Rational® XDE™ Diretrizes da Estrutura do Modelo para J2EE™

Rational Software White Paper
TP 154, 05/03

Índice Analítico

1. Introdução

Este documento fornece recomendações sobre como representar e estruturar os artefatos do modelo RUP® no Rational XDE™, Java Platform Edition. Logicamente, a decisão de modelar esses artefatos RUP no XDE é uma questão específica do projeto. Neste documento, comentamos sobre aqueles modelos XDE que oferecem suporte de automação e aqueles que não oferecem, porque isso pode influenciar em sua decisão.

Como existem todos os modelos XDE dentro de projetos XDE, a seção Estrutura do Projeto XDE fornece recomendações sobre quais projetos XDE devem ser criados e quais modelos XDE devem ser criados nesses projetos.

Tanto o RUP como o XDE utilizam o termo "model" e o mapeamento entre os modelos RUP e os modelos XDE nem sempre é de um para um. Na seção Mapeamento do Modelo RUP para o Modelo XDE , está descrito o mapeamento dos modelos RUP para modelos XDE.

A estrutura de cada um dos artefatos do modelo RUP em seus arquivos do modelo XDE está, portanto, descrita em sua própria seção.

2. Escopo

Este documento destaca a descrição das estruturas do arquivo de modelo XDE recomendadas, não no processo para desenvolver conteúdo dos artefatos RUP associados. Este documento também não descreve a heurística detalhada para definir os projetos XDE que contêm os modelos XDE descritos. Para obter informações sobre como definir, desenvolver e modelar os conteúdos dos artefatos RUP, consulte o RUP. Para obter informações adicionais sobre projetos, consulte a documentação do IDE.

Este documento não descreve um exemplo completo, mas utiliza exemplos selecionados que enfatizam os pontos a servem abrangidos; entretanto, todos os exemplo são consistentes uns com os outros e são tirados de modelos XDE reais.

Esta versão do documento não trata do desenvolvimento de biblioteca de tags.

As estruturas do projeto e do modelo descritas neste documento são apenas recomendações e podem ser substituídas por qualquer número de estruturas igualmente válidas.

3. Estrutura do Projeto XDE

O objetivo deste documento é demonstrar como estruturar os modelos XDE. Entretanto, como todos os modelos XDE existem dentro de projetos XDE, é importante fornecermos uma breve introdução sobre a estrutura do projeto no qual existem as estruturas do modelo recomendadas.

Para um aplicativo corporativo J2EE que está sendo desenvolvido por várias pessoas, recomendamos a criação dos seguintes projetos e modelos XDE.
Nota: Se forem utilizados os assistentes para "criar projeto" do XDE, muitos modelos serão criados automaticamente na criação do projeto. De fato, se você estiver utilizando WSS AD XDE, se criar um *Projeto de Modelagem de Aplicativo Corporativo,* a maior parte dessa estrutura de projeto múltipla será criada automaticamente, inclusive muitos modelos. O XDE também fornece gabaritos de modelo para antecipar o conteúdo dos modelos.

Projeto XDE	Descrição	Modelos XDE "<nome do modelo recomendado>" (<tipo de arquivo XDE: gabarito do modelo>]
Projeto do Aplicativo (projeto Modelagem Básica XDE)	O Projeto do Aplicativo representa todo o aplicativo. Contém os arquivos de modelo XDE que descrevem o aplicativo por inteiro.	- "Modelo de Caso de Uso" (Rational XDE: Modelo de Caso de Uso) - "Modelo de Análise" (Rational XDE: Modelo de Análise) - "Modelo de Design Global" (Rational XDE: Modelo de Design) - "Modelo de Implementação Global" (Rational XDE: Modelo em Branco) - "Modelo de Implantação EAR" (Java: Modelo de Implantação EAR)
Projeto de Modelagem de Dados (Projeto de Modelagem de Dados XDE)	O Projeto de Modelagem de Dados contém os recursos necessários para modelar os dados do aplicativo, assim como aplicar a engenharia ida e volta de um Modelo de dados para/de um banco de dados.	- "Modelo de Dados Lógicos" (Dados: Modelo de Dados Lógicos) - "Modelo de Dados Físicos" (Dados: *arquivo de modelo de dados físicos específicos do fornecedor*) [1] - "Modelo de Domínio" (Dados: *arquivo de modelo de domínio específico do fornecedor*)
Projetos EJB (Projeto de Modelagem XDE EJB)	Os projetos EJB contêm os recursos necessários para implementar EJB(s). Os elementos contidos são empacotados e implantados como um módulo EJB (arquivo .EJB-JAR). Podem ser definidos projetos EJB separados para EJBs individuais ou conjuntos de EJB (cada projeto EJB pode conter no máximo um Modelo de Código Java). A recomendação é criar um projeto EJB para cada EJB-JAR que deve ser criado. Se os projetos separados são definidos, então o nome do projeto deve refletir seu conteúdo.[2]	- "Modelo do Código EJB" (Java: Modelo do Código EJB) - "Modelo de Implantação EJB" (Java: Modelo de Implantação EJB)
Projetos da Web (Projeto de Modelagem da Web XDE)	Projetos da Web representam os recursos da Web do aplicativo. Os elementos contidos são empacotados e implantados em um arquivo archive da Web (arquivo WAR). Projetos da Web separados podem ser definidos para áreas específicas da apresentação. A	- "Modelo de Código Java" (Java: Modelo de Código Java 1.3/1.4) - "O Modelo de Biblioteca de Tags JSP" (Web: Modelo de Biblioteca de Tags JSP)[4] - "Modelo de Diretório Virtual" (Web: Modelo de Diretório Virtual[5]

[1] Rational XDE fornece suporte para banco de dados físico para vários fornecedores de banco de dados. Um gabarito específico do fornecedor existe para cada fornecedor de banco de dados suportado pelo XDE.

[2] Se você estiver utilizando XDE para WSAD, quando criar os Projetos EJB (Modelagem) adicionais, o assistente pedirá um Projeto de Aplicativo para hospedar o EAR. Você deve reutilizar o nome do Projeto de Aplicativo (Modelagem) acima.

[3] Se você estiver utilizando XDE para WSAD, quando criar os Projetos da Web (Modelagem) adicionais, o assistente pedirá um Projeto de Aplicativo para hospedar o EAR. Você deve reutilizar o nome do Projeto de Aplicativo (Modelagem) acima.

[4] Podem ser vários modelos de Biblioteca de Tags por projeto. De fato, você precisa de um modelo separado para cada arquivo .tld. Em todo caso, esta versão do documento não trata do desenvolvimento de biblioteca de tags.

[5] Podem ser vários modelos de Diretório Virtual por projeto da Web XDE.

	recomendação é criar um projeto da Web para cada WAR que precise ser produzido. Se os projetos separados são definidos, então o nome do projeto deve refletir seu conteúdo. [3]	- "Modelo de Implantação da Web" (Web: Modelo de Implantação da Web)

Em exemplo desse tipo de projeto e da organização do modelo está mostrado na Figura 1 (observe os nomes de modelo exclusivos).

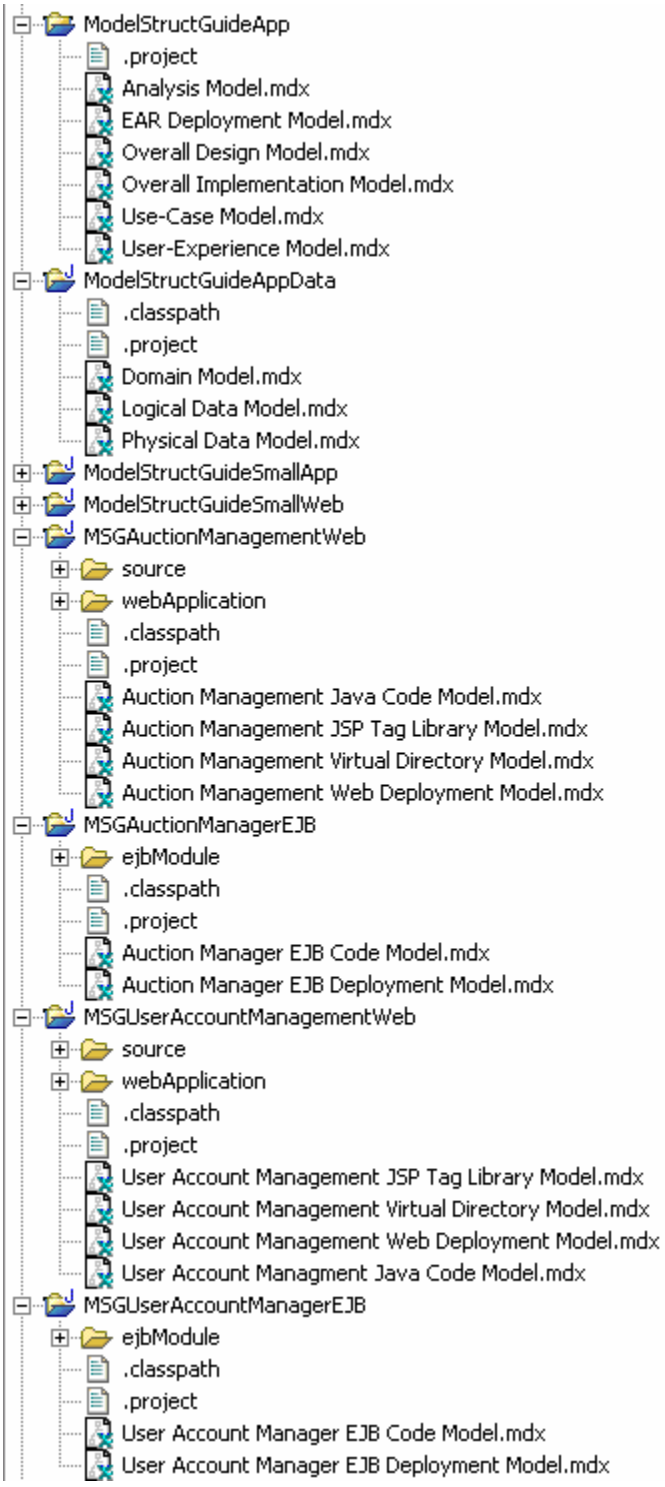

Figura 1:Exemplo de Projeto XDE e Organização de Modelo

Alternativamente, se o aplicativo for realmente pequeno e for desenvolvido por uma única pessoa, a estrutura do projeto acima pode ser simplificada para dois projetos, um que contenha os elementos de todo o aplicativo e os elementos não-Web, e outro que contenha os elementos da Web. Além de reduzir o número de projetos, o número

de modelos pode ser reduzido também. Por exemplo, para um projeto pequeno e de uma pessoa só, são possíveis as seguintes simplificações.

- Um Modelo de Análise Separado não é mantido. A análise e o design são desempenhados nos modelos de ida e volta XDE.
- Um "Modelo de Design Global" e um "Modelo de Implementação Global" não são mantidos. O projeto é pequeno o suficiente, de maneira que é possível ter uma visão geral examinando diretamente os modelos de ida e volta XDE. Também, as Realizações de Caso de Uso são mantidas no modelo de código EJB e são incluídas aos elementos no modelo do Diretório Virtual.
- Um Modelo de Dados Lógico separado não é mantido. Um esquema de dados físicos é desenvolvido diretamente no "Modelo de Dados Físicos".

Uma "estrutura de projeto pequeno" está resumida na tabela a seguir.

Projeto XDE	**Descrição**	**Modelos XDE "<recommended model name>" (<XDE file type: model template>]**
Projeto do Aplicativo (Projeto de Modelagem XDE EJB)	O Projeto do Aplicativo representa os aspectos não-Web do aplicativo. Contém os modelos que descrevem o aplicativo como um todo, o Modelo de Dados, e os modelos específicos ao EJB.	- "Modelo de Caso de Uso" (Rational XDE: Modelo de Caso de Uso) - "Modelo de Dados Físicos" (Dados: *arquivo de modelo de dados físicos específicos do fornecedor*) - "Modelo do Código EJB" (Java: Modelo do Código EJB) - "Modelo de Implantação EJB" (Java: Modelo de Implantação EJB) - "Modelo de Implantação EAR" (Java: Modelo de Implantação EAR)
Projeto da Web (Projeto de Modelagem da Web XDE)	Projetos da Web representam os recursos da Web do aplicativo. Os elementos contidos são empacotados e implantados em um arquivo archive da Web (arquivo WAR).	- "Modelo de Código Java" (Java: Modelo de Código Java 1.3/1.4) - "O Modelo de Biblioteca de Tags JSP" (Web: Modelo de Biblioteca de Tags JSP)[6] - "Modelo de Diretório Virtual" (Web: Modelo de Diretório Virtual[7] - "Modelo de Implantação da Web" (Web: Modelo de Implantação da Web)

[6] Podem ser vários modelos de Biblioteca de Tags por projeto. De fato, você precisa de um modelo separado para cada arquivo .tld. Em todo caso, esta versão do documento não trata do desenvolvimento de biblioteca de tags.

[7] Podem ser vários modelos de Diretório Virtual por projeto da Web XDE.

Um exemplo de um projeto pequeno e da organização do modelo está mostrado na Figura 2.

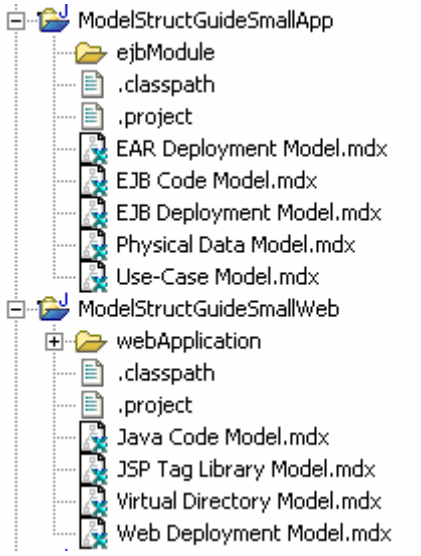

Figura 2: Exemplo de Projeto XDE Pequeno e Organização de Modelo

A atual seleção do número de projetos e os arquivos de modelo individuais são uma escolha de arquitetura e podem variar para aplicativos diferentes. Entretanto, não importa quantos projetos são definidos, pode haver apenas um arquivo de modelo Java XDE por projeto. Para obter informações adicionais sobre projetos e os arquivos de modelo XDE que eles podem conter, consulte a documentação XDE.

Também, é recomendado que os *nomes de modelo XDE sejam exclusivos em todos os projetos XDE*. Isso é extremamente importante ao tentar resolver referências entre os modelos XDE. Para obter informações adicionais sobre referências entre modelos e resolvê-las, consulte a documentação XDE.

Para os exemplos deste documento, o projeto e a estrutura do modelo estão mostrados na Figura 1 é utilizado. Observe que vários projetos EJB e da Web foram definidos. Para obter o fundamento de vários projetos EJB e da Web, consulte a seção Subsistemas de Implementação .

4. Modelo RUP para Mapeamento de Modelo XDE

Antes de descrever como representar os artefatos do modelo RUP em XDE, é importante considerar a confusão entre um "modelo RUP" e um "modelo XDE", porque são coisas diferentes e o mapeamento dos modelos RUP para os modelos XDE associados nem sempre é de um para um (fechado, mas não de um-para-um). Como o "modelo" é utilizado em ambos, no RUP e no XDE, a suposição inicial é de que eles devam ser os mesmos. Entretanto, os modelos no RUP divergem-se no que diz respeito ao processo (análise x design x implementação, etc.), onde os modelos em XDE divergem-se no que diz respeito a desenvolvimento (modelos de código separados para descrever a estrutura de empacotamento da linguagem de programação versus uma estrutura de diretório virtual, modelos de código separados para diferentes linguagens de programação e ambientes de desenvolvimento, etc.). Para esclarecer essa confusão, no contexto deste white paper, o termo “model” é explicitamente qualificado com “RUP” ou “XDE”.

A tabela a seguir resume o mapeamento do modelo RUP para o modelo XDE. Os modelos XDE são aqueles modelos introduzidos na seção Estrutura do Projeto XDE . A estrutura de cada um dos modelos XDE está descrita nas seções posteriores deste white paper.

Modelo RUP	**<XDE Project>: < XDE Model Name>**
Modelo de Caso de Uso	Projeto do Aplicativo: Modelo de Caso de Uso
Modelo de Análise	Projeto do Aplicativo: Modelo de Análise
Modelo de Design	Projeto do Aplicativo: Modelo de Design Global [As Classes de Design em cada um dos arquivos de modelo de ida e volta XDE (veja abaixo)]
Modelo de Dados	Modelos de dados XDE: - Projeto de Modelagem de Dados: Modelo de Dados Lógicos - Projeto de Modelagem de Dados: Modelo de Dados Físicos *Específico do Fornecedor* - Projeto de Modelagem de Dados: Modelo de Domínio *Específico do Fornecedor*
Modelo de Implementação	Projeto do Aplicativo: Modelo de Implementação Global Modelos de ida e volta XDE [8] - Projeto EJB: Modelos de Código EJB - Projetos da Web: Modelos de Código Java - Projetos da Web: Modelos de Biblioteca de Tags JSP - Projetos da Web: Modelos de Diretório Virtual
Modelo de Implantação	Modelos de implantação XDE [9] - Projeto de Aplicativo: Modelo de Implantação EAR"[10] - Projetos EJB: Modelos de Implantação EJB - Projetos da Web: Modelos de Implantação da Web

[8] Para abreviar, utilizarei "modelos de ida e volta XDE" em todo este white paper para representar esses modelos XDE.

[9] Para abreviar, utilizarei "modelos de implantação XDE" em todo este white paper para representar esses modelos de XDE.

[10] O Modelo de Implantação EAR "cruza" os modelos de implantação XDE individual. Contém diagramas que descrevem os nós de implantação e suas conexões. Também contém diagramas que mapeam os arquivos archive individuais, definidos nos modelos de implantação individual, para os nós de implantação.

5. Modelo de Caso de Uso

A estrutura recomendada do "Modelo de Caso de Uso" está apresentada na Figura 3.

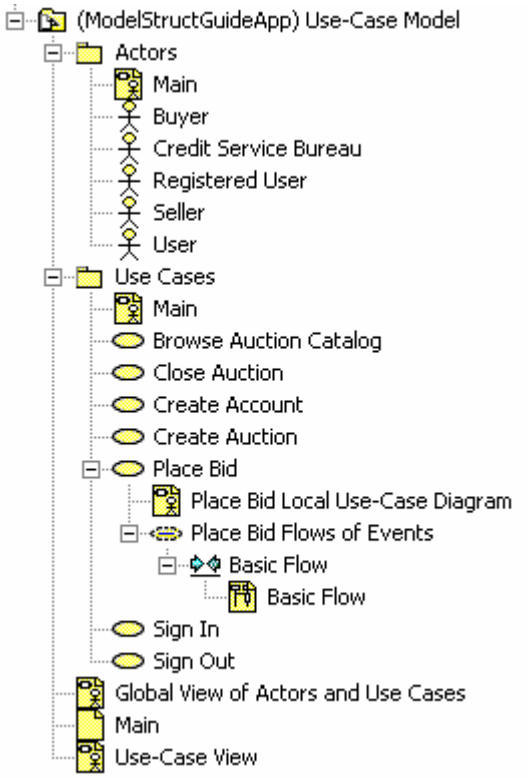

Figura 3: Estrutura do "Modelo de Caso de Uso"

O "Modelo de Caso de Uso" está particionado em dois pacotes: " Agentes" e "Casos de Uso".

Além dos diagramas do **Modelo Caso de Uso** que contêm os **Agentes** e os **Casos de Uso**, diagramas adicionais podem ser utilizados para diferenciar aspectos dos **Casos de Uso**. Podem ser incluídos os seguintes elementos de modelo suplementar "sob" o elemento do modelo de **Caso de Uso** no **Modelo de Caso de Uso**, conforme mostrado em Figura 3:

- O "Diagrama de Caso de Uso de Licitação de Local" contém a "Licitação de Local" **Caso de Uso** e os **Agentes** que participam deste **Caso de Uso**.
- A instância de colaboração "Fluxos de Eventos da Licitação de Local" contém as instâncias de colaboração que descrevem graficamente os fluxos dos eventos descritos na descrição de caso de uso (ou seja, as interações entre os **Agentes** e o **Caso de Uso**). As instâncias de colaboração de caso de uso não devem ser confundidas com **Realizações de Caso de Uso**, descritas nas seções Modelo de Análise e Realizações de Caso de Uso do Design , como as instâncias de colaboração no "Modelo de Caso de Uso" são estritamente "caixa preta" e não descrevem interações dos elementos dentro do aplicativo.
- O gráfico de atividade "Fluxos de Eventos da Licitação de Local" contém os diagramas de atividade que descrevem graficamente os fluxos dos eventos contidos na descrição de caso de uso.

No exemplo mostrado em Figura 3, o diagrama "Visão Global de Agentes e Casos de Uso" em Figura 3 contém todos os **Casos de Uso** e **Agentes** e seus relacionamentos, diferentes dos diagramas "Principais" , que contém apenas os elementos nos pacotes em que existem os diagramas "Principais" . Se houver muitos **Agentes** e **Casos de Uso**, as informações no diagrama "Visão Global dos Agentes e Casos de Uso" podem ser expressadas utilizando-se vários diagramas.

O diagrama "visualização de casos de uso" representa a Visualização de Casos de Uso da arquitetura de software. Para obter informações adicionais sobre Visualizações arquiteturais, consulte RUP.

Se desejado, podem ser criados pacotes adicionais nos pacotes "Agentes" e "Casos de Uso" para organizar melhor os elementos contidos no modelo, conforme mostrado na Figura 4.

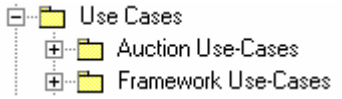

Figura 4: Particionamento do Pacote de Casos de Uso Adicional

6. Modelo de Análise

O Modelo de Análise é onde residem as **Classes de Análise** e a análise de **Realizações de Caso de Uso** .

Nota: Se é ou não necessário manter um **Modelo de Análise** e **Modelo de Design** separados, essa será uma decisão específica do projeto. Se um **Modelo de Análise** for criado, mas não mantido, então as **Classes de Análise** serão mostradas dentro da partição apropriada do **Modelo do Design** [11] e redefinido. Outra opção é criar as **Classes de Análise** e analisar as **Realizações de Caso de Uso** no **Modelo de Design** e então e expandi-las em seus formulários de design a partir de lá. Consulte a seção Modelo de Design para obter mais informações sobre como o **Modelo de Design** é representado no XDE.

A estrutura recomendada para o **Modelo de Análise** está mostrada na Figura 5.

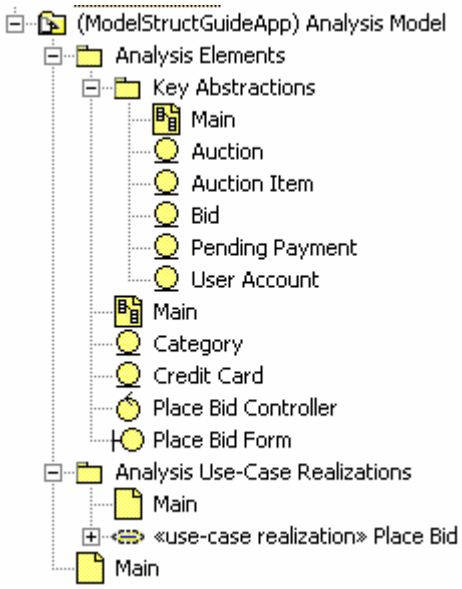

Figura 5: Estrutura do Modelo de Análise

O pacote "Elementos de Análise" contém as **Classes de Análise**. Instâncias das **Classes de Análise** aparecem nos diagramas no pacote "Análise das Realizações de Caso de Uso".

Além das **Classes de Análise**, pacotes podem ser definidos no pacote de "Elementos de Análise" para particionar ainda mais as **Classes de Análise** contidas (consulte o pacote "Abstrações de Tecla em Figura 5). Esse particionamento adicional é opcional, especialmente se não for mantido um **Modelo de Análise** . Nesses casos, as **Classes de Análise** podem ser consideradas "transitórias" (ou seja, existem apenas até serem expandidas nos elementos de design), assim sua organização não é considerada crítica. Uma exceção possível é a abstração de tecla **Classes de Análise**.

Conforme mostrado em Figura 5, o pacote "Abstrações de Teclas" contém as **Classes de Análise** que são consideradas para representar as abstrações de teclas do sistema. Conforme observado anteriormente, este pacote é

[11] Como você verá na seção posterior, a partição "do Modelo do Design " correta só pode ser um pacote em um dos modelos de ida e volta XDE, pois o design dos elementos específicos à tecnologia é realizado nos modelos de ida e volta.

opcional. Uma alternativa é representar as abstrações de tecla em um diagrama de classe no pacote "Elementos de Análise" . Entretanto, criando um pacote separado é possível oferecer uma categorização mais explícita das **Classes de Análise** como abstrações de tecla. De fato, mesmo se não for mantido um **Modelo de Análise** em sua totalidade, alguns projetos podem escolher manter a abstração de tecla **Classes de Análise**. Nesses casos, é útil definir um pacote separado para conter as **Classes de Análise** que são mantidas.
Nota: As abstrações de tecla também aparecem no diagrama "Visualização Lógica: Abstrações de Tecla" no "Modelo de Design Global". Consulte a seção Modelo de Design para obter informações adicionais.

O pacote "Realizações de Caso de Uso de Análise" contém as **Realizações de Caso de Uso**em nível de análise, o que descreve como os **Casos de Uso** são desempenhados em termos de **Classes de Análise** no pacote "Elementos de Análise". Cada uma das análises de **Realizações de Caso de Uso** concebe um **Caso de Uso** no **Modelo de Caso de Uso**, possui o mesmo nome que o **Caso de Uso** , e deve ter a estrutura conforme mostrada na Figura 6.

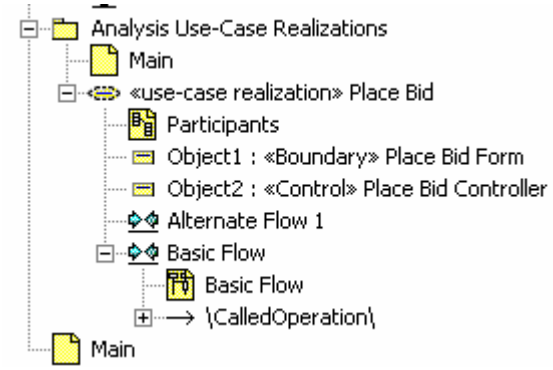

Figura 6: "Realização do Caso de Uso de Análise" Estrutura do Pacote

O diagrama "Participantes" mostra as **Classes de Análise** (do pacote "Elementos de Análise") participando na **Realização de Caso de Uso** (ou seja, aquelas **Classes de Análise** cujas instâncias aparecem nos diagramas de interação) e os relacionamentos que suportam a colaboração descrita nos diagramas de interação.

As instâncias de interação de "fluxo" ("Fluxo Básico" e "Fluxo Alternado 1") contêm diagramas de seqüência que descrevem os fluxos de **Casos de Uso** dos eventos. Deve haver uma instância de interação para cada fluxo de eventos de caso de uso significativo. Os diagramas de seqüência nas instâncias de interação descrevem o fluxo entre as **Classes de Análise** participantes durante a execução do **Caso de Uso**associado.

7. Modelo de Design

O **Modelo de Design RUP** é representado por vários modelos XDE – "Modelo de Design Global" e os elementos de design de ida-e-volta que residem em modelos de ida-e-volta XDE separados (elementos de design em ida-e-volta são elementos de design detalhados que participam da engenharia de ida-e-volta). Dessa forma, a automação disponível nos modelos de ida-e-volta individuais podem ser alavancados. Por exemplo, os padrões XDE EJB podem ser utilizados para criar as classes que especificam o EJB.

O "Modelo de Design Global" descreve o design do aplicativo como um todo e contém elementos que expandem vários modelos de ida-e-volta XDE. Contém as partições lógicas que inspiram a organização dos modelos de ida-e-volta individuais, bem como as **Realizações de Caso de Uso** que combinam tudo (as **Realizações de Caso de Uso** descrevem a colaboração entre os elementos de design dos diferentes modelos de ida-e-volta). O "Modelo de Design Global" contém diagramas que se referem aos elementos de design de ida-e-volta. Para obter informações sobre modelos de ida e volta XDE individuais, consulte a seção Modelo de Implementação .

Outra possibilidade é representar o **Modelo de Design** e o **Modelo de Implementação** no mesmo modelo de código XDE. Isso só é possível se você tiver apenas um idioma de implementação de destino e sua equipe for pequena. Para obter um exemplo de uma estrutura de projeto pequeno, consulte a seçãoEstrutura de Projeto XDE .

Manter o "Modelo de Design Global" é opcional, mas pode ser uma boa idéia para diagramas organizacionais, alto nível de abstração, etc., bem como para fornecer um local para elementos de design enquanto se resolve qual mecanismo de implementação aplicar.

A estrutura recomendada do “Modelo de Design Global” está mostrada na Figura 7.

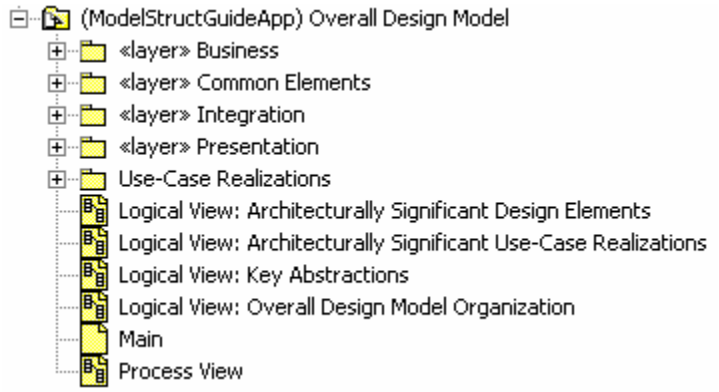

Figura 7: Estrutura de Modelo de Design Global

O Modelo de Design Global contém os seguintes pacotes:

- Os pacotes «layer» contêm (ou contêm diagramas que fazem referência) os elementos de design do sistema(**Classes de Design**, **Interfaces**, e **Subsistema de Design**). Essa estrutura representa uma estratégia de particionamento particular que nós descrevemos na seção Camadas de Design .
- O pacote de “Realizações de Caso de Uso ” contém as Realizações de Caso de Uso em nível de design. A estrutura interna das Realizações de Caso de Uso está discutida com maiores detalhes nas seções Realizações de Caso de Uso do Design .

Os diagramas que representam as visualizações arquiteturais incluem “Visualização” no nome do diagrama. Para obter informações adicionais sobre Visualizações arquiteturais, consulte RUP.

O diagrama “Visualização Lógica: Abstrações de Tecla ” contém as abstrações de tecla do sistema. Existem várias opções para manter essas abstrações de tecla:

- É mantido um **Modelo de Análise** completo. Nesse caso, o diagrama “visualização lógica: Abstrações de Tecla” contém as **Classes de Análise** do **Modelo de Análise** que representa as abstrações de tecla do sistema.
- Um **Modelo de Análise** é mantido, isto é, apenas as abstrações de tecla. Nesse caso, o diagrama “visualização lógica: Abstrações de Tecla” contém as **Classes de Análise** do **Modelo de Análise** que representa as abstrações de tecla do sistema.
- Nenhuma parte do **Modelo de Análise** é mantida. Neste caso, as **Classes de Análise** que representam as abstrações de tecla podem ser mantidas em um pacote no **Modelo de Design**, chamado de “Abstrações de Tecla”

Para obter informações adicionais sobre **Modelo de Análise**, consulte a seção Modelo de Análise .

7.1 Camadas de Design

Os pacotes «layer» contém os elementos de design do sistema (ou seja, **Classes de Design**, **Interfaces**, e **Subsistemas de Design**) que originam-se das **Classes de Análise**. Os pacotes «layer» podem conter qualquer número de subpacotes que particionam ainda mais os elementos de design contidos. O design **Realizações de Caso de Uso** (contido no pacote “Realizações de Caso de Uso” do “Modelo do Design” está discutido sob o título na seção Realizações de Caso de Uso do Design) estão escritos em termos de elementos de design contidos nesses pacotes.

O **Modelo de Design** pode seguir qualquer número de estratégias de particionamento. A estratégia de particionamento descrita nesta seção está mostrada em Figura 8.

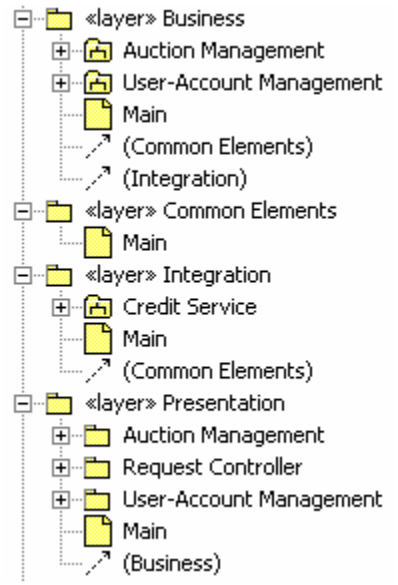

Figura 8: Exemplo de Particionamento de Pacote de Design

Neste exemplo, os pacotes de primeiro nível são considerados camadas, onde cada camada possui uma responsabilidade específica. Os pacotes do segundo nível particionam ainda mais os elementos do pacote de camadas pela funcionalidade dos negócios.

O pacote da camada "Apresentação" é responsável por lidar com as interações com o usuário final. Em um aplicativo J2EE, os elementos de design que podem residir no pacote da camada de "Apresentação" incluem as páginas HTML, Java Server Pages (JSPs) e servlets. Você pode dividir ainda mais o pacote da camada "Apresentação" em subpacotes para agrupar elementos que pertencem a um conjunto relacionado de **Casos de Uso**; por exemplo, o pacote "Gerenciamento de Leilão" em Figura 8.

O pacote da camada "Negócios" é responsável por desempenhar qualquer processamento de negócios. Na estrutura de "Modelo de Design " apresentada neste documento, pacote de camada "Negócios" é composto de um conjunto de pacotes de subsistema, um por função de negócios principal (por exemplo, os pacotes de subsistema "Gerenciamento de Leilão" e "Gerenciamento de Conta do Usuário" na Figura 8). Os pacotes **Subsistema de Design** estão descritos em maiores detalhes sob o título, na seção Subsistemas de Design .

O pacote de camada "Integração" é responsável por fornecer acesso aos recursos de backend, incluindo bancos de dados e sistemas externos. Na estrutura **Modelo de Design** apresentada neste documento, o pacote da camada "Integração" também é compactado de pacotes de subsistemas de design, um por sistema externo (por exemplo, o pacote de subsistema "Serviço de Crédito" na Figura 8). Os pacotes **Subsistema de Design** estão descritos em maiores detalhes sob o título, na seção Subsistemas de Design .

O pacote de camada "Elementos Comuns" contém os elementos que são compartilhados através das camadas.

Novamente, a estrutura descrita nesta seção pode ser substituída por uma estrutura diferente que reflete uma estratégia de particionamento diferente.

7.2 Subsistemas de Design

Subsistemas de Design são representados pelos pacotes de subsistemas no "Modelo de Design Global". Cada pacote de subsistema de design deve ter a mesma estrutura. As partes específicas da estrutura varia dependendo do nível de detalhamento que estão sendo capturados pelo **Subsistema de Design**.

Um exemplo de uma estrutura de **Subsistemas de Design** mais formal e rigorosa está mostrado na Figura 9.

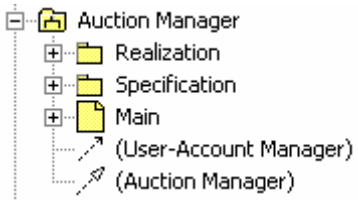

Figura 9: Estrutura de Subsistema de Design

Esta estrutura de pacote de subsistema de design suporta a definição de pacotes de "Especificação" e "Realização" separados dentro do pacote do subsistema de design. Essa estrutura foi influenciada pelo título do livro *UML Components: A Simple Process for Specifying Component-Based Software* escrito por J. Cheesman e J. Daniels. Uma estrutura de pacote de subsistema de design que não contém essas partições pode ser utilizada sem impactar as outras estruturas de arquivo de modelo definidas neste documento. Cada um dos pacotes "Especificação" e "Realização" está discutido nas próximas seções.

7.2.1 Especificação do Subsistema

O pacote "Especificação" contém uma descrição das interfaces do **Subsistema de Design**.[12] Um exemplo de uma especificação de subsistema está mostrado na Figura 10.

Figura 10: Exemplo de Especificação do Subsistema de Design

7.2.2 Realização do Subsistema

O pacote "Realização" contém uma descrição de como a especificação do **Subsistema de Design** é realizada. Um exemplo do pacote de "Realização" de um pacote de subsistema de design está mostrado na Figura 11.

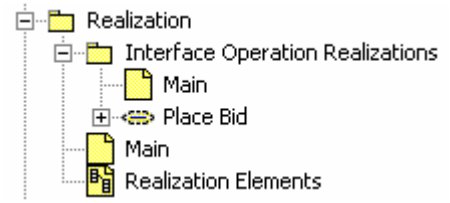

Figura 11: Exemplo de Realização de Subsistema de Design

O diagrama "Elementos de Realização" contém referências aos elementos de design que realizam o subsistema. Os próprios elementos de design podem residir no pacote "Realização" ou podem residir em um modelo de código XDE separado, onde participam da engenharia de ida e volta. Para obter informações adicionais, consulte a seção Modelos de Ida e volta XDE .

[12] Neste exemplo, você pode questionar a necessidade de um pacote separado apenas para a interface. Entretanto, em um projeto real o pacote deve ser mantido porque contém referências a documentos que descrevem o subsistema e, em particular, restrições da interface, como condições prévias e condições póstumas nas operações.

O pacote "Realizações de Operação de Interface" contém instâncias de colaboração que descrevem como os elementos do subsistema realizam as operações importantes das interfaces do **Subsistema de Design** (no pacote "Especificação"). Há uma instância de colaboração por operação de interface de subsistema importante.[13] Um exemplo de um pacote de "Realizações de Operação de Interface" está mostrado na Figura 12.

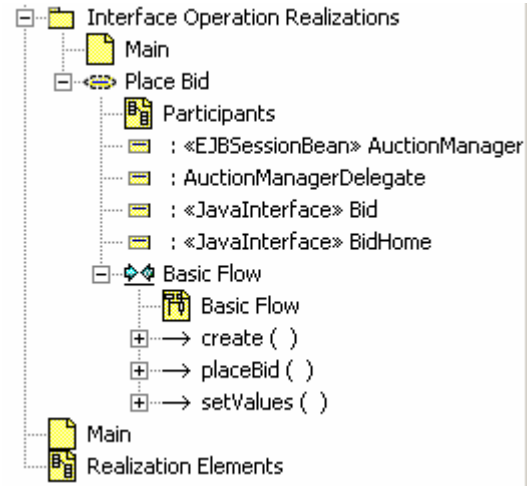

Figura 12: Exemplo de Pacote de Realizações da Operação de Interface

Assim como com as **Realizações de Caso de Uso** em nível de análise (discutidas anteriormente na seção Modelo de Análise) e as **Realizações de Caso de Uso em nível de design** (discutido posteriormente na seção Realizações de Caso de Uso do Design), cada realização de operação de interface contém um diagrama de classes com os elementos do subsistema que participam na realização (o diagrama "Participantes" na Figura 12), assim como os diagramas de interação que descrevem como esses participantes colaboram para realizar a operação de interface do subsistema (o diagrama "Fluxo Básico" em Figura 12).

7.3 Realizações de Caso de Uso do Design

O pacote "Realizações de Caso de Uso" contém as **Realizações de Caso de Uso** em nível de design. Cada uma das **Realizações de Caso de Uso** está associada a um **Caso de Uso** no **Modelo de Caso de Uso**, possui o mesmo nome que o **Caso de Uso** e deve ter a estrutura mostrada na Figura 16.

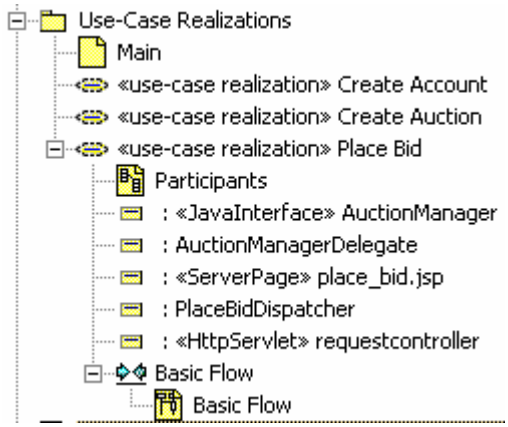

Figura 13: Estrutura de Realização do Caso de Uso do Design

O diagrama de **Realização de Caso de Uso** "Participantes" mostra os elementos de design que participam na **Realização de Caso de Uso** (ou seja, aqueles elementos de design cuja instâncias aparecem nos diagramas de

[13] Nem todas as operações precisam ser definidas neste nível. Algumas operações mais simples não precisam de uma instância de colaboração separada.

interação **Realização de Caso de Uso**) e os relacionamentos que suportam as colaborações descritas nos diagramas de interação.

O diagrama "Fluxo Básico" é um exemplo de um diagrama de interação que descreve o fluxo entre os elementos de design que participam durante a execução do **Caso de Uso**associado. Deve haver uma instância de interação para cada fluxo de evento no **Caso de Uso**.

É importante observar que os diagramas de **Realização de Caso de Uso** podem (e geralmente contêm) conter referências aos elementos de design que residem fisicamente em modelos de ida-e-volta XDE separados. A **Realização de Caso de Uso** é onde são demonstrados os elementos de colaboração em modelos de ida-e-volta separados.

8. Modelo de Dados

O **Modelo de Dados** RUP é representado por vários arquivos de modelo XDE:

- **Modelo de Dados Lógicos** (opcional). Representa o Modelo de Dados Lógicos, que é uma visualização independente do aplicativo do design lógico do banco de dados.
- **Modelo de Dados Físicos**. Representa um Modelo de Dados Físicos específico do fornecedor do banco de dados. Contém os elementos do modelo detalhados para definir as características específicas das tabelas do banco de dados. O "Modelo de Dados Físicos" arquivo de modelo XDE também inclui os artefatos de implementação específicos do banco de dados para implementar as tabelas em um banco de dados específico do fornecedor.
- **Modelo de Domínio** (opcional). Representa os tipos de dados específicos do fornecedor do banco de dados que podem ser utilizados para definir tipos de dados consistentes através do "Modelo de Dados Físicos".

A separação dos arquivos do modelo XDE fornece a flexibilidade opcional para automação suportada entre o **Modelo de Design**, o **Modelo de Dados** e o banco de dados físico.

Cada um desses arquivos de modelo XDE está descrito em maior detalhes abaixo.

8.1 Modelo de Dados Lógicos (Opcional)

O Modelo de Dados Lógicos pode ser utilizado em situações em que o projeto precisa criar uma representação de dados lógicos de entidades chave e os relacionamentos são importantes para o design do banco de dados. Criar um Modelo de Dados Lógicos XDE é opcional desde que a equipe de design do banco de dados possa, em lugar disso, transformar **Classes de Design** persistentes no **Modelo de Design** em tabelas no **Modelo de Dados** para criar a estrutura de design de banco de dados físico inicial diretamente no Modelo de Dados Físicos XDE (consulte a seção Modelo de Dados Físicos abaixo).

O Modelo de Dados Lógicos XDE pode ser particionado em pacotes de área de assunto , na medida em que for necessário. Os pacotes de área de assunto definem os agrupamentos lógicos das classes de entidade. O Modelo de Dados Lógicos também pode conter um pacote de "Elementos Comuns" que contêm elementos de modelo que cruzam as áreas de assunto.

Os diagramas com "Visualização" no nome são utilizados para documentar a Visualização de Dados da arquitetura. O diagrama "Visualização de Dados: Organização do Modelo de Dados Lógicos Global" é utilizado para documentar a organização de dados de alto nível do Modelo de Dados Lógico, como expresso nas principais partições (ou seja, pacotes) do Modelo de Dados Lógicos XDE. A "Visualização de Dados: Elementos de Dados Lógicos de Tecla" é utilizada para documentar os elementos lógicos de tecla do **Modelo de Dados**. Para obter informações adicionais sobre as visualizações de arquitetura, consulte RUP.

Um exemplo das estrutura recomendada para o Modelo de Dados Lógicos está mostrado na Figura 14.

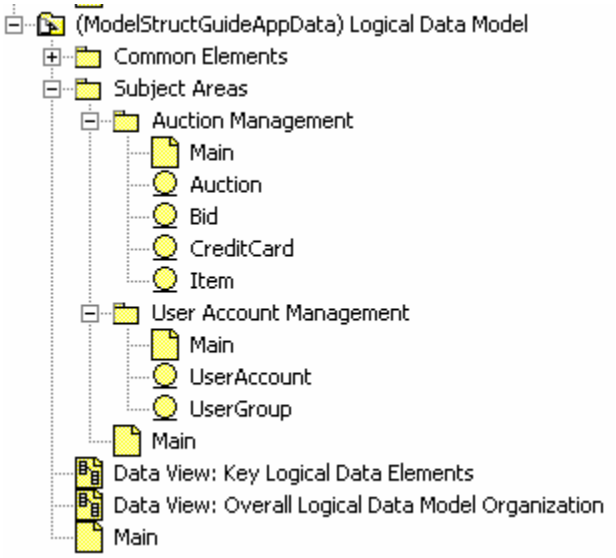

Figura 14: Estrutura de Modelo de Dados Lógicos

Neste exemplo, há dois pacotes de área de assunto, "Gerenciamento de Leilão" e "Gerenciamento de Conta do Usuário". Cada pacote de área de assunto contém as classes de entidade que compreendem juntas o Modelo de Dados Lógicos. Não é um mapeamento direto para as estruturas do pacote no **Modelo do Design** embora possa haver alguma similaridade.

8.2 Modelo de Dados Físicos

O Modelo de Dados Físicos contém a tabela de banco de dados detalhada e os designs de procedimentos armazenados que são utilizados para implementar o banco de dados através do recursos de engenharia de redirecionamento do Modelador de Dados XDE. O Modelo de Dados Físicos também consiste nos elementos de modelo utilizados para definir a configuração do armazenamento físico do banco de dados. Em geral, os elementos de modelo incluem os bancos de dados e tabelas de espaços que compreendem o layout físico das tabelas do banco de dados na mídia de armazenamento de destino.

Ao se criar o Modelo de Dados Físicos, o Designer de Banco de Dados precisa selecionar o banco de dados de destino apropriado. Bancos de dados suportados incluem: DB2 MVS, DB2 UDB, Oracle, Sybase e SQL Server. XDE padronizará o nome do arquivo de modelo XDE para o banco de dados selecionado. No exemplo "Modelo de Dados Físicos" neste documento, o nome do arquivo de modelo XDE foi atualizado para "Modelo de Dados Físicos". Um Designer de Banco de Dados pode escolher se aceita ou não o nome padrão ao criar o "Modelo de Dados Físicos".

Um exemplo da estrutura recomendada para o Modelo de Dados Físicos está mostrado na Figura 15.

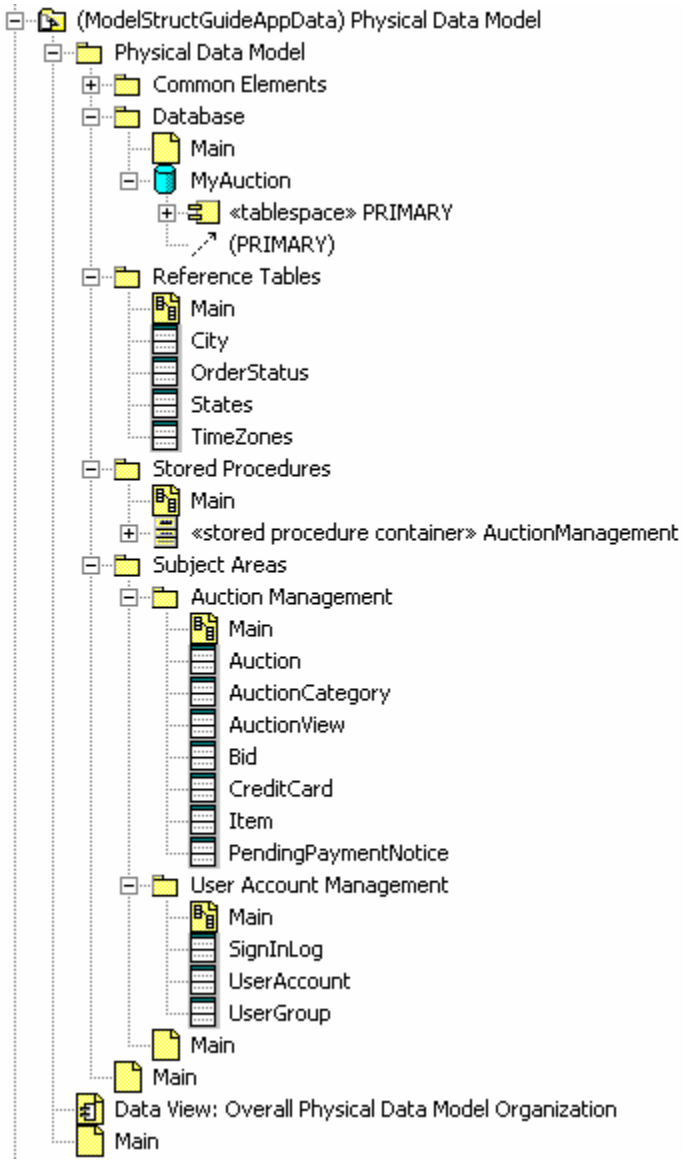

Figura 15: Estrutura de "Modelo de Dados Físicos"

O pacote "Elementos Comuns" contém as tabelas de banco de dados e visualizações que atravessam as áreas de assunto.

O pacote "Banco de Dados" contém os elementos de modelo que definem a configuração do armazenamento físico do banco de dados. Contém os bancos de dados e tabela de banco de dados que compreender o layout físico das tabelas de banco de dados na mídia de armazenamento de destino. As tabelas de espaços são utilizadas para agrupar logicamente tabelas dentro de um banco de dados. Para obter instruções sobre como definir tabelas de espaço, consulte RUP. O pacote "Banco de Dados" pode ser particionado em pacotes de nível inferior, conforme necessário, dependendo da complexidade do aplicativo.

No exemplo mostrado em Figura 15, o pacote "Banco de Dados" contém um único banco de dados, MyAuction, seu espaço de tabelas associado, PRIMARY, e os relacionamentos de realização de tabela. O espaço de tabelas pode ser nomeado com qualquer nome apropriado a um projeto de banco de dados. Para o banco de dados MyAuction, apenas um espaço de tabelas denominado PRIMARY é definido.Quando a engenharia de redirecionamento é realizada, são criadas as tabelas vinculadas ao banco de dados através do relacionamento de realização com o espaço de tabelas do banco de dados (ou em um banco de dados ou em um DDL).

O pacote “Tabelas de Referências” contém as tabelas de dados estáticos que mantêm “constantes” informações de dados necessárias ao aplicativo.

O pacote “Procedimentos Armazenados” contém todas as classes que representam os procedimentos armazenados de banco de dados («contêiner de procedimento armazenado » classes e as operações associadas de «procedimento armazenado»). Procedimentos armazenados que estão relacionados a uma única tabela podem ser empacotados ou no pacote de “Procedimentos Armazenados” ou no pacote “Área de Assunto”com a tabela das referências de procedimento armazenado, dependendo se você deseja ou não representar uma visualização “cêntrica do procedimento armazenado” ou uma visualização “cêntrica da tabela” [14].

O pacote “Áreas de Assunto” contém pacotes que agrupam logicamente conjuntos relacionados de tabelas e visualizações[15]. É recomendado que as visualizações sejam criadas no pacote de área de assunto juntamente com as tabelas. Essa recomendação é puramente por questões organizacionais. Pode ser útil para manter visualizações na área de assunto onde elas são utilizadas, colocando-as nas mesmas áreas de assunto que as tabelas. Neste exemplo mostrado na Figura 15, há dois pacotes de área de assunto, “Gerenciamento de Leilão” e “Gerenciamento de Conta do Usuário”. O número dos pacotes de área de assunto depende da complexidade do aplicativo. Entretanto, em geral, os pacotes de área de assunto no Modelo de Dados Lógicos “inspiram” os pacotes de área de assunto no Modelo de Dados Físicos. As áreas de assunto no Modelo de Dados Lógicos são abstrações das áreas de assunto no Modelo de Dados Físicos.

As tabelas nos pacotes de área de assunto contém as colunas e acionadores definidos para a tabela. As tabelas são criadas através de uma das seguintes

- funções de transformação de classe XDE em tabela.
- Engenharia reversa XDE, uma função de banco de dados existente[16].
- Criação manual pelo Designer de Banco de Dados.

Quando a engenharia reversa atua em um banco de dados existente, são criados pacotes de esquema no Modelo de Dados Físicos XDE. Os nomes desses pacotes são baseados no proprietário do banco de dados[17] do banco de dados que está sendo submetido à engenharia reversa. É recomendado que as tabelas da engenharia reversa sejam movidas para dentro dos pacotes de área de assunto com o pacote de “Áreas de Assunto” e que os pacotes do esquema em engenharia reversa sejam excluídos. O fato de mover as tabelas dentro dos pacotes da área de assunto organiza funcionalmente as tabelas para permitir que o Designer do Banco de Dados atualize as tabelas conforme necessário.

Os diagramas com “Visualização” no nome são utilizados para documentar a Visualização de Dados da arquitetura. O diagrama “Visualização de Dados: Organização do Modelo de Dados Físicos Geral” é utilizado para documentar a organização de dados de alto nível, conforme expresso nas principais partições (ou seja, pacotes) do Modelo de Dados Físicos XDE. Para obter informações adicionais sobre as visualizações de arquitetura, consulte RUP.

---

[14] Uma visualização cêntrica da tabela permite uma melhor compreensão do design do banco de dados / operação em uma só visualização. Uma visualização cêntrica de procedimento armazenado simplifica os atos de encontrar e alterar/manter do procedimento armazenado.

[15] Algumas pessoas podem questionar o uso de pacotes de área de assunto no modelo de dados físicos, já que isso requer manutenção adicional para conservar os pacotes de área de assunto do banco de dados físico e lógico. As áreas de assunto no modelo de dados físicos estão aqui para consistência com o modelo de dados lógicos (se ele for utilizado) e mais para o caso em que o modelo de dados físico é “grande” e não há modelo de dados lógicos. Neste caso os pacotes de área de assunto podem ser utilizados para gerenciar as tabelas geradas da transformação de Classe em Tabela.

[16] Normalmente o banco de dados é obtido em engenharia reversa uma vez, e então todas as atualizações futuras são sincronizadas utilizando-se as funções de Comparação e Sincronia do XDE.

[17] Dentro do XDE, o proprietário do banco de dados é capturado como uma propriedade do componentes do <<database>>. No interior da propriedade Local, como parte da cadeia de conexão, há um atributo de esquema. Quando a engenharia reversa atua em um banco de dados, geralmente é o proprietário do banco de dados.

8.3 Modelo de Domínio (Opcional)

O Modelo de Domínio é um modelo XDE opcional utilizado para armazenar tipos de dados definidos pelo usuário para o banco de dados. Os domínios permitem que os Designers de Banco de Dados reutilizem propriedades do elemento através do design do banco de dados. Um domínio é utilizado pelo Designer de Banco de Dados para documentar consistentemente as propriedades de uma coluna através do banco de dados. O nome da coluna é definido na tabela; o domínio é utilizado para definir a *TypeExpression* da coluna.

Um exemplo da estrutura recomendada para o Modelo de Domínio [18] está mostrado na Figura 14.

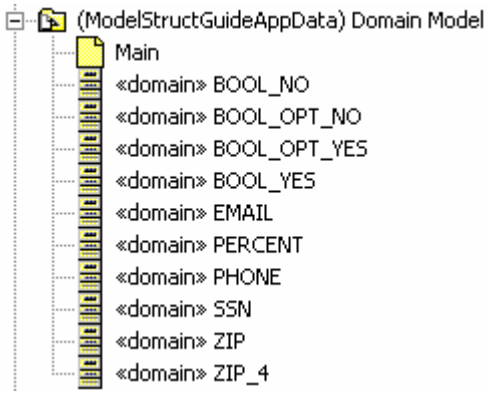

Figura 16: Estrutura do "Modelo do Domínio"

Neste exemplo, está demonstrado os valores de domínio do Servidor SQL organizados dentro do pacote de "Domínio do Servidor SQL " . Nos casos em que o Designer do Banco de Dados define um grande número de domínios, pode ser necessário ao Designer do Banco de Dados organizar os domínios utilizando pacotes sob o pacote de "Domínio do Servidor SQL" .

9. Modelo de Implementação

O Modelo de **Implementação RUP** é representado por vários modelos XDE – o "Modelo de Implementação Global" e vários modelos de ida e volta XDE [19]. A utilização de vários modelos de ida e volta permite diferentes preocupações de implementação a serem representadas com maior eficiência, como as descritas na estrutura de pacote Java versus a estrutura de diretório. Também, a automação disponível nos modelos de ida e volta individuais pode ser alavancada (ou seja, o XDE fornece automação para a criação de elementos de modelo específicos à tecnologia de implementação, e a sincronização desses elementos de modelo com código-fonte utilizando a engenharia de ida e volta).

O "Modelo de Implementação Global" descreve a implementação do aplicativo como um todo e contém elementos que expandem vários modelos de ida-e-volta. Contém diagramas que se referem aos elementos nos elementos de design em ida e volta, e geralmente não "possuem" eles próprios nenhum elemento de modelo. O "Modelo de Implementação Global" não participa de nenhuma engenharia de ida e volta XDE.

Manter o "Modelo de Implementação Global" é opcional; entretanto, ele pode ser útil para descrever a estrutura global da implementação, incluindo as unidades de integração (definidas em RUP como **Subsistemas de Implementação**), assim como para documentar a Visualização da Implementação da arquitetura. **Subsistemas de**

[18] Dentro do XDE, são suportados vários fornecedores de bancos de dados, inclusive DB2, Oracle, Sybase e SQL Server. Ao se criar um Modelo de Dados XDE de Domínio, o Designer de Banco de Dados criará o Modelo de Dados XDE de Domínio selecionando o banco de dados do fornecedor apropriado. O XDE criará uma lista padrão de domínios para o fornecedor de banco de dados selecionado.

[19] modelos de ida e volta XDE são modelos híbridos. São utilizados para representar ambos os modelos, **Modelo de Design RUP** e **Modelos de Implementação RUP**. Os elementos de modelo nos modelos de ida e volta XDE representam as **Classes de Design** Rup (classes que mapeam diretamente para classes físicas são consideradas **Classes de Design**) e arquivos de implementação física. As estruturas dos modelos de ida e volta XDE representam as estruturas de diretório físico.

Implementação são discutidos com maior detalhe ma seção Subsistemas de Implementação

O número de modelos de ida e volta XDE depende do projeto XDE definido e da organização do modelo (para obter informações adicionais, consulte a seção Estrutura do Projeto XDE). Entretanto, uma opção é definir projetos separados (e modelos de ida e volta) para cada **Subsistema de Implementação**. Para obter informações sobre como representar o **Subsistema de Implementação** no XDE, consulte a seção Subsistemas de Implementação . Alternativamente, como mencionado anteriormente, se você tiver um único idioma de implementação de destino e sua equipe for pequena, pode escolher utilizar um único modelo de ida e volta XDE para representar ambos os modelos, **Modelo de Design** e **Modelo de Implementação** RUP.

Um exemplo de um "Modelo doe Implementação Global" está mostrado na Figura 17.

Figura 17: Estrutura de Modelo de Implementação Global

Conforme mostrado em Figura 17, o "Modelo de Implementação Global" só contém diagramas. Os diagramas que representam as visualizações arquiteturais incluem "Visualização" no nome do diagrama. Para obter informações detalhadas sobre visualização arquitetural, consulte o RUP.

O diagrama "visualização de implementação: Elementos de Implementação Implantáveis" faz referência aos arquivo archive que serão implantados em nós nos modelos de implantação XDE. Os próprios arquivos archive estão contidos fisicamente nos modelos de implantação. Para obter informações adicionais, consulte a seção Modelo de Implantação .

O diagrama de "Visualização de Implementação: Subsistemas de Implementação" faz referência aos Subsistemas de Implementação do aplicativo. Os relacionamentos de dependência podem ser projetados entre os **Subsistemas de Implementação** no diagrama. Essas dependências representam as importações de **Subsistema de Implementação** , as quais determinam a ordem em que os **Subsistemas de Implementação** devem ser integrados. Para obter informações sobre como representar os Subsistema de Implementação no XDE, consulte a seção Subsistemas de Implementação .

O diagrama "Visualização de Implementação: Estrutura de Modelo de Implementação" contém referências a todos os modelos XDE utilizados para representar o **Modelo de Implementação**e seus relacionamentos.
Nota: Quando projetos individuais são definidos para cada um dos **Subsistemas de Implementação**, então o conteúdo do diagrama "Visualização de Implementação: Estrutura de Modelo de Implementação" é redundante com o diagrama "Visualização de Implementação: Subsistema de Implementação" , e pode ser omitido. Para obter informações adicionais sobre os Subsistemas de Implementação, consulte a seção Subsistemas de Implementação .

9.1 Subsistemas de Implementação

Os **Subsistemas de Implementação** RUP são unidade de integração[20]. Como tal, eles alinham-se muito bem com os módulos J2EE (**Subsistemas de Implementação** podem ser empacotados e implantados nos módulos J2EE). Como uma forma possível de configurar a estrutura do projeto XDE é definir um projeto XDE para cada archive J2EE, em seguida, os **Subsistemas de Implementação** identificados podem ser utilizados para conduzir quais projetos XDE devem ser definidos para suportar o design detalhado e a implementação. Especialmente, um **Subsistema de Implementação RUP** pode ser representado no XDE como um projeto XDE. Essa é a abordagem utilizada nessas diretrizes, na qual os **Subsistemas de Implementação** são representados utilizando projetos XDE.

Os **Subsistemas de Implementação** para os exemplos nestas diretrizes são os seguintes:

[20] As diretrizes para identificar **Subsistemas de Implementação** não estão dentro do escopo deste documento. Para obter informações adicionais, consulte o RUP.

- Um subsistema de implementação para cada subsistema de design no “Modelo de Design Global”. Por exemplo: Gerenciador de Conta e Gerenciador de Leilão.
- Um Subsistema de Implementação para os elementos de apresentação do Gerenciamento de Leilão.
- Um subsistema de implementação para os elementos da apresentação do Gerenciamento de Conta do Usuário.

Os projetos XDE associados e os modelos são mostrados na Figura 18.

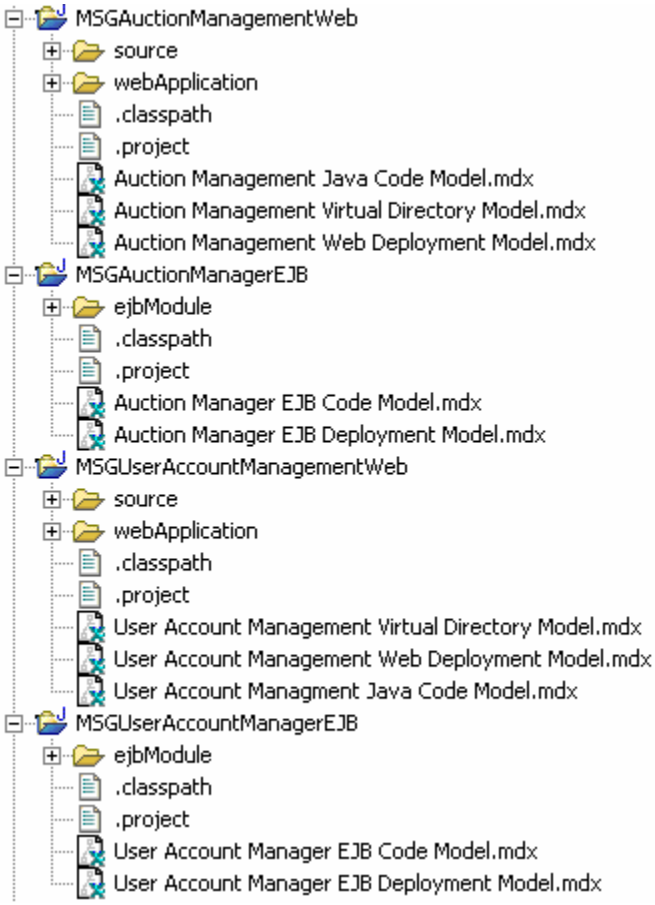

Figura 18: Subsistemas de Implementação como Projetos XDE

Quando vários projetos da Web e EJB são definidos, é recomendado que os nomes dos projetos e arquivos de modelo contidos sejam exclusivos. Isso torna muito mais fácil interpretar os diagramas nos modelos, bem como resolver as referências entre modelos. No exemplo mostrado em Figura 18, o nome do **subsistema de implementação** foi utilizado no nome do projeto, bem como o nome de cada um dos modelos contidos.

Alternativamente, se o projeto for pequeno, um **Subsistema de Implementação RUP** pode ser representado no XDE como pacote em um modelo XDE. Figura 19 fornece um exemplo do caso em que cada Subsistema de Implementação é representado utilizando um pacote XDE (os pacotes “auctionmanager” e “useraccountmanager”).

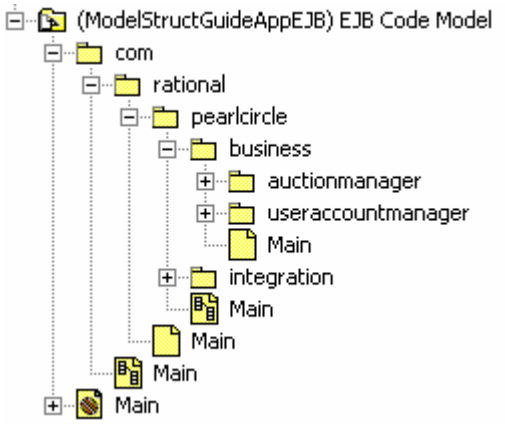

Figura 19: Subsistema de Implementaçõess como Pacotes

9.2 Modelos de Ida e Volta XDE

Esta seção fornece exemplos dos seguintes modelos de ida e volta XDE.

- (Projeto EJB) Modelo de Código EJB (consulte a seção Projeto EJB: Modelo de Código EJB)
- (Projeto da Web) Modelo de Código Java (consulte a seção Projeto da Web: Modelo de Código Java)
- (Projeto da Web) Modelo de Diretório Virtual (consulte a seção Projeto da Web: Modelo de Diretório Virtual)

Em geral, ao estruturar seus modelos de ida e volta XDE, recomenda-se alinhar a estrutura o mais próximo possível da estrutura lógica, conforme descrito na seção “Modelo de Design Global” (descrito na seção Modelo de Design), levando-se em consideração as forças da implementação, logicamente. Alinhando-se a estrutura do **Modelo do Design** e do **Modelo de Implementação** o mais próximo possível, a rastreabilidade entre eles torna-se implícita e mais direta para se manter. Isso é importante, desde que o mapeamento entre o **Modelo de Design** e o **Modelo de Implementação** seja algo que precise ser mantido e gerenciado (como parte da manutenção e gerenciamento da arquitetura).

As **Classes de Design** que são parte dos modelos de ida e volta XDE podem ser criadas através de um dos seguintes métodos

- automação XDE para criar elementos dependentes de tecnologia de implementação, como EJBs, servlets, etc., ou criando os elementos a partir do rascunho ou transformando os elementos de elementos independentes de tecnologia como as **Classes de Análise**.
- Engenharia reversa XDE de uma implementação existente
- Criação Manual.

9.2.1 Projeto EJB: Modelo de Código EJB

Um Modelo de Código EJB contém os recursos Java necessários para implementar EJBs (ou seja, classes de bean de implementação, classes de interface home, classes de interface remota, etc.).

A propriedade Raiz de Origem do Modelo de Código EJB deve ser definida para o diretório em que o código fonte será posicionado. Por exemplo, a propriedade Raiz de Origem do Modelo de Código EJB pode ser definida

para o subdiretório “ejbModule” do projeto EJB [21].

Um exemplo de um Modelo de Código EJB está mostrado na Figura 20.

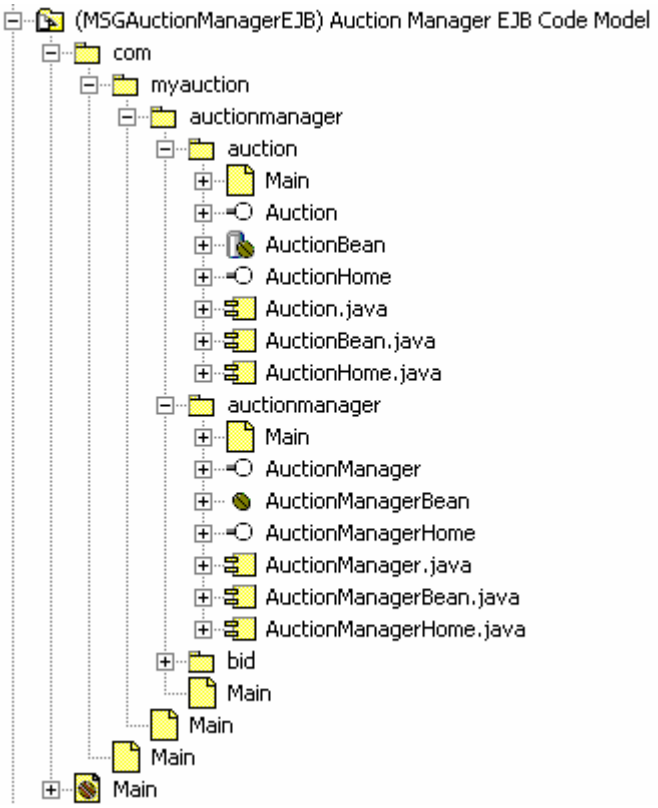

Figura 20: Exemplo de Modelo de Código EJB

O Modelo de Código EJB mostrado na Figura 20 contém os recursos Java que implementam o Subsistema de Design do Gerenciador de Leilão . É a contrapartida de implementação para o pacote de Subsistema de Design do “Gerenciador de Leilão” do pacote da camada de “Negócios” no “Modelo de Design Global ” discutido na seção Camadas de Design .

Como ilustrado na Figura 20, a estrutura do Modelo de Código EJB segue a convenção de utilizar o Nome de domínio como nome do pacote inicial Java. O Nome de domínio do Aplicativo de Amostra é “www.myauction.com”. Portanto, os pacotes que contêm os elementos de implementação são colocados dentro do pacote “myauction” dentro do pacote “ com” . Como resultado, todos os elementos Java dentro do pacote “myauction” terão um nome completo que é prefixado com “com.myauction”. Por exemplo, o nome completo do pacote “leilão” é “com.myauction.business.auctionmanager.auction”. A convenção de usar o nome de domínio como nome de pacote Java inicial garante que os nomes de classe Java serão exclusivos, mesmo se for incorporada uma biblioteca de classe Java de outro fabricante.

Dentro do pacote “myauction” , a estrutura reflete a estrutura do “Modelo de Design Global” (discutida na seção Modelo de Design). Há um pacote para a camada de design que contém o pacote de Gerenciador de Leilão **Subsistema de Design** (“negócios”) e, em seguida, há um pacote que representa o Gerenciador de Leilão **Subsistema de Design** (pacote “auctionmanager”). Pacote adicionais também são definidos no Modelo de Código EJB para coletar elementos de modelo relacionados (ou seja, os pacotes “auctionmanager”, “leilão”, e “licitação”). Nota: Como a linguagem de programação Java não permite espaços nos nomes de pacotes, um nome de pacote Java não pode ser idêntico ao nome do pacote “Modelo de Design Global” associado.

[21] Se você criar o projeto utilizando o assistente para criar projeto XDE, o subdiretório do projeto será criado automaticamente e a propriedade Raiz de Origem do modelo de código será definida automaticamente.

Conforme mostrado em Figura 20, um Modelo de Código EJB não contém apenas a representação visual dos arquivos de código-fonte (os elementos .java), mas também contém as classes que especificam esse elementos de implementação. Essas classes representam as **Classes de Design** que desenvolveram até o ponto em que elas possam ser implementadas e, no caso de XDE, passaram pela engenharia de ida e volta. Nota: XDE oferece padrões que criam automaticamente todas as classes utilizadas para especificar um EJB.

9.2.2 Projeto da Web: Modelo de Código Java

Um Modelo de Código Java de Projeto da Web contém os recursos da Web Java (ou seja, JavaBeans, servlets, classes ajudantes, etc.).

A propriedade Raiz de Origem do Modelo de Código Java do Projeto da Web deve ser definida para o diretório em que o código-fonte será posicionado. Por exemplo, a propriedade Raiz de Origem do Modelo de Código Java do Projeto da Web pode ser definida para o subdiretório "Origem Java Source" do projeto da Web[22].

Um exemplo de um Modelo de Código Java do Projeto da Web" está mostrado em Figura 21.

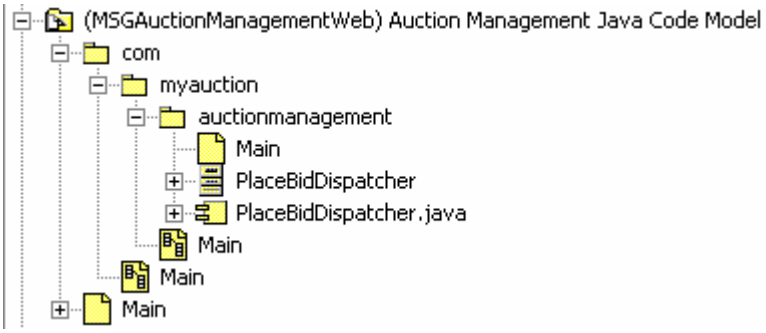

Figura 21: Exemplo de Modelo de Código Java do Projeto da Web

O Modelo de Código Java mostrado em Figura 21 contém os recursos Java que implementam os elementos de design da apresentação do Gerenciamento de Leilão. É a contrapartida da implementação Java para o pacote de "Gerenciamento de Leilão" do pacote da camada "Apresentação" no "Modelo de Design Global" discutido na seção Camadas de Design).

Como ilustrado na Figura 21, a estrutura do Modelo de Código Java do Projeto da Web segue a convenção de usar o nome de domínio como nome de pacote Java inicial. O Nome de domínio do Aplicativo de Amostra é "www.myauction.com". Portanto, os pacotes que contêm os elementos de implementação são colocados dentro do pacote "myauction" dentro do pacote " com" . Como resultado, todos os elementos Java dentro do pacote "myauction" terão um nome completo que é prefixado com "com.myauction". Por exemplo, o nome completo do pacote "auctionmanagement" é "com.myauction.presentation.auctionmanagement". A convenção de usar o nome de domínio como nome de pacote Java inicial garante que os nomes de classe Java serão exclusivos, mesmo se for incorporada uma biblioteca de classe Java de outro fabricante.

Dentro do pacote "myauction" , a estrutura reflete a estrutura do "Modelo de Design Global" (discutida na seção Modelo de Design). Há um pacote para a camada de design que contém os elementos de apresentação de Gerenciamento de Leilão (o pacote "apresentação"). Então há um pacote que representa os elementos da apresentação de Gerenciamento de Leilão (o pacote "auctionmanagement").
Nota: Como a linguagem de programação Java não permite espaços nos nomes de pacote, um nome de pacote Java pode não ser idêntico ao nome do pacote "Modelo de Design Global" associado.

Conforme mostrado em Figura 21, O Modelo de Código Java do Projeto da Web não apenas contém a representação visual dos arquivos de código-fonte (os elementos .java), mas também contém as classes que especificam esses elementos de implementação. Essas classes representam as **Classes de Design** que desenvolveram até o ponto em que elas possam ser implementadas e, no caso de XDE, passaram pela engenharia de ida e volta.

[22] Se você criar o projeto utilizando o assistente para criação do projeto XDE, o subdiretório do projeto será criado automaticamente e a propriedade Raiz de Origem do modelo de código será definida automaticamente.

9.2.3 Projeto da Web: Modelo de Diretório Virtual

Um Modelo de Diretório Virtual contém recurso da Web de código-fonte não-Java (por exemplo, páginas JSPs e HTML). Pode haver vários modelos de Diretório Virtual por projeto da Web XDE. Os Muitos Modelos de Diretório Virtual suportam o desenvolvimento de aplicativos J2EE que possuem muitos diretórios virtuais. Em tais aplicativos, o site da Web resultante é fisicamente dividido em diretórios que não possuem raiz comum. Por exemplo, em uma loja a varejo on-line, o endereço www.mystore.com seria utilizado para apresentar o catálogo, ao passo que o endereço order.mystore.com seria utilizado para monitorar o status de pedidos.

A propriedade de Raiz de Origem do Modelo de Diretório Virtual deve ser definida para o diretório em que serão posicionados os recursos da Web que serão implantados para o contêiner de Web. Por exemplo, a propriedade Raiz de Origem do Modelo de Diretório Virtual deve ser definida para o subdiretório "Conteúdo da Web" do projeto da Web[23].

Um exemplo de um Modelo de Diretório Virtual está mostrado na Figura 22.

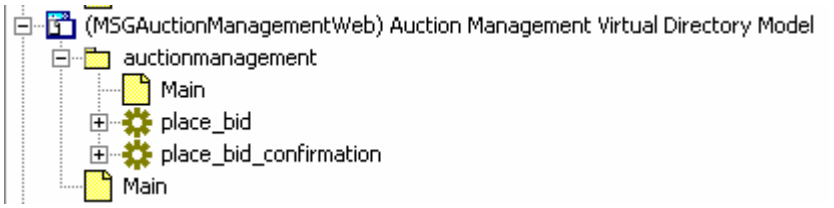

Figura 22: Estrutura do Modelo de Diretório Virtual

O Modelo de Diretório Virtual mostrado na Figura 22 contém recurso de código-fonte não-Java que implementam os elementos de design da apresentação de Gerenciamento de Leilão. É a contrapartida da implementação de código-fonte não-Java para o pacote "Gerenciamento de Leilão" do pacote da camada de "Apresentação" no "Modelo de Design Global" discutida na seção Camadas de Design .

Neste exemplo, a estrutura do Modelo de Diretório Virtual reflete a estrutura do "Modelo de Design Global" (discutida na seção Camadas de Design). Existe um pacote para cada pacote no "Modelo de Design Global", cujo conteúdo será implantado pelos elementos não-Java a serem implantados no contêiner de Web, como as páginas JSPs e HTML. Devido às restrições impostas pela forma de nomear os diretórios acessados pelo servidor da Web, os nomes dos diretórios sob o diretório virtual nem sempre são idênticos àqueles no "Modelo de Design Global".

Conforme mostrado em Figura 22, o Modelo de Diretório Virtual contém a representação visual das classes que especificam os elementos de implementação. Essas classes representam as **Classes de Design** que desenvolveram até o ponto em que elas possam ser implementadas e, no caso de XDE, passaram pela engenharia de ida e volta. Diferentemente, o Modelo de Código EJB e o Modelo de Código Java do Projeto da Web, o XDE não produz uma representação visual dos arquivos de código-fonte associados no Modelo de Diretório Virtual. O nome do arquivo de código-fonte associado é mantido como uma propriedade da classe.

10. Modelo de Implantação

O **Modelo de Implantação** RUP é representado por vários modelos XDE – o "Modelo de Implantação EAR" e um conjunto de modelos de implantação XDE individual, um para todo projeto XDE que contém elementos a serem implantados. O uso de vários modelos de implantação alavanca a automação da implantação do XDE (XDE fornece automação e refinamento dos arquivos archive J2EE e descritores de implantação, e a sincronização desses elementos de modelo com os elementos de modelo cuja origem é empacotada no archive).

Esta seção fornece exemplos dos seguintes modelos de implantação:

- (Projeto do Aplicativo) Modelo de Implantação EAR (consulte a seção Modelo de Implantação EAR)
- (Projeto EJB) Modelo de Implantação EJB (consulte a seção Modelo doe Implantação EJB)
- (Projeto da Web) Modelo de Implantação da Web (consulte a seção Modelo de Implantação da Web)

[23] Se você criar o projeto utilizando o assistente para criar projeto XDE, o subdiretório do projeto será criado automaticamente e a propriedade de Raiz de Origem do Modelo de Diretório Virtual será definida automaticamente.

A seguir estão alguns itens gerais valiosos sobre a implantação no XDE:

- Descritores de implantação não são representados explicitamente nos modelos de implantação XDE como arquivos (artefatos UML). Em vez disso, são representados pelos conteúdos dos modelos de implantação XDE. XDE utiliza os elementos contidos nos modelos de implantação, e os valores de propriedade atribuídos a esses elementos, para determinar as informações de conteúdo gravadas nos arquivos de descritor de implantação do modelo.
- O XDE utiliza um Modelo de Implantação EAR para todas as implantações, mesmo quando você implanta apenas um único EJB-JAR ou WAR. Isso reflete uma limitação de muitos servidores de aplicativo, que precisam de um EAR para todas as implantações. Então, mesmo se você estiver modelando apenas aplicativos do EJB-JAR ou WAR, ainda será necesário utilizar um Modelo de Implantação EAR para implantação, para os servidores de aplicativo suportados pelo XDE. Entretanto, o XDE pode *exportar* qualquer archive para o sistema de arquivos. Essa capacidade pode ser utilizada para implantar em servidores de aplicativos que o XDE não oferece suporte. Depois de exportar o archive, você pode chamar as ferramentas específicas do servidor para completar a implantação.
- Arquivos de archive J2EE diferentes podem ser definidos para diferentes ambientes de implantação (teste, produção, etc.). Esses archives podem ser definidos no mesmo modelo de implantação XDE ou em modelos de implantação XDE separados. A automação XDE suporte ambos, mas define os modelos de implantação padrão para projetos e archives padrões por modelo de implantação, mas você pode ajustar essas definições, já que são modeladas nos modelos de implantação XDE. Qualquer que seja a abordagem escolhida, os modelos de implantação EJB devem ser definidos em projetos EJB, e os modelos de implantação da Web devem estar nos projetos da Web ().[24]

10.1 Modelo de Implantação EAR

O "Modelo de Implantação EAR" contém a representação visual do arquivo archive do Aplicativo J2EE (arquivo .EAR), informações de descritor de implantação EAR e o nó em que o EAR deve ser implantado. O "Modelo de Implantação EAR" também pode conter diagramas que mostram quais módulos J2EE (de outros modelos de implantação) estão contidos no EAR.

O "Modelo de Implantação EAR" também pode ser utilizado para descrever a configuração de implantação do aplicativo como um todo, assim como a visualização da implantação da arquitetura. O "Modelo de Implantação EAR" pode conter diagramas que mostram todos os nós de implantação e suas conexões, assim como quais archives devem ser implantados para quais nós. Tais diagramas devem fazer referência a elementos (nós e archives) de todos os modelos de implantação individuais.

[24] Um Modelo de Implantação EJB deve estar em um Projeto EJB, mas não precisa ser o *mesmo* projeto que o Modelo de Código EJB correspondente. De fato, você pode "misturar e combinar" EJBs de diferentes modelos de códigos em um Modelo de Implantação. O mesmo se aplica a modelos da Web.

Um exemplo de um “Modelo de Implantação EAR” está mostrado na Figura 23.

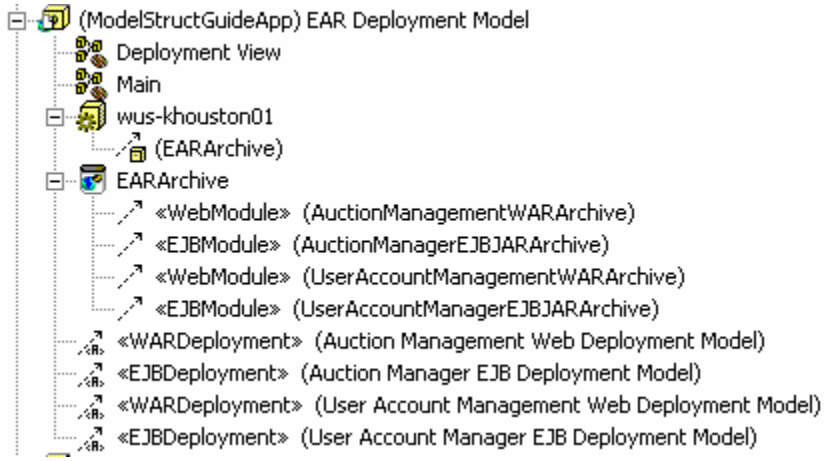

Figura 23: Modelo de Implantação EAR

O diagrama de “Visualização da Implantação” representa a visualização da implantação da arquitetura. Inclui os nós de implantação, suas interconexões e quais archives J2EE devem ser implantados nesses nós. Em alguns casos, esse diagrama pode incluir apenas o archive do Aplicativo J2EE e o nó do servidor de aplicativo, no qual ele será implantado. Entretanto, no caso em que são implantados archives do Módulo J2EE independentes para especificar nós, este diagrama deve mostrar quais archives são implantados em quais nós. Para obter informações sobre as visualizações arquiteturais, consulte o RUP.

10.2 Modelo de Implantação EJB

O Modelo de Implantação EJB contém componente(s) EJB, o artefato EJB-JAR, e as informações de descritor de implantação EJB. O Modelo de Implantação EJB também pode conter diagramas que mostram quais elementos de implementação (do Modelo de Código EJB) estão contidos no EJB-JAR. Para ser preciso, no Modelo de Implantação EJB, os componentes EJB são utilizados para representar os elementos de implementação, e são os componentes EJB que são mapeados para o EJB-JAR.

Um exemplo de um Modelo de Implantação EJB está mostrado na Figura 24.

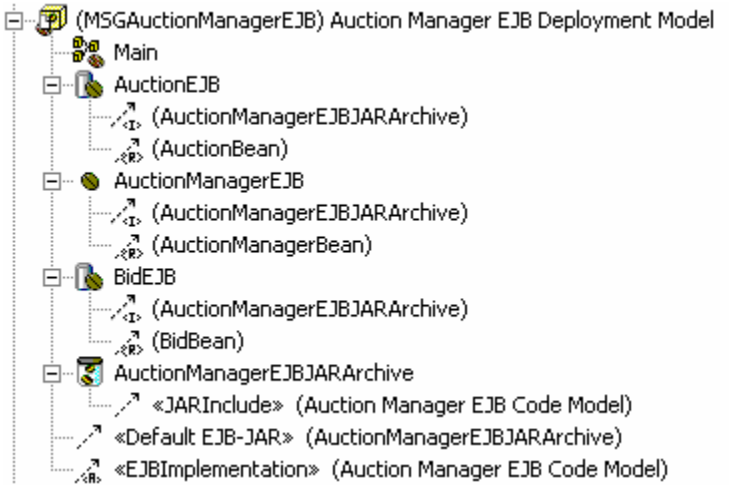

Figura 24: Modelo de Implantação EJB Exemplo

10.3 Modelo de Implantação da Web

O Modelo de Implantação da Web contém componente(s) da Web, o arquivo archive WAR e informações de descritor de implantação da Web. O Modelo de Implantação da Web também pode conter diagramas que mostram

quais elementos de implementação (dos modelos de ida e volta do projeto da Web) estão contidos no WAR. Para ser preciso, no Modelo de Implantação da Web, os componentes da Web são utilizados para representar os elementos de implementação, e são os componentes da Web que são mapeados para o WAR.

Um exemplo de um Modelo de Implantação da Web está mostrado na Figura 25.

Figura 25: Modelo de Implantação da Web

Rational®
the software development company

Duas Sedes:
Rational Software
18880 Homestead Road
Cupertino, CA 95014
Tel: (408) 863-9900

Rational Software
20 Maguire Road
Lexington, MA 02421
Tel: (781) 676-2400

Sem custo: (800) 728-1212
E-mail: info@rational.com
Web: www.rational.com
Localização Internacional:www.rational.com/worldwide