sch98b]. As estruturas de acesso são mais elaboradas e incluem mecanismos como um índice de WebApps, um mapa do site ou um *tour* orientado.

Assim que as classes de navegação forem definidas, o OOHDM “estrutura um espaço de navegação por meio do agrupamento de objetos de navegação em conjuntos denominados contextos” [Sch98b]. Um *contexto* inclui uma descrição da estrutura de navegação local, restrição imposta ao acesso de objetos de conteúdo e métodos (operações) necessários para efetuar o acesso a objetos de conteúdo. É desenvolvido um template de contextos (análogo aos cartões CRC discutidos no Capítulo 6) que pode ser usado para acompanhar os requisitos de navegação de cada categoria de usuário através dos vários contextos definidos no OOHDM. Desse modo, vêm à tona caminhos de navegação específicos (aquilo que denominamos WoN na Seção 13.8.1).

13.10.3 Projeto da interface abstrata e implementação

A ação de *projeto da interface abstrata* especifica os objetos de interface que o usuário vê à medida que ocorre a interação da WebApp. Um modelo formal de objetos de interface, denominado *visão abstrata de dados* (*abstract data view, ADV*), é usado para representar o relacio-

o uso de elementos gráficos e decisões estéticas relacionadas. Um conjunto de diretrizes para design gráfico fornece a base para uma abordagem ao projeto.

O projeto de conteúdo define o layout, a estrutura e um resumo para todo o conteúdo apresentado como parte da WebApp e estabelece as relações entre os objetos de conteúdo. O projeto de conteúdo começa com a representação dos objetos de conteúdo, suas associações e relações. Um conjunto de rudimentos de navegação estabelece a base para o projeto de navegação.

O projeto da arquitetura identifica a estrutura de hipermídia geral para a WebApp e engloba tanto a arquitetura de conteúdo quanto a arquitetura da WebApp. Os estilos da arquitetura para conteúdo incluem estruturas lineares, em grade, hierárquicas e em rede. A arquitetura da WebApp descreve uma infraestrutura que permite a um sistema ou aplicação baseados na Web atingir os objetivos de seu domínio de aplicação.

Projeto de navegação representa um fluxo de navegação entre objetos de conteúdo e para todas as funções da WebApp. A semântica da navegação é definida descrevendo-se um conjunto de unidades semânticas de navegação. Cada unidade é composta por formas de navegação e links e nós de navegação. A sintaxe de navegação representa os mecanismos usados para efetuar a navegação descrita como parte da semântica.

O projeto de componentes desenvolve a lógica de processamento detalhada para implementar os componentes funcionais que implementam uma função de WebApp completa. As técnicas de projeto descritas no Capítulo 10 se aplicam à criação de componentes para WebApps.

O método para projeto de hipermídia orientado a objetos (OOHDM) é um dos vários métodos propostos para projeto de WebApps. O OOHDM sugere um processo de projeto que abrange os projetos conceitual, de navegação e da interface abstrata, bem como a implementação.

PROBLEMAS E PONTOS A PONDERAR

13.1. Por que o “ideal artístico” é uma filosofia de projeto insuficiente ao se construírem WebApps modernas? Existe algum caso em que o ideal artístico é a filosofia a ser seguida?

13.2. Neste capítulo escolhemos uma ampla gama de atributos de qualidade para WebApps. Escolha os três que você acredita serem os mais importantes e defenda um argumento que explique por que cada um deve ser enfatizado no trabalho de projeto para WebApps.

13.3. Acrescente pelo menos cinco outras questões ao projeto de WebApps check-list de qualidade apresentado na Seção 13.1.

13.4. Você é um projetista de WebApps da *FutureLearning Corporation*, uma empresa de aprendizagem à distância. Você pretende implementar um “mecanismo de aprendizagem” baseado na Internet que deixará à disposição conteúdo de cursos para os estudantes. O mecanismo de aprendizagem fornece a infraestrutura básica para transmissão de conteúdo de aprendizagem sobre qualquer tema (os projetistas de conteúdo prepararão o conteúdo apropriado). Desenvolva um protótipo de projeto da interface para o mecanismo de aprendizagem.

13.5. Qual o site mais esteticamente atraente que você já visitou até hoje e por quê?

13.6. Considere o objeto de conteúdo **Pedido**, gerado assim que o usuário do **CasaSeguraGarantida.com** tenha completado a escolha de todos os componentes e esteja pronto para finalizar sua compra. Desenvolva uma descrição UML para **Pedido** com todas as representações de projeto apropriadas.

13.7. Qual a diferença entre arquitetura de conteúdo e arquitetura de uma WebApp?

13.8. Reconsiderando o “mecanismo de aprendizagem” da *FutureLearning* descrito no Problema 13.4, escolha uma arquitetura de conteúdo que seja apropriada para a WebApp. Discuta a razão para ter feito tal escolha.

13.9. Use UML para desenvolver três ou quatro representações de projeto para objetos de conteúdo que seriam encontrados enquanto o “mecanismo de aprendizagem” descrito no Problema 13.4 é desenhado.

13.10. Pesquise mais sobre a arquitetura MVC e decida se seria ou não a arquitetura de Web App apropriada para o “mecanismo de aprendizagem” discutido no Problema 13.4.

13.11. Qual a diferença entre sintaxe de navegação e navegação semântica?

13.12. Defina duas ou três NSUs para a WebApp **CasaSeguraGarantida.com**. Descreva cada uma delas com certo nível de detalhe.

13.13. Redija um breve artigo sobre um método de projeto de hipermídias que não seja o OOHDM.

LEITURAS E FONTES DE INFORMAÇÃO COMPLEMENTARES

Van Duyne e seus colegas (*The design of Sites*, 2. ed., Prentice Hall, 2007) escreveram um livro completo que cobre os mais importantes aspectos do processo de projeto de WebApps. Modelos de processo de projeto e padrões de projeto são vistos em detalhe. Wodtke (*Information Architecture*, New Riders Publishing, 2003), Rosenfeld e Morville (*Information Architecture for the World Wide Web*, O'Reilly & Associates, 2002) e Reiss (*Practical Information Architecture*, Addison-Wesley, 2000) tratam arquitetura de conteúdo e outros tópicos.

Embora tenham sido escritos centenas de livros sobre “web design”, poucos discutem quaisquer métodos técnicos significativos para a realização do trabalho de projeto. Na melhor das hipóteses, são apresentadas várias diretrizes úteis para projeto de WebApps, são mostrados exemplos de páginas Web e programação Java que valem a pena, e são discutidos detalhes técnicos importantes para a implementação de WebApps modernas. Entre as várias ofertas dessa categoria temos livros como os de Sklar (*Principles of Web Design*, 4. ed., Course Technology, 2008), McIntire (*Visual Design for the Modern Web*, New Riders Press, 2007), Niederst (*Web Design in a Nutshell*, 3. ed., O'Reilly, 2006), Eccher (*Advanced Professional Web Design*, Charles River Media, 2006), Cederholm (*Bulletproof Web Design*, New Riders Press, 2005) e Shelly e seus colegas (*Web Design*, 2. ed., Course Technology, 2005). A discussão enciclopédica de Powell [Pow02] bem como a aprofundada discussão de Nielsen [Nie00] sobre projeto também são livros que valem a pena fazer parte de qualquer biblioteca.

Livros como os de Beard (*The Principles of Beautiful Web Design*, SitePoint, 2007), Clarke e Holzschlag (*Transcending CSS: The Fine Art of Web Design*, New Riders Press, 2006) e Golbeck (*Art Theory for Web Design*, Addison Wesley, 2005) enfatizam o projeto estético e devem ser lidos por profissionais com pouca experiência no assunto.

A visão ágil de projeto (e outros tópicos) para WebApps é apresentada por Wallace e seus colegas (*Extreme Programming for Web Projects*, Addison-Wesley, 2003). Conallen (*Building Web Applications with UML*, 2. ed., Addison-Wesley, 2002) e Rosenberg e Scott (*Applying Use-Case Driven Object Modeling with UML*, Addison-Wesley, 2001) apresentam exemplos detalhados de WebApps modeladas com o emprego da UML.

Também são mencionadas técnicas de projeto no contexto de livros escritos para ambientes de desenvolvimento específicos. Os leitores interessados devem examinar obras sobre HTML, CSS, J2EE, Java, .NET, XML, Perl, Ruby on Rails, Ajax e uma série de aplicações para criação de WebApps (*Dreamweaver*, *HomePage*, *Frontpage*, *GoLive*, *MacroMedia Flash* etc.) para obter úteis informações sobre design.

Uma ampla gama de fontes de informação sobre projeto de WebApps se encontra à disposição na Internet. Uma lista atualizada de referências na Web, relevante para o projeto de WebApps, pode ser encontrada no site www.mhhe.com/engcs/compsci/pressman/professional/olc/ser.htm.