

INSTITUTO DE COMPUTAÇÃO  
UNIVERSIDADE ESTADUAL DE CAMPINAS

**Requisitos para um Ambiente de  
Desenvolvimento Baseado em Componentes e  
Centrado na Arquitetura de Software**

*Rodrigo Teruo Tomita  
Cecília Mary Fischer Rubira*

Technical Report - IC-05-23 - Relatório Técnico

September - 2005 - Setembro

The contents of this report are the sole responsibility of the authors.  
O conteúdo do presente relatório é de única responsabilidade dos autores.



# Requisitos para um Ambiente de Desenvolvimento Baseado em Componentes e Centrado na Arquitetura de Software

Rodrigo Teruo Tomita\*

Cecília Mary Fischer Rubira†

## Resumo

A construção de software através da integração planejada de componentes reutilizáveis, conhecido como desenvolvimento baseado em componentes (DBC), tem conquistado ampla aceitação para o desenvolvimento de sistemas de informação grandes e complexos. A abordagem de arquitetura de software é complementar ao paradigma DBC, com a responsabilidade pela integração dos componentes de forma a se obter as qualidades desejadas para o sistema final. Por isso, os principais processos de DBC são também centrados na arquitetura do software. Os ambientes de desenvolvimento existentes apóiam, em geral, a modelagem UML e a implementação de sistemas orientados a objetos e implementação de componentes. Entretanto, eles não integram a modelagem de arquiteturas de software e DBC. Nesse relatório técnico são descritos os requisitos para ambiente Bellatrix, que é uma extensão do Eclipse para DBC, centrado no processo UML Components e baseado na arquitetura do software. Tais requisitos são descritos em termos de casos de uso e protótipos de interface com usuário.

---

\*Apoio AUTBANK Consultoria e Projetos Ltda.

†Apoio parcial CNPq/Brasil, processo no. 351592/97-0



## Sumário

<b>1</b>	<b>Introdução</b>	<b>7</b>
<b>2</b>	<b>Casos de Uso do Ambiente</b>	<b>9</b>
2.1	Visão Geral . . . . .	9
2.2	Especificar Interface (Pacote 1) . . . . .	10
2.2.1	Criar interface (UC1.1) . . . . .	10
2.2.2	Excluir interface (UC1.2) . . . . .	11
2.2.3	Editar informações de uma interface (UC1.3) . . . . .	11
2.2.4	Criar operação numa interface (UC1.4) . . . . .	12
2.2.5	Excluir operação de uma interface (UC1.5) . . . . .	12
2.2.6	Editar informações de uma operação (UC1.6) . . . . .	13
2.2.7	Copiar ou mover operação de uma interface para outra (UC1.7) . . . . .	13
2.2.8	Definir relação de generalização e especialização entre interfaces (UC1.8)	13
2.3	Especificar Componente (Pacote 2) . . . . .	14
2.3.1	Criar componente (UC2.1) . . . . .	14
2.3.2	Excluir componente (UC2.2) . . . . .	15
2.3.3	Editar informações de um componente (UC2.3) . . . . .	15
2.3.4	Associar interface provida ou requerida ao componente (UC2.4) . . . . .	15
2.3.5	Desassociar interface do componente (UC2.5) . . . . .	16
2.3.6	Criar porta num componente (UC2.6) . . . . .	16
2.3.7	Remover porta de um componente (UC2.7) . . . . .	16
2.3.8	Associar interface provida ou requerida a uma porta (UC2.8) . . . . .	17
2.3.9	Desassociar interface de uma porta (UC2.9) . . . . .	17
2.3.10	Mover interface (UC2.10) . . . . .	17
2.3.11	Editar informações da associação interface / componente (UC2.11) . . . . .	18
2.3.12	Definir como conector (UC2.12) . . . . .	18
2.3.13	Remover definição de conector (UC2.13) . . . . .	18
2.4	Especificar Arquitetura (Pacote 3) . . . . .	18
2.4.1	Criar arquitetura (UC3.1) . . . . .	18
2.4.2	Excluir arquitetura (UC3.2) . . . . .	19
2.4.3	Editar informações de uma arquitetura (UC3.3) . . . . .	20
2.4.4	Inserir componente ou conector a uma arquitetura (UC3.4) . . . . .	20
2.4.5	Remover componente ou conector de uma arquitetura (UC3.5) . . . . .	20
2.4.6	Conectar a interface requerida de um elemento à interface provida de outro (UC3.6) . . . . .	21
2.4.7	Conectar porta de um elemento à porta de outro (UC3.7) . . . . .	21
2.4.8	Excluir conexão entre interfaces (UC3.8) . . . . .	22
2.4.9	Mapear interface interna a uma interface externa (UC3.9) . . . . .	22

2.4.10	Exportar arquitetura para xACME (UC3.10)	23
2.5	Implementar Componente (Pacote 4)	23
2.5.1	Criar implementação de componente (UC4.1)	24
2.5.2	Excluir implementação de componente (UC4.2)	24
2.5.3	Editar informações da implementação de componente (UC4.3)	25
2.5.4	Associar implementação a um componente (UC4.4)	25
2.5.5	Gerar esqueleto de código do componente (UC4.5)	25
2.5.6	Codificar componente (UC4.6)	26
2.6	Testar implementação do componente (Pacote 5)	26
2.6.1	Criar caso de teste (UC5.1)	27
2.6.2	Editar caso de teste (UC5.2)	27
2.6.3	Excluir caso de teste (UC5.3)	27
2.6.4	Executar caso de teste (UC5.4)	28
2.6.5	Criar conjunto de testes (UC5.5)	28
2.6.6	Editar conjunto de testes (UC5.6)	28
2.6.7	Excluir conjunto de testes (UC5.7)	29
2.6.8	Executar conjunto de testes (UC5.8)	29
2.7	Usar assistentes do processo (Pacote 7)	29
2.7.1	Criar interfaces de sistema (UC7.1)	29
2.7.2	Criar interfaces de negócio (UC7.2)	30
2.7.3	Criar componentes de negócio (UC7.3)	31
2.7.4	Associar interfaces de sistema a componentes (UC7.4)	31
2.7.5	Criar componente de sistema (UC7.5)	32
2.7.6	Criar dependência entre um componente e uma interface provida (UC7.6)	32
2.8	Usar repositório de componentes (Pacote 8)	32
2.8.1	Depositar um artefato novo (UC8.1)	33
2.8.2	Depositar uma nova versão de um artefato (UC8.2)	33
2.8.3	Pesquisar por artefato (UC8.3)	34
2.8.4	Recuperar artefato (UC8.4)	34
2.8.5	Remover artefato (UC8.5)	34
2.8.6	Editar meta-dados de um artefato (UC8.6)	35
2.8.7	Recuperar histórico do artefato (UC8.7)	35
2.8.8	Obter estatísticas de uso dos artefatos (UC8.8)	36
2.9	Incluir repositório (UC8.9)	36
2.10	Remover repositório (UC8.10)	36
2.11	Editar informações do repositório (UC8.11)	37
<b>3</b>	<b>Protótipos de Interface com Usuário</b>	<b>39</b>
3.1	Especificação de interfaces	39
3.2	Especificação de componentes	55
3.3	Especificação de arquiteturas	77
3.4	Implementação de componentes	99
3.5	Assistentes do processo	109

<i>Requisitos para um Ambiente DBC centrado na Arquitetura</i>	5
3.6 Uso do repositório de componentes . . . . .	126
<b>A Mapeamento das Atividades do Processo UML Components</b>	<b>137</b>
A.1 Processo UML Components . . . . .	137
A.2 Mapeamento das Atividades . . . . .	138



## 1 Introdução

O desenvolvimento de software baseado em componentes [8] (DBC) e arquitetura de software [2] são abordagens complementares para auxiliar desenvolvedores a lidar com a crescente complexidade dos sistemas de software. O DBC visa a redução dos custos e prazos de desenvolvimento através da montagem de novos sistemas integrando componentes reutilizáveis já existentes. Componentes de software reutilizáveis são, em geral, desenvolvidos sem o conhecimento do contexto específico de todos os sistemas onde são empregados. Cabe, assim, à arquitetura do software a responsabilidade pela integração desses componentes de forma a se obter as qualidades desejadas para o sistema em desenvolvimento. Essa complementariedade se evidencia nos principais processos de DBC, tais como Kobra [1], UML Components [3] e Catalysis [6], que são também centrados na arquitetura do software.

A eficiência de um processo de desenvolvimento de software depende da existência de ferramentas de software adequadas. Nesse contexto, foi proposto o ambiente BELLATRIX [5, 9], uma extensão do ambiente Eclipse [4] para DBC, centrado no processo UML Components [3] e baseado na arquitetura do software. Um dos principais objetivos do BELLATRIX é facilitar a transição de um projeto de arquitetura de software para a implementação do sistema, de acordo com o paradigma DBC.

Com o intuito de se produzir uma especificação completa do ambiente BELLATRIX, foram realizados trabalhos para elicitación de seus requisitos, ou seja, as atividades às quais ele deveria dar suporte. A proposta inicial do ambiente [5] lista as seguintes atividades como requisitos básicos num ambiente DBC:

1. Modelagem de arquiteturas de software em dois níveis de abstração: linhas de produtos e sistemas específicos;
2. Especificação de interfaces, componentes e conectores;
3. Implementação de componentes e conectores, com a geração automática de código a partir do projeto arquitetural;
4. Testes unitários de componentes e conectores;
5. Criação, manutenção, versionamento e busca dos artefatos produzidos, utilizando repositórios de componentes;
6. Integração de sistemas, a partir de componentes e conectores armazenados nos repositórios de componentes;
7. Testes de integração de configurações de componentes e conectores.

Ainda, outros trabalhos [7] citam como principais requisitos para um ambiente de engenharia de software baseado em componentes:

1. Integração das ferramentas que compõem o ambiente, incluindo o uso de uma plataforma comum, compartilhamento de dados, mesma apresentação ao usuário, controle integrado das ferramentas e existência de processo guiando o uso das mesmas;

2. Apoio a atividades de engenharia de software baseada em componentes, incluindo o desenvolvimento de componentes “para reuso” e desenvolvimento de aplicações “com reuso”;
3. Reusabilidade, ou seja, o ambiente deve promover o reuso de artefatos e evitar duplicação de esforços;
4. Integridade referencial, garantindo que os artefatos produzidos sejam consistentes entre si;
5. Gestão de configuração de software, oferecendo controle sobre as mudanças e versões dos artefatos produzidos;
6. Múltiplas visões da informação;
7. Segurança de acesso;
8. Independência de tecnologia e linguagem de programação.

O ambiente BELLATRIX é centrado no processo UML Components. Assim, visando oferecer um apoio adequado a esse processo, foi feito um mapeamento detalhado das atividades do mesmo, com base em sua bibliografia de referência [3]. Nesse mapeamento, o processo foi modelado num conjunto de diagramas de atividades UML, que também foram usados como base na extração de requisitos do ambiente.

Neste trabalho detalhamos os requisitos funcionais do BELLATRIX, apresentando os casos de uso do ambiente e os protótipos de interface com o usuário que retratam esses casos de uso. O documento está organizado da seguinte maneira: na Seção 2 apresentamos os casos de uso do ambiente. Na Seção 3 mostramos os protótipos de interface com o usuário do ambiente. Finalmente, o Apêndice A contém os diagramas de atividades do mapeamento do processo UML Components.

## 2 Casos de Uso do Ambiente

Nessa seção apresentamos os casos de uso especificados para o ambiente BELLATRIX. Neste documento chamamos de *área de trabalho* o local onde as informações sobre os sistemas modelados pelo usuário são armazenadas. Esse local não é compartilhado e é diferente do *repositório de componentes*. Ainda, a visão do usuário sobre os itens existentes nesse local podem ser filtradas ou escondidas, não sendo uma visão direta de seu sistema de arquivos.

### 2.1 Visão Geral

Nesta seção apresentamos uma visão geral dos casos de uso do ambiente. Os casos de uso foram organizados em pacotes de acordo com os requisitos iniciais do ambiente [5], facilitando a visualização dos diagramas apresentados e a descrição dos mesmos. A Figura 1 mostra o diagrama de casos de uso.

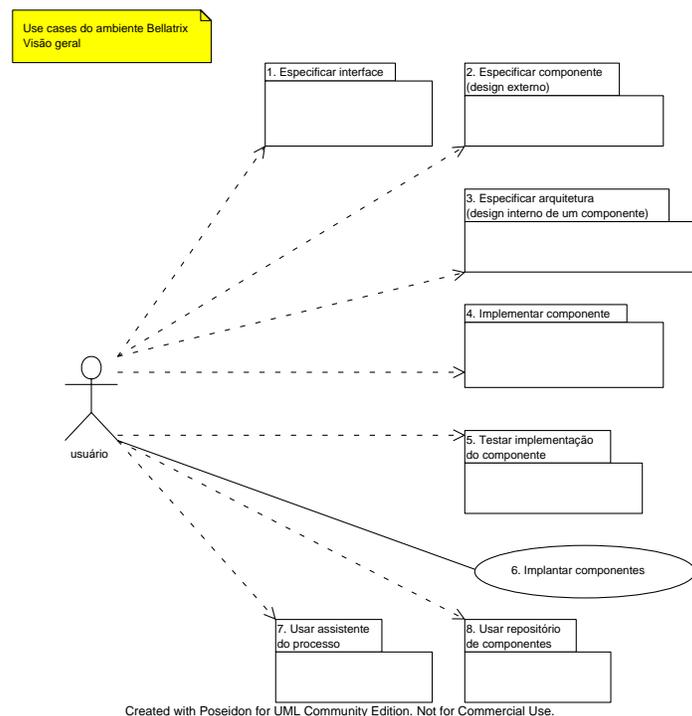


Figura 1: Visão geral dos casos de uso do ambiente.

A seguir, descrevemos os elementos desse diagrama:

1. **Especificar interface** Neste pacote estão agrupados os casos de uso referentes às atividades de especificação de interfaces externas dos componentes de uma arquitetura de software.
2. **Especificar componente** Neste pacote estão os casos de uso referentes às atividades

de especificação de componentes e conectores, mais especificamente, seu projeto “externo”.

3. **Especificar arquitetura** Neste pacote estão os casos de uso referentes às atividades de modelagem da arquitetura de componentes, mais especificamente, o projeto “interno” de um componente.
4. **Implementar componente** Neste pacote estão os casos de uso referentes às atividades de implementação dos componentes especificados no ambiente.
5. **Testar implementação do componente** Neste pacote estão os casos de uso referentes às atividades de especificação de casos teste de um componente.
6. **Implantar componentes** O usuário pode implantar (*deploy*) a implementação dos componentes.
7. **Usar assistente do processo** Neste pacote estão os casos de uso relacionados com o apoio ao processo UML Components.
8. **Usar repositório de componentes** Neste pacote estão os casos de uso básicos relacionados com o uso do repositório a partir do ambiente.

## 2.2 Especificar Interface (Pacote 1)

Nesta seção, apresentamos os casos de uso do pacote (1) Especificar interface. Na Figura 2 apresentamos o diagrama com os casos de uso desse pacote.

A seguir, mostramos a descrição e os passos desses casos de uso:

### 2.2.1 Criar interface (UC1.1)

**Descrição:** O usuário irá criar uma nova especificação de interface, que será aberta para edição.

Passos:

1. O caso de uso se inicia quando o usuário seleciona a opção para criar uma nova especificação de interface.
2. O usuário informa o local em que a especificação de interface deve ser criada, dentro da área de trabalho do ambiente.
3. O usuário informa o nome da interface a ser criada.
4. O sistema cria uma representação da interface em sua área de trabalho no local especificado pelo usuário e a abre para edição. O sistema atualiza a visão do usuário da área de trabalho.

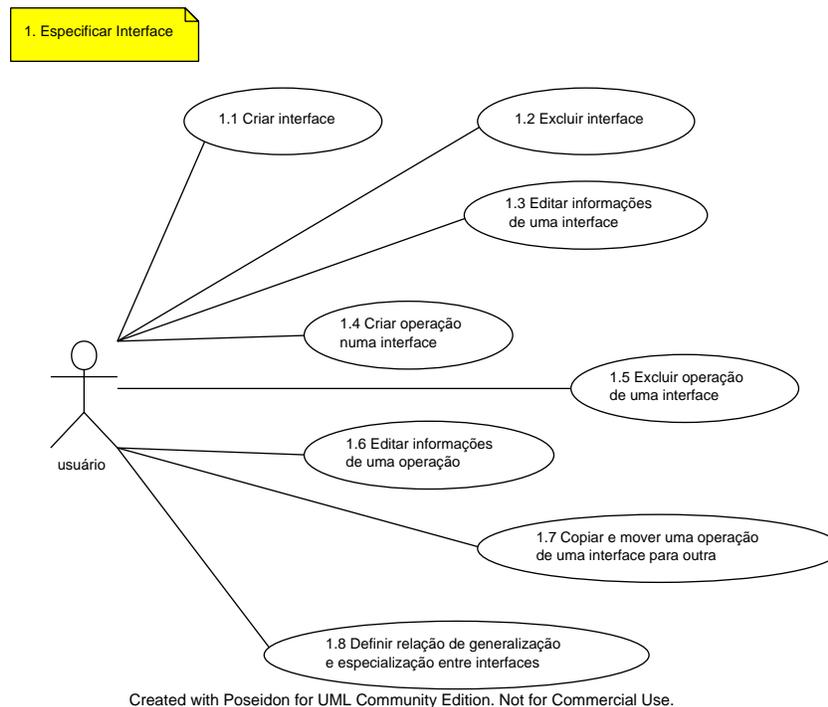


Figura 2: Casos de uso do pacote especificar interface.

### 2.2.2 Excluir interface (UC1.2)

**Descrição:** O usuário irá excluir uma especificação de interface da área de trabalho do ambiente.

Passos:

1. O caso de uso se inicia quando o usuário seleciona a opção para excluir uma especificação de interface.
2. O usuário informa a especificação de interface a excluir da área de trabalho.
3. O usuário confirma a exclusão da interface.
4. Se a especificação da interface estiver aberta para edição, o sistema a fecha.
5. O sistema exclui a especificação de interface da área de trabalho e atualiza a visão do usuário do mesmo.

### 2.2.3 Editar informações de uma interface (UC1.3)

**Descrição:** O usuário irá editar informações de uma especificação de interface. Tais informações podem ser nome, descrição, meta-informações, entre outros.

Passos:

1. A interface deve estar disponível na área de trabalho do ambiente. Caso a interface não esteja aberta para edição:
  - (a) o usuário deve selecionar a especificação da interface em sua área de trabalho e abri-la.
  - (b) o sistema deve abrir a especificação da interface para edição, exibindo-a para o usuário.
2. O usuário seleciona e edita as informações que desejar.
3. O sistema atualiza a representação gráfica da interface com as novas informações. Essas informações devem ser persistidas juntamente com a especificação da interface.

#### 2.2.4 Criar operação numa interface (UC1.4)

**Descrição:** O usuário irá criar uma operação em uma especificação de interface previamente criada e aberta para edição. Uma interface pode ter várias operações.

Passos:

1. O caso de uso se inicia quando o usuário seleciona a opção de “incluir nova operação” na interface na qual ele deseja incluir uma operação.
2. O usuário informa os detalhes da operação: nome, parâmetros e seus tipos, tipo de retorno, exceções, entre outros.
3. O sistema atualiza a representação gráfica da especificação da interface para exibir a nova operação.

#### 2.2.5 Excluir operação de uma interface (UC1.5)

**Descrição:** O usuário irá excluir uma operação pré-existente de uma especificação de interface. Essa especificação de interface deve estar aberta para edição e deve conter pelo menos uma operação.

Passos:

1. O caso de uso se inicia quando o usuário seleciona a operação a ser excluída e seleciona a opção para excluir a operação.
2. O usuário confirma a exclusão da operação.
3. O sistema exclui a operação selecionada após confirmação do usuário e atualiza a representação gráfica da interface.

### 2.2.6 Editar informações de uma operação (UC1.6)

**Descrição:** O usuário irá editar as informações de uma operação. Tais informações podem incluir nome, descrição, assinatura, pré e pós condições, meta-informação, entre outros. A especificação da interface que contém a operação a ser editada deve estar aberta para edição.

Passos:

1. O caso de uso se inicia quando o usuário seleciona uma operação na especificação de interface e edita suas informações.
2. Sistema atualiza a representação gráfica com as informações da operação. As informações devem ser persistidas juntamente com a especificação da interface.

### 2.2.7 Copiar ou mover operação de uma interface para outra (UC1.7)

**Descrição:** O usuário deseja copiar ou mover uma operação existente em uma interface para outra interface aberta para edição.

Passos:

1. O caso de uso se inicia quando o usuário seleciona uma operação na interface de origem e escolhe a opção Copiar (para copiar) ou Recortar (para mover).
2. O usuário seleciona a interface de destino da operação e seleciona a opção Colar.
3. Se a opção do usuário foi Copiar, o sistema copia a operação copiada da primeira para a segunda interface, preservando a operação original.
4. Se a opção do usuário foi Recortar, o sistema deve copiar a operação copiada da primeira para a segunda interface e em seguida excluir a operação original.

### 2.2.8 Definir relação de generalização e especialização entre interfaces (UC1.8)

**Descrição:** O usuário deseja criar relações de generalização e especialização entre especificações de interface criadas anteriormente. A interface a ser definida como sub-tipo deve estar aberta para edição.

Passos:

1. O caso de uso se inicia quando o usuário seleciona uma interface e escolhe a opção de definir supertipo.
2. O usuário informa ao sistema qual o supertipo da interface.
3. O sistema define a interface informada como supertipo da primeira interface e atualiza sua representação gráfica.

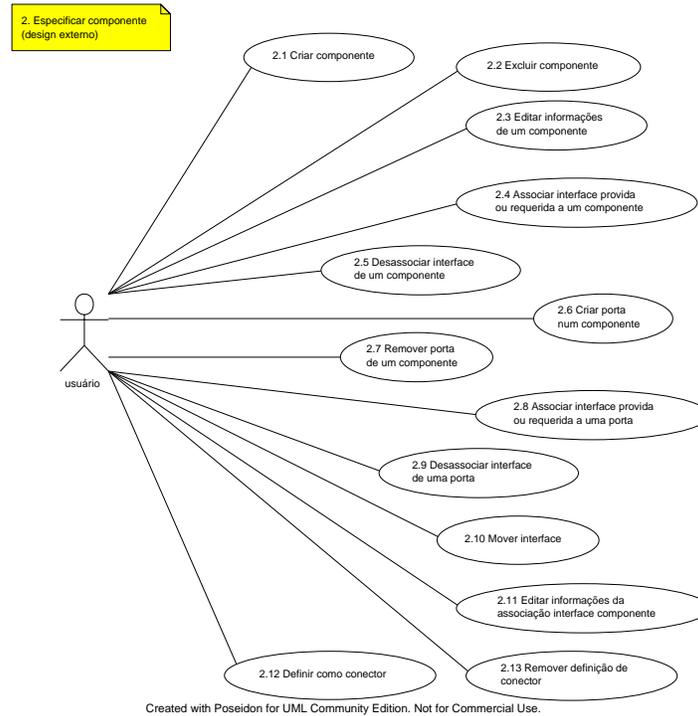


Figura 3: Casos de uso do pacote Especificar componente.

## 2.3 Especificar Componente (Pacote 2)

Nesta seção, apresentamos os casos de uso do pacote (2) Especificar componente. Na Figura 3 apresentamos o diagrama com os casos de uso desse pacote.

A seguir, mostramos a descrição e os passos desses casos de uso:

### 2.3.1 Criar componente (UC2.1)

**Descrição:** O usuário irá criar uma nova especificação de componente que será aberta para edição.

Passos:

1. O caso de uso se inicia quando o usuário seleciona a opção para criar uma nova especificação de componente.
2. O usuário informa o local em que a especificação de componente deve ser criada, dentro da área de trabalho do ambiente.
3. O usuário informa o nome da componente a ser criado.
4. O sistema cria uma representação do componente em sua área de trabalho no local especificado pelo usuário e o abre para edição. O sistema atualiza a visão do usuário da área de trabalho.

### 2.3.2 Excluir componente (UC2.2)

**Descrição:** O usuário irá excluir uma especificação de componente da área de trabalho do ambiente.

Passos:

1. O caso de uso se inicia quando o usuário seleciona a opção para excluir uma especificação de componente.
2. O usuário informa a especificação de componente a excluir da área de trabalho.
3. O usuário confirma a exclusão da interface.
4. Se a especificação do componente estiver aberta para edição, o sistema a fecha.
5. O sistema exclui a especificação de componente da área de trabalho e atualiza a visão do usuário da mesma. As interfaces associadas ao componente não são excluídas.
6. Se existir uma arquitetura associada a esse componente, o sistema exclui também a arquitetura.

### 2.3.3 Editar informações de um componente (UC2.3)

**Descrição:** O usuário editará informações sobre a especificação de um componente. Tais informações podem ser nome, descrição, entre outras.

Passos:

1. O componente a ser editado deve estar disponível na área de trabalho do ambiente e aberto para edição. Caso o componente não esteja aberto:
  - (a) O usuário deve selecionar a especificação de componente em sua área de trabalho e abri-la.
  - (b) O sistema deve abrir a especificação de componente para edição, exibindo-a para o usuário.
2. O usuário seleciona e edita as informações que desejar.
3. O sistema atualiza a representação gráfica do componente com as novas informações. Essas informações devem ser persistidas juntamente com a especificação do componente.

### 2.3.4 Associar interface provida ou requerida ao componente (UC2.4)

**Descrição:** O usuário associará uma especificação de interface já existente à especificação do componente. Essa interface pode ser associada como uma interface provida ou requerida. Um componente pode ter várias interfaces associadas a ele. O componente em questão deve estar aberto para edição.

Passos:

1. O caso de uso se inicia quando o usuário seleciona a opção de associar interface à especificação do componente.
2. O sistema exibe uma lista das interfaces que estão contidas na área de trabalho do ambiente. Devem haver meios para filtrar essa lista.
3. O usuário seleciona a interface e informa se ela é uma interface provida ou requerida do componente.
4. O sistema atualiza a representação gráfica do componente para exibir a interface associada. A especificação do componente deve persistir essa associação.

### 2.3.5 Desassociar interface do componente (UC2.5)

**Descrição:** O usuário irá desfazer uma associação entre a especificação do componente e uma interface, seja ela provida ou requerida. O componente deve estar aberto para edição e deve conter pelo menos uma associação com uma interface.

Passos:

1. O caso de uso se inicia quando o usuário seleciona uma interface associada a um componente e escolhe a opção para desassociar essa interface.
2. O sistema desfaz a associação, atualizando a representação gráfica do componente.

### 2.3.6 Criar porta num componente (UC2.6)

**Descrição:** O usuário irá criar uma porta em uma especificação de componente previamente criada e aberta para edição. Um componente pode ter várias operações.

Passos:

1. O caso de uso se inicia quando o usuário seleciona um componente e escolhe a opção para incluir uma nova porta nesse componente.
2. O sistema atualiza a representação gráfica da especificação da interface para exibir a nova operação.

### 2.3.7 Remover porta de um componente (UC2.7)

**Descrição:** O usuário irá excluir uma porta existente numa especificação de componente. Essa especificação deve estar aberta para edição e deve conter pelo menos uma porta.

Passos:

1. O caso de uso se inicia quando o usuário seleciona a porta a ser excluída e seleciona a opção para excluir a porta.
2. O usuário confirma a exclusão da porta.
3. O sistema exclui a porta selecionada após confirmação do usuário e atualiza a representação gráfica do componente.

### 2.3.8 Associar interface provida ou requerida a uma porta (UC2.8)

**Descrição:** O usuário associará uma especificação de interface já existente à especificação do componente através de uma porta que esse componente possua. Essa interface pode ser associada como uma interface provida ou requerida. Uma porta pode ter várias interfaces associadas a ela. O componente em questão deve estar aberto para edição.

Passos:

1. O caso de uso se inicia quando o usuário seleciona uma porta na especificação do componente e escolhe a opção de associar uma interface à porta.
2. O sistema exibe uma lista das interfaces que estão contidas na área de trabalho do ambiente. Devem haver meios para filtrar essa lista.
3. O usuário seleciona a interface e informa se ela é uma interface provida ou requerida.
4. O sistema atualiza a representação gráfica do componente para exibir a porta com a interface associada. A especificação do componente deve persistir essa associação.

### 2.3.9 Desassociar interface de uma porta (UC2.9)

**Descrição:** O usuário irá desfazer uma associação entre a porta de um componente e uma interface, seja ela provida ou requerida. O componente deve estar aberto para edição e deve conter pelo menos uma porta com uma associação com uma interface.

Passos:

1. O caso de uso se inicia quando o usuário seleciona uma interface associada a um componente e escolhe a opção para desassociar essa interface.
2. O sistema desfaz a associação, atualizando a representação gráfica do componente.

### 2.3.10 Mover interface (UC2.10)

**Descrição:** O usuário pode mover uma interface previamente associada a um componente para uma porta e vice-versa. Ainda, ele pode mover uma interface associada a uma porta para outra porta. A especificação do componente em questão deve estar aberta para edição.

Passos:

1. O caso de uso se inicia quando o usuário seleciona uma interface previamente associada a um componente ou porta e escolhe a opção para mover a interface.
2. O usuário seleciona o componente ou porta para o qual ele deseja que a interface seja movida.
3. O sistema associa a interface selecionada ao novo elemento (componente ou porta), desassociando-a do anterior. O sistema também atualiza a representação gráfica dessas associações.

### **2.3.11 Editar informações da associação interface / componente (UC2.11)**

**Descrição:** O usuário irá editar informações da associação entre uma interface, seja ela provida ou requerida, e um componente ou porta. Tal associação deve existir previamente.

Passos:

1. O caso de uso se inicia quando o usuário seleciona uma associação entre uma interface e um componente ou porta e edita informações sobre a associação.
2. O sistema atualiza a representação gráfica da associação com as novas informações.

### **2.3.12 Definir como conector (UC2.12)**

**Descrição:** O usuário irá definir uma especificação de componente como sendo um conector.

1. O caso de uso se inicia quando o usuário seleciona uma especificação de componente e escolhe a opção para defini-la como conector.
2. O sistema marca essa especificação de componente como sendo agora um conector e atualiza a representação gráfica do componente.

### **2.3.13 Remover definição de conector (UC2.13)**

**Descrição:** O usuário irá definir uma especificação de conector como sendo um componente.

1. O caso de uso se inicia quando o usuário seleciona uma especificação de conector e escolhe a opção para defini-la como componente.
2. O sistema marca essa especificação de componente como sendo agora um componente e atualiza a representação gráfica do conector.

## **2.4 Especificar Arquitetura (Pacote 3)**

Nesta seção, apresentamos os casos de uso do pacote (3) Especificar arquitetura. Na Figura 4 apresentamos o diagrama com os casos de uso desse pacote.

A seguir, mostramos a descrição e os passos desses casos de uso:

### **2.4.1 Criar arquitetura (UC3.1)**

**Descrição:** O usuário criará uma nova arquitetura, contendo suas informações básicas, que será aberta para edição. Essa arquitetura estará relacionada a um componente (será a decomposição de um componente).

Passos:

1. O caso de uso se inicia quando o usuário seleciona um componente e escolhe a opção para criar uma nova arquitetura para aquele componente.



Figura 4: Casos de uso do pacote Especificar arquitetura.

2. O usuário informa o local em que a arquitetura deve ser criada, dentro da área de trabalho do ambiente.
3. O usuário informa o nome e a descrição (opcional) da arquitetura a ser criada.
4. O sistema cria uma representação da arquitetura em sua área de trabalho no local especificado pelo usuário, associa essa arquitetura ao componente selecionado e a abre para edição.

#### 2.4.2 Excluir arquitetura (UC3.2)

**Descrição:** O usuário excluirá uma arquitetura previamente criada de sua área de trabalho. O componente que contém a arquitetura não é excluído.

Passos:

1. O caso de uso se inicia quando o usuário seleciona a opção para excluir uma arquitetura.
2. O usuário informa a especificação de interface a excluir da área de trabalho.
3. O usuário confirma que deseja excluir a arquitetura.
4. Se a arquitetura está aberta para edição, o sistema fecha a arquitetura.

5. O sistema exclui a arquitetura da área de trabalho e atualiza a visão do usuário do mesmo. O sistema não exclui os componentes e conectores incluídos na arquitetura.

#### **2.4.3 Editar informações de uma arquitetura (UC3.3)**

**Descrição:** O usuário edita as informações de uma arquitetura pré-existente.

Passos:

1. A arquitetura a ser editada deve estar disponível na área de trabalho do ambiente e aberta para edição. Caso a arquitetura não esteja aberta:
  - (a) O usuário deve selecionar a arquitetura em sua área de trabalho e abri-la.
  - (b) O sistema deve abrir a arquitetura para edição, exibindo-a para o usuário.
2. O usuário seleciona e edita as informações que desejar.
3. O sistema atualiza a representação gráfica da arquitetura com as novas informações. Essas informações devem ser persistidas juntamente com a arquitetura.

#### **2.4.4 Inserir componente ou conector a uma arquitetura (UC3.4)**

**Descrição:** O usuário escolhe uma especificação de componente ou conector existente na área de trabalho e o adiciona a uma arquitetura. Essa arquitetura deve estar aberta para edição. É possível incluir quantos componentes ou conectores se desejar em uma arquitetura. Também é possível incluir uma mesma especificação mais de uma vez na mesma arquitetura.

Passos:

1. O caso de uso se inicia quando o usuário seleciona a opção para incluir uma especificação de componente ou conector.
2. O sistema apresenta uma lista contendo os componentes e conectores existentes na área de trabalho. Essa lista deve ser filtrável.
3. O usuário escolhe o componente ou conector que deseja inserir na arquitetura e confirma a inserção.
4. O sistema insere o componente ou conector na arquitetura e atualiza a representação gráfica da arquitetura, mostrando o elemento inserido, seu nome e interfaces.

#### **2.4.5 Remover componente ou conector de uma arquitetura (UC3.5)**

**Descrição:** O usuário retira uma especificação de componente ou conector de uma arquitetura. A arquitetura deve estar aberta para edição e possuir pelo menos uma especificação de componente ou conector.

Passos:

1. O caso de uso se inicia quando o usuário seleciona a especificação de componente ou conector que deseja remover da arquitetura e escolhe a opção para removê-la.
2. O usuário confirma que deseja remover o componente ou conector da arquitetura.
3. O sistema retira o componente ou conector da arquitetura, bem como suas interfaces e todas as suas conexões. O componente e suas interfaces não são excluídos da área de trabalho, apenas desassociados da arquitetura.

#### **2.4.6 Conectar a interface requerida de um elemento à interface provida de outro (UC3.6)**

**Descrição:** O usuário conecta uma interface requerida de um elemento à interface provida de outro elemento. Ambos os elementos devem estar na mesma arquitetura. A arquitetura deve estar aberta para edição. As interfaces não devem pertencer ao mesmo elemento. Pelo menos uma das interfaces deve estar associada a um componente ou a uma porta ligada a um componente.

Passos:

1. O caso de uso se inicia quando o usuário seleciona a opção para conectar interfaces.
2. O usuário seleciona uma interface de um componente ou conector na arquitetura.
3. O usuário seleciona outra interface de um outro componente ou conector na arquitetura.
4. O sistema verifica se as interfaces são compatíveis (iguais ou subtipo).
5. Se uma das interfaces selecionadas está associada a um conector e a outra está associada a um componente, o sistema cria uma conexão entre essas interfaces.
6. Se ambas as interfaces selecionadas estão associadas a componentes, o sistema cria um conector que provê/requer interfaces do mesmo tipo das interfaces selecionadas (para cada interface provida é criada uma interface requerida no conector e vice-versa) e conecta essas interfaces às interfaces dos componentes. Esse conector é criado com um nome padrão.
7. O sistema atualiza a representação gráfica da arquitetura para exibir as novas conexões.

#### **2.4.7 Conectar porta de um elemento à porta de outro (UC3.7)**

**Descrição:** O usuário conecta a porta de um elemento à porta de outro elemento. Ambos os elementos devem estar na mesma arquitetura. A arquitetura deve estar aberta para edição. Pelo menos uma das portas deve estar associada a um componente.

1. O caso de uso se inicia quando o usuário seleciona a opção para conectar portas.

2. O usuário seleciona uma porta de um componente ou conector na arquitetura.
3. O usuário seleciona outra porta de um outro componente ou conector na arquitetura.
4. O sistema verifica se as portas são compatíveis (se para cada interface provida de uma porta existe uma interface requerida compatível na outra porta e vice-versa).
5. Se uma das portas selecionadas está associada a um conector e a outra está associada a um componente, o sistema cria uma conexão entre as interfaces compatíveis dessas portas.
6. Se ambas as portas selecionadas estão associadas a componentes, o sistema cria um conector que contém portas compatíveis com as portas selecionadas e conecta as interfaces dessas portas às portas dos componentes. Esse conector é criado com um nome padrão.
7. O sistema atualiza a representação gráfica da arquitetura para exibir as novas conexões.

#### 2.4.8 Excluir conexão entre interfaces (UC3.8)

**Descrição:** O usuário exclui conexões entre interfaces de componentes e conectores.

1. O caso de uso se inicia quando o usuário seleciona uma conexão entre interfaces e escolhe a opção para excluir conexão entre interfaces.
2. O usuário confirma que deseja excluir a conexão.
3. O sistema exclui a conexão entre as interfaces e atualiza a representação gráfica das interfaces.

#### 2.4.9 Mapear interface interna a uma interface externa (UC3.9)

**Descrição:** O usuário mapeia uma interface associada a um componente da arquitetura a uma interface “externa”, ou seja, uma interface que está associada ao componente que contém a arquitetura. A arquitetura deve estar aberta para edição. A interface a ser mapeada deve ser compatível com uma interface presente na especificação do componente que contém a arquitetura, tanto com respeito ao seu tipo quanto com respeito a ser provida ou requerida. A interface a ser mapeada não pode estar associada a outra interface dentro da arquitetura.

1. O caso de uso se inicia quando o usuário seleciona a interface “interna” a ser mapeada e escolhe a opção para mapear a interface numa interface externa.
2. O sistema exibe as interfaces “externas” da arquitetura (estas são as interfaces providas e requeridas do componente que contém a arquitetura).
3. O usuário seleciona uma interface “externa”.

4. O sistema cria uma conexão entre as duas interfaces, exibindo o mapeamento entre elas ao usuário.

#### 2.4.10 Exportar arquitetura para xACME (UC3.10)

**Descrição:** O usuário exporta uma arquitetura existente para a ADL xAcme.

1. O caso de uso se inicia quando o usuário seleciona uma arquitetura e escolhe a opção para exportar para xAcme.
2. O usuário informa o local em que o arquivo xAcme deve ser criado.
3. O usuário informa o nome e da arquitetura xAcme a ser criada.
4. O sistema cria uma representação da arquitetura no formato da ADL xAcme no local especificado pelo usuário.

### 2.5 Implementar Componente (Pacote 4)

Nesta seção, apresentamos os casos de uso do pacote (4) Implementar componente. Na Figura 5 apresentamos o diagrama com os casos de uso desse pacote.

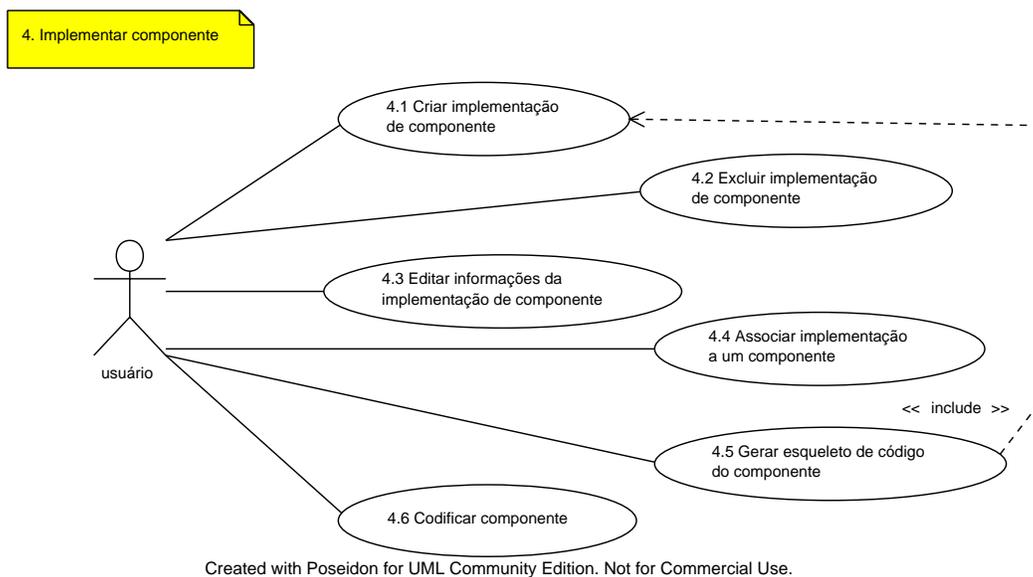


Figura 5: Casos de uso do pacote Implementar componente.

A seguir, mostramos a descrição e os passos desses casos de uso:

### 2.5.1 Criar implementação de componente (UC4.1)

**Descrição:** O usuário cria uma nova representação de implementação de componente, associada a uma especificação de componente.

Passos:

1. O caso de uso se inicia quando o usuário seleciona uma especificação de componente e escolhe a opção para criar uma nova implementação daquele componente.
2. O usuário informa o local em que a representação da implementação de componente deve ser criada, dentro da área de trabalho do ambiente.
3. O usuário informa o nome e a descrição da representação de implementação de componente a ser criada. Ainda, o usuário pode, opcionalmente, informar o local físico da implementação real do componente (fontes e binários).
4. O sistema cria uma representação da implementação de componente em sua área de trabalho no local especificado pelo usuário, associa essa representação de implementação à especificação do componente.
5. Se essa implementação do componente for criada no contexto de uma arquitetura, o sistema associa essa implementação com a arquitetura em questão.
6. O sistema atualiza a visão do usuário.

### 2.5.2 Excluir implementação de componente (UC4.2)

**Descrição:** O usuário exclui uma representação de implementação de componente previamente criada de sua área de trabalho.

Passos:

1. O caso de uso se inicia quando o usuário seleciona a opção para excluir uma implementação de componente.
2. O usuário informa a representação de implementação de componente a excluir da área de trabalho.
3. O usuário informa se deseja ou não excluir os códigos-fonte e/ou binários associados a essa implementação.
4. O usuário confirma que deseja excluir a implementação.
5. O sistema exclui a implementação da área de trabalho e atualiza a visão do usuário do mesmo. O sistema não exclui a especificação do componente.

### 2.5.3 Editar informações da implementação de componente (UC4.3)

**Descrição:** O usuário edita as informações de uma representação de implementação de componente pré-existente.

Passos:

1. A implementação a ser editada deve estar disponível na área de trabalho do ambiente e aberta para edição. Caso a implementação não esteja aberta:
  - (a) O usuário deve selecionar a implementação em sua área de trabalho e abri-la.
  - (b) O sistema deve abrir a implementação para edição, exibindo-a para o usuário.
2. O usuário seleciona e edita as informações que desejar.
3. O sistema atualiza a visão do usuário da implementação com as novas informações.

### 2.5.4 Associar implementação a um componente (UC4.4)

**Descrição:** O usuário associa uma representação de implementação de componente pré-existente à especificação de um componente também já existente no contexto de uma arquitetura. A representação de implementação do componente, a especificação de componente e a implementação do componente ao qual arquitetura está associada devem estar disponíveis na área de trabalho.

Passos:

1. O caso de uso de inicia quando o usuário seleciona especificação de componente dentro de uma arquitetura.
2. O usuário seleciona a implementação que deve estar associada ao componente.
3. O sistema verifica se a especificação e a implementação são compatíveis e registra a associação na implementação do componente que contém a arquitetura.

### 2.5.5 Gerar esqueleto de código do componente (UC4.5)

**Descrição:** O usuário cria esqueletos de código de um componente a partir de sua especificação.

Passos:

1. O caso de uso de inicia quando o usuário seleciona uma especificação de componente e escolhe a opção para gerar esqueleto de código.
2. O sistema cria uma representação da implementação do componente (Inclui *Criar implementação do componente (UC4.1)*).
3. O sistema cria esqueletos de código seguindo a especificação do componente e o modelo de componentes utilizado no local informado pelo usuário, dentro de sua área de trabalho.

### 2.5.6 Codificar componente (UC4.6)

**Descrição:** O usuário codifica um componente. Esse componente pode ser um esqueleto de código gerado pelo ambiente, um componente pré-existente em sua área de trabalho ou mesmo um componente recuperado do repositório de componentes.

Passos:

1. O caso de uso se inicia quando o usuário abre a implementação do componente para edição.
2. O usuário edita a implementação do componente, criando ou editando arquivos, dependendo do modelo de componentes utilizado.
3. O sistema provê o apoio à implementação, na forma de ferramentas de verificação sintática, editores, compiladores e depuradores.
4. O sistema grava a implementação do componente.

### 2.6 Testar implementação do componente (Pacote 5)

Nesta seção, apresentamos os casos de uso do pacote (5) Testar implementação do componente. Na Figura 6 apresentamos o diagrama com os casos de uso desse pacote.

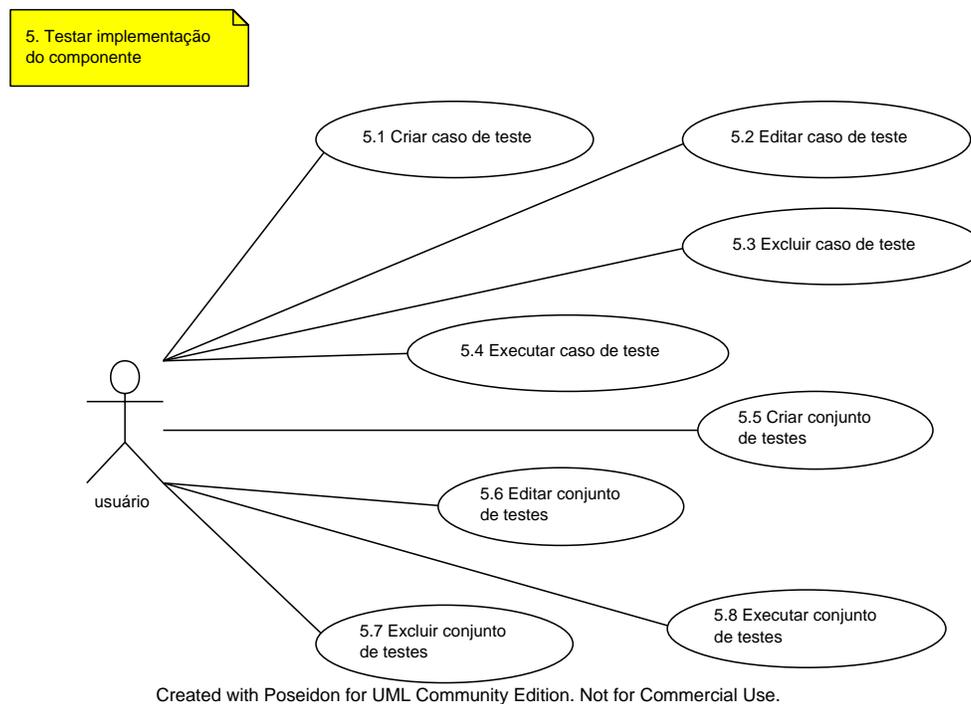


Figura 6: Casos de uso do pacote Testar implementação do componente.

A seguir, mostramos a descrição e os passos desses casos de uso:

### 2.6.1 Criar caso de teste (UC5.1)

**Descrição:** O usuário cria um novo caso de teste unitário para uma especificação de componente.

Passos:

1. O caso de uso se inicia quando o usuário seleciona uma especificação de componente e escolhe a opção para criar um novo caso de teste para aquele componente.
2. O usuário informa o local em que o caso de teste do componente deve ser criado, dentro da área de trabalho do ambiente.
3. O usuário informa o nome e a descrição do caso de teste.
4. O sistema cria o caso de teste em sua área de trabalho no local especificado pelo usuário, registrando a especificação do componente a qual esse caso de teste se refere.
5. O sistema atualiza a visão do usuário da área de trabalho.

### 2.6.2 Editar caso de teste (UC5.2)

**Descrição:** O usuário edita um caso de teste pré-existente.

Passos:

1. O caso de uso se inicia quando o usuário seleciona um caso de teste e escolhe a opção para editar esse caso de teste.
2. O usuário informa quais métodos das interfaces providas do componente serão exercitadas, em que ordem e com quais valores de parâmetros.
3. O usuário informa quais são os valores de retorno dos métodos das interfaces requeridas do componente.
4. O sistema salva o caso de teste, registrando os valores informados.

### 2.6.3 Excluir caso de teste (UC5.3)

**Descrição:** O usuário exclui um caso de teste unitário de um componente. Esse caso de teste deve existir em sua área de trabalho.

Passos:

1. O caso de uso se inicia quando o usuário seleciona um caso de teste em sua área de trabalho e escolhe a opção para excluir o caso de teste.
2. O usuário confirma que deseja excluir o caso de teste.
3. Sistema exclui o caso de teste e atualiza a visão do usuário da área de trabalho.

#### 2.6.4 Executar caso de teste (UC5.4)

**Descrição:** O usuário executa um caso de teste unitário de um componente. Esse caso de teste deve existir em sua área de trabalho.

Passos:

1. O caso de uso se inicia quando o usuário seleciona um caso de teste em sua área de trabalho e escolhe a opção para executar o caso de teste.
2. O sistema executa o caso de teste e exibe os resultados para o usuário.

#### 2.6.5 Criar conjunto de testes (UC5.5)

**Descrição:** O usuário cria um novo conjunto de testes. Deve haver pelo menos um caso de teste de componente criado em sua área de trabalho.

Passos:

1. O caso de uso se inicia quando o usuário escolhe a opção para criar um novo conjunto de testes.
2. O usuário informa o local em que o conjunto de testes de componente deve ser criado, dentro da área de trabalho do ambiente.
3. O usuário informa o nome e a descrição do conjunto de testes.
4. O sistema exibe uma lista com os casos de teste disponíveis na área de trabalho.
5. O usuário informa os casos de teste que irão compor esse conjunto de testes.
6. O sistema cria o conjunto de testes em sua área de trabalho no local especificado pelo usuário, registrando os casos de teste que o compõem.
7. O sistema atualiza a visão do usuário da área de trabalho.

#### 2.6.6 Editar conjunto de testes (UC5.6)

**Descrição:** O usuário edita as informações de um conjunto de testes criado anteriormente. O conjunto de testes a ser editado deve estar disponível na área de trabalho do ambiente e aberto para edição.

Passos:

1. O usuário seleciona e edita as informações que desejar, que podem ser o nome e descrição do conjunto de testes, ou ainda quais casos de teste compõem esse conjunto.
2. O sistema registra as alterações do usuário e atualiza a visão do usuário com as novas informações.

### 2.6.7 Excluir conjunto de testes (UC5.7)

**Descrição:** O usuário exclui um conjunto de testes unitários de um componente. Esse conjunto de testes deve existir em sua área de trabalho.

Passos:

1. O caso de uso se inicia quando o usuário seleciona um conjunto de testes em sua área de trabalho e escolhe a opção para excluir o conjunto de testes.
2. O usuário confirma que deseja excluir o conjunto de testes.
3. Sistema exclui o conjunto de testes e atualiza a visão do usuário da área de trabalho. Os casos de teste que compõem o conjunto de testes não são excluídos.

### 2.6.8 Executar conjunto de testes (UC5.8)

**Descrição:** O usuário executa um conjunto de testes unitários. Esse conjunto de testes deve existir em sua área de trabalho.

Passos:

1. O caso de uso se inicia quando o usuário seleciona um conjunto de testes em sua área de trabalho e escolhe a opção para executar o conjunto de testes.
2. O sistema executa cada caso de teste que compõe o conjunto de testes e exibe os resultados para o usuário.

## 2.7 Usar assistentes do processo (Pacote 7)

Nesta seção, apresentamos os casos de uso do pacote (7) Usar assistentes do processo, onde estão os casos de uso relacionados com o apoio ao processo UML Components. Na Figura 7 apresentamos o diagrama com os casos de uso desse pacote.

A seguir, mostramos a descrição e os passos desses casos de uso:

### 2.7.1 Criar interfaces de sistema (UC7.1)

**Descrição:** O usuário importa os casos de uso criados em uma ferramenta UML, criando uma interface de sistema para cada caso de uso. A ferramenta UML deve exportar seus diagramas em formato XMI.

1. O caso de uso se inicia quando o usuário seleciona a opção para importar casos de uso e criar interfaces de sistema.
2. O usuário informa ao sistema o local do arquivo XMI a ser importado.
3. O sistema lê o arquivo XMI e apresenta ao usuário uma lista com os casos de uso encontrados nesse arquivo e uma lista de interfaces de sistema a serem criadas.

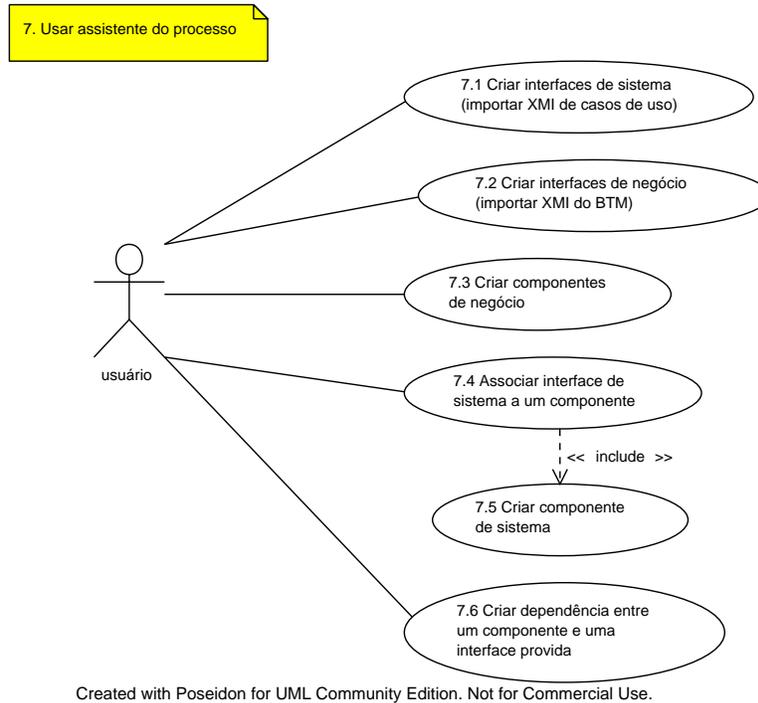


Figura 7: Casos de uso do pacote Usar assistentes do processo.

4. O usuário pode alterar os nomes sugeridos pelo sistema para as interfaces a serem criadas.
5. O usuário informa ao sistema o local na área de trabalho a serem criadas as interfaces.
6. Após a confirmação do usuário, o sistema cria as especificações de interface na área de trabalho e atualiza a visão do usuário do mesmo.

### 2.7.2 Criar interfaces de negócio (UC7.2)

**Descrição:** O usuário importa o *Business Type Model* criado em uma ferramenta UML, criando uma interface de negócio para cada *core type*. O *business type model* deve ser um diagrama de classes em formato XMI e as *core types* devem utilizar o estereótipo *core*, de acordo com o UML Components.

1. O caso de uso se inicia quando o usuário seleciona a opção para importar o *business type model* e criar interfaces de negócio.
2. O usuário informa ao sistema o local do arquivo XMI a ser importado.
3. O sistema lê o arquivo XMI e apresenta ao usuário uma lista com os *core types* encontrados nesse arquivo e uma lista de interfaces de negócio a serem criadas.

4. O usuário pode alterar os nomes sugeridos pelo sistema para as interfaces a serem criadas.
5. O usuário informa ao sistema o local na área de trabalho a serem criadas as interfaces.
6. Após a confirmação do usuário, o sistema cria as especificações de interface na área de trabalho e atualiza a visão do usuário do mesmo.

### 2.7.3 Criar componentes de negócio (UC7.3)

**Descrição:** O usuário cria uma especificação de componente de negócio para cada interface de negócio.

1. O caso de uso se inicia quando o usuário seleciona a opção para criar componentes de negócio.
2. O sistema apresenta uma lista de interfaces presentes na área de trabalho. Essa lista deve ser filtrável.
3. O usuário seleciona as interfaces para as quais ele deseja criar componentes e confirma.
4. O usuário pode alterar os nomes sugeridos pelo sistema para os componentes a serem criados.
5. O usuário informa ao sistema o local na área de trabalho a serem criados os componentes.
6. Para cada interface selecionada, o sistema cria uma especificação de componente na área de trabalho, seguindo o nome dado pelo usuário. Cada componente provê a interface selecionada.

### 2.7.4 Associar interfaces de sistema a componentes (UC7.4)

**Descrição:** O usuário escolhe suas interfaces de sistema e as associa a especificações de componente (existentes ou não).

1. O caso de uso se inicia quando o usuário seleciona a opção para associar interfaces de sistema a componentes.
2. O sistema exibe uma lista de interfaces disponíveis na área de trabalho. Essa lista deve ser filtrável.
3. O usuário seleciona as interfaces de sistema que ele deseja associar. O usuário deve selecionar pelo menos uma interface.
4. O sistema exibe uma lista de componentes disponíveis na área de trabalho. Essa lista deve ser filtrável.
5. O usuário seleciona os componentes de sistema que ele deseja especificar.

6. Para cada interface selecionada:
  - (a) O usuário pode criar um novo componente de sistema (Inclui *Criar componente de sistema (UC7.5)*) e associar essa interface a ele.
  - (b) O usuário pode associá-la a um componente selecionado.
7. Para cada componente selecionado ou criado, o sistema edita sua especificação para incluir as interfaces associadas pelo usuário como interfaces providas do componente.

### 2.7.5 Criar componente de sistema (UC7.5)

**Descrição:** O usuário cria uma especificação de componente de sistema.

1. O caso de uso se inicia quando o usuário seleciona a opção para criar componente de sistema.
2. O usuário informa o nome e o local na área de trabalho para a especificação de componente a ser criada.
3. O sistema cria uma especificação de componente na área de trabalho, seguindo o nome dado pelo usuário.

### 2.7.6 Criar dependência entre um componente e uma interface provida (UC7.6)

**Descrição:** O usuário, através da noção de dependência utilizada no UML Components, conecta dois componentes criando automaticamente interfaces requeridas e conectores. O usuário deve estar editando uma arquitetura.

1. O caso de uso se inicia quando o usuário, editando uma arquitetura, seleciona a opção para criar dependência entre um componente e uma interface provida.
2. O usuário informa os componente que é origem da dependência e a interface provida da qual ele depende.
3. O sistema inclui a interface provida selecionada como interface requerida na especificação do componente.
4. O sistema cria um conector que provê e requer a interface selecionada e conecta essas interfaces na interface provida selecionada e na interface requerida recém criada no componente.

## 2.8 Usar repositório de componentes (Pacote 8)

Nesta seção, apresentamos os casos de uso do pacote (8) Usar repositório de componentes. Na Figura 8 apresentamos o diagrama com os casos de uso desse pacote.

A seguir, mostramos a descrição e os passos desses casos de uso:

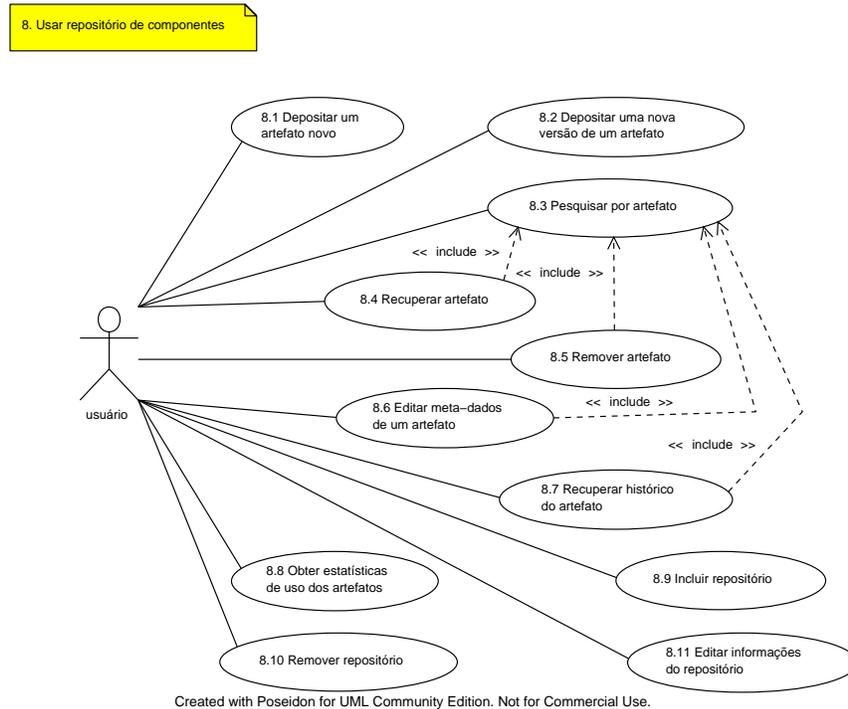


Figura 8: Casos de uso do pacote Usar repositório de componentes.

### 2.8.1 Depositar um artefato novo (UC8.1)

**Descrição:** O usuário adiciona um artefato novo no repositório. O usuário deve ter autorização para realizar essa tarefa no repositório.

Passos:

1. O caso de uso se inicia quando o usuário escolhe a opção para inserir um novo artefato no repositório.
2. O usuário informa a localização do artefato na área de trabalho, o local no repositório no qual o artefato será inserido e meta-informações sobre o artefato.
3. O sistema copia o artefato especificado para o local informado no repositório de componentes.

### 2.8.2 Depositar uma nova versão de um artefato (UC8.2)

**Descrição:** O usuário deposita uma nova versão de um artefato no repositório. O artefato deve existir no repositório. O usuário deve ter recuperado o artefato anteriormente do repositório e deve possuir autorização para realizar essa tarefa.

Passos:

1. O caso de uso se inicia quando o usuário seleciona o artefato e escolhe a opção de depositá-lo no repositório.
2. O usuário informa meta-informações sobre a nova versão do artefato.
3. O sistema copia a nova versão do artefato para o repositório de componentes, registrando as meta-informações e seu histórico.

### 2.8.3 Pesquisar por artefato (UC8.3)

**Descrição:** O usuário pesquisa por artefatos depositados previamente no repositório.

Passos:

1. O caso de uso se inicia quando o usuário escolhe a opção para pesquisar artefatos no repositório.
2. O sistema oferece uma lista de características que um artefato pode ter.
3. O usuário preenche a lista com as características que ele deseja no artefato e aciona o sistema para pesquisar.
4. O sistema retorna uma lista de artefatos que satisfazem as características dadas pelo usuário.

### 2.8.4 Recuperar artefato (UC8.4)

**Descrição:** O usuário recupera um artefato pré-existente do repositório. O usuário deve ter autorização para realizar essa tarefa no repositório.

Passos:

1. O caso de uso se inicia quando o usuário escolhe a opção de recuperar artefato.
2. Inclui caso de uso *Pesquisar por Artefato (UC8.3)*.
3. O usuário seleciona o artefato a ser recuperado e confirma.
4. O sistema encontra o artefato e o copia para a área de trabalho do usuário.

### 2.8.5 Remover artefato (UC8.5)

**Descrição:** O usuário remove um artefato pré-existente do repositório. O usuário deve ter autorização para realizar essa tarefa no repositório.

Passos:

1. Se o usuário recuperou o artefato a ser excluído do repositório anteriormente:
  - (a) o caso de uso se inicia quando ele seleciona sua versão local do artefato e escolhe a opção para remover o artefato do repositório.

2. Senão:
  - (a) o caso de uso se inicia quando o usuário seleciona a opção para remover artefatos do repositório.
  - (b) Inclui caso de uso *Pesquisar por Artefato (UC8.3)*.
  - (c) O usuário seleciona o artefato que deseja remover.
3. O usuário confirma a exclusão do artefato.
4. O sistema exclui o artefato do repositório.

### 2.8.6 Editar meta-dados de um artefato (UC8.6)

**Descrição:** O usuário edita os meta-dados de um artefato depositado no repositório. O usuário deve ter autorização para realizar essa tarefa no repositório.

Passos:

1. Se o usuário recuperou anteriormente o artefato cujos meta-dados deseja editar:
  - (a) o caso de uso se inicia quando ele seleciona sua versão local do artefato e escolhe a opção para editar meta-dados do artefato.
2. Senão:
  - (a) o caso de uso se inicia quando o usuário seleciona a opção para editar meta-dados de um artefato.
  - (b) Inclui caso de uso *Pesquisar por Artefato (UC8.3)*.
  - (c) O usuário seleciona o artefato cujos meta-dados deseja editar.
3. O usuário informa os novos meta-dados.
4. O sistema atualiza os meta-dados do artefato no repositório.

### 2.8.7 Recuperar histórico do artefato (UC8.7)

**Descrição:** O usuário visualiza o histórico de versões de um artefato depositado no repositório. O usuário deve ter autorização para realizar essa tarefa no repositório.

Passos:

1. Se o usuário recuperou anteriormente o artefato cujo histórico deseja ver:
  - (a) o caso de uso se inicia quando ele seleciona sua versão local do artefato e escolhe a opção para visualizar o histórico do artefato.
2. Senão:

- (a) o caso de uso se inicia quando o usuário seleciona a opção para visualizar o histórico de um artefato.
  - (b) Inclui caso de uso *Pesquisar por Artefato (UC8.3)*.
  - (c) O usuário seleciona o artefato cujo histórico deseja consultar.
3. O sistema exibe o histórico de versões do artefato.

### 2.8.8 Obter estatísticas de uso dos artefatos (UC8.8)

**Descrição:** O usuário obtém estatísticas de uso sobre os artefatos depositados no repositório. O usuário deve ter autorização para realizar essa tarefa no repositório.

Passos:

1. O usuário seleciona a opção de obter estatísticas.
2. O usuário informa o tipo de estatística que deseja obter e sobre quais componentes ele deseja obter essas estatísticas.
3. O sistema exibe as estatísticas para o usuário, baseado nas informações dadas.

### 2.9 Incluir repositório (UC8.9)

**Descrição:** O usuário inclui um repositório para uso, criando uma referência para ele, e se autentica para utilizá-lo.

Passos:

1. O caso de uso se inicia quando o usuário seleciona a opção para incluir repositório.
2. O usuário informa o nome e localização do repositório.
3. O usuário informa sua identificação e senha.
4. O sistema valida as informações do repositório e do usuário. Caso elas sejam válidas, inclui o repositório e concede as autorizações que couberem ao usuário.

### 2.10 Remover repositório (UC8.10)

**Descrição:** O usuário exclui um repositório previamente registrado no ambiente. O repositório em si e seu conteúdo não é afetado por essa operação.

Passos:

1. O caso de uso se inicia quando o usuário seleciona o repositório e escolhe a opção para excluí-lo.
2. O usuário confirma que deseja excluir o repositório.
3. O sistema exclui o registro do repositório do ambiente.

### **2.11 Editar informações do repositório (UC8.11)**

**Descrição:** O usuário edita informações de um repositório previamente registrado no ambiente.

Passos:

1. O caso de uso se inicia quando o usuário seleciona o repositório e escolhe a opção para editar suas informações.
2. O usuário edita as informações sobre o repositório.
3. O sistema valida as informações editadas. Caso elas sejam válidas, registra as alterações, atualizando a visão do usuário com as novas informações.



### **3 Protótipos de Interface com Usuário**

Nessa seção apresentamos os protótipos de interface gráfica com usuário (interfaces homem-computador ou IHC) produzidos para o projeto do ambiente BELLATRIX. Tais protótipos foram desenvolvidos em papel, apresentados para discussões com o grupo de pesquisas e com potenciais usuários e posteriormente digitalizados. Identificamos como vantagens dessa abordagem a maior participação do usuário, auxiliando tanto o projeto de interface homem-máquina propriamente dito, quanto a extração de requisitos e seu registro. Ainda, os protótipos procuram exercitar os casos de uso do ambiente, formando cenários de uso.

As seguintes seções estão organizadas do seguinte modo: na Seção 3.1 são apresentados protótipos relacionados à especificação de interfaces. Na Seção 3.2 são apresentados protótipos relativos à especificação de componentes. Na Seção 3.3 mostramos os protótipos de especificação de arquiteturas. Na seção 3.4 estão os protótipos de interface de implementação de componentes. Na Seção 3.5 são apresentados os protótipos dos assistentes para o processo UML Components e na Seção 3.6, os protótipos relativos à interface com o repositório de componentes a partir do ambiente.

#### **3.1 Especificação de interfaces**

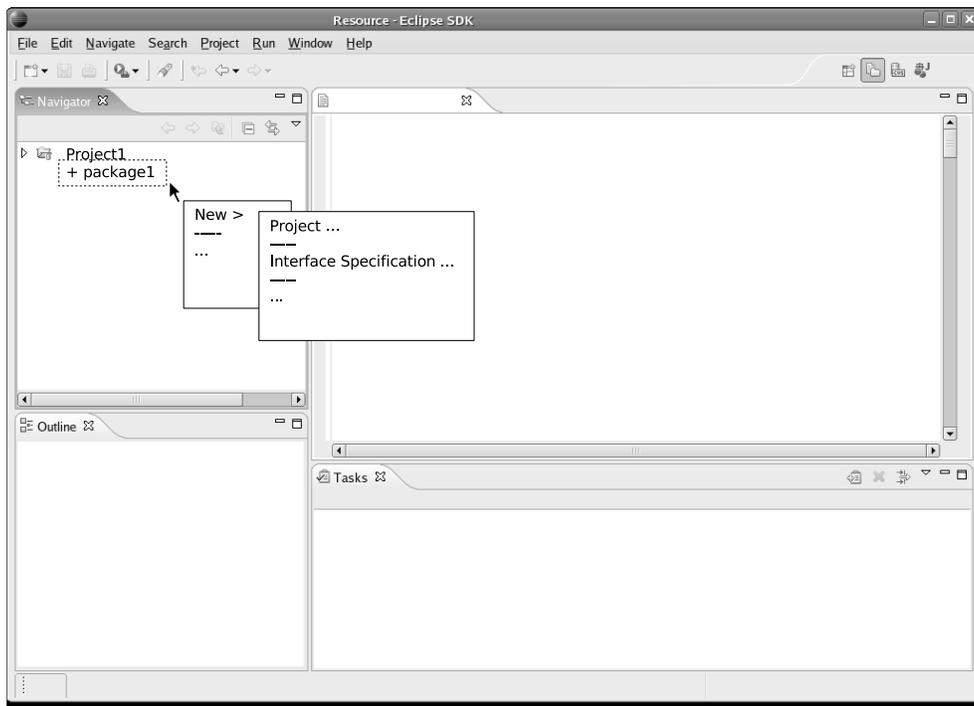


Figura 9: Criar interface (1 - menu)

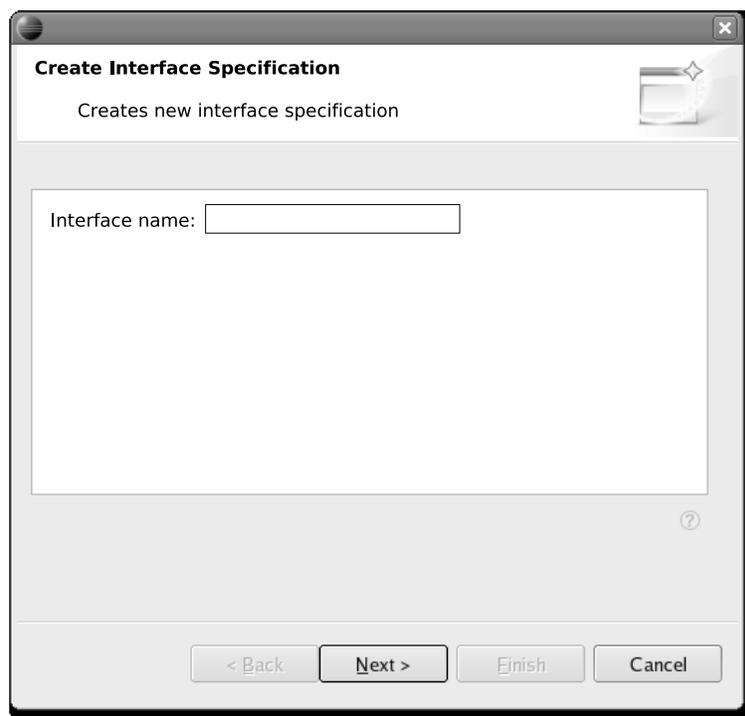


Figura 10: Criar interface (2 - assistente)

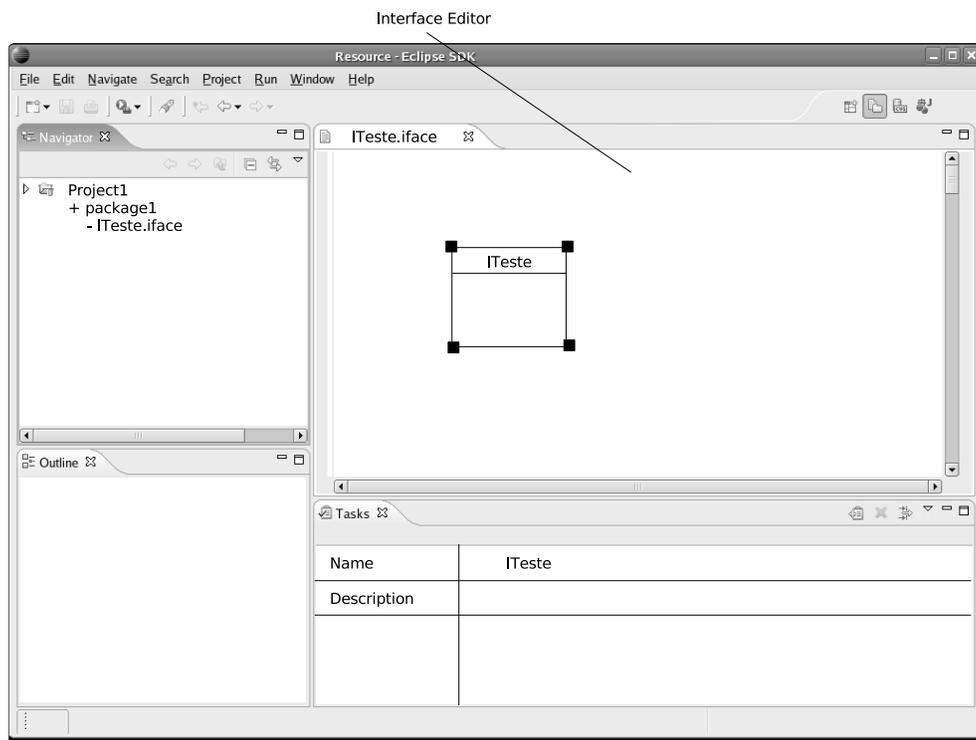


Figura 11: Criar interface (3 - interface criada)

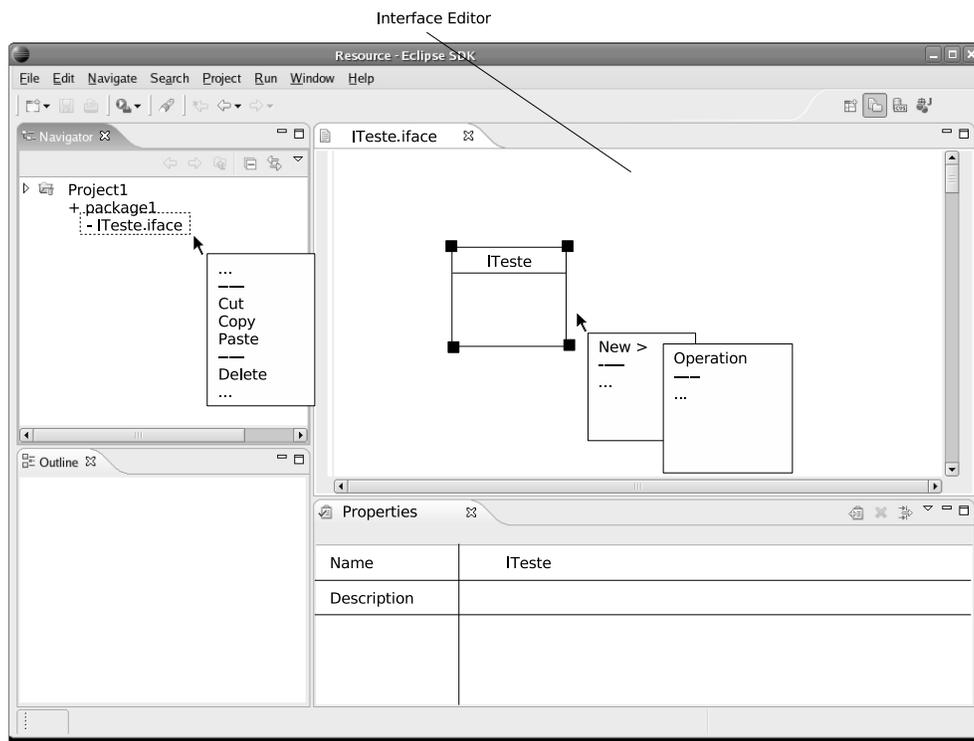


Figura 12: Excluir interface; Editar informações de uma interface; e Criar operação numa interface

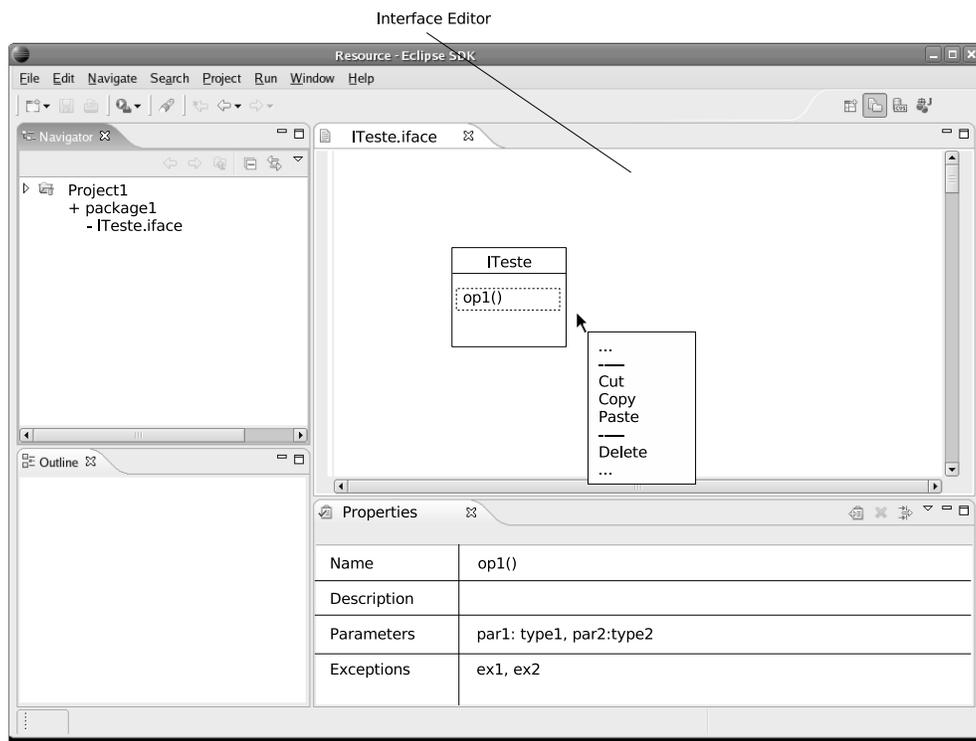


Figura 13: Excluir operação de uma interface; Editar informações de uma operação (tela propriedades); e Copiar e mover uma operação de uma interface para outra

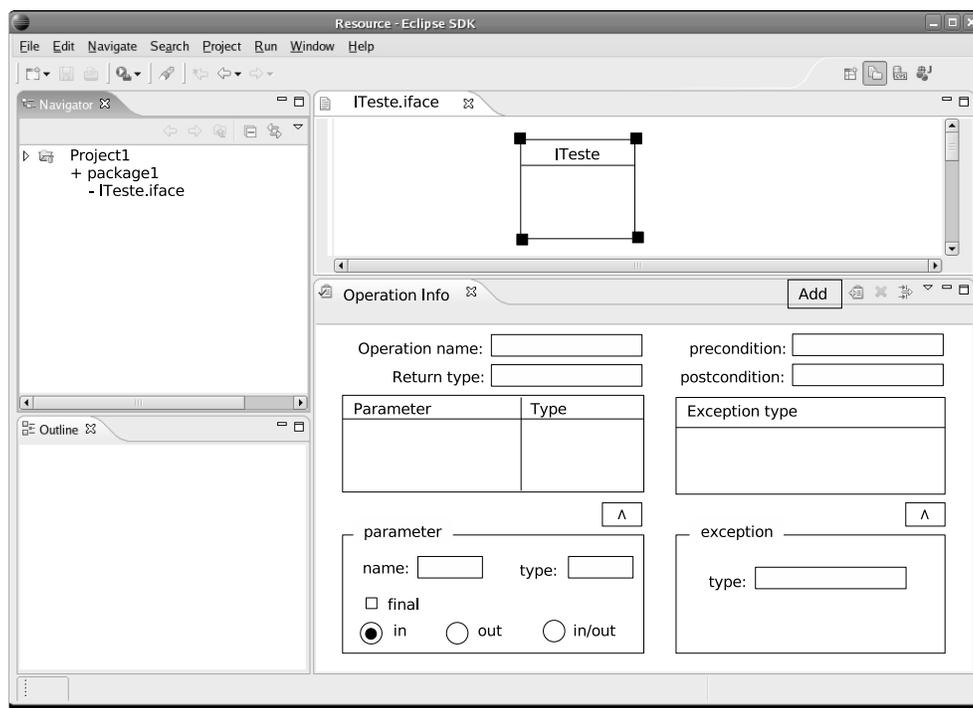


Figura 14: Criar operação numa interface; e Editar informações de uma operação (1 - tela formulário)

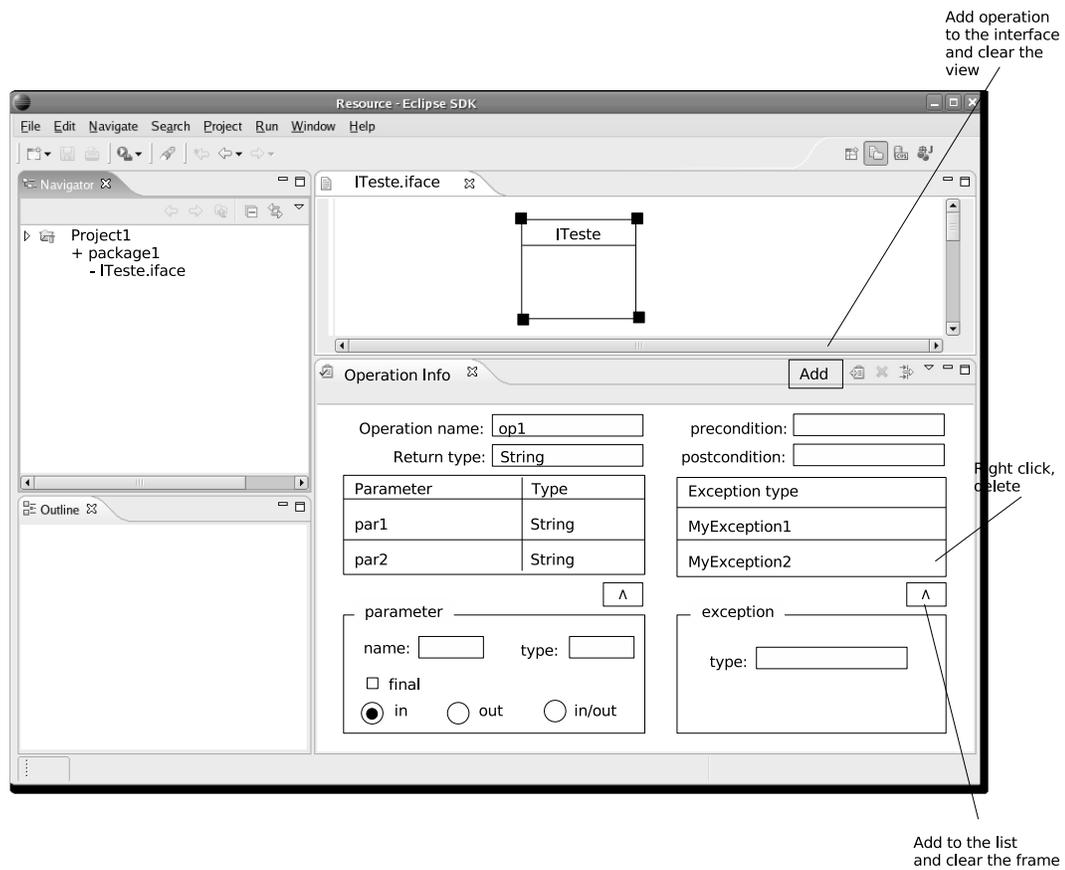
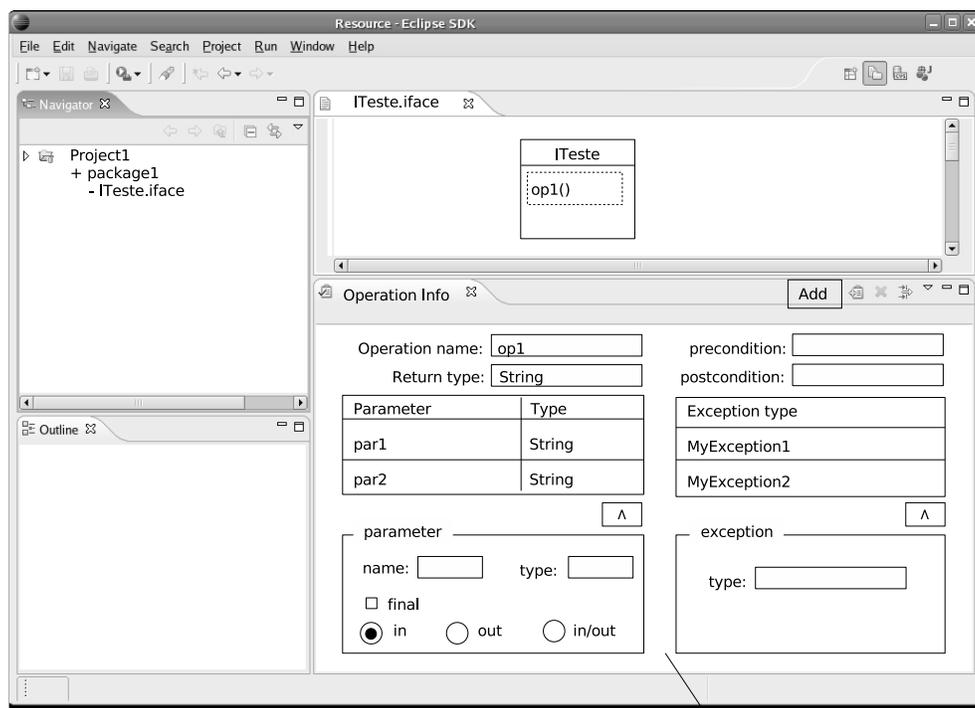


Figura 15: Criar operação numa interface; e Editar informações de uma operação (2 - tela formulário preenchida)



Changing data in this view now will edit operation info.

Figura 16: Criar operação numa interface; e Editar informações de uma operação (3 - tela formulário e operação criada)

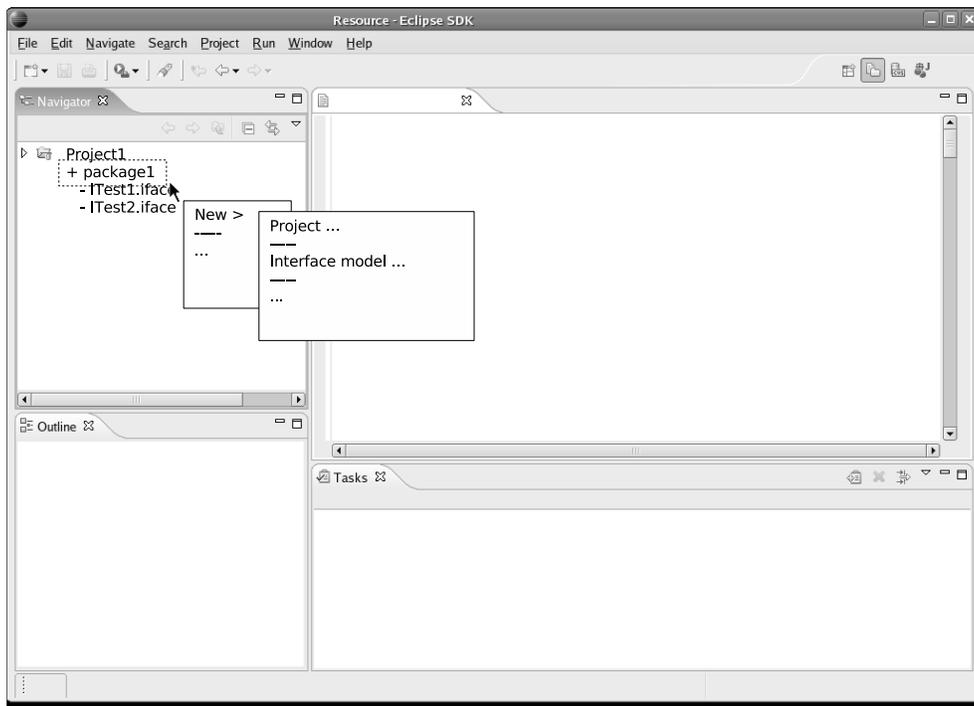


Figura 17: Menu para criar modelo de interfaces (1)

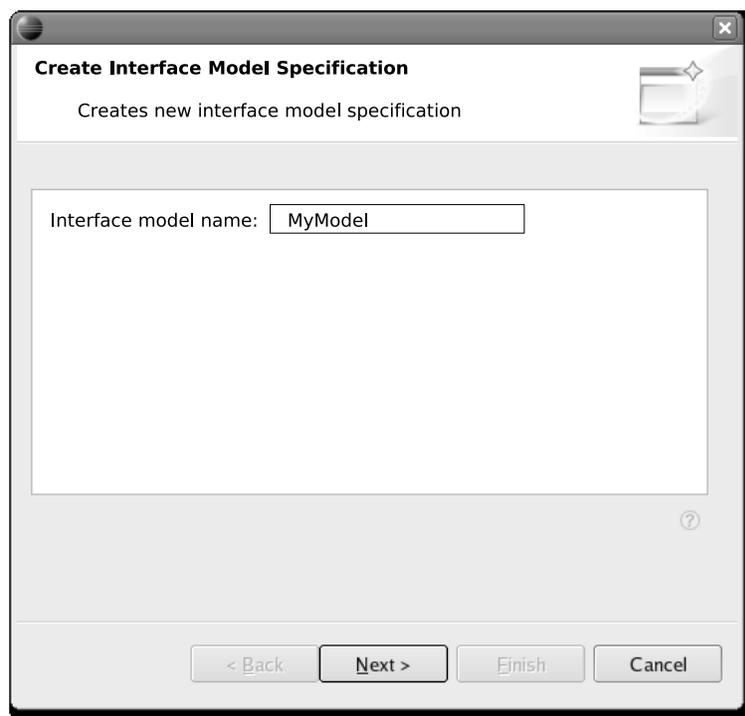


Figura 18: Assistente para definir nome do modelo de interfaces (2)

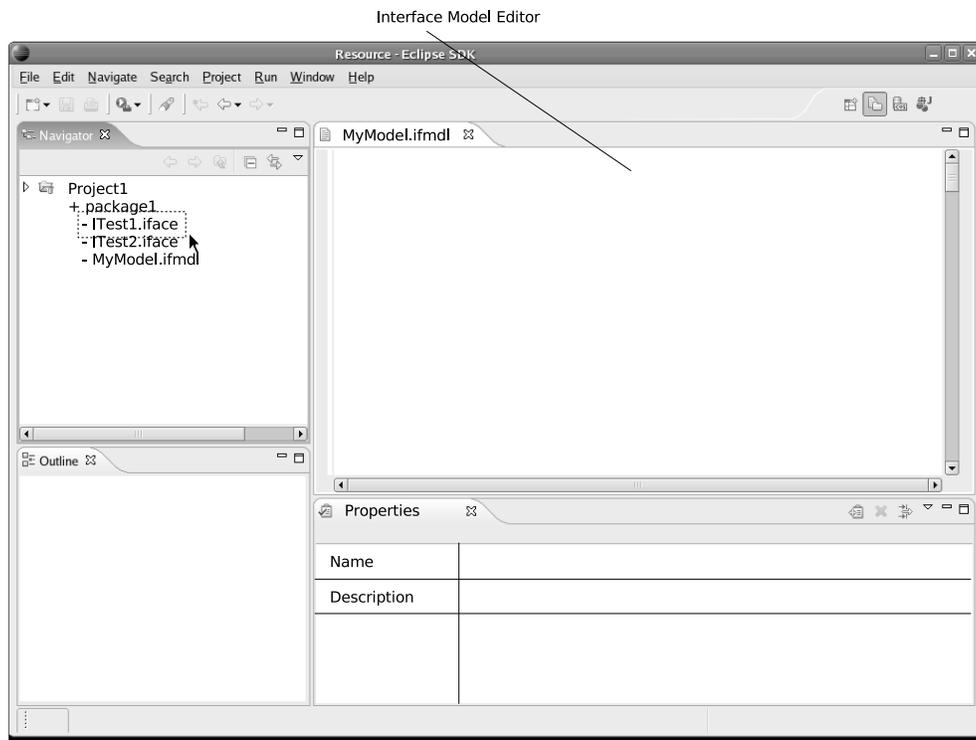


Figura 19: Modelo criado (3)

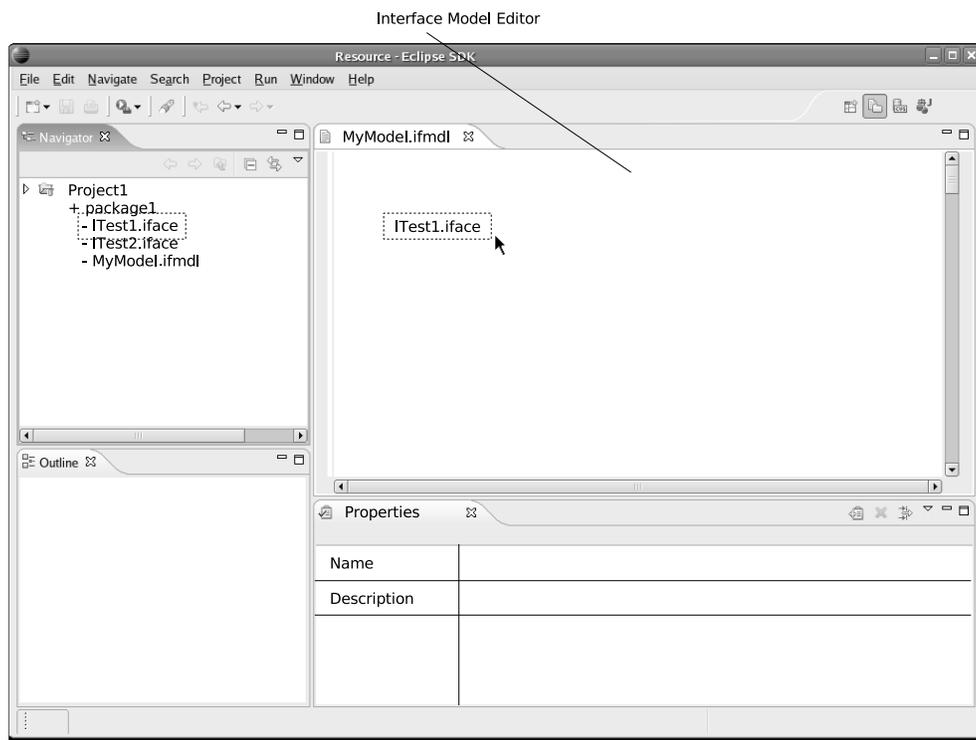


Figura 20: Arrastando nova interface para o modelo (1)

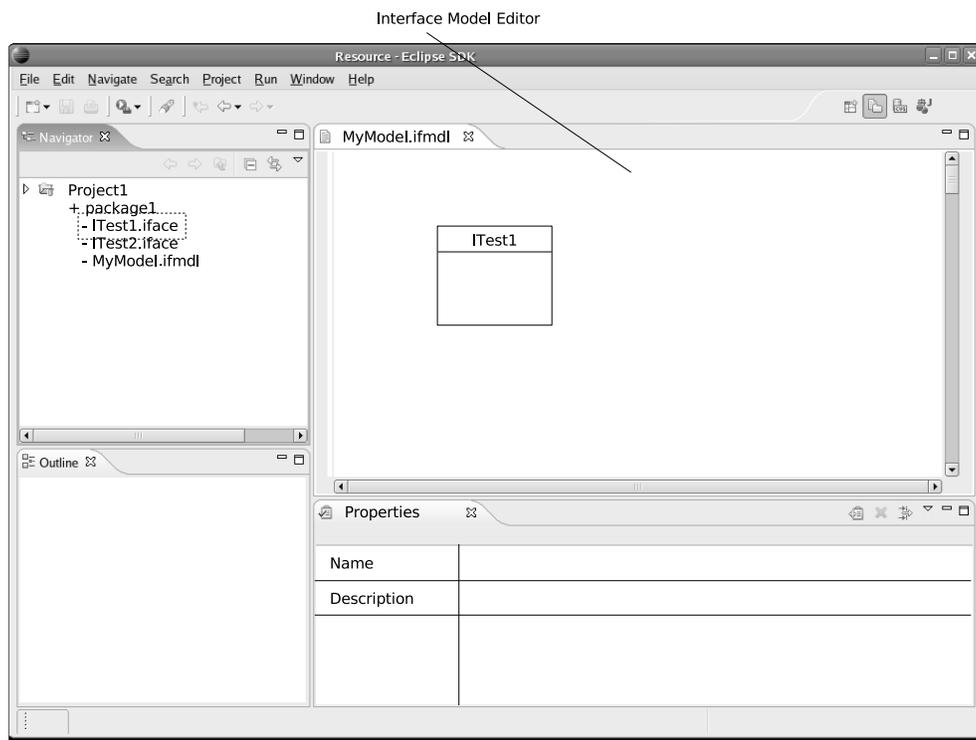


Figura 21: Interface incluída no modelo (2)

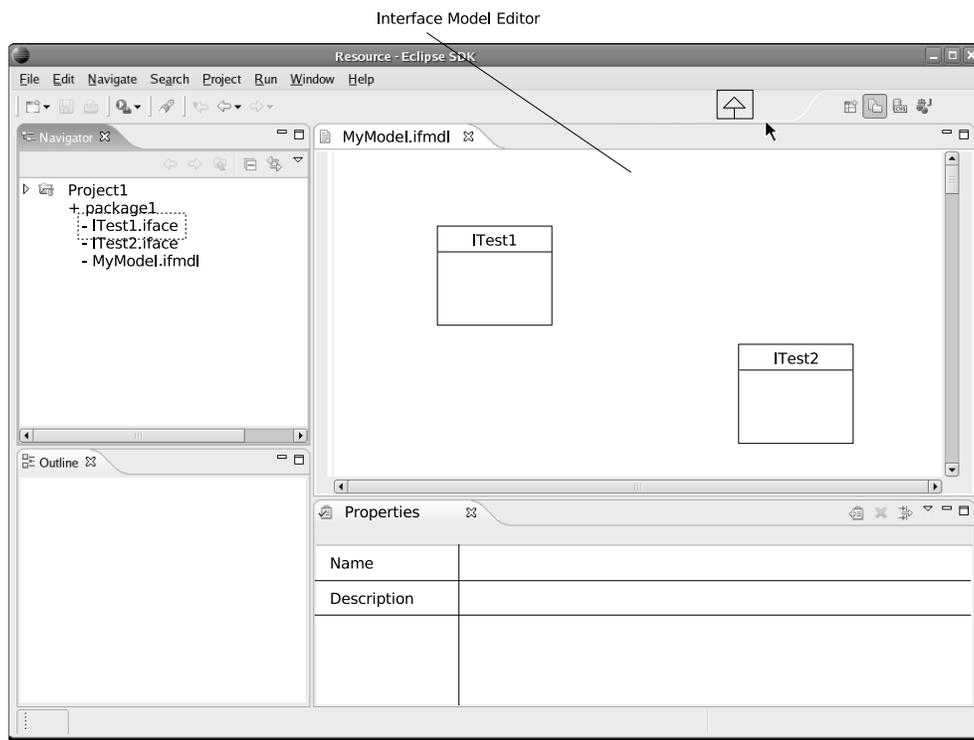


Figura 22: Definir relação de generalização e especialização entre interfaces (1)

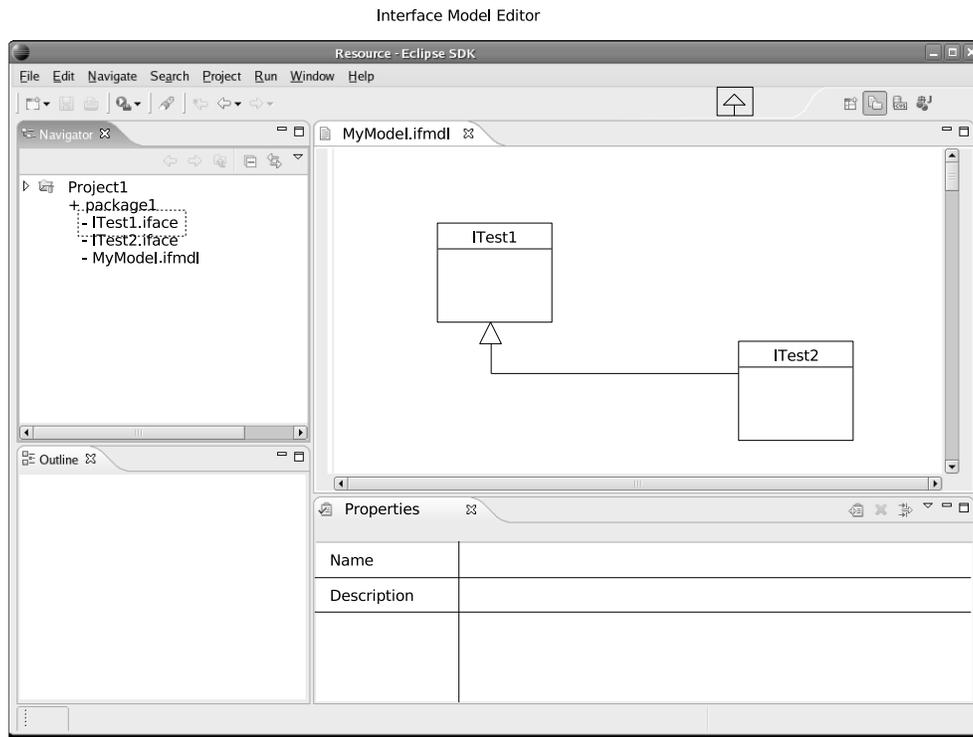


Figura 23: Definir relação de generalização e especialização entre interfaces (2)

### 3.2 Especificação de componentes

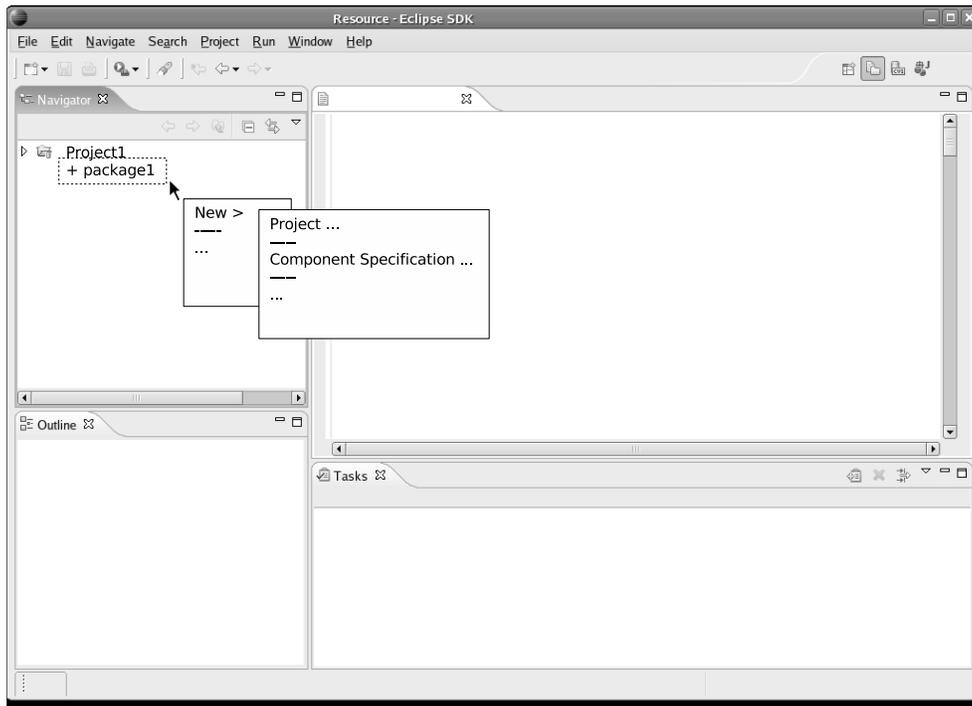


Figura 24: Criar componente (1 - menu)

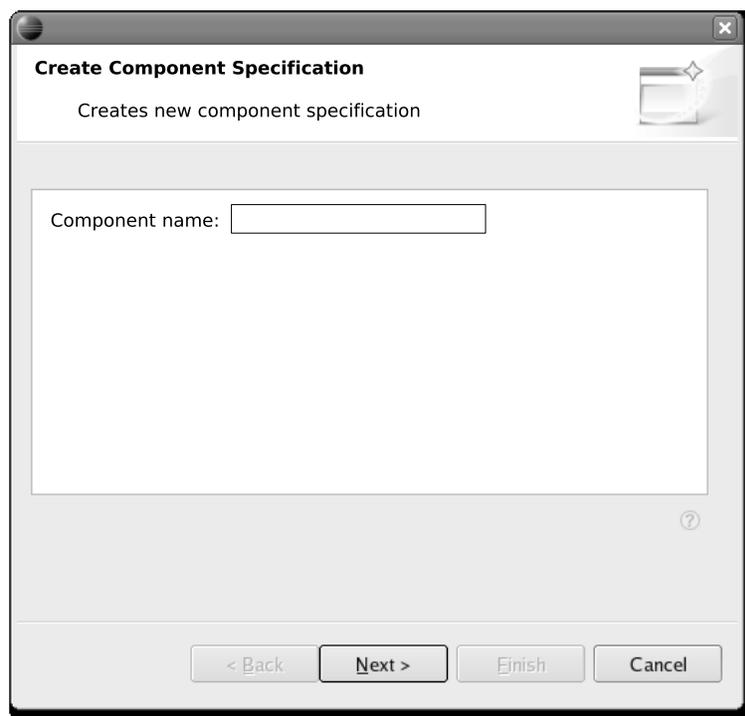


Figura 25: Criar componente (2 - assistente)

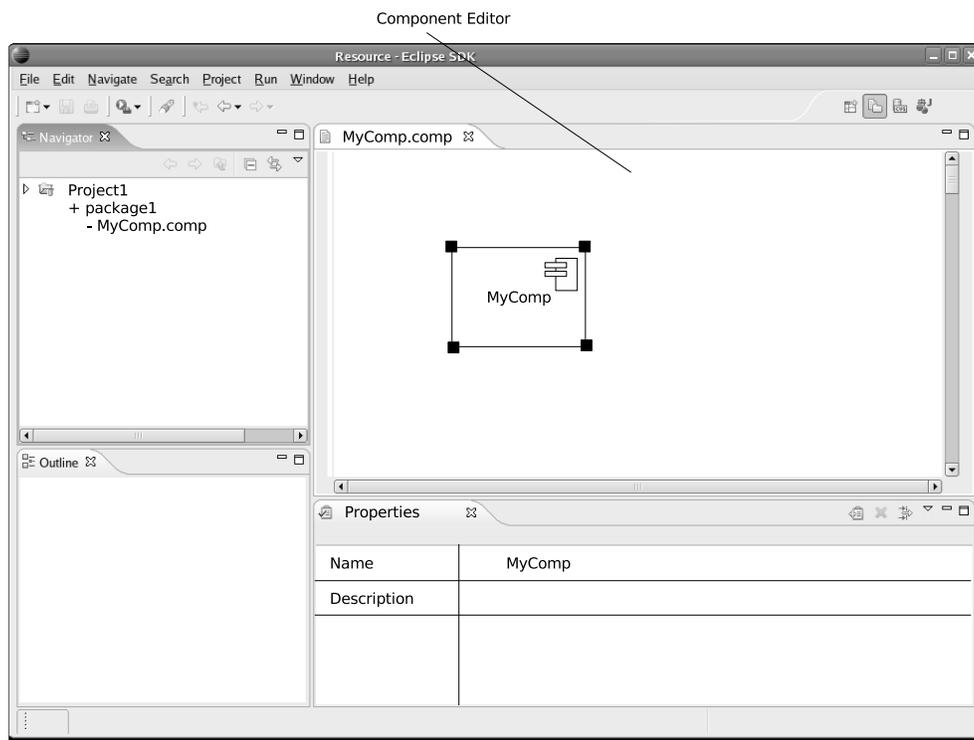


Figura 26: Criar componente (3 - componente criado)

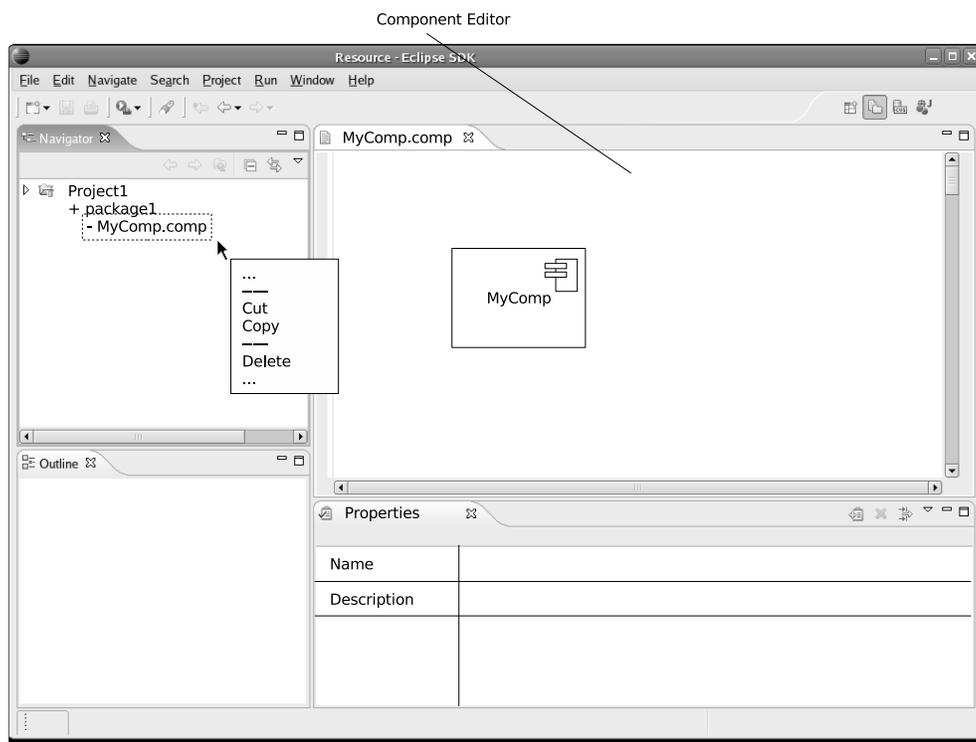


Figura 27: Excluir componente

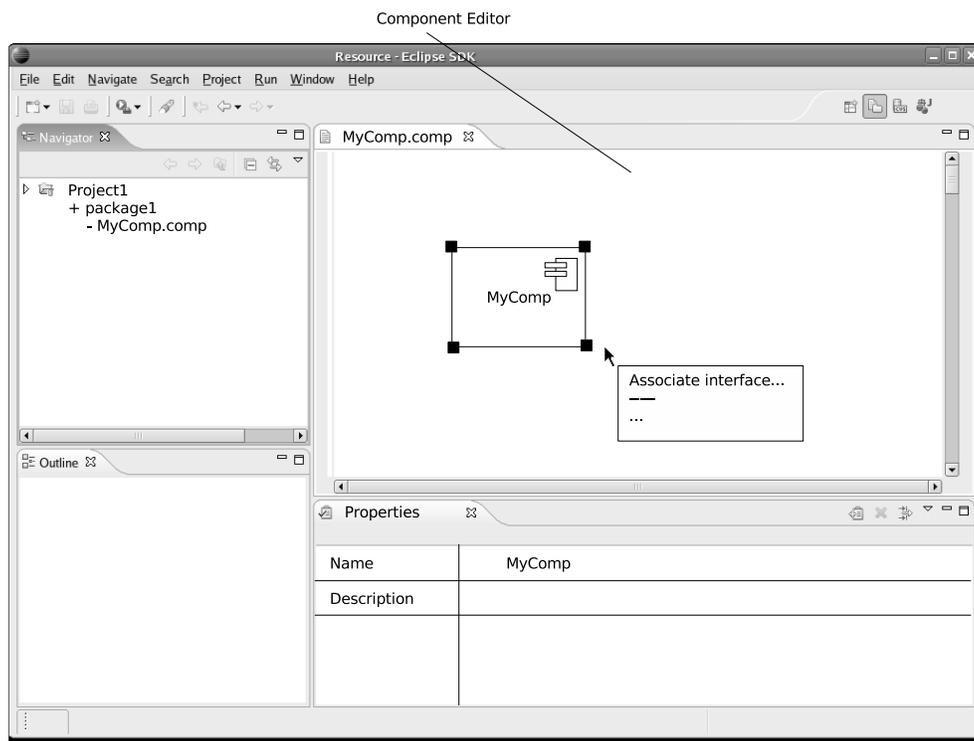


Figura 28: Editar informações de um componente; e Associar interface provida ou requerida a um componente (1 - menu)

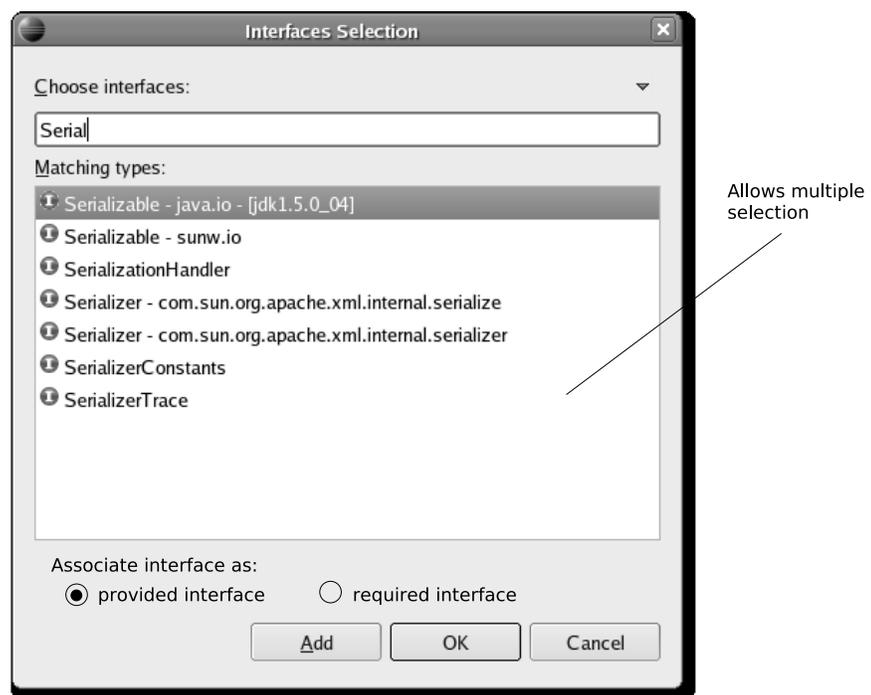


Figura 29: Associar interface provida ou requerida a um componente (2 - selecionar interface)

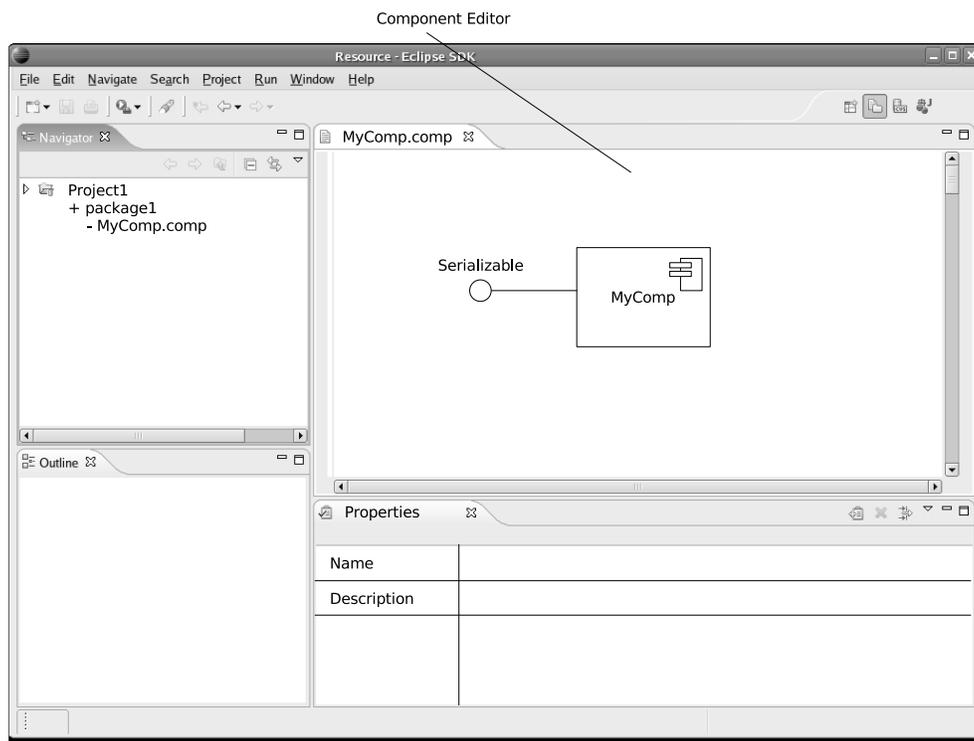


Figura 30: Associar interface provida ou requerida a um componente (3 - interface provida associada)

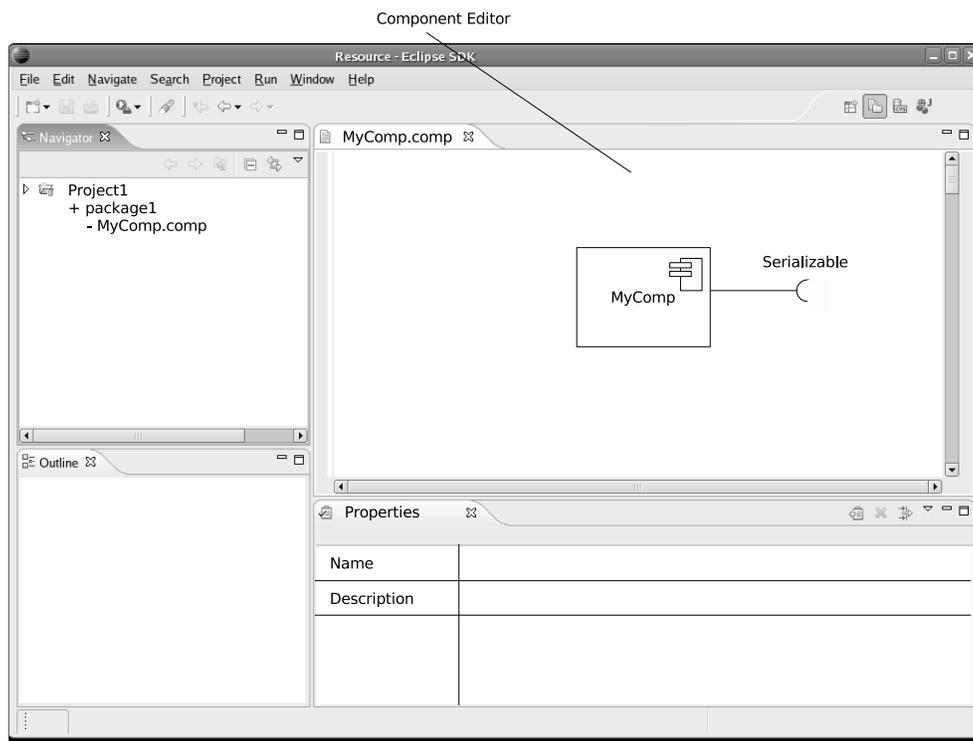


Figura 31: Associar interface provida ou requerida a um componente (3 - interface requerida associada)

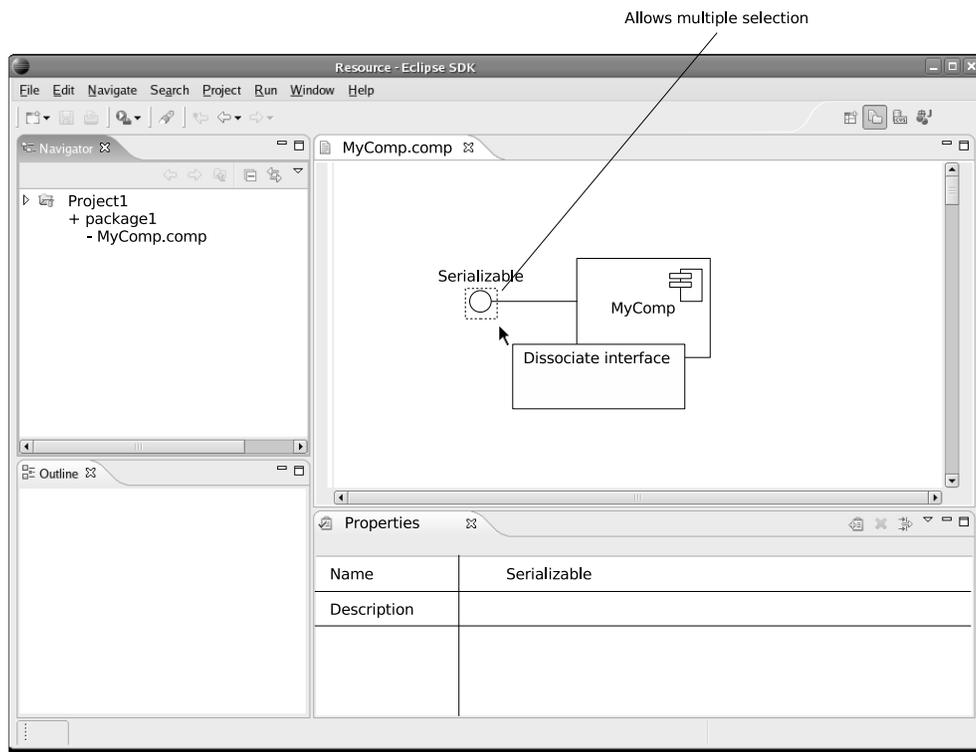


Figura 32: Desassociar interface de um componente

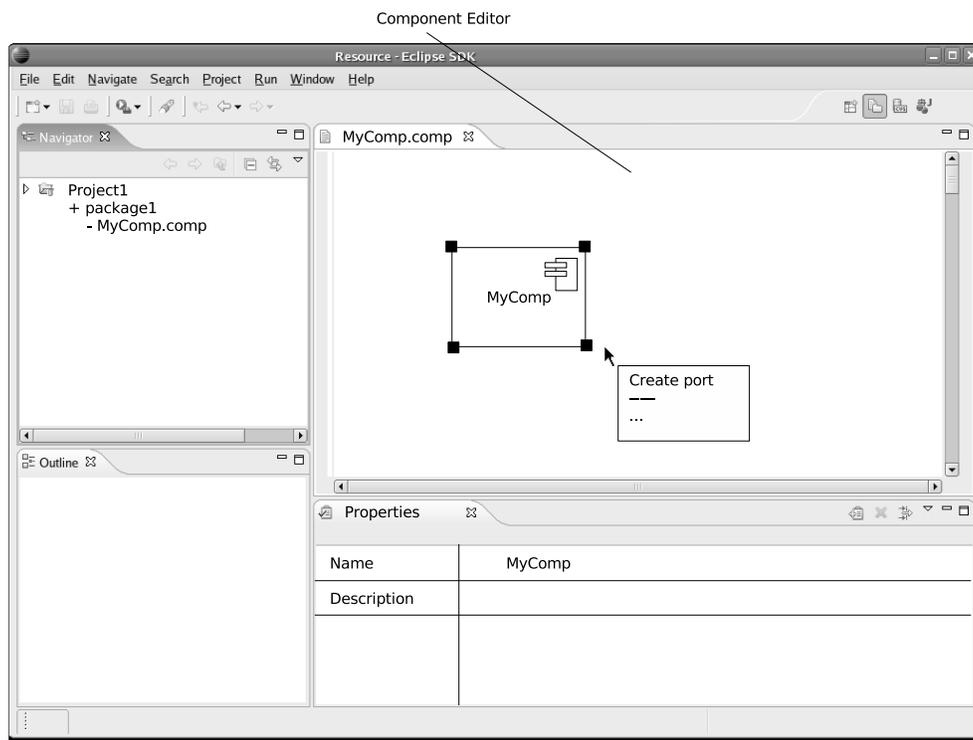


Figura 33: Criar porta num componente (1 - menu)

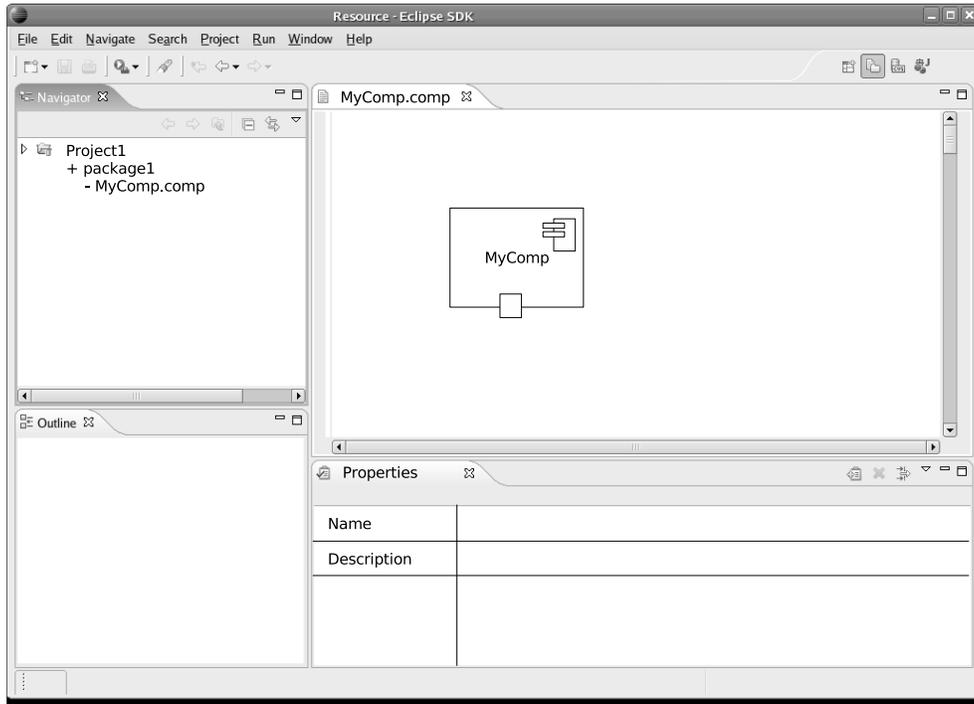


Figura 34: Criar porta num componente (2 - porta criada)

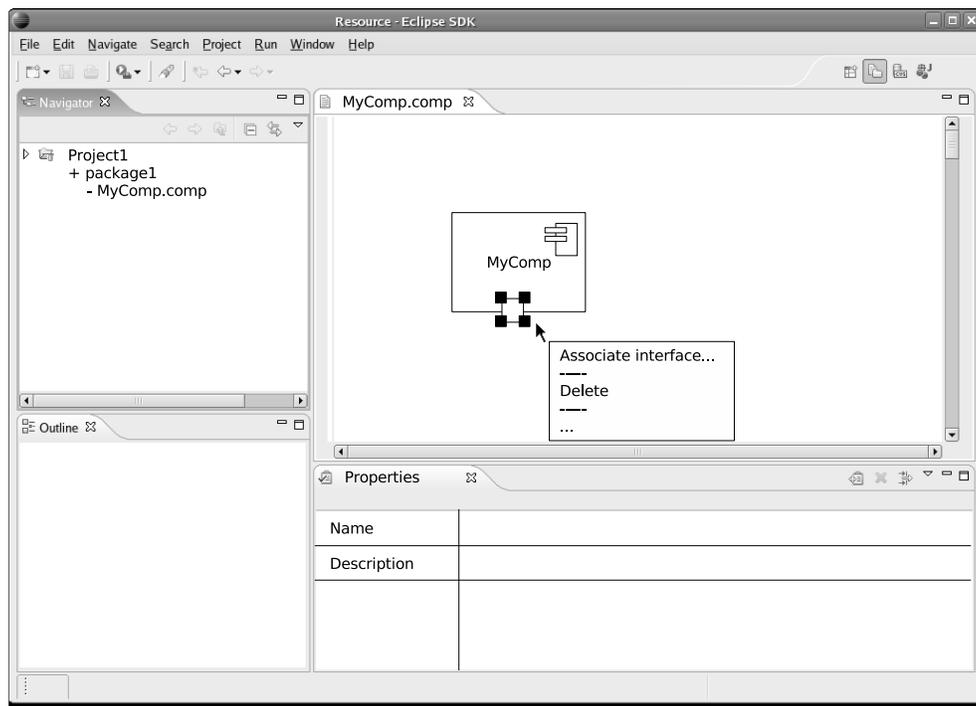


Figura 35: Remover porta de um componente; e Associar interface provida ou requerida a uma porta (1 - menu)

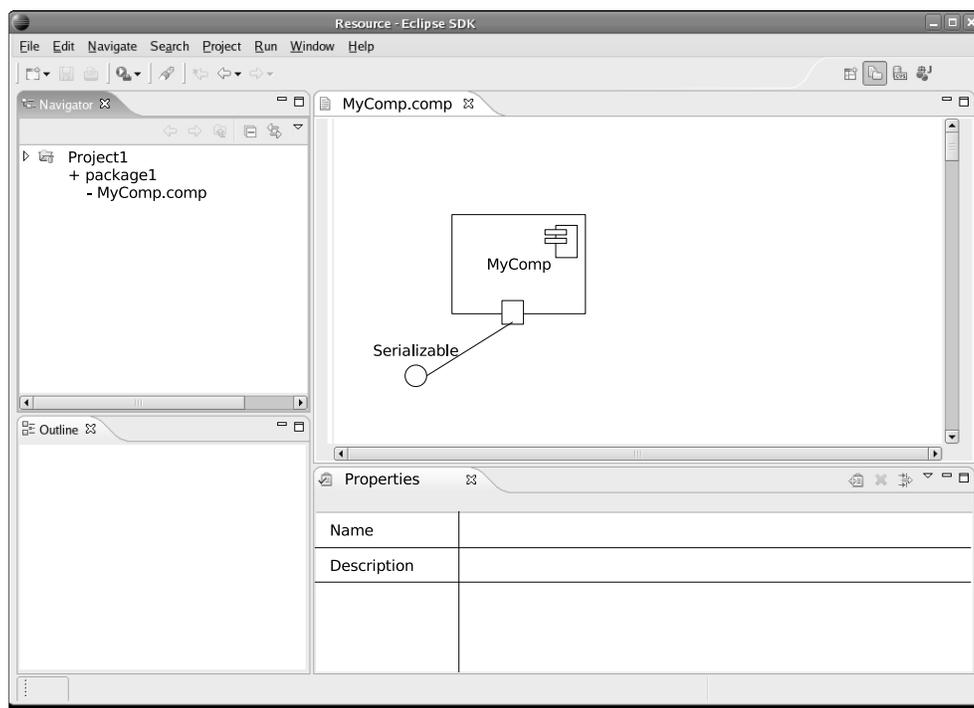


Figura 36: Associar interface provida ou requerida a uma porta (2 - interface provida associada)

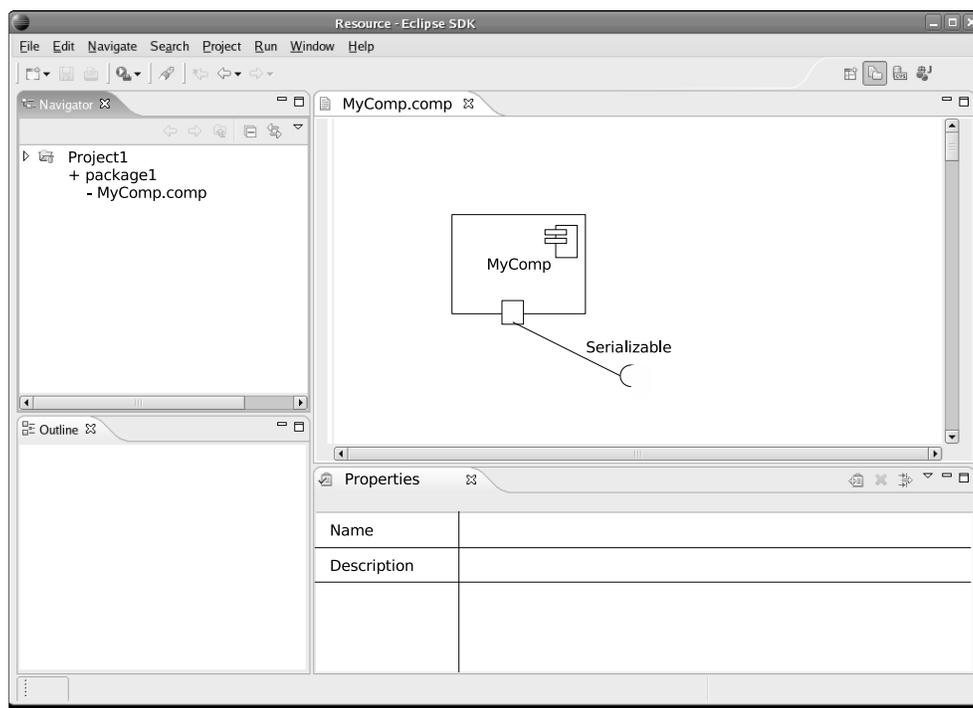


Figura 37: Associar interface provida ou requerida a uma porta (2 - interface requerida associada)

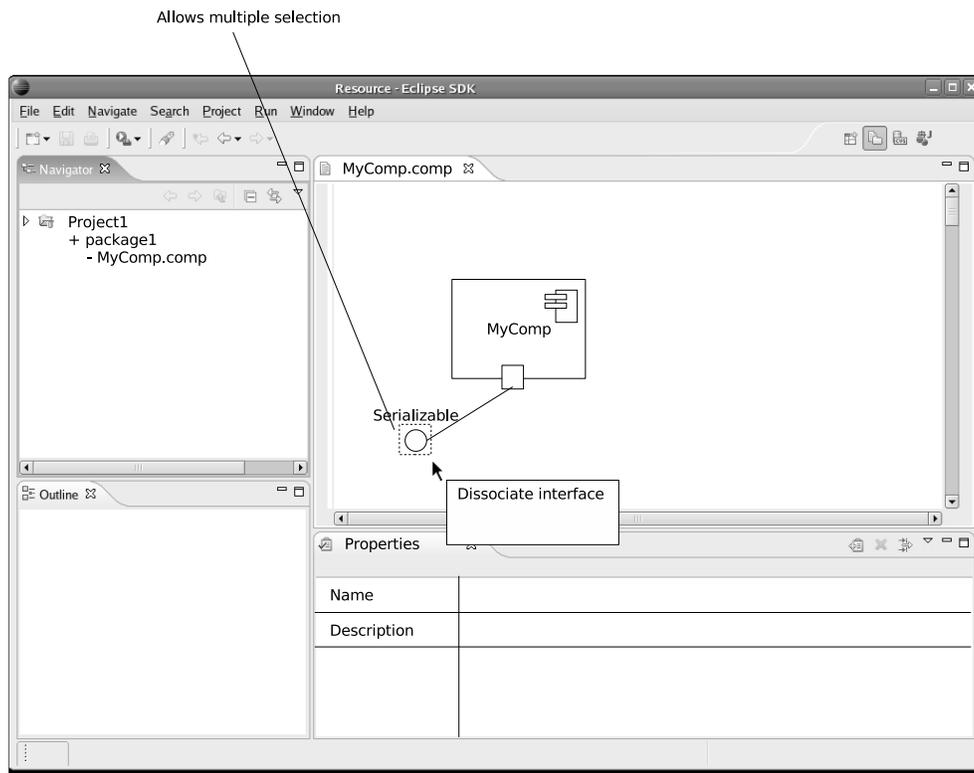


Figura 38: Desassociar interface de uma porta

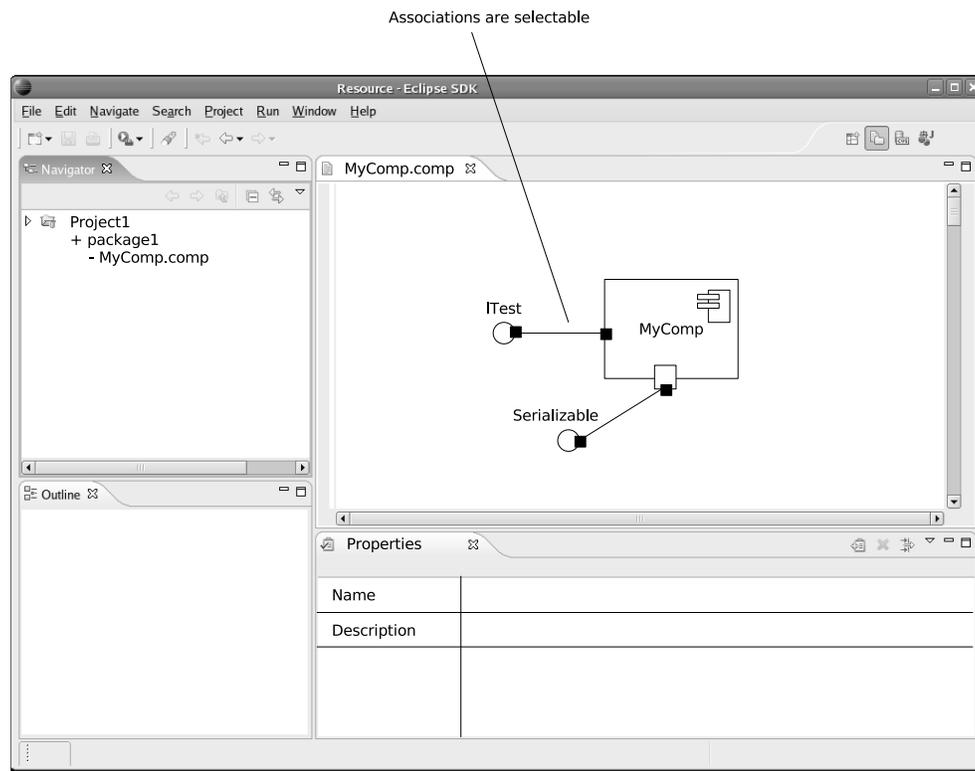


Figura 39: Editar informações da associação interface componente

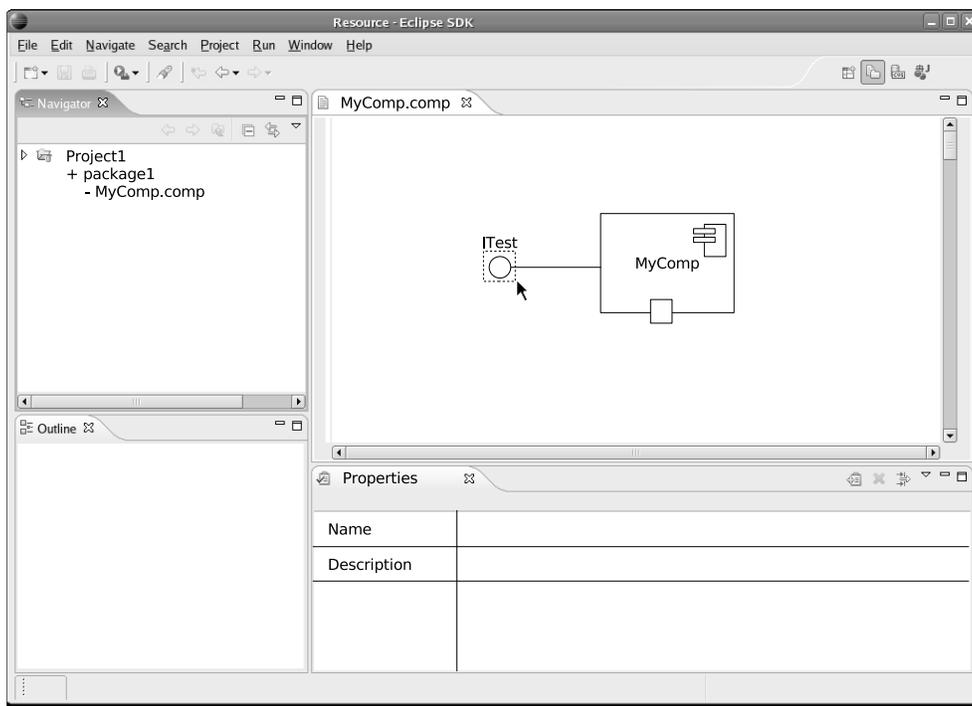


Figura 40: Mover interface (1 - interface selecionada)

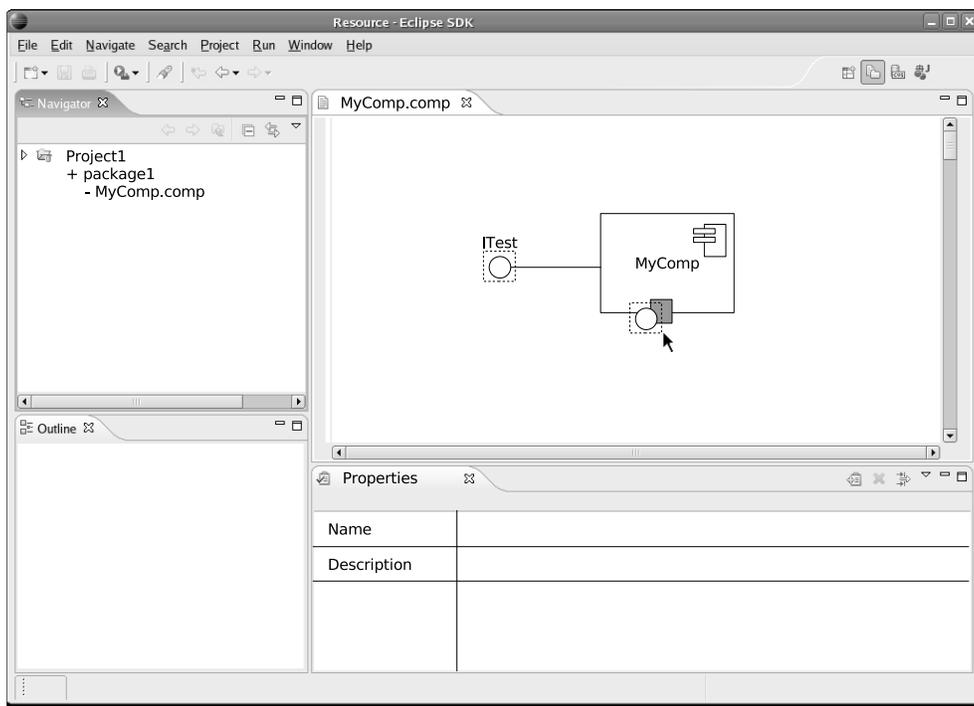


Figura 41: Mover interface (2 - arrastando interface)

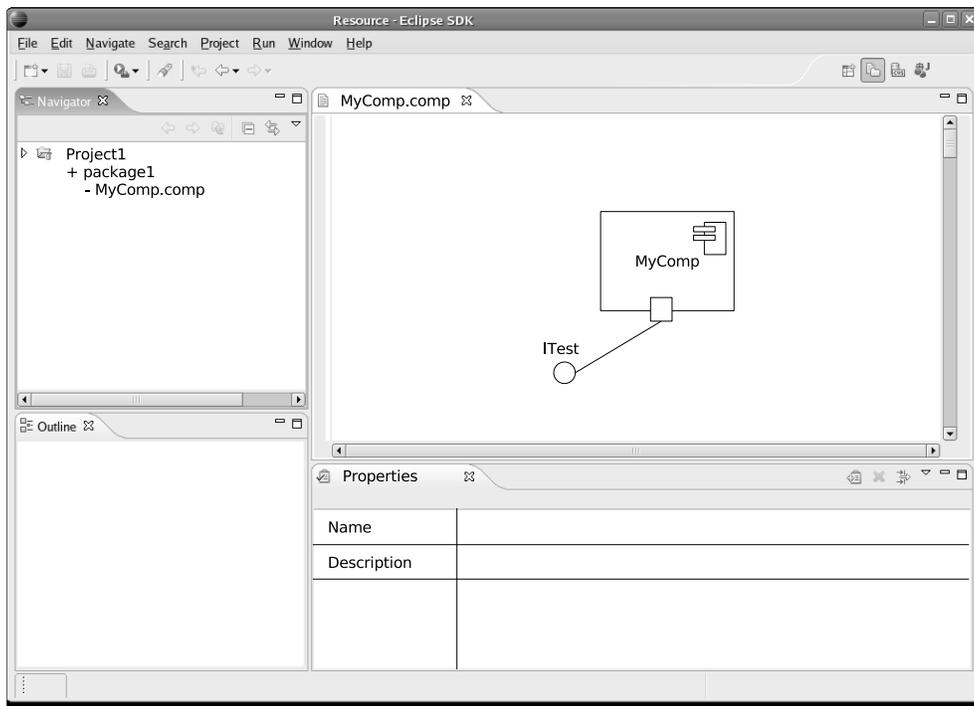


Figura 42: Mover interface (3 - interface movida)

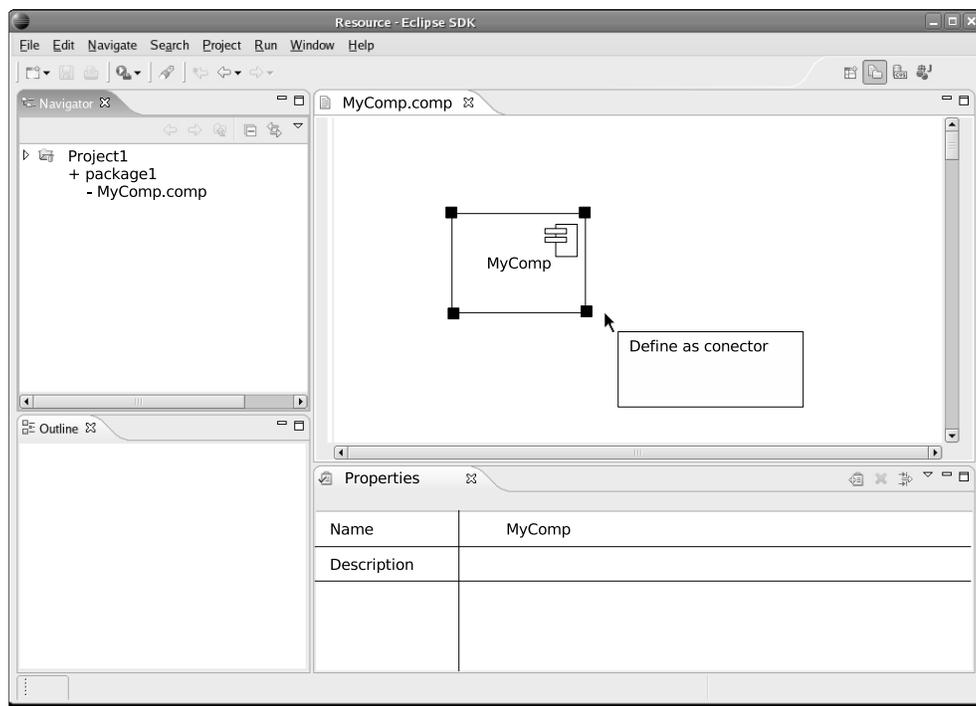


Figura 43: Definir como conector (1 - menu)

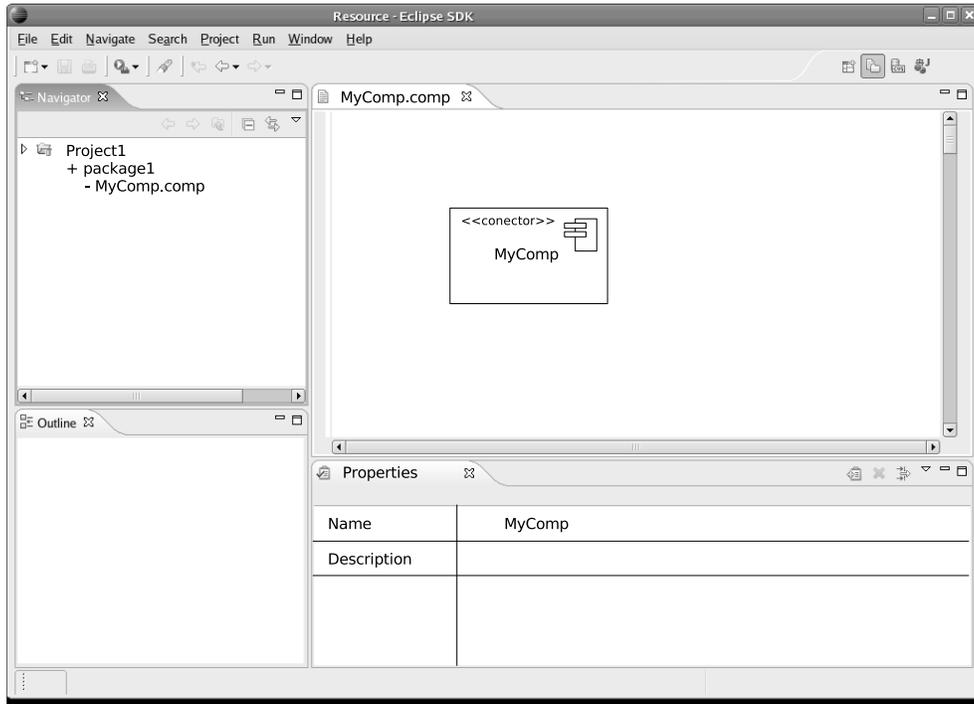


Figura 44: Definir como conector (2 - definido)

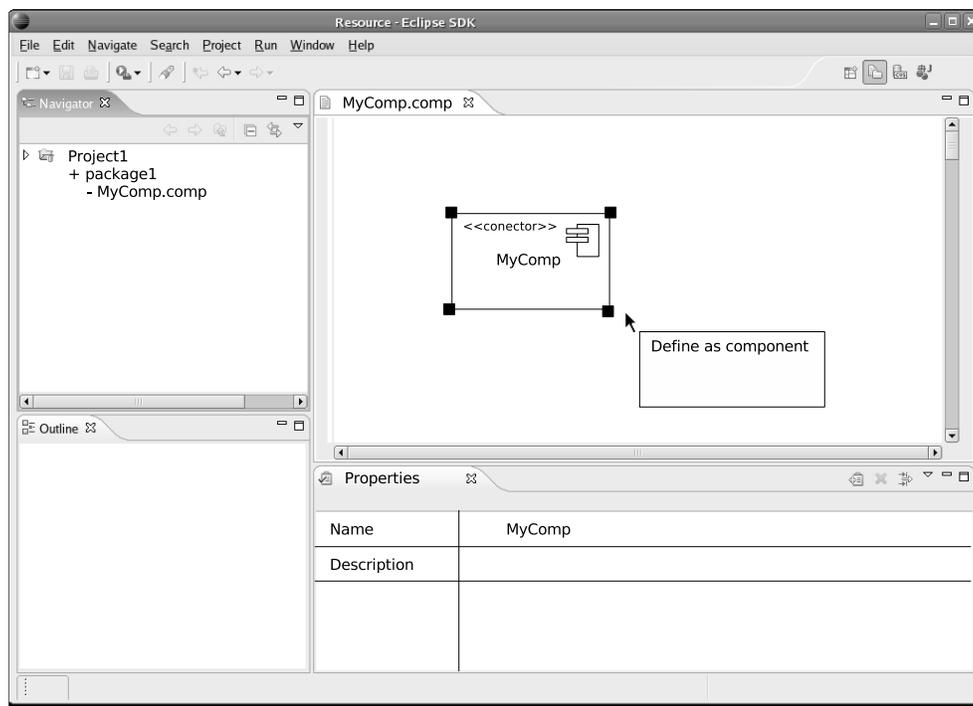


Figura 45: Remover definição de conector (1 - menu)

### 3.3 Especificação de arquiteturas

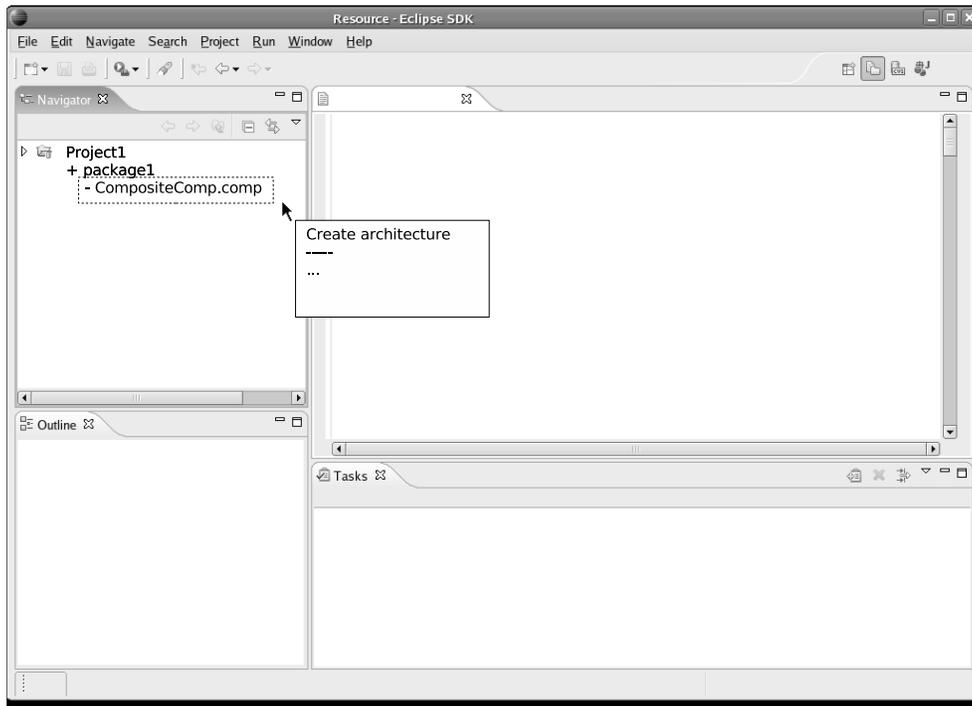


Figura 46: Criar arquitetura (1 - menu)

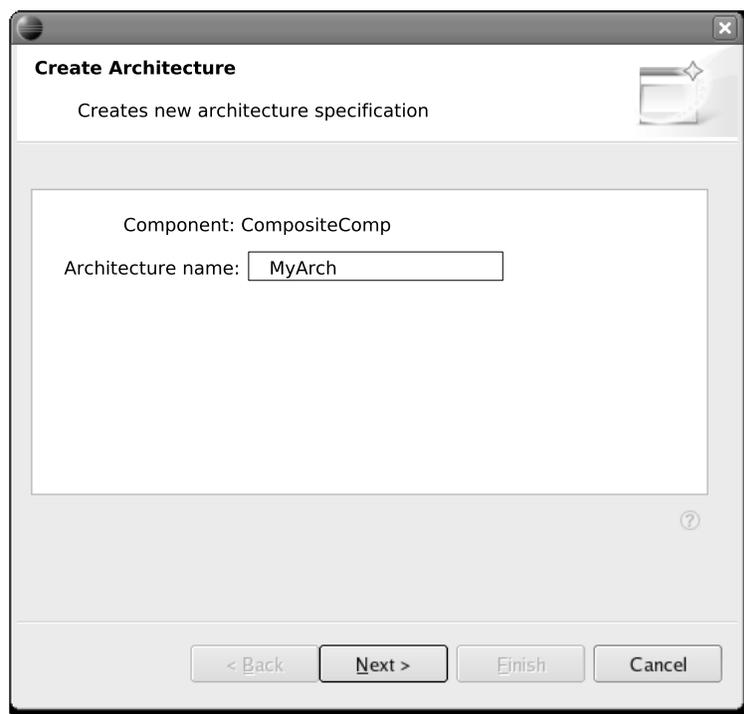


Figura 47: Criar arquitetura (2 - assistente)

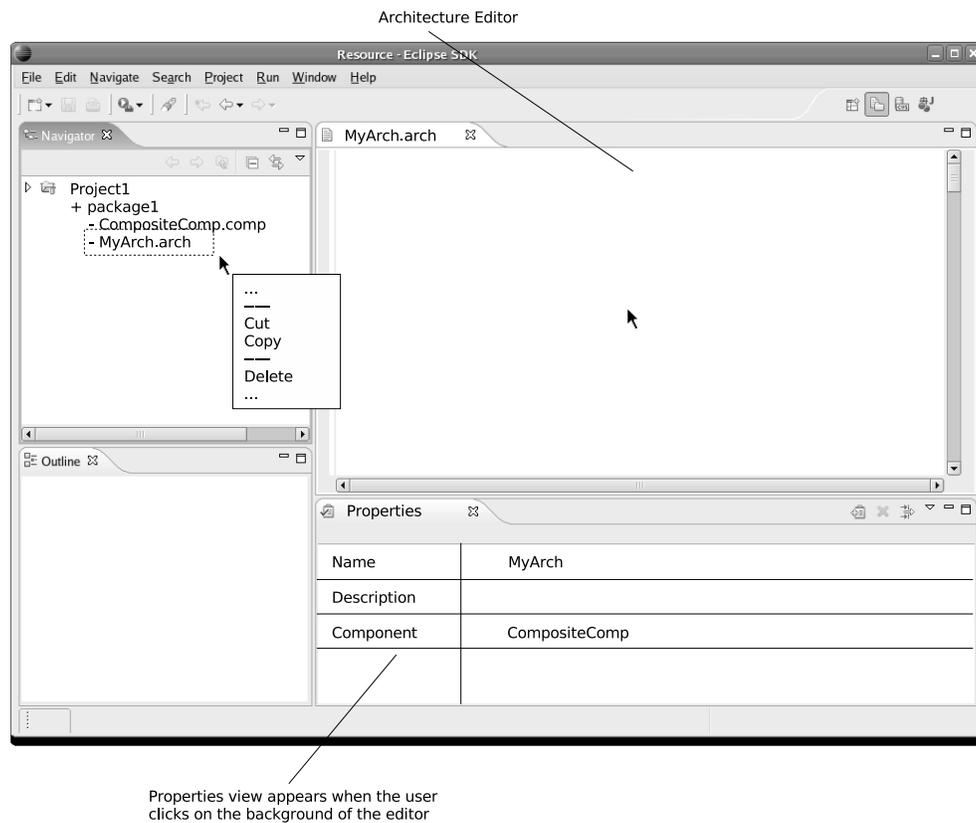


Figura 48: Criar arquitetura (3 - arquitetura criada); Excluir arquitetura; e Editar informações de uma arquitetura

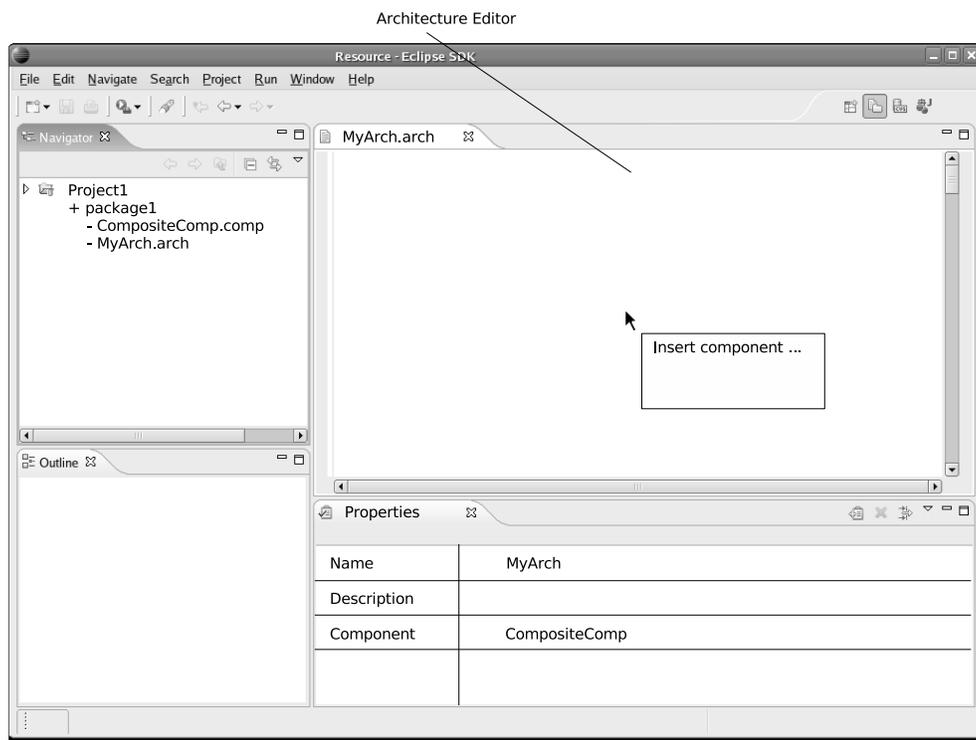


Figura 49: Inserir componente ou conector a uma arquitetura (1 - menu)

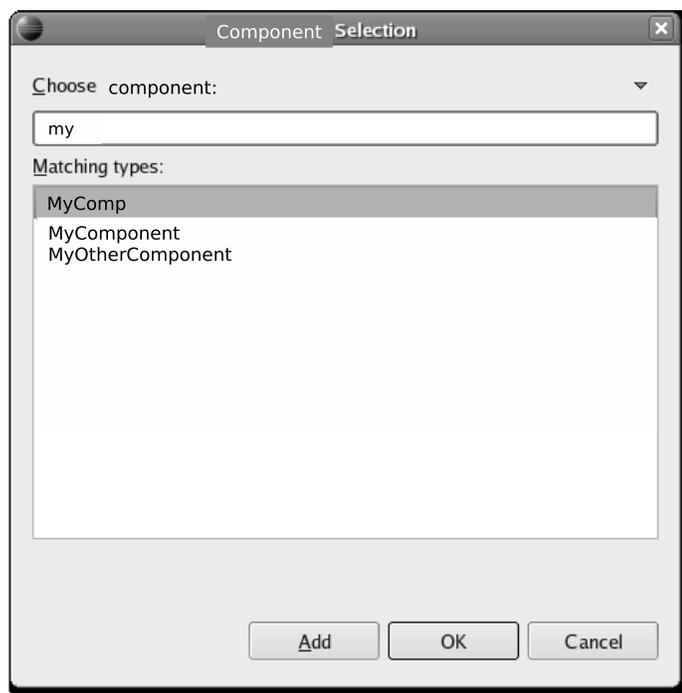


Figura 50: Inserir componente ou conector a uma arquitetura (2 - selecionar componente)

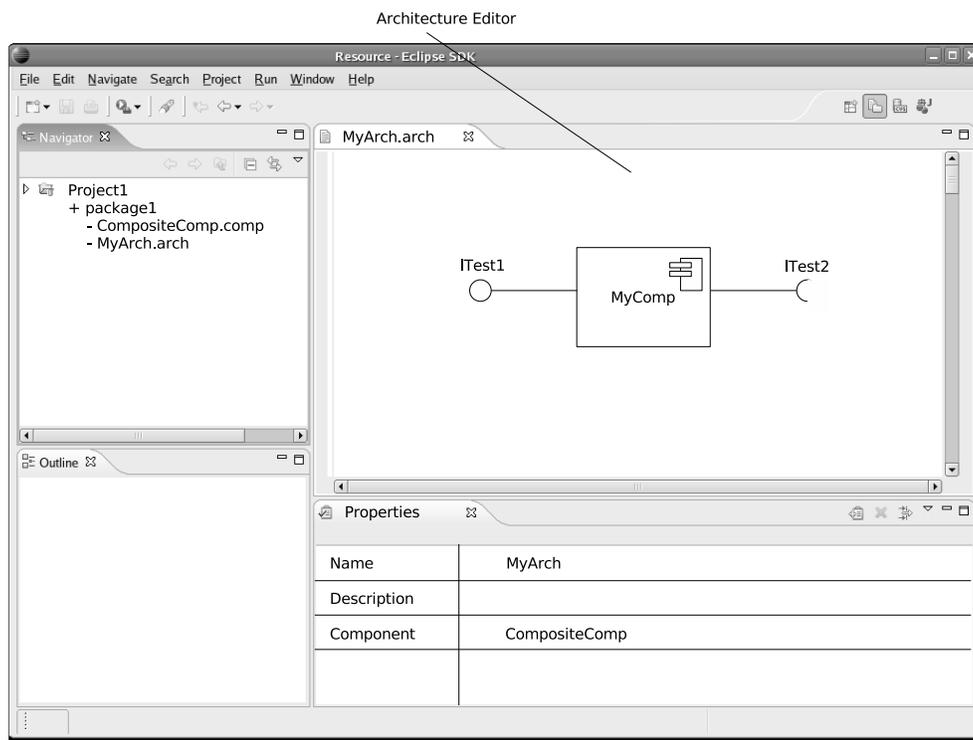


Figura 51: Inserir componente ou conector a uma arquitetura (3 - componente inserido)

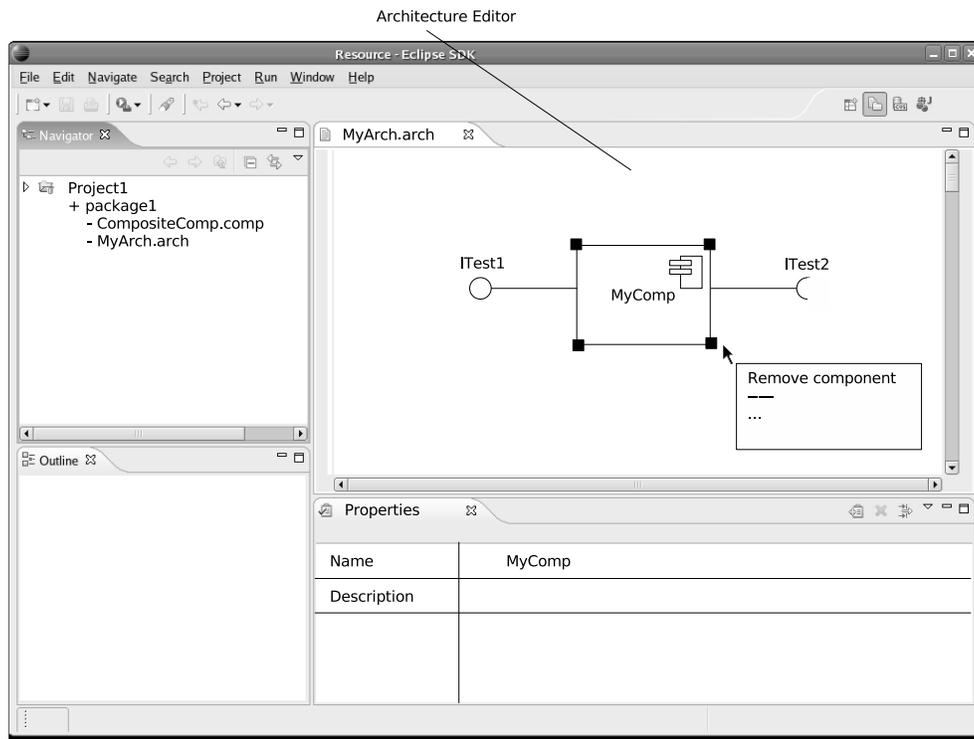


Figura 52: Remover componente ou conector de uma arquitetura

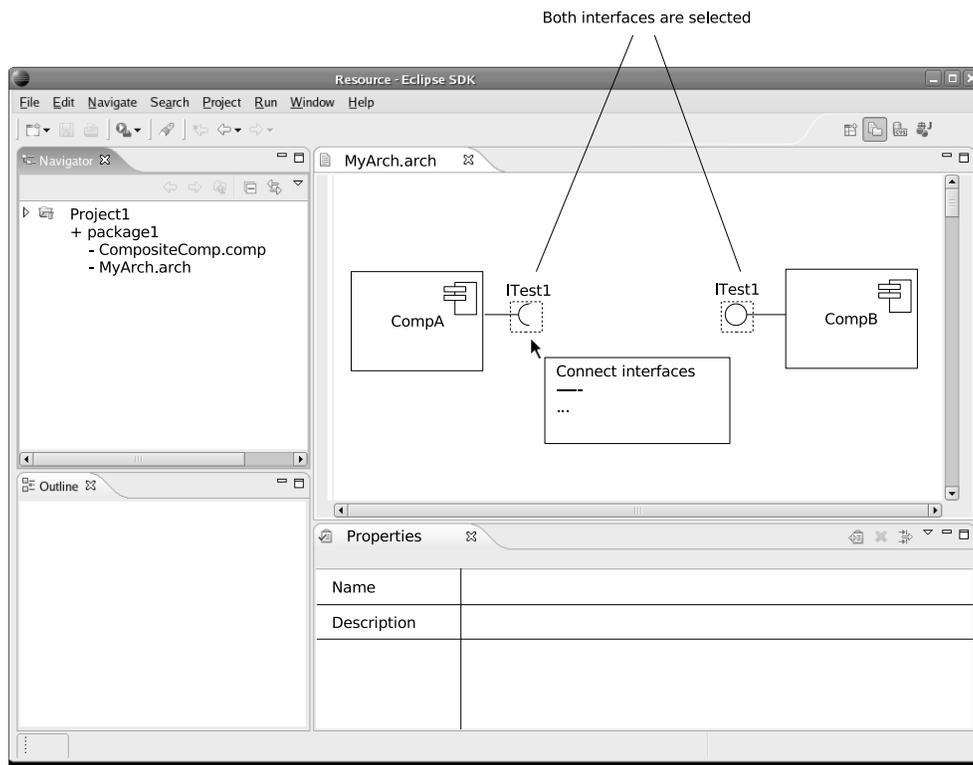


Figura 53: Conectar interface requerida de um elemento à interface provida de outro (1 - menu, conectando componentes)

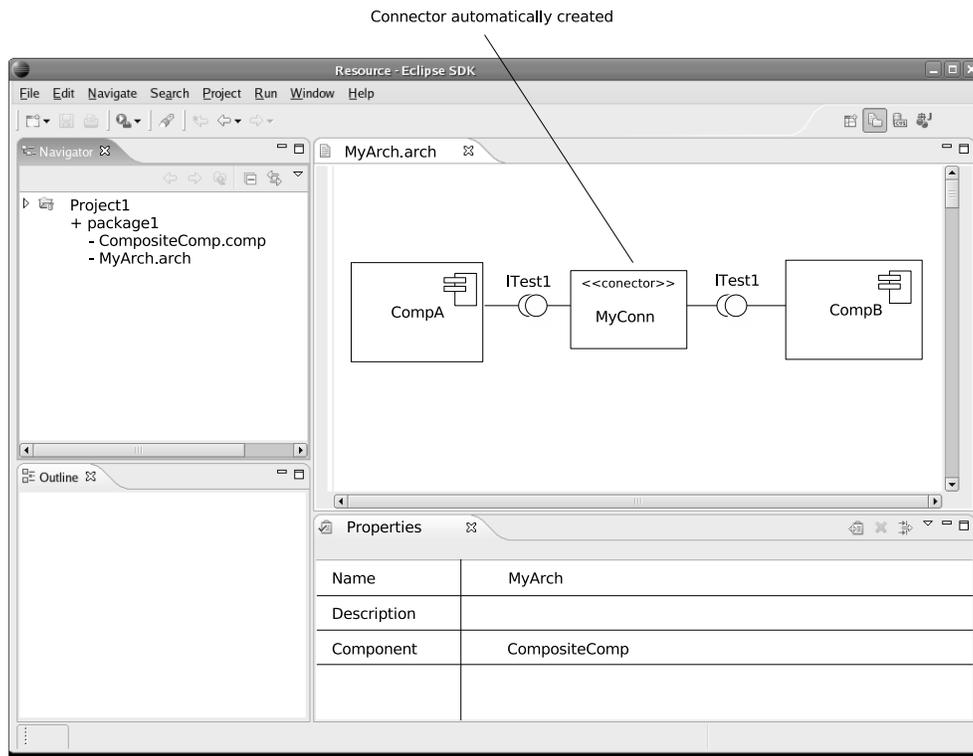


Figura 54: Conectar interface requerida de um elemento à interface provida de outro (2 - conector automaticamente criado)

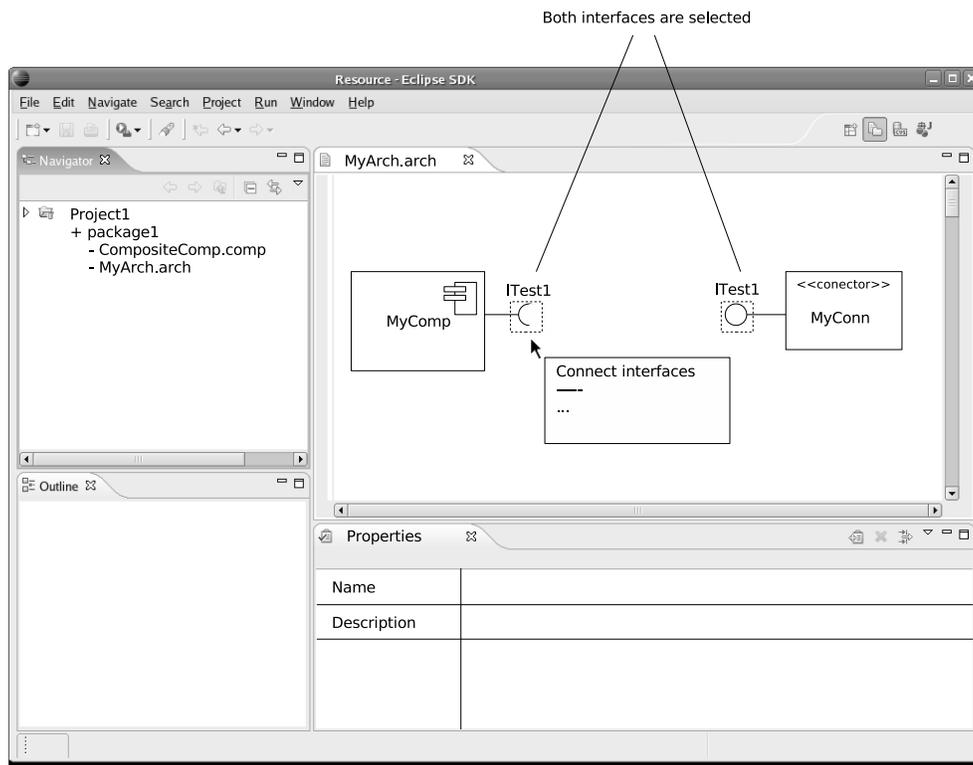


Figura 55: Conectar interface requerida de um elemento à interface provida de outro (1 - menu, conectando componente e conector)

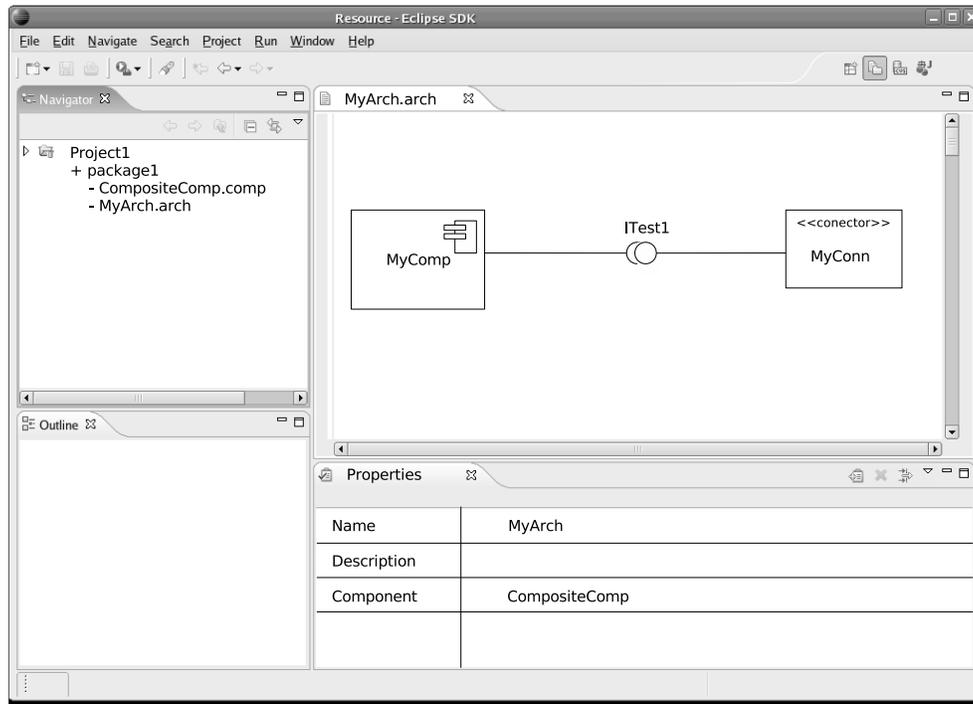


Figura 56: Conectar interface requerida de um elemento à interface provida de outro (2 - componente e conector ligados)

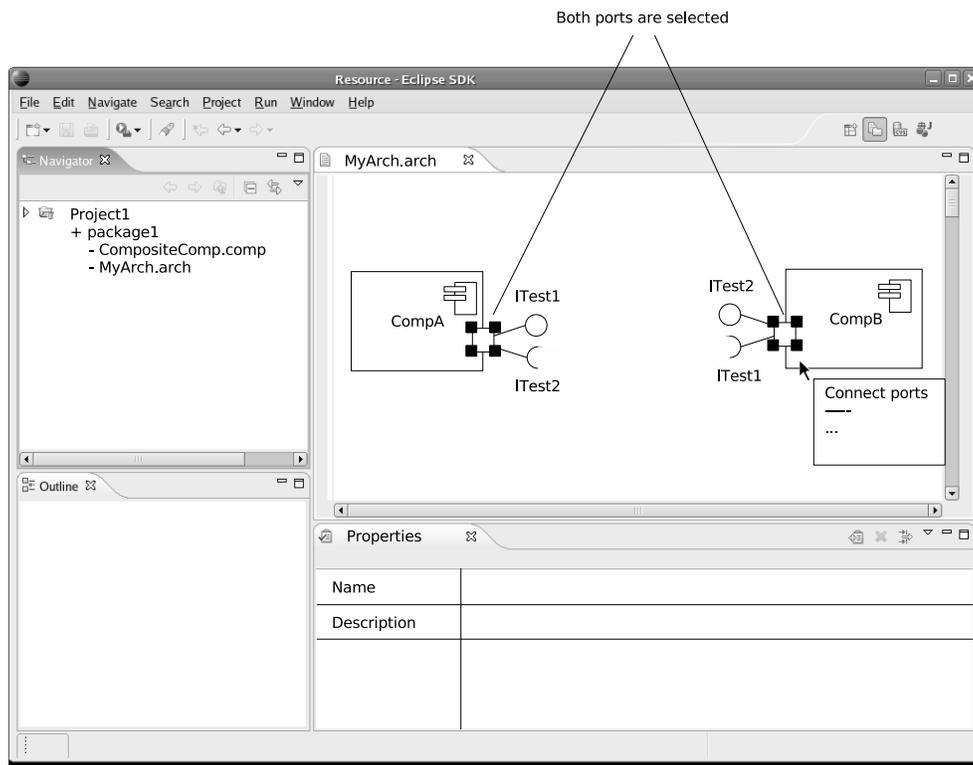


Figura 57: Conectar porta de um elemento à porta de outro (1 - menu, conectando portas de componentes)

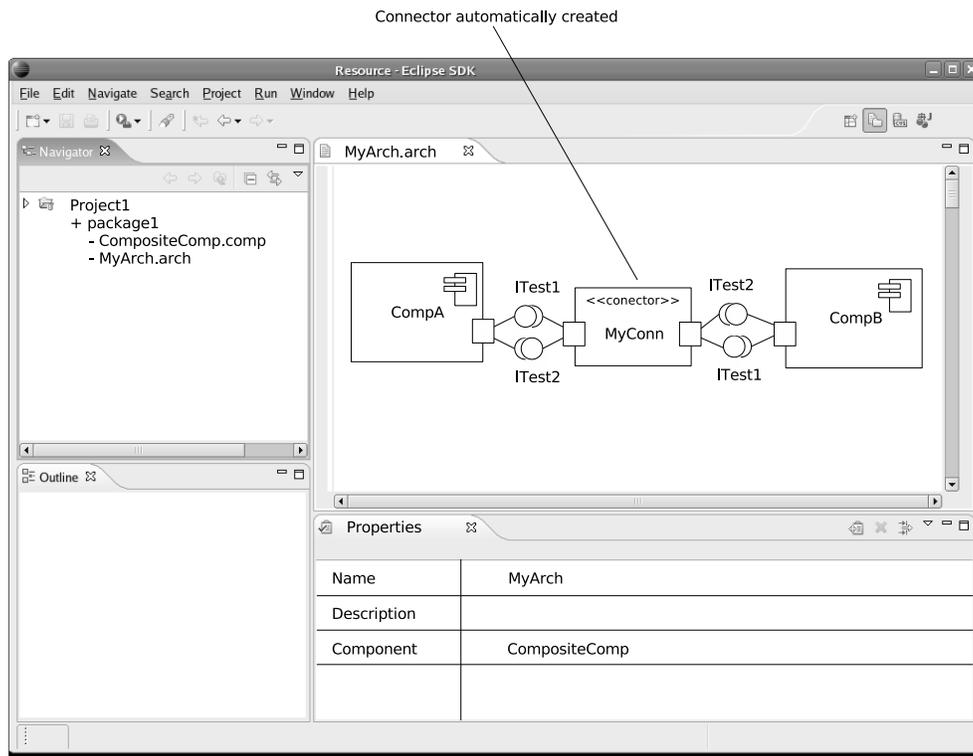


Figura 58: Conectar porta de um elemento à porta de outro (2 - conector automaticamente criado)

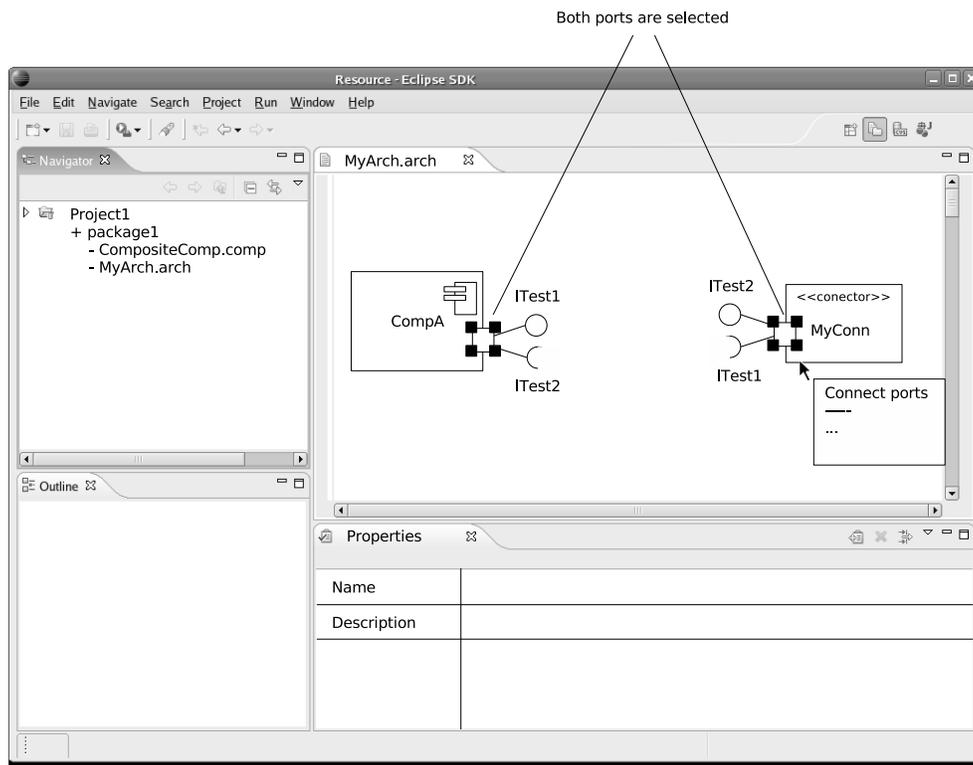


Figura 59: Conectar porta de um elemento à porta de outro (1 - menu, conectando portas de componente e conector)

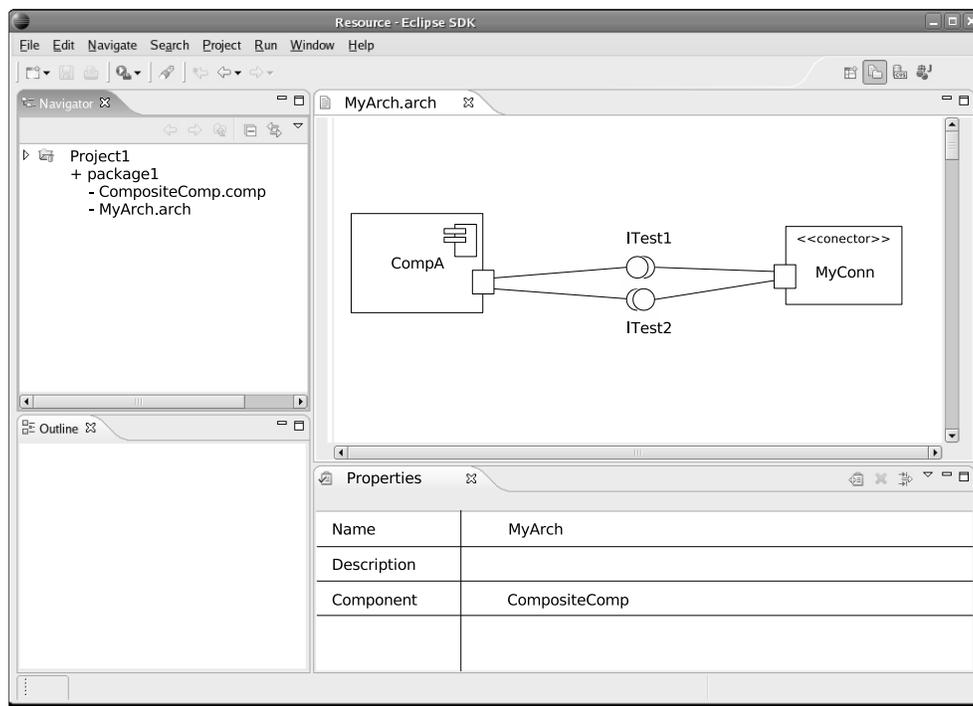


Figura 60: Conectar porta de um elemento à porta de outro (2 - portas conectadas)

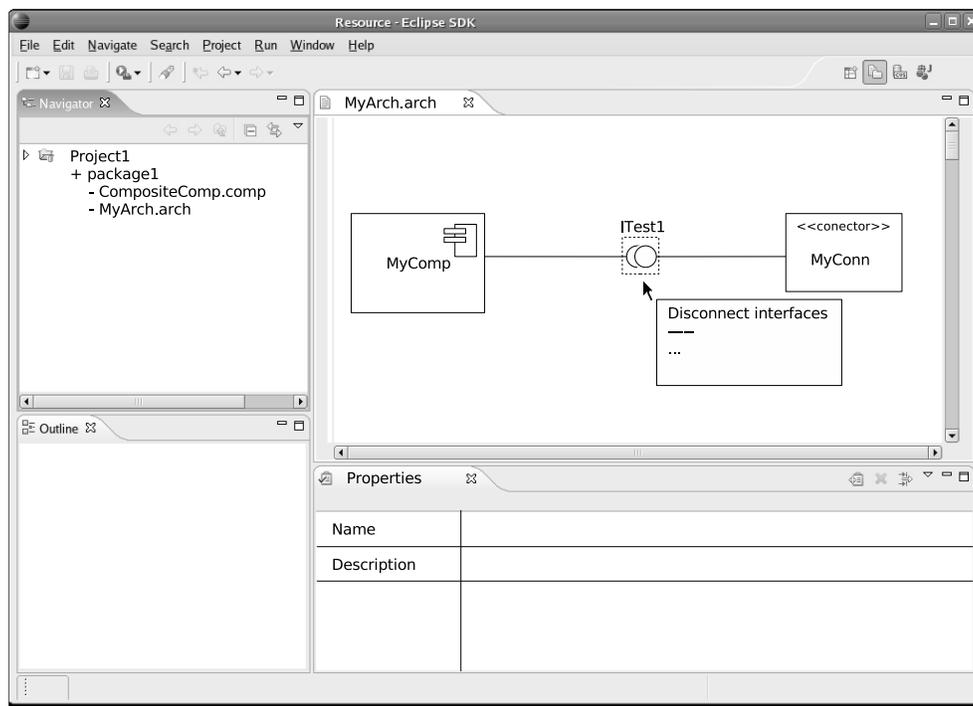


Figura 61: Excluir conexão entre interfaces (1 - menu)

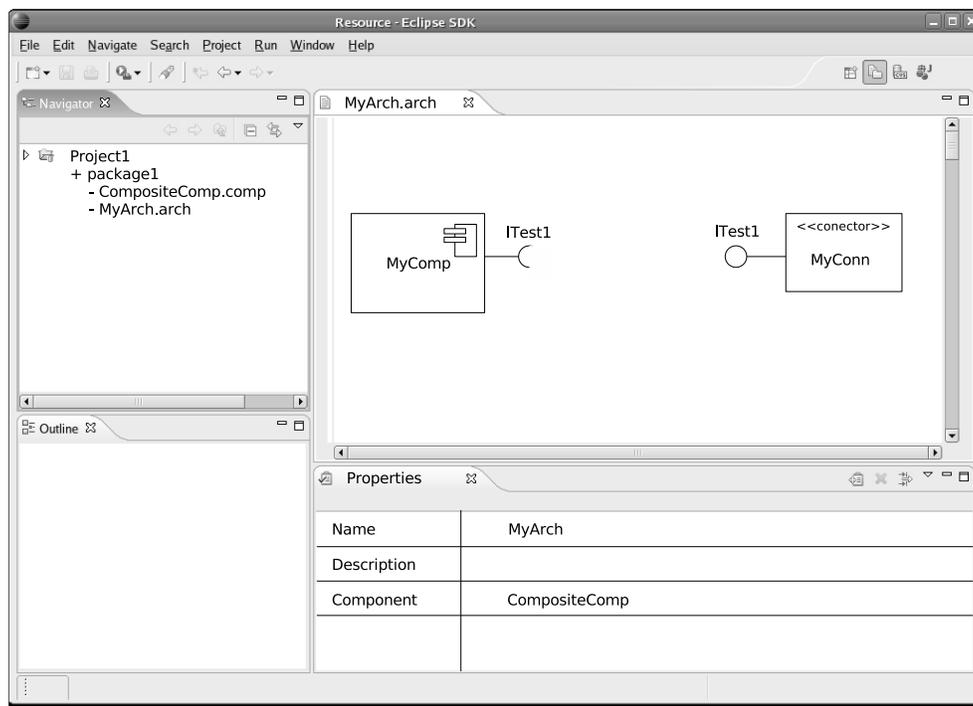


Figura 62: Excluir conexão entre interfaces (2 - interfaces desconectadas)

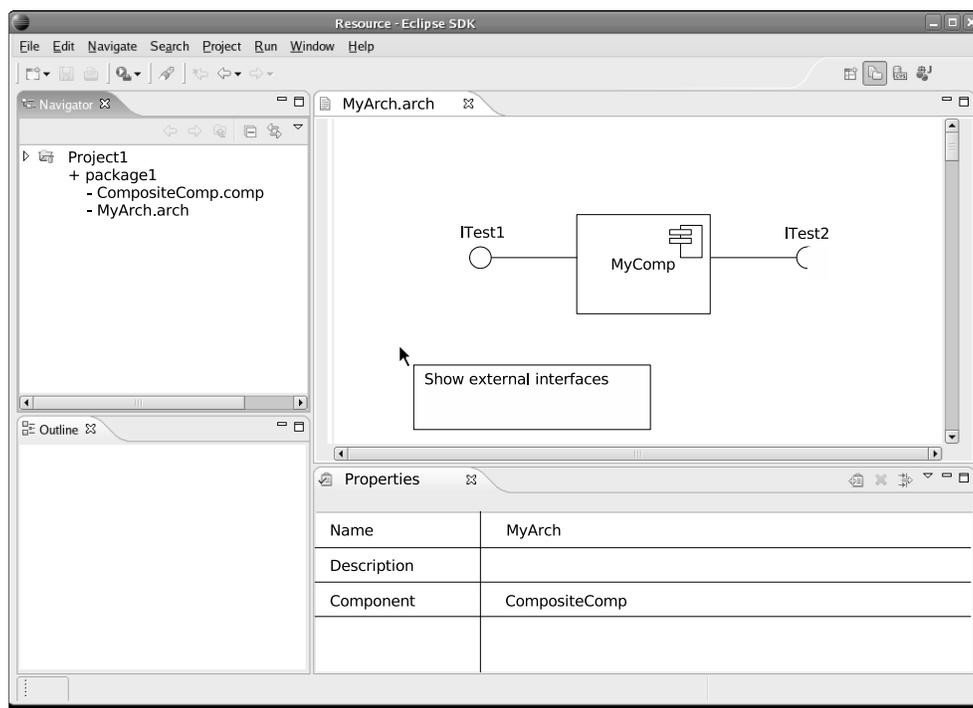


Figura 63: Mapear interface interna a interface externa (1 - menu para exibir interfaces externas)

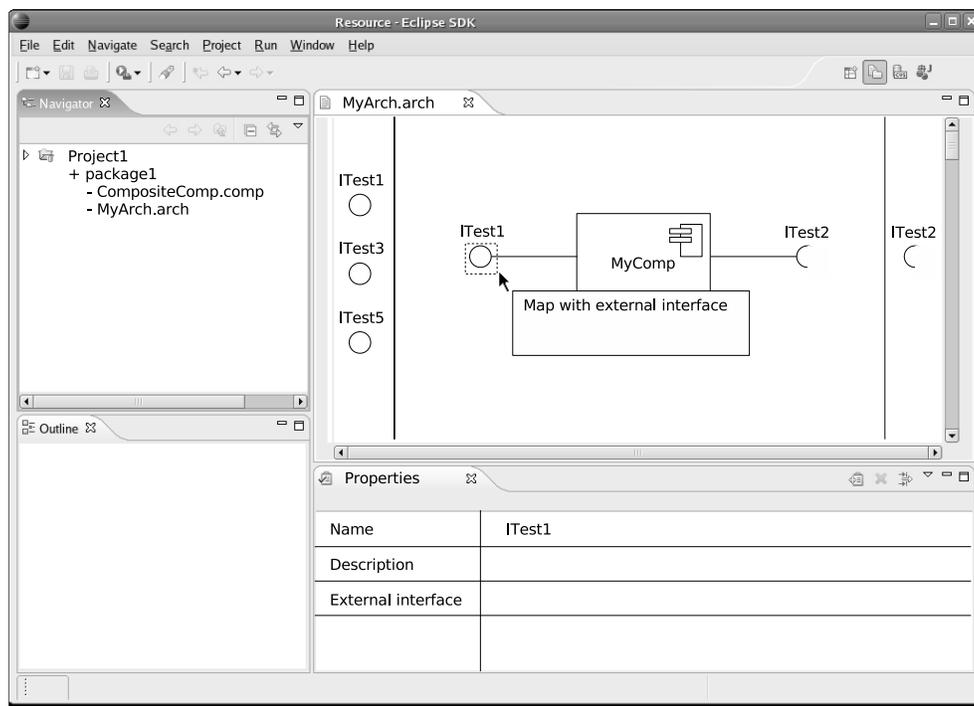


Figura 64: Mapear interface interna a interface externa (2 - menu para mapear interfaces)

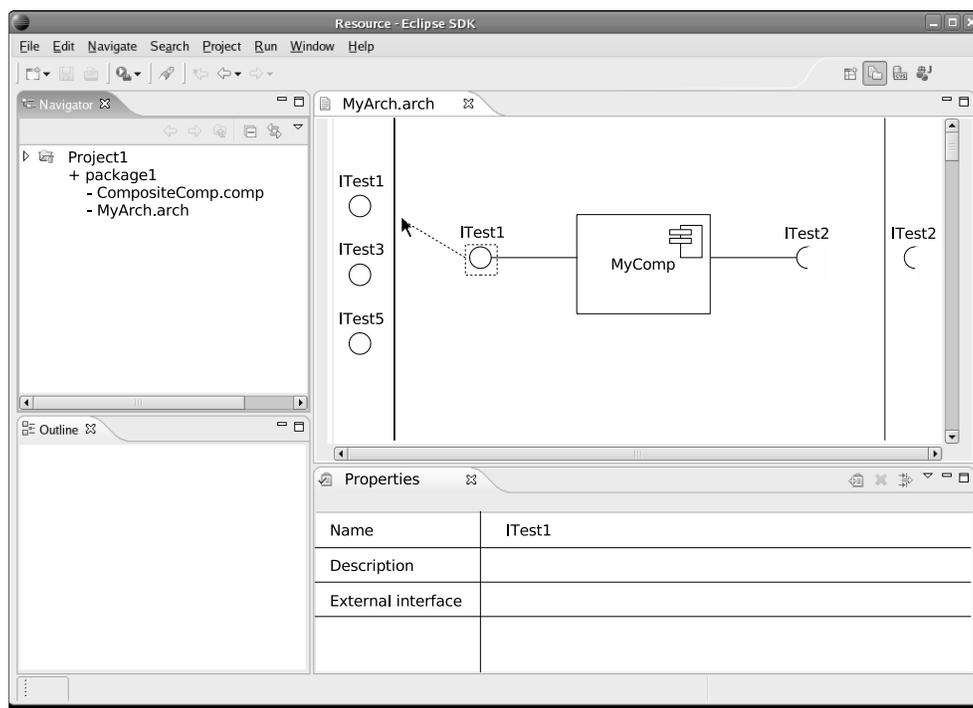


Figura 65: Mapear interface interna a interface externa (3 - arrastando linha para mapear interfaces)

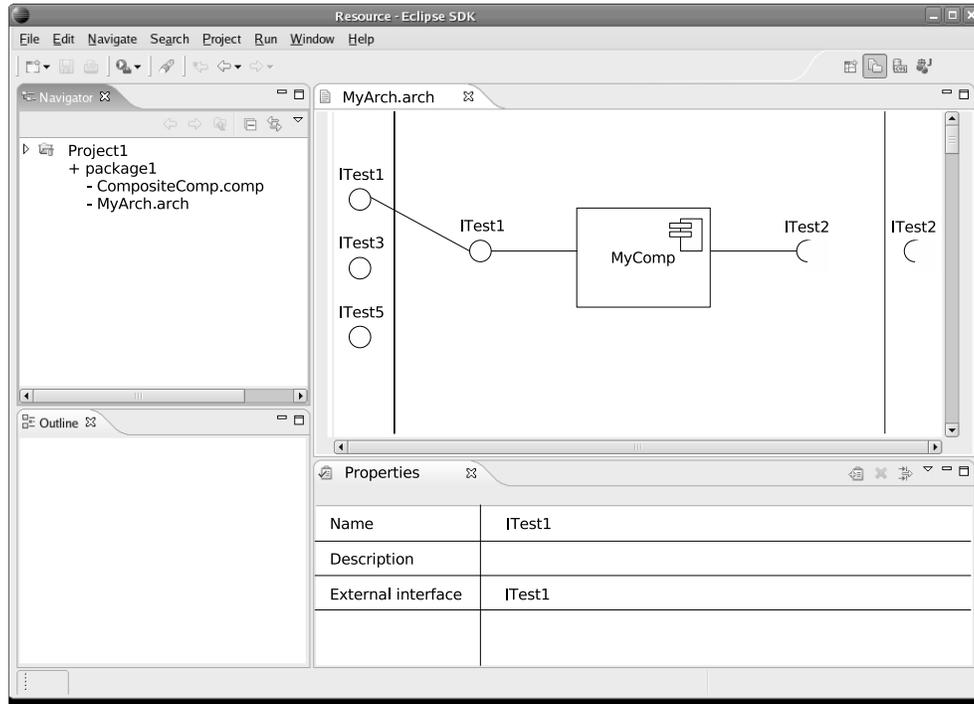


Figura 66: Mapear interface interna a interface externa (4 - interfaces mapeadas)

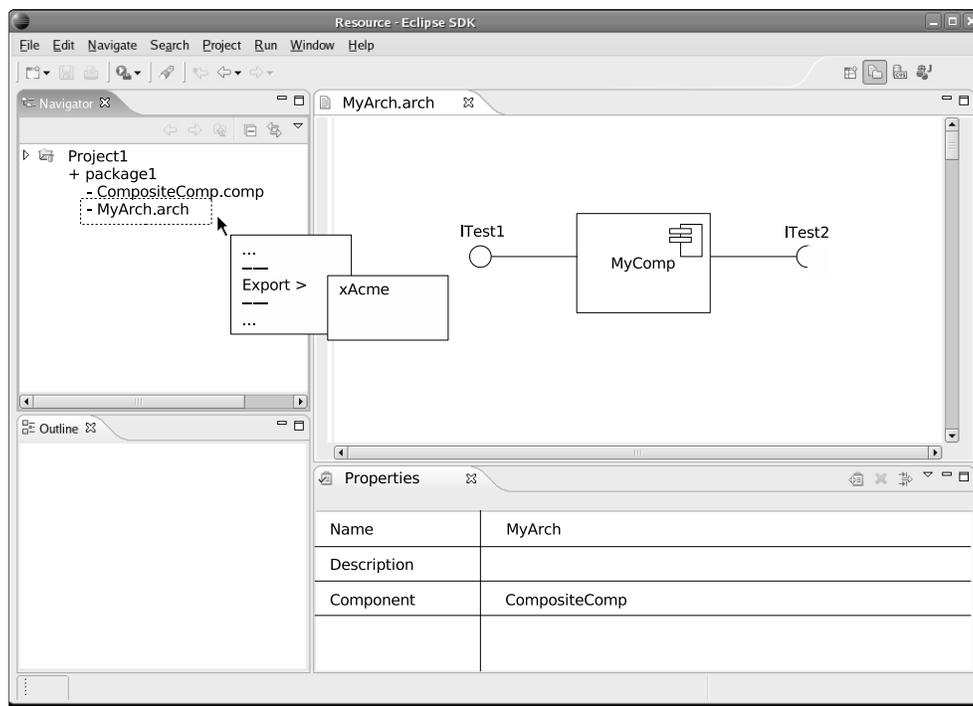


Figura 67: Exportar arquitetura para xAcme (menu)

### 3.4 Implementação de componentes

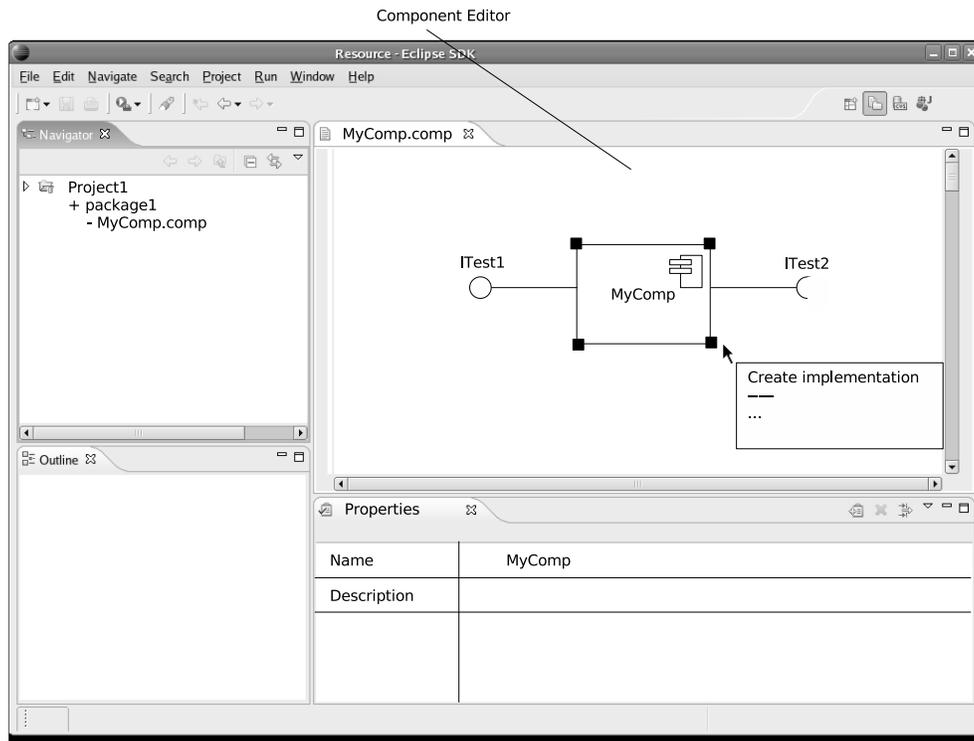


Figura 68: Criar implementação de componente (1 - menu no editor de componentes)

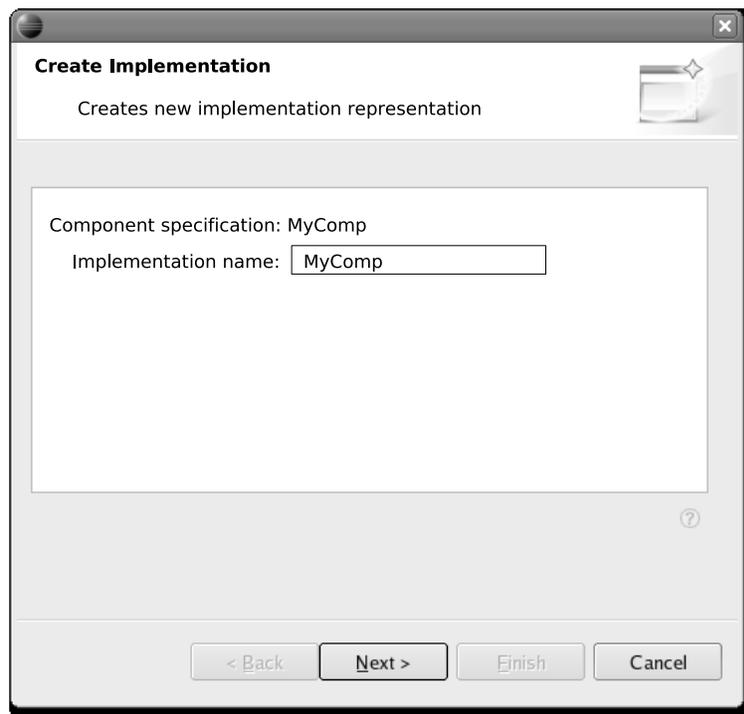


Figura 69: Criar implementação de componente (2 - assistente)

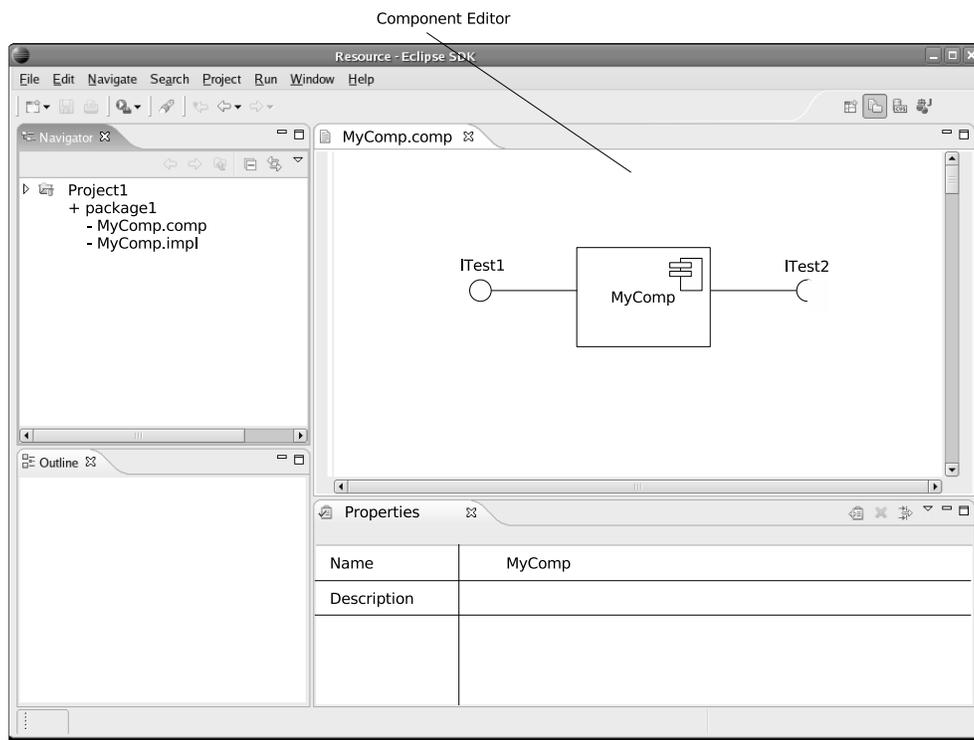


Figura 70: Criar implementação de componente (3 - descritor de implementação criado)

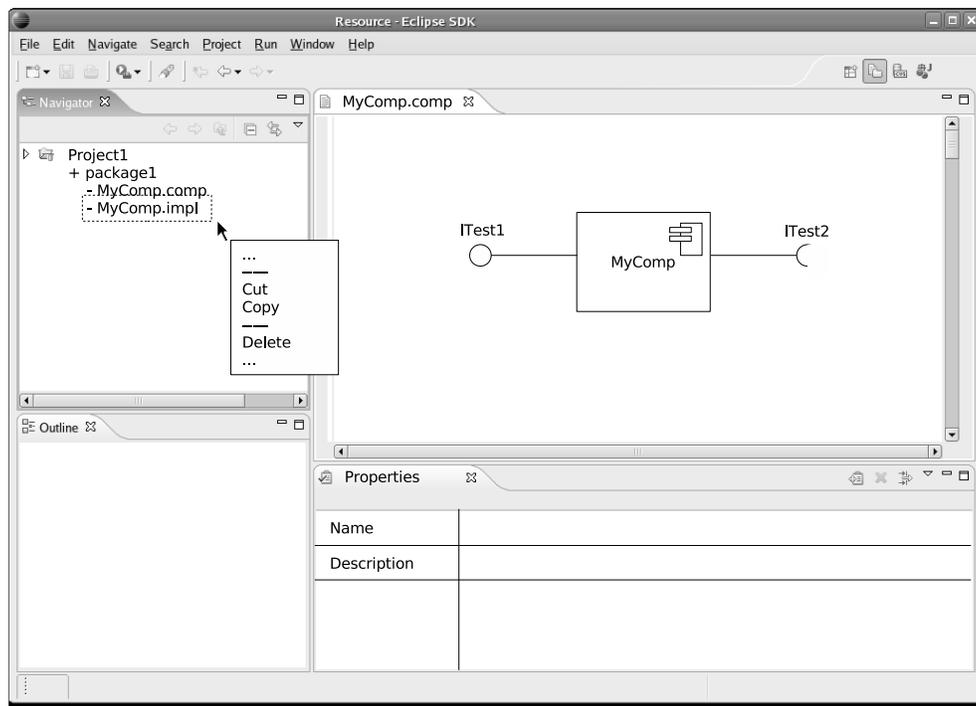


Figura 71: Excluir implementação de componente

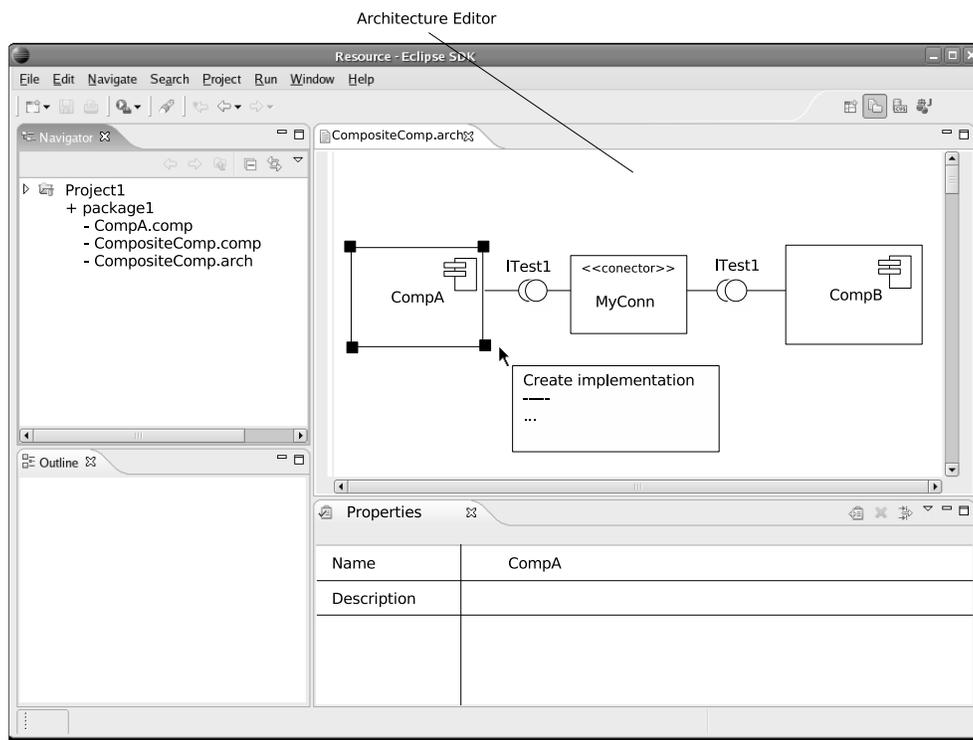


Figura 72: Criar implementação de componente (1 - menu no editor de arquiteturas)

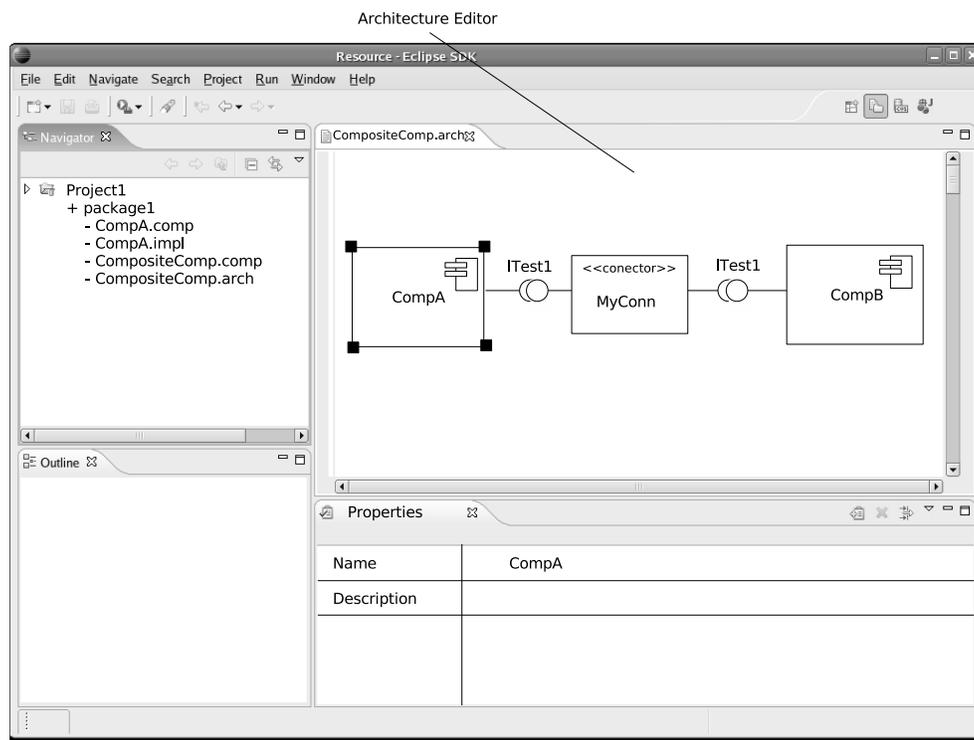


Figura 73: Criar implementação de componente (2 - descritor de implementação criado)

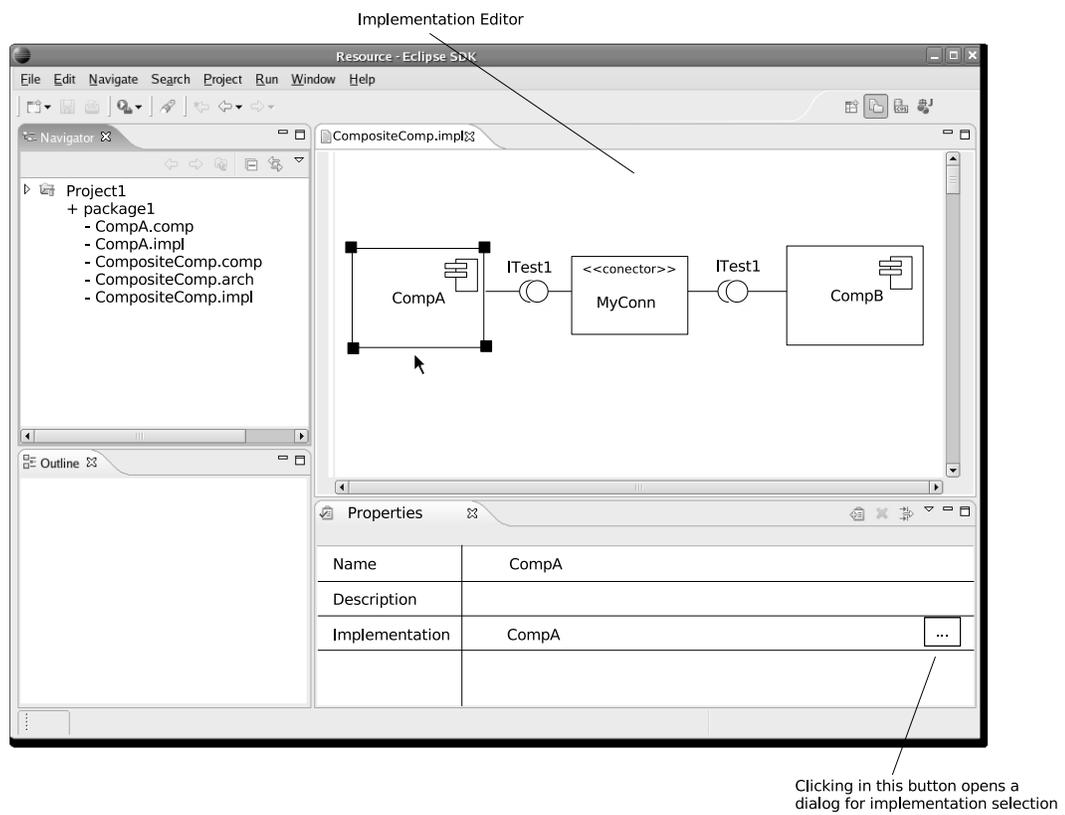


Figura 74: Associar implementação a um componente

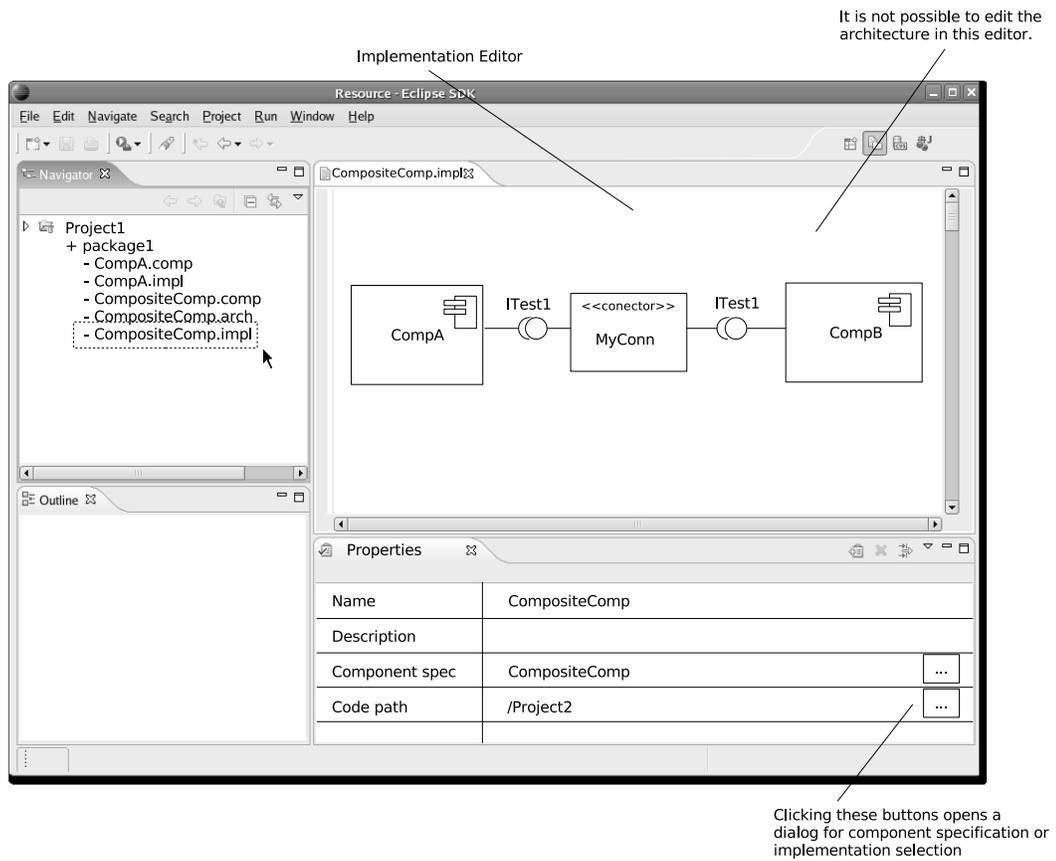


Figura 75: Editar informações da implementação de componente

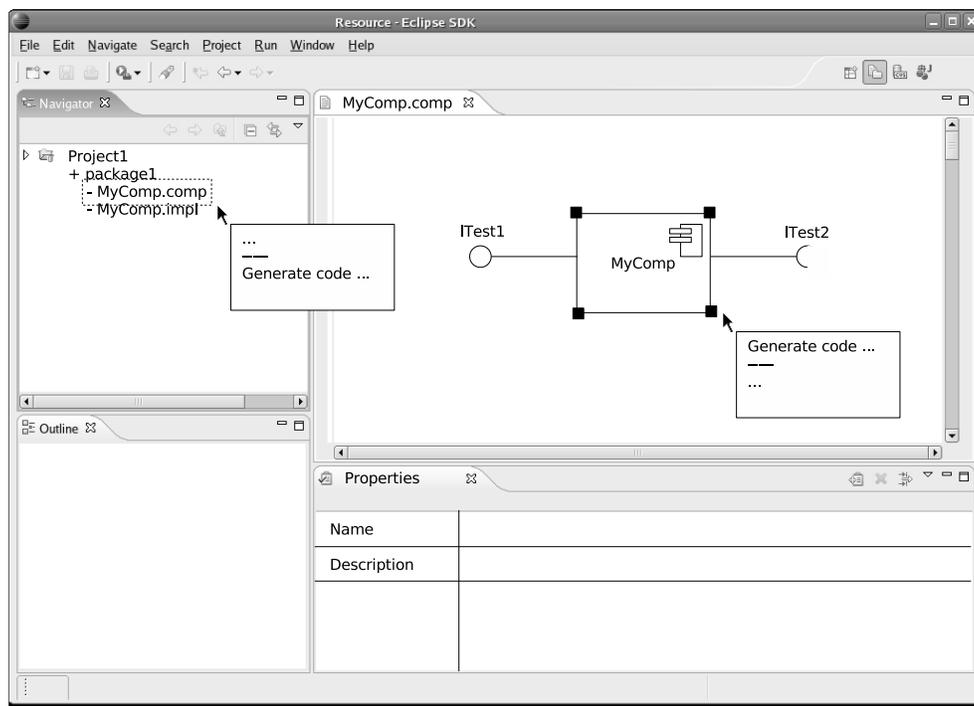


Figura 76: Gerar esqueleto de código (menu na área de trabalho e no editor de componentes)

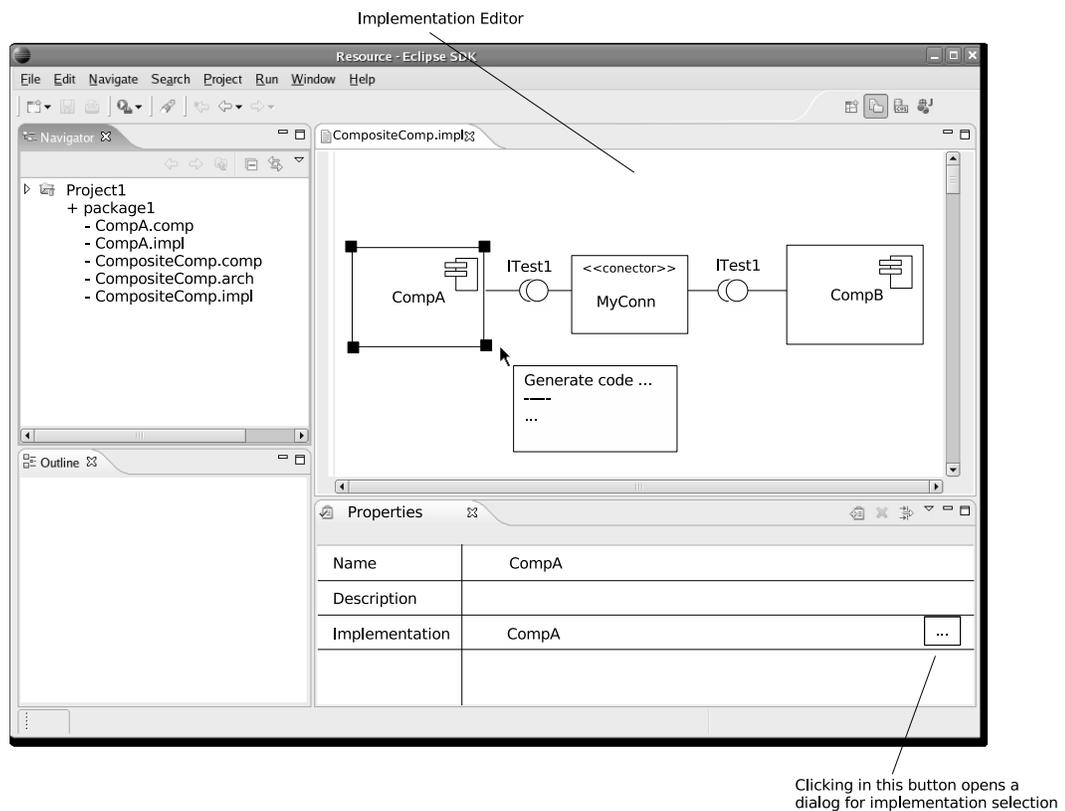


Figura 77: Gerar esqueleto de código (menu no editor de implementação)

### 3.5 Assistentes do processo

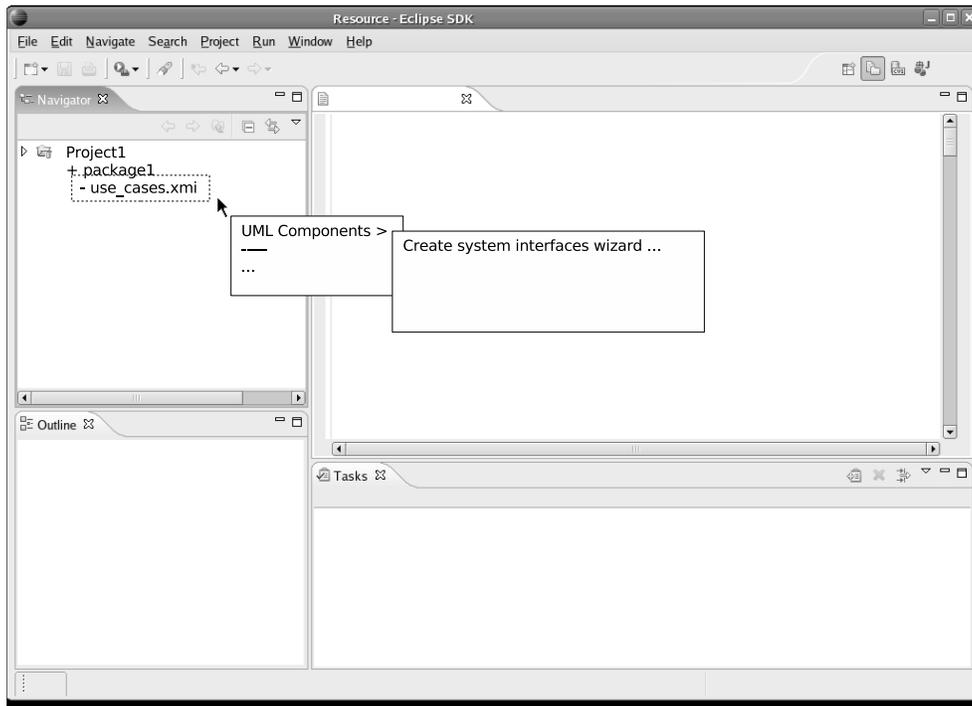


Figura 78: Criar interfaces de sistema (1 - menu para importar XMI de casos de uso)

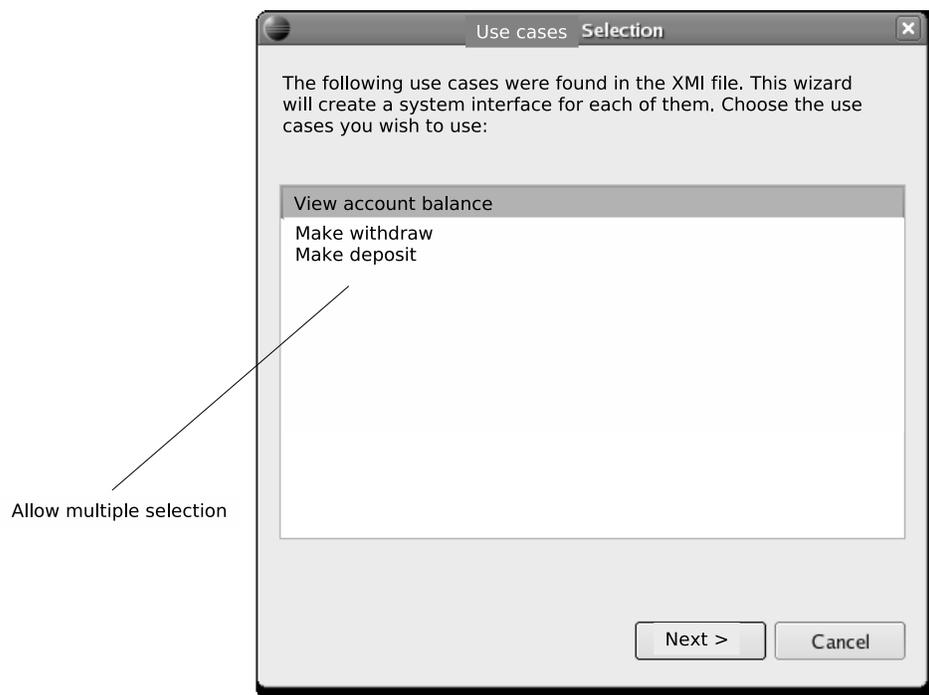


Figura 79: Criar interfaces de sistema (2 - selecionar casos de uso a serem importados)

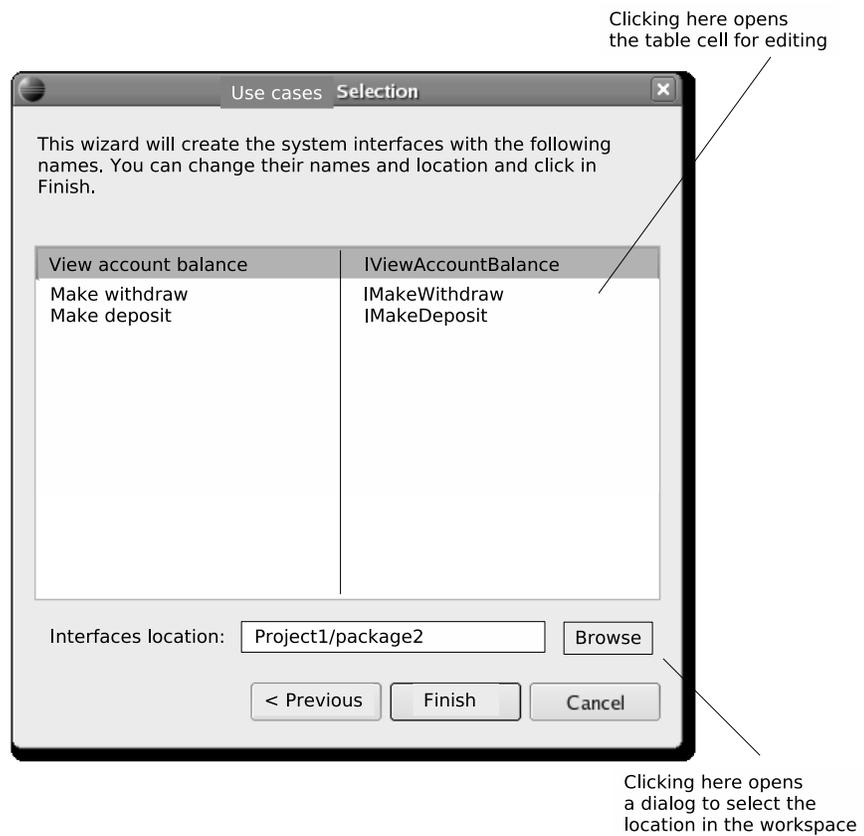


Figura 80: Criar interfaces de sistema (3 - informar nomes das interfaces)

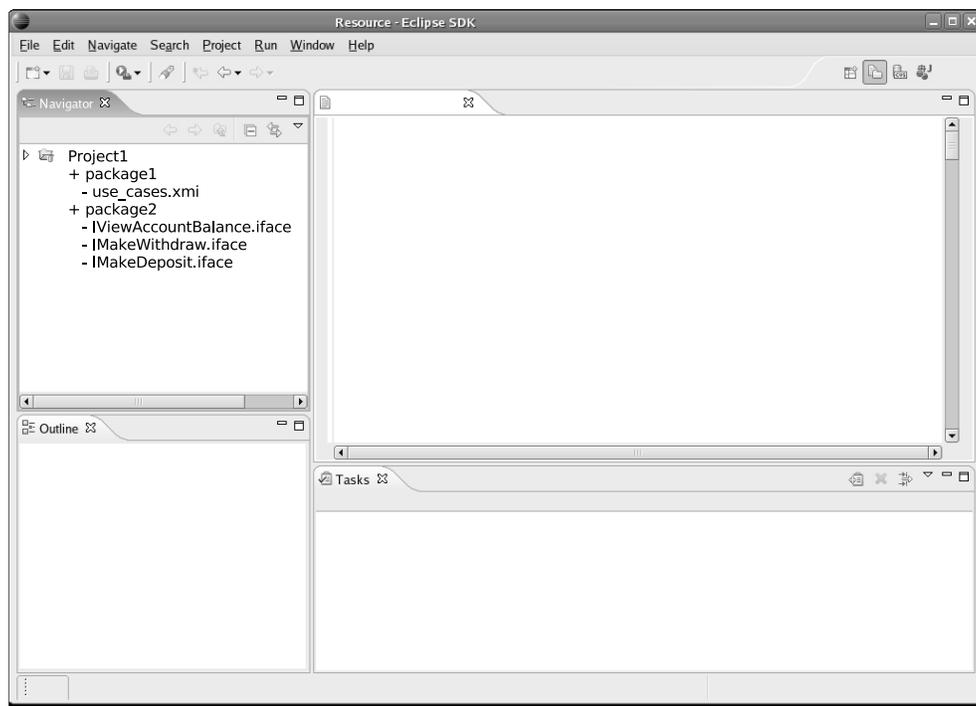


Figura 81: Criar interfaces de sistema (4 - interfaces criadas)

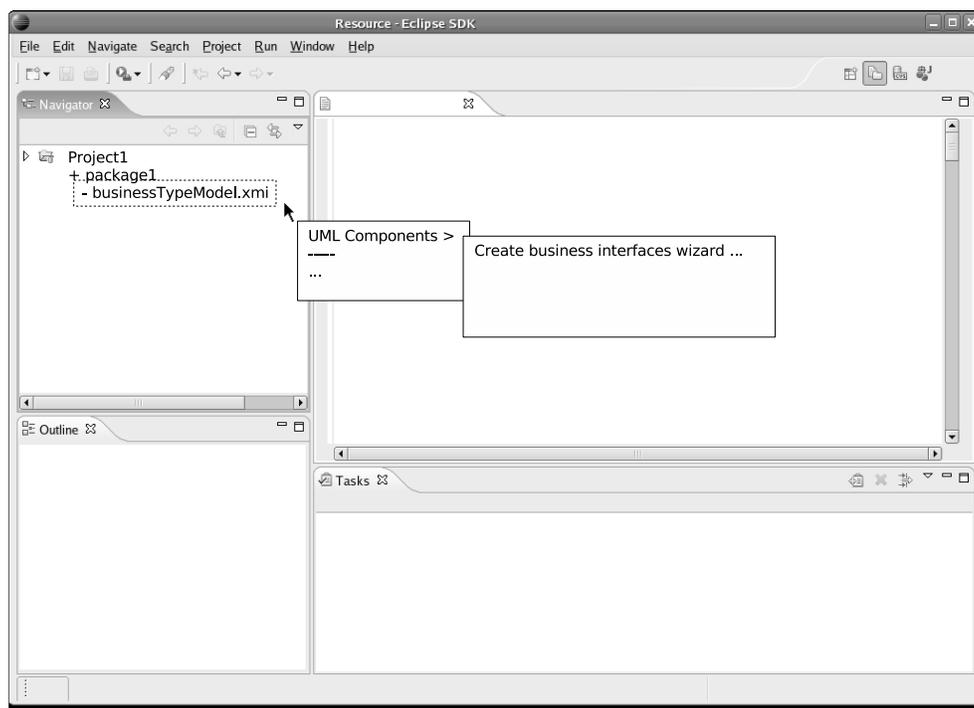


Figura 82: Criar interfaces de negócios (1 - menu para importar XMI de diagrama de classes / *Business Type Model*)

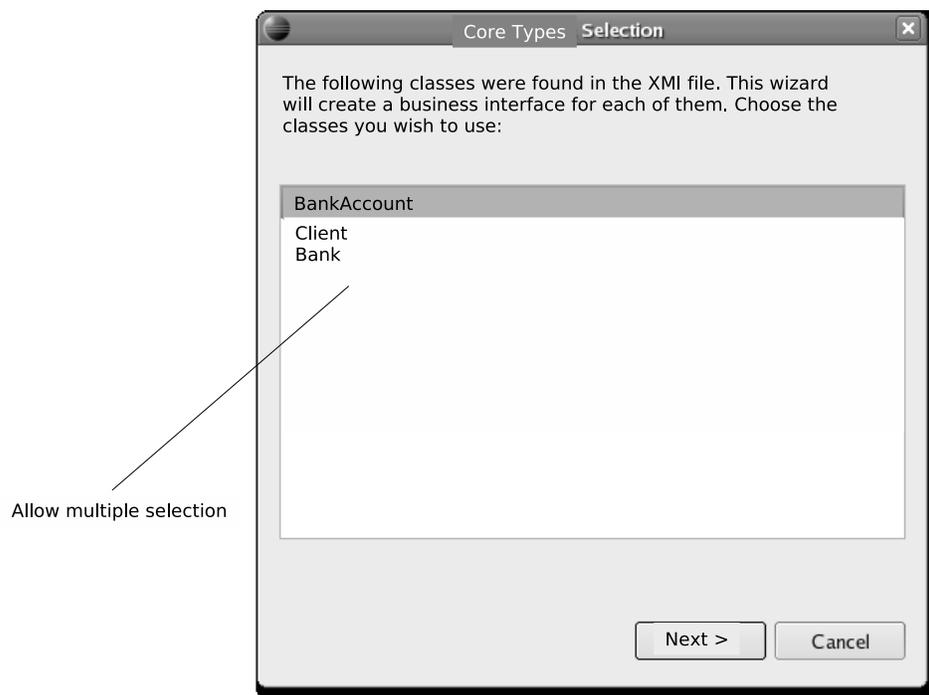


Figura 83: Criar interfaces de negócios (2 - selecionar classes / *core types*)

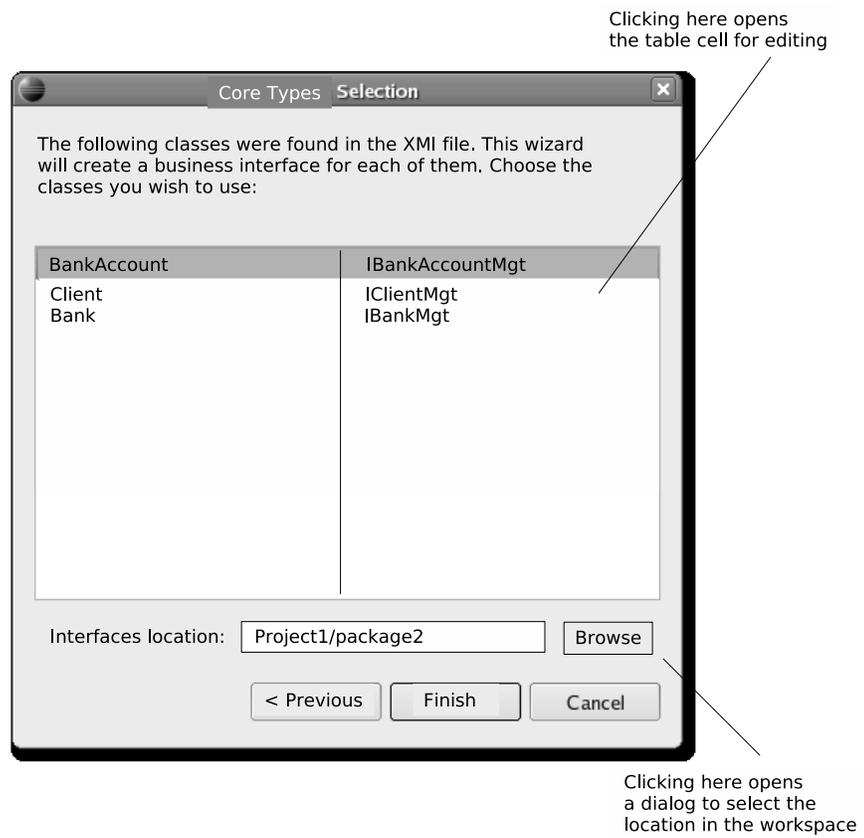


Figura 84: Criar interfaces de negócios (3 - informar nomes das interfaces)

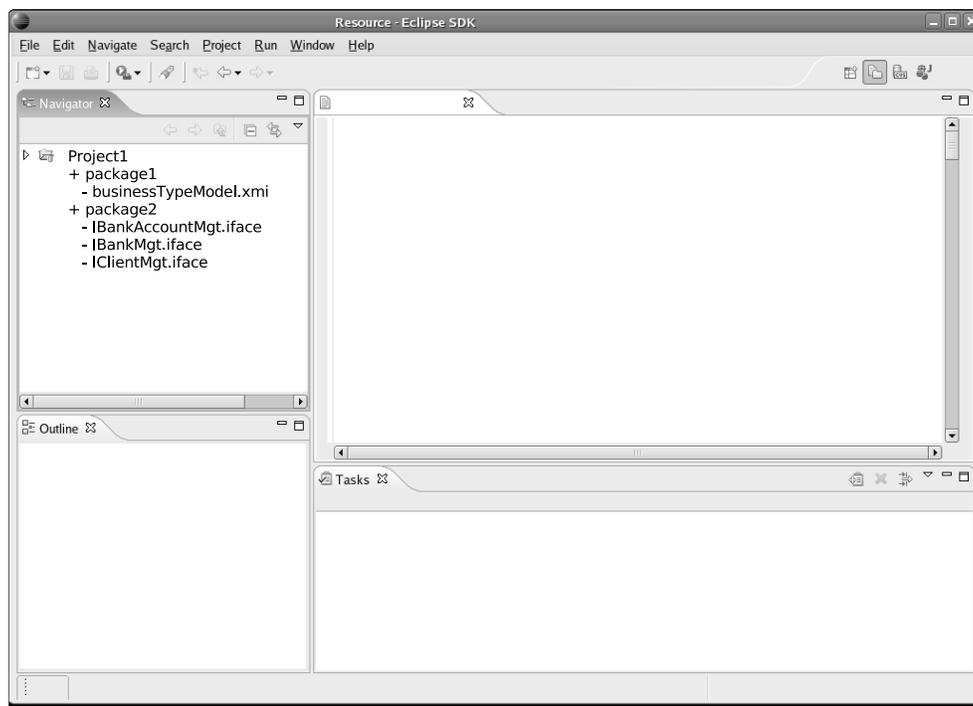


Figura 85: Criar interfaces de negócios (4 - interfaces criadas)

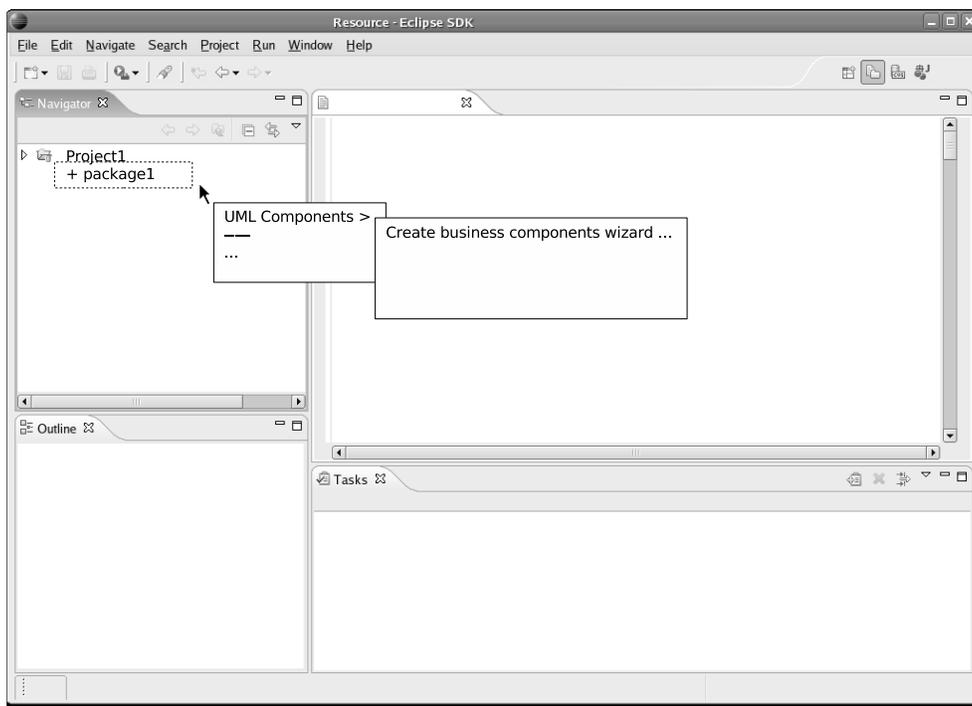


Figura 86: Criar componentes de negócios (1 - menu)

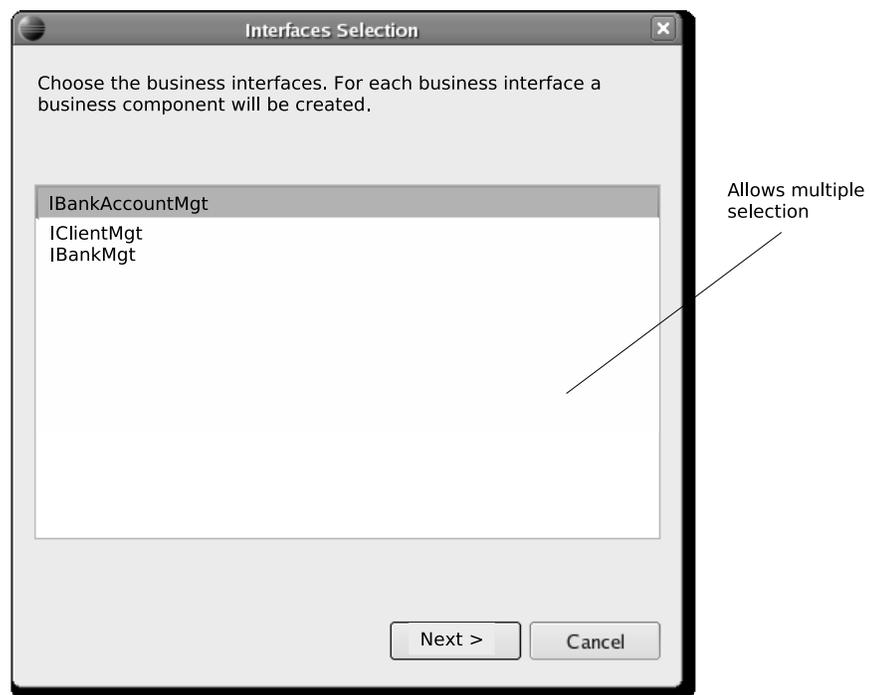


Figura 87: Criar componentes de negócios (2 - selecionar interfaces)

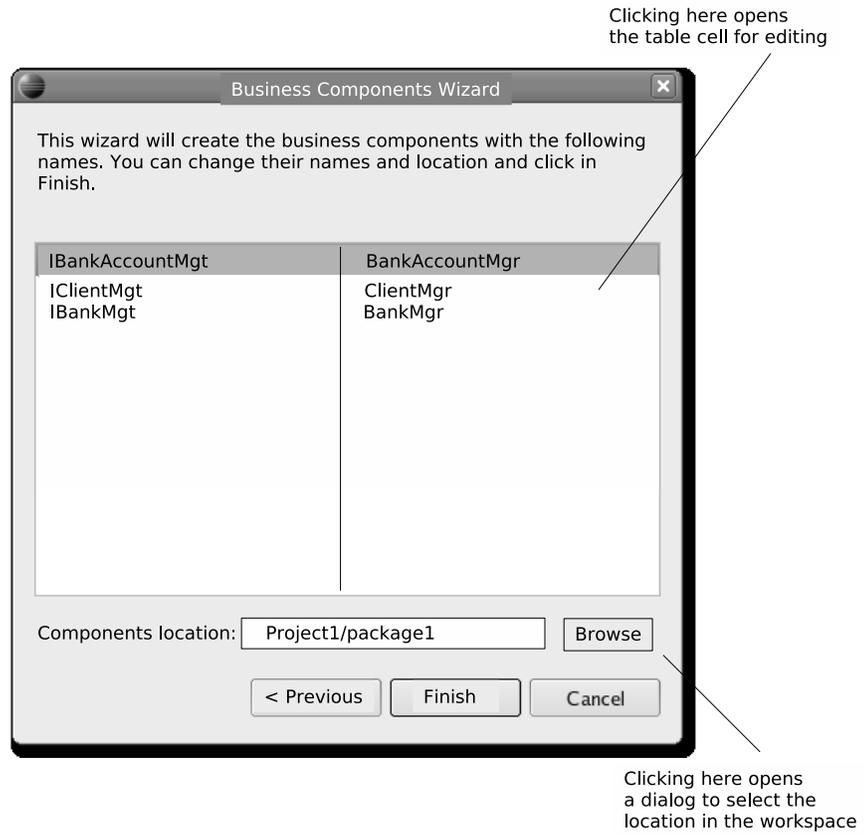


Figura 88: Criar componentes de negócios (3 - informar nomes dos componentes)

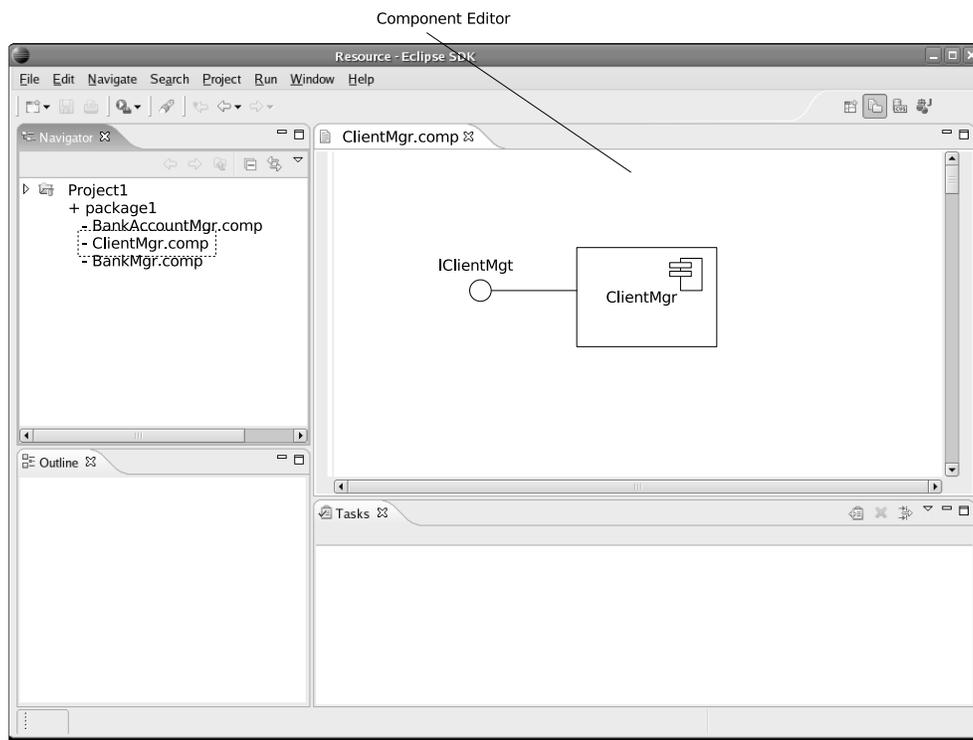


Figura 89: Criar componentes de negócios (4 - componentes criados)

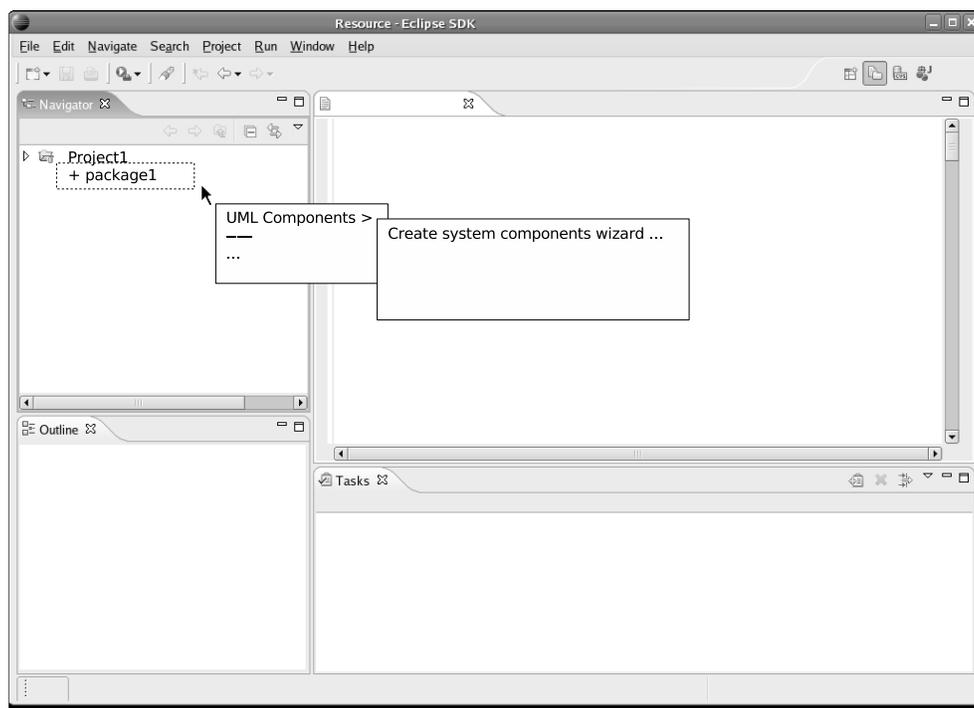


Figura 90: Criar componentes de sistema e Associar interfaces de sistema a componentes (1 - menu)

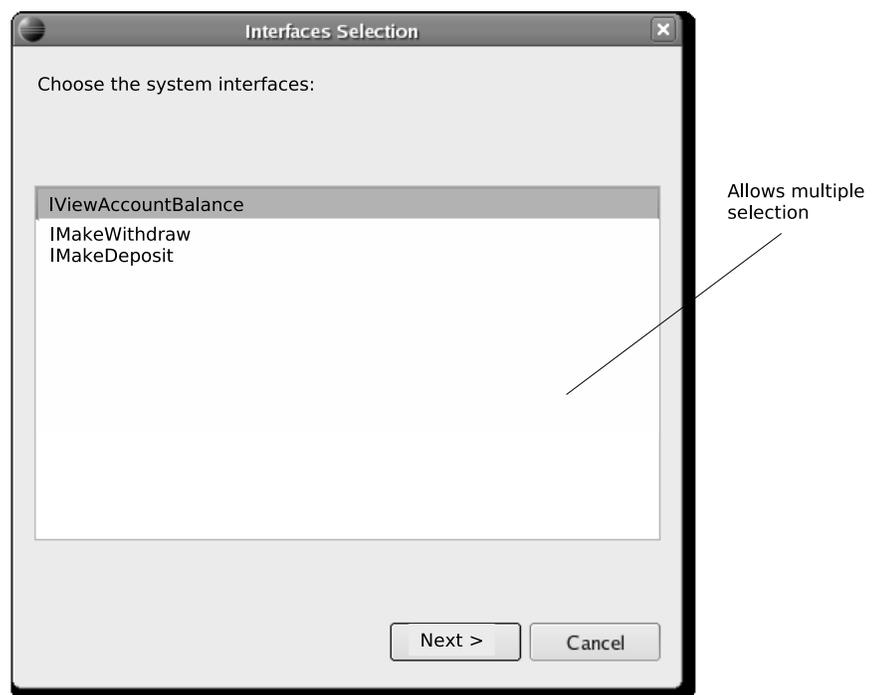


Figura 91: Criar componentes de sistema e Associar interfaces de sistema a componentes (2 - selecionar interfaces, pelo menos uma interface deve ser selecionada)

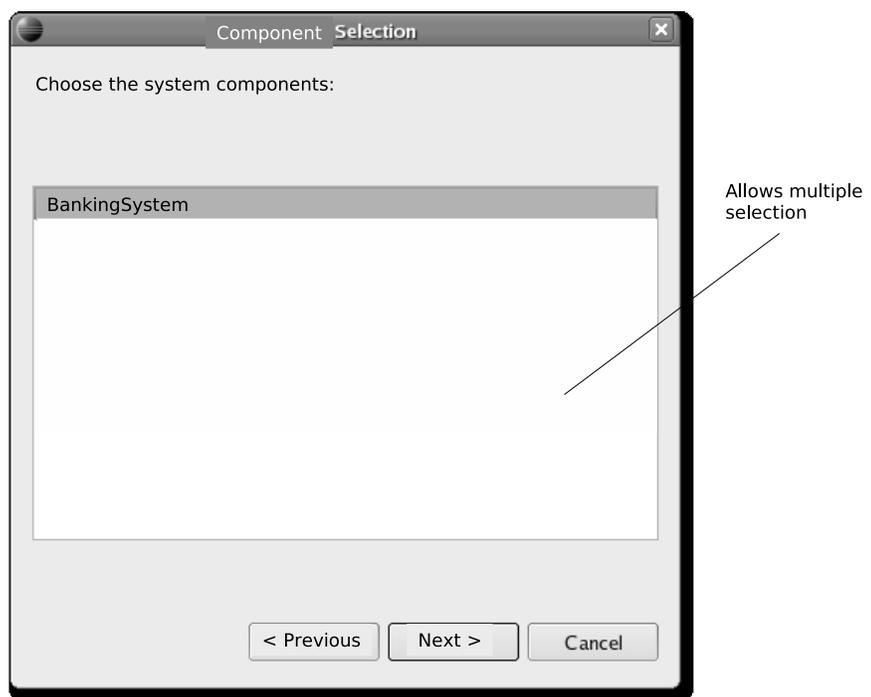


Figura 92: Criar componentes de sistema e Associar interfaces de sistema a componentes (3 - selecionar componentes, opcional)

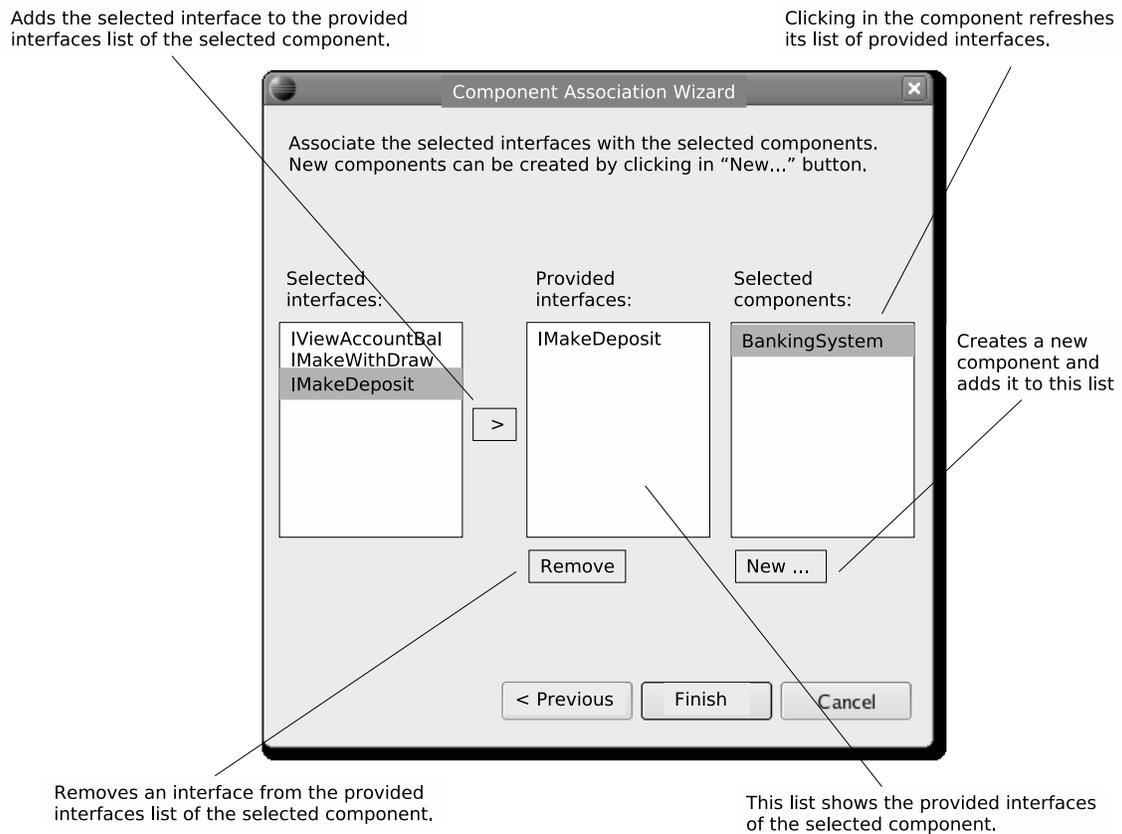


Figura 93: Criar componentes de sistema e Associar interfaces de sistema a componentes (4 - associando as interfaces aos componentes)

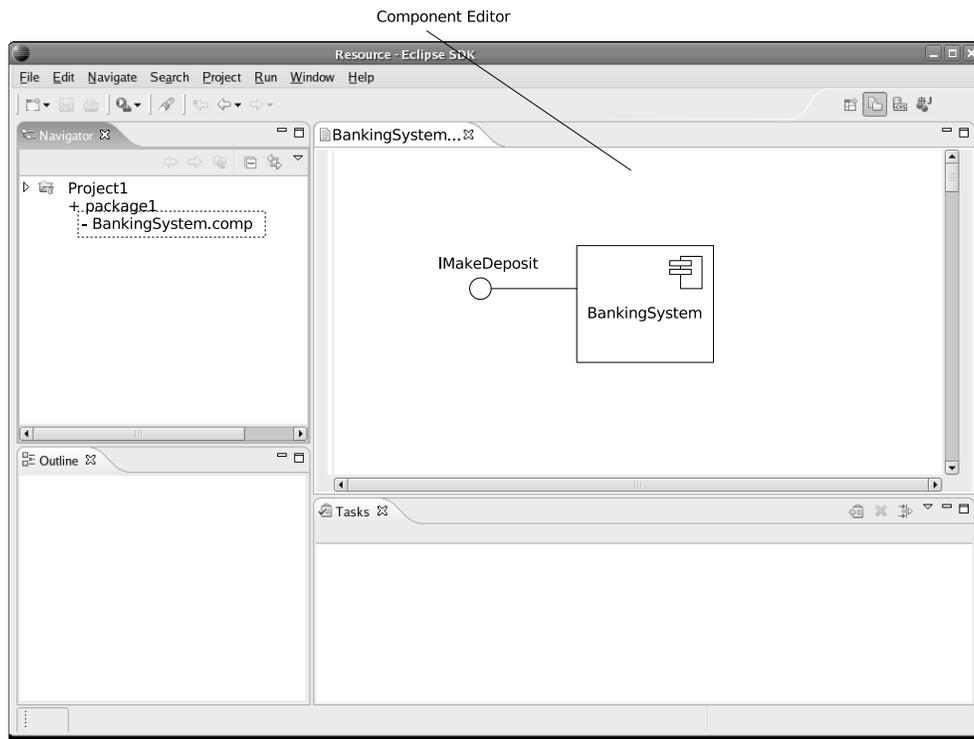


Figura 94: Criar componentes de sistema e Associar interfaces de sistema a componentes (5 - componentes criados e/ou associados)

### 3.6 Uso do repositório de componentes

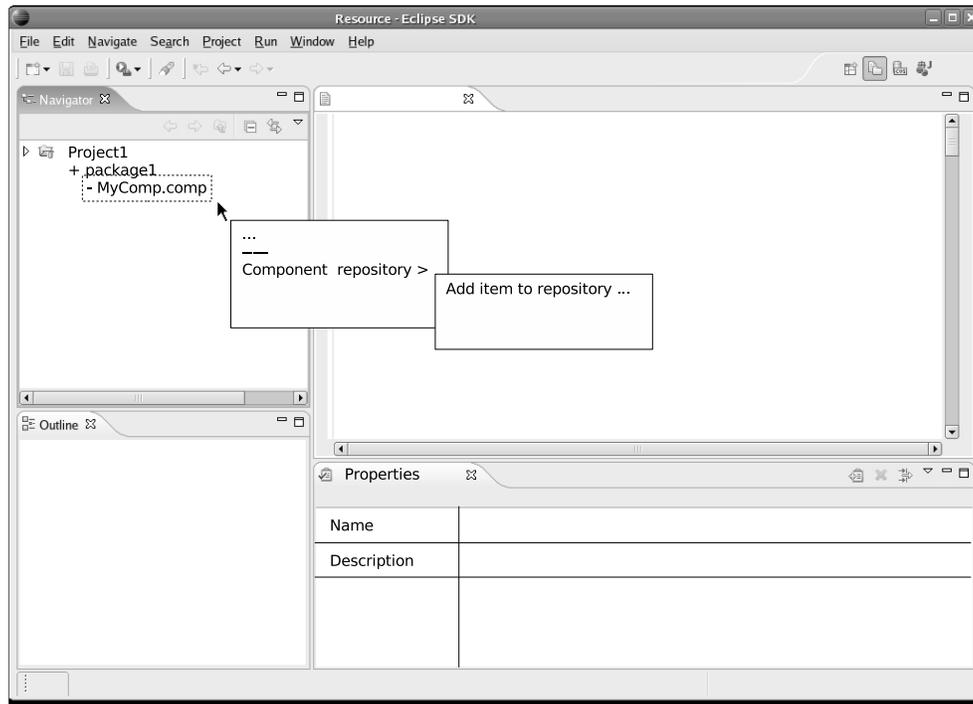


Figura 95: Depositar um artefato novo (1 - menu)

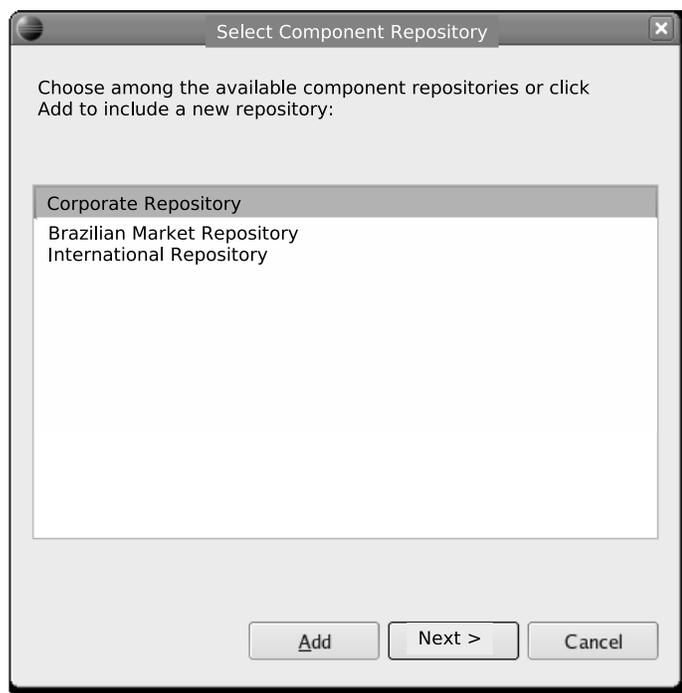
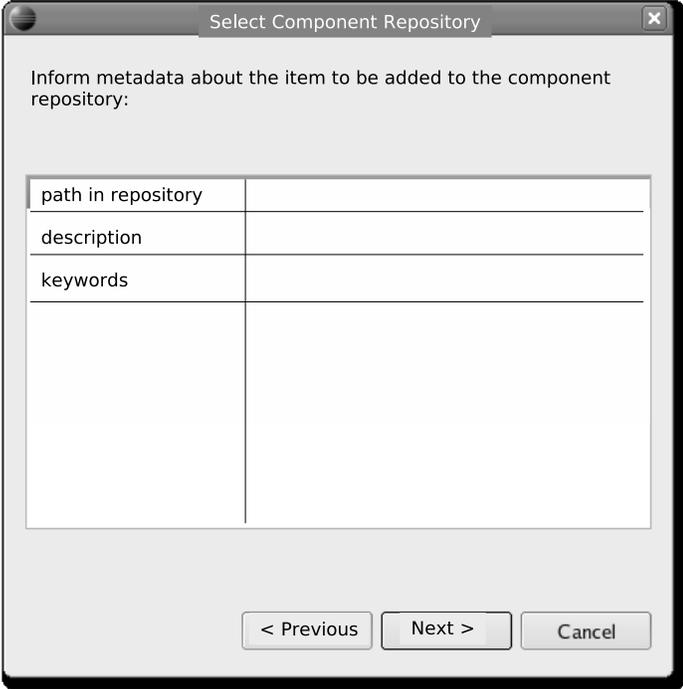


Figura 96: Depositar um artefato novo (2 - selecionar repositório)



Select Component Repository

Inform metadata about the item to be added to the component repository:

path in repository	
description	
keywords	

< Previous    Next >    Cancel

Figura 97: Depositar um artefato novo (3 - inserir meta-informação)

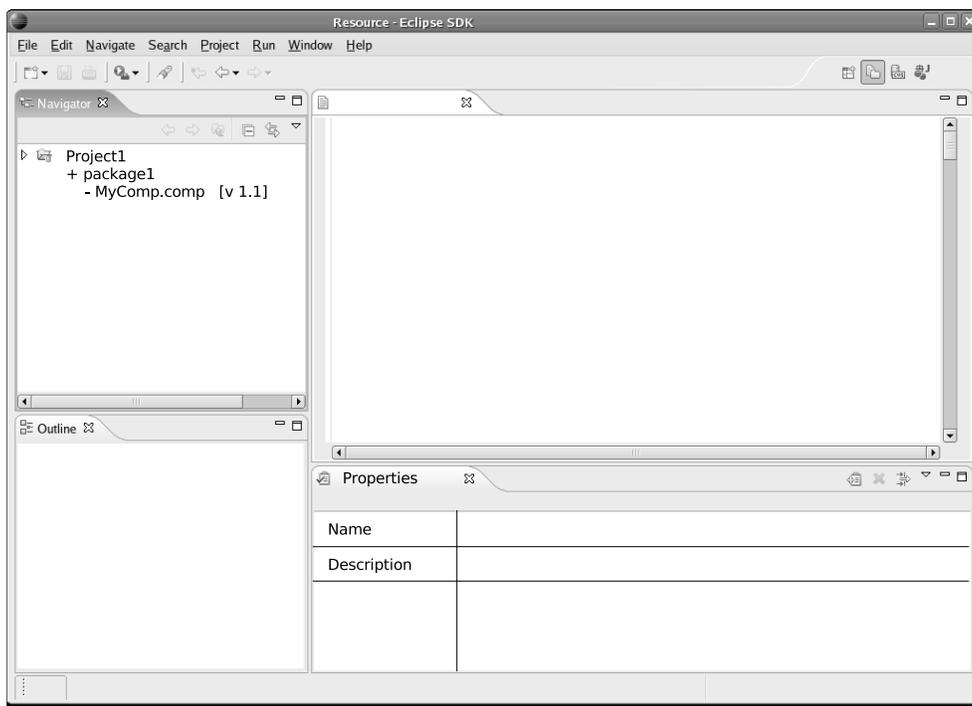


Figura 98: Depositar um artefato novo (4 - artefato inserido)

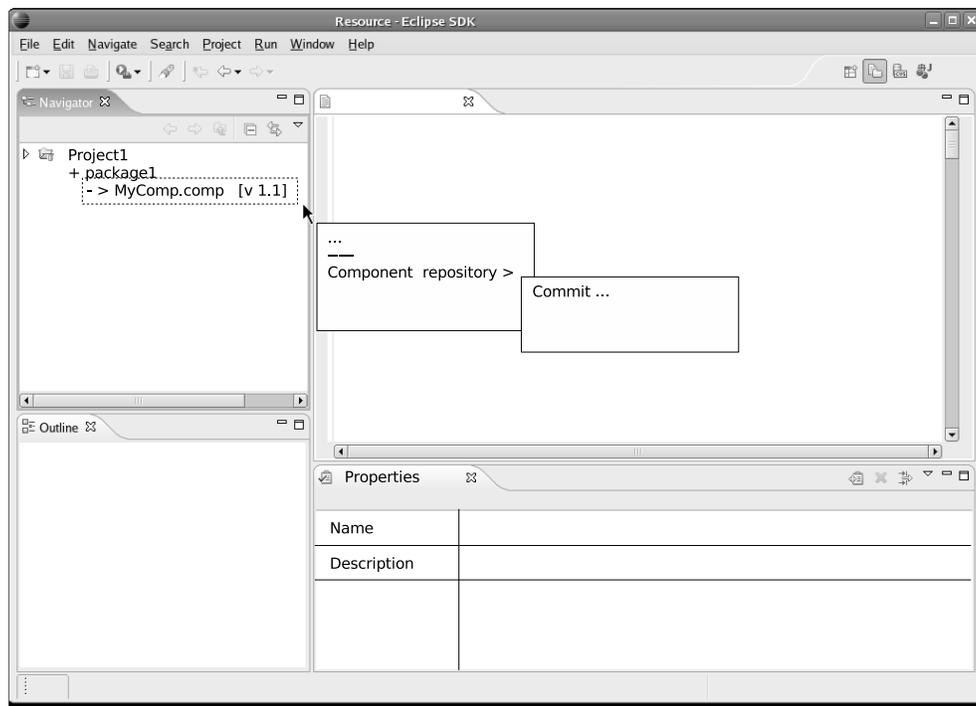


Figura 99: Depositar uma nova versão de um artefato

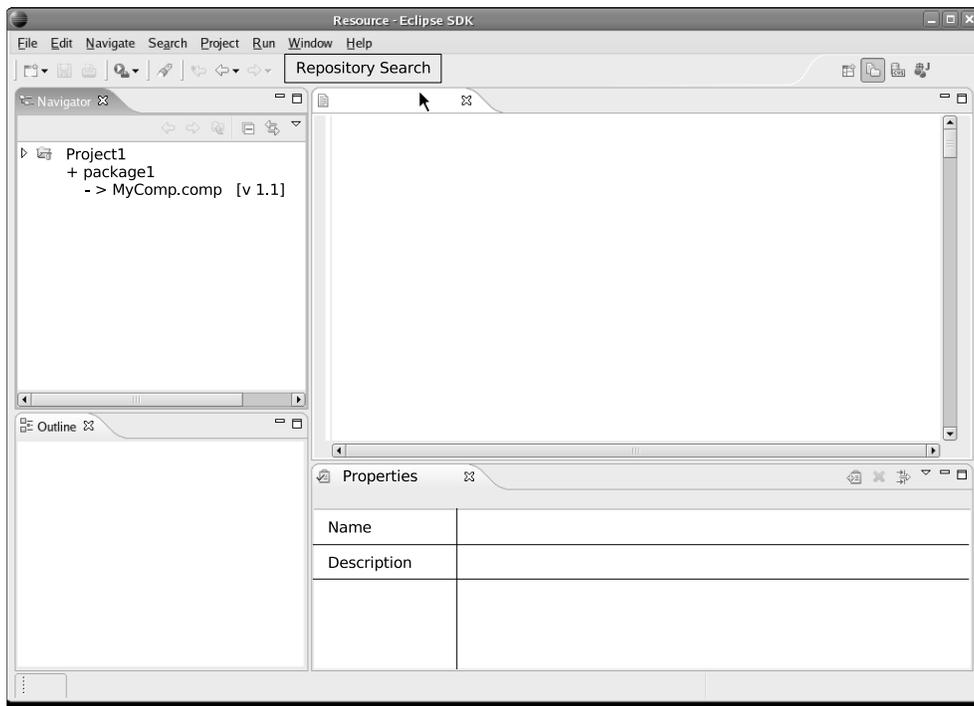


Figura 100: Pesquisar por artefato (1 - botão)

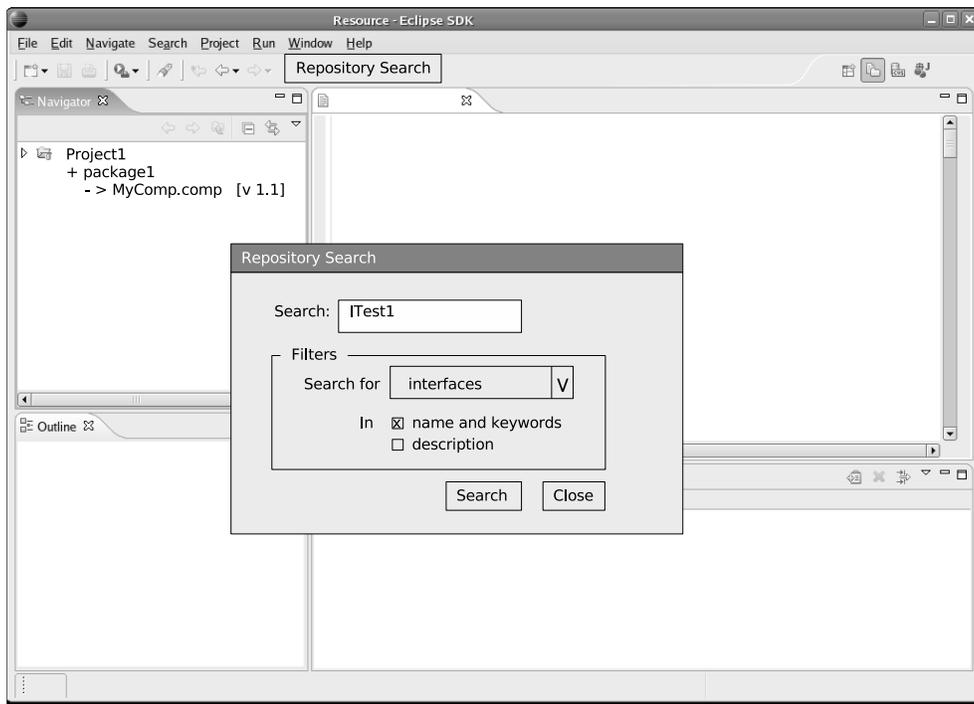


Figura 101: Pesquisar por artefato (2 - inserir critério de busca)

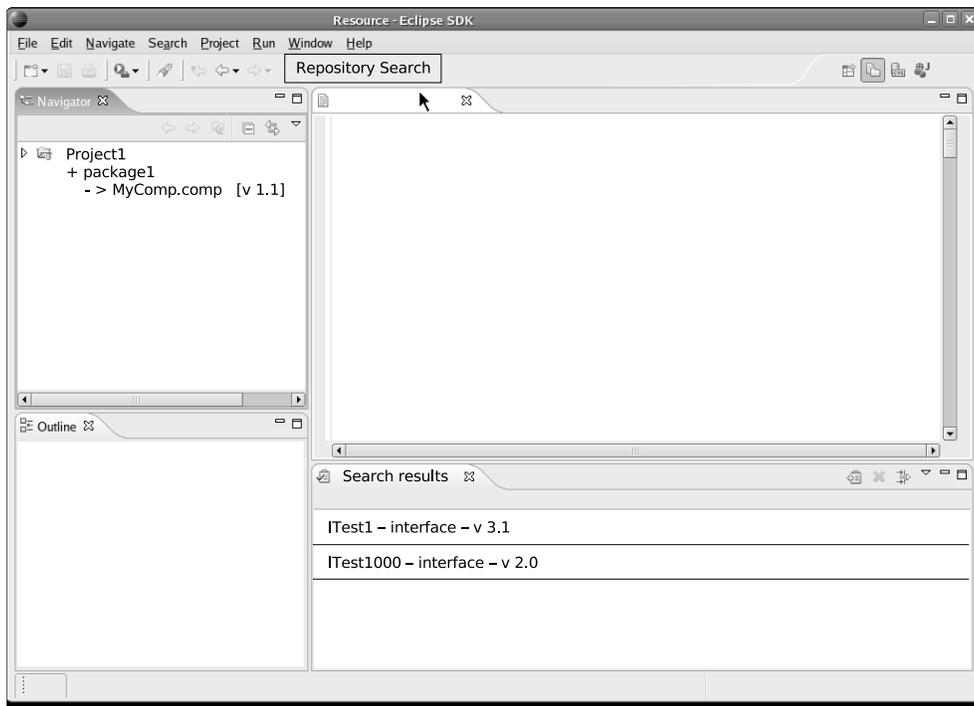


Figura 102: Pesquisar por artefato (3 - resultados)

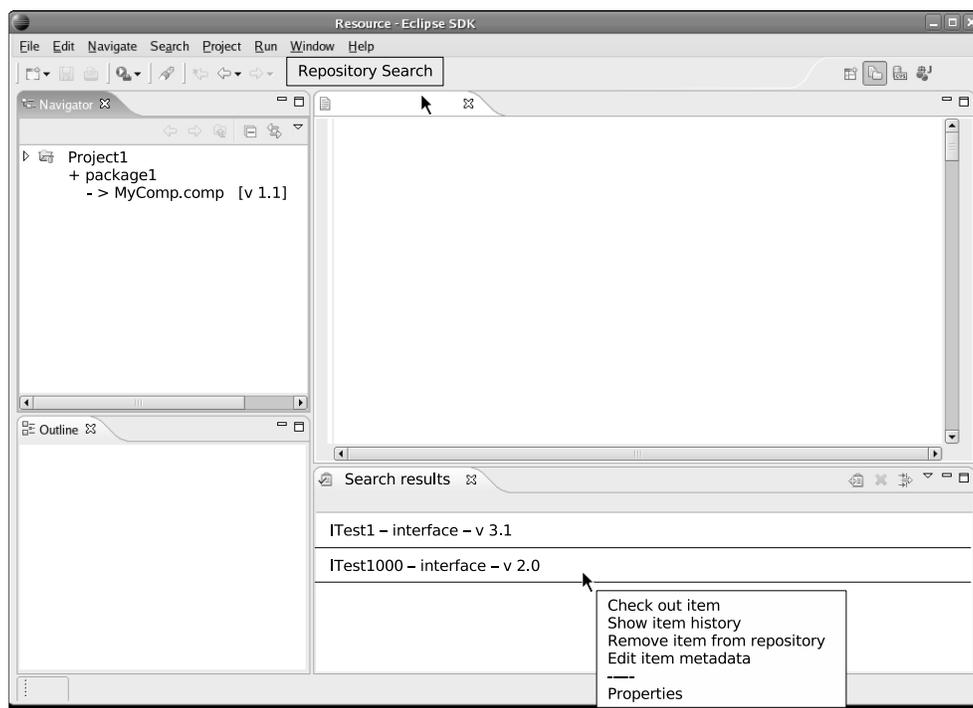


Figura 103: Recuperar artefato; Remover artefato; Editar meta-dados de um artefato; Recuperar histórico do artefato (menu em itens não recuperados anteriormente)

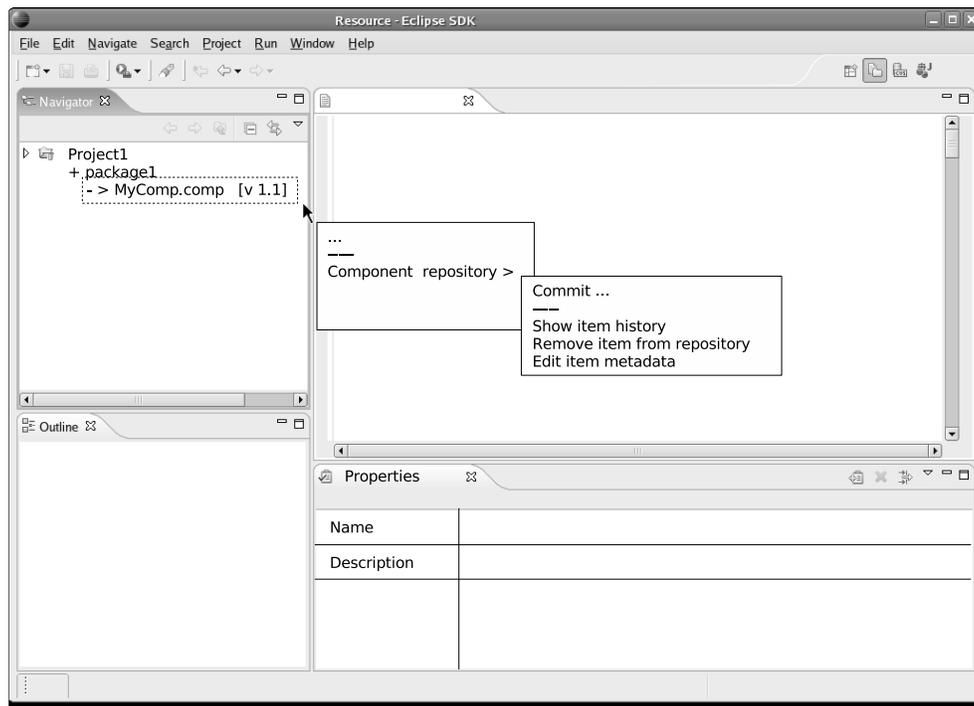


Figura 104: Depositar uma nova versão de um artefato; Remover artefato; Editar metadados de um artefato; Recuperar histórico do artefato (menu em itens recuperados anteriormente)



## A Mapeamento das Atividades do Processo UML Components

Essa seção tem como objetivo fazer um mapeamento detalhado das atividades descritas pelo processo de desenvolvimento baseado em componentes *UML Components* [3]. Ele foi produzido visando um melhor entendimento do processo, bem como uma representação clara que sirva de referência rápida ao mesmo. Ainda, desejava-se uma lista de atividades a partir da qual pudessem ser extraídos requisitos para um ambiente de desenvolvimento baseado em componentes, ou seja, quais atividades poderiam ou não ser apoiadas por tal ambiente.

Assim, foram feitos estudos sobre o processo e suas atividades foram modeladas em diagramas de atividades UML. Ainda, atividades envolvendo aspectos importantes de arquiteturas de *software*, como conectores e interfaces requeridas foram acrescentadas. Acreditamos que esta seja uma representação do processo mais objetiva de ser seguida, mas que não substitui o livro em que foi proposto [3], sendo um guia de referência rápida. Ainda, este livro foca na fase de Especificação, fato que os autores deixam claro, havendo pouca descrição para as fases de Provisionamento e Montagem. Também há pouco apoio para o reuso de componentes em sua descrição.

Nas próximas subseções, apresentamos uma visão geral do processo *UML Components* e os diagramas de atividades obtidos.

### A.1 Processo UML Components

O processo *UML Components* [3] é um processo de desenvolvimento de *software* baseado em componentes, que trata do problema de especificar e arquitetar sistemas baseados em componentes. Ele não é um processo de desenvolvimento completo, já que não inclui atividades relacionadas ao processo gerencial, e enfatiza as etapas de análise, projeto e, em menor grau, implementação.

As fases definidas pelo método são: (1) especificação, onde é gerado um conjunto de especificações de componentes e uma arquitetura de componentes, (2) provisionamento, que garante que os componentes necessários estão disponíveis, seja através de seu desenvolvimento, aquisição ou reuso e (3) montagem, que compõe os diversos componentes entre si e com artefatos de software pré-existentes, incluindo a interface com o usuário. Uma visão geral do processo é apresentada na Figura 105.

A fase de especificação ainda é dividida nas seguintes sub-fases: (a) identificação de componentes, cujo objetivo é identificar um conjunto inicial de interfaces (e o nome de seus métodos) e componentes e combiná-los em uma arquitetura inicial, (b) interação de componentes, onde através de modelos de interação, são descobertas mais operações com suas assinaturas completas e as interfaces podem ser refinadas, agrupadas ou divididas e (c) especificação de componentes, em que é feita uma especificação detalhada das interfaces, incluindo suas restrições, em termos de pré e pós-condições. Essa fase é ilustrada na Figura 106.

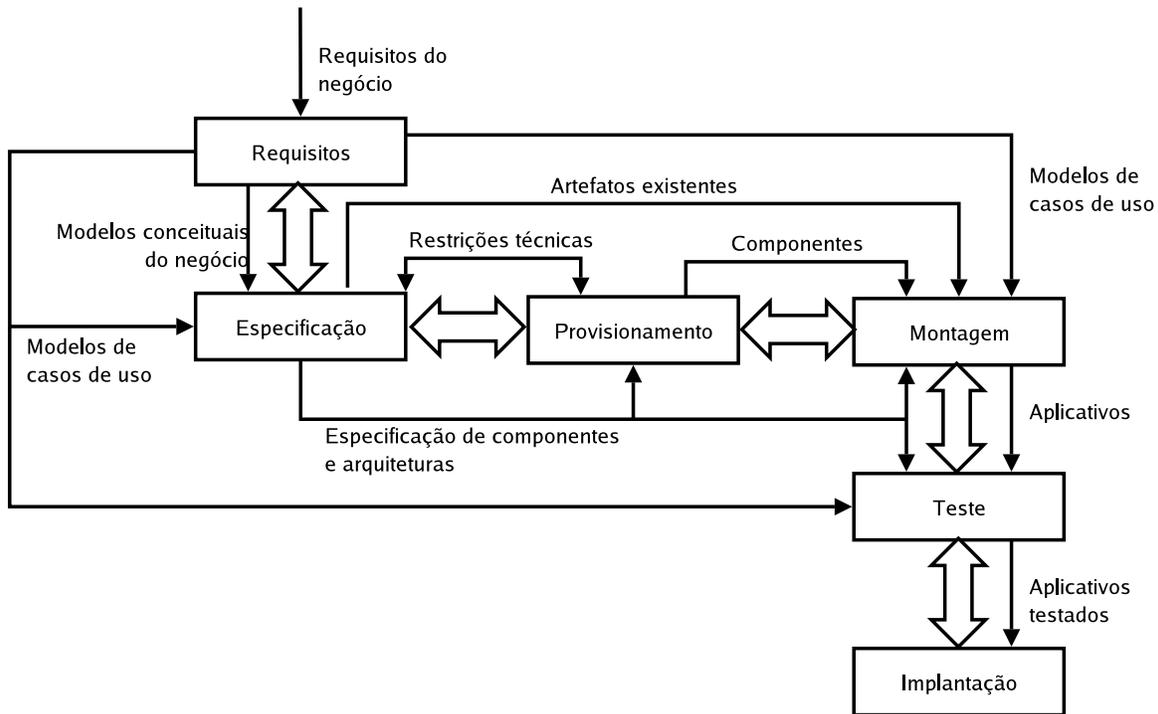


Figura 105: Fases (*workflows*) do processo

## A.2 Mapeamento das Atividades

As tarefas descritas no *UML Components* por seus autores foram identificadas e modeladas em diagramas de atividades, cobrindo as fases de Requisitos, Especificação, Provisionamento e Montagem, mostradas na Figura 105. Ainda, cada tarefa da fase de Especificação ilustrada na Figura 106 foi representada em um diagrama, havendo casos em que diagramas foram agrupados para melhorar seu entendimento. A identificação das tarefas está presente em uma nota no próprio diagrama, onde preservamos o nome original em inglês para facilitar a referência à descrição original. Algumas atividades não descritas originalmente no *UML Components* foram incluídas, em especial no que tange à inserção de interfaces requeridas e conectores, conceitos importantes para a descrição da arquitetura de *software*.

Desse modo, apresentamos seguintes diagramas de atividades:

- Requisitos (Requirements Definition), Figura 107.
- Especificação (Specification):
  - Component Identification, Identify System Interfaces & Ops, Figura 108.
  - Component Identification, Develop Business Type Model e Identify Business Interfaces, Figura 109.
  - Component Identification, Create Inicial Comp Specs & Architecture, Figura 110.

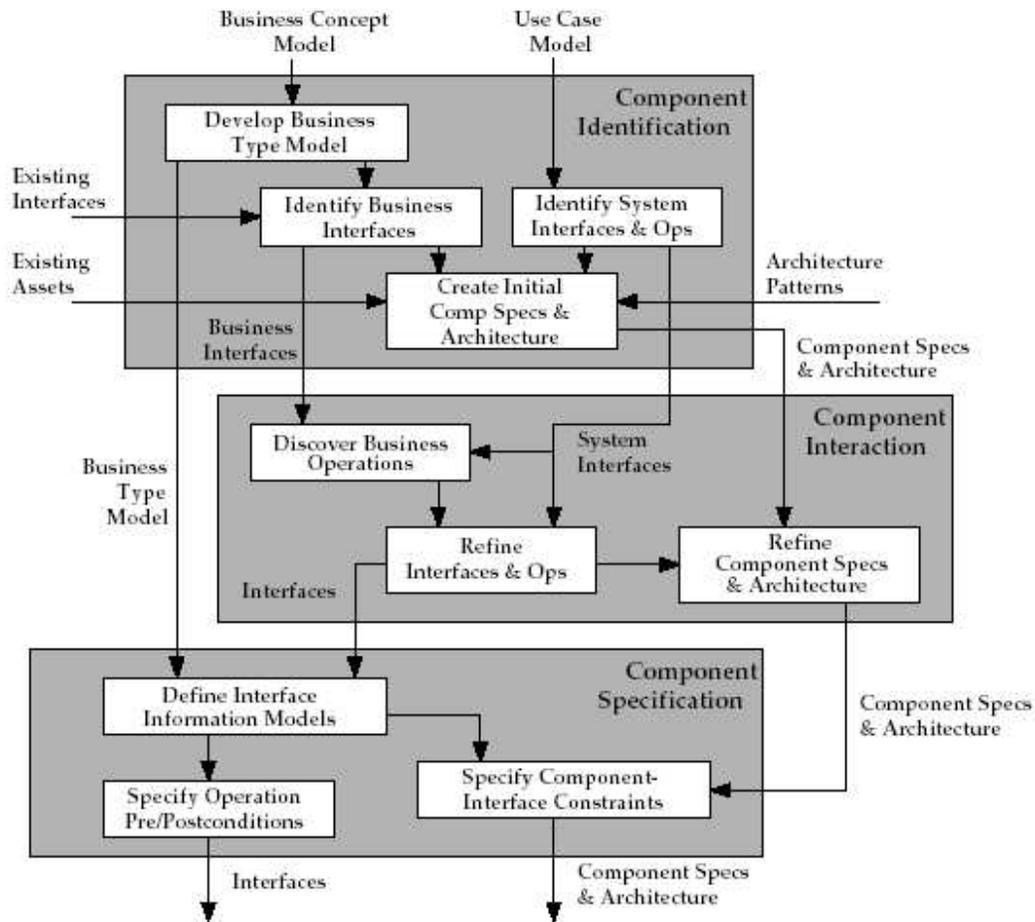


Figura 106: Sub-fases de especificação de componentes (*Specification workflow*)

- Component Interaction, Discover Business Operations, Figura 111.
  - Component Interaction, Refine Interfaces & Ops, Figura 112.
  - Component Interaction, Refine Comp Specs & Architecture, Figura 113.
  - Component Specification, Define Interface Information Model e Specify Operation Pre/Postconditions, Figura 114.
  - Component Specification, Specify Component-interface Constraints, Figura 115.
- Provisionamento (Provisioning), Figura 116 e Figura 117.
  - Montagem (Assembly), Figura 118.

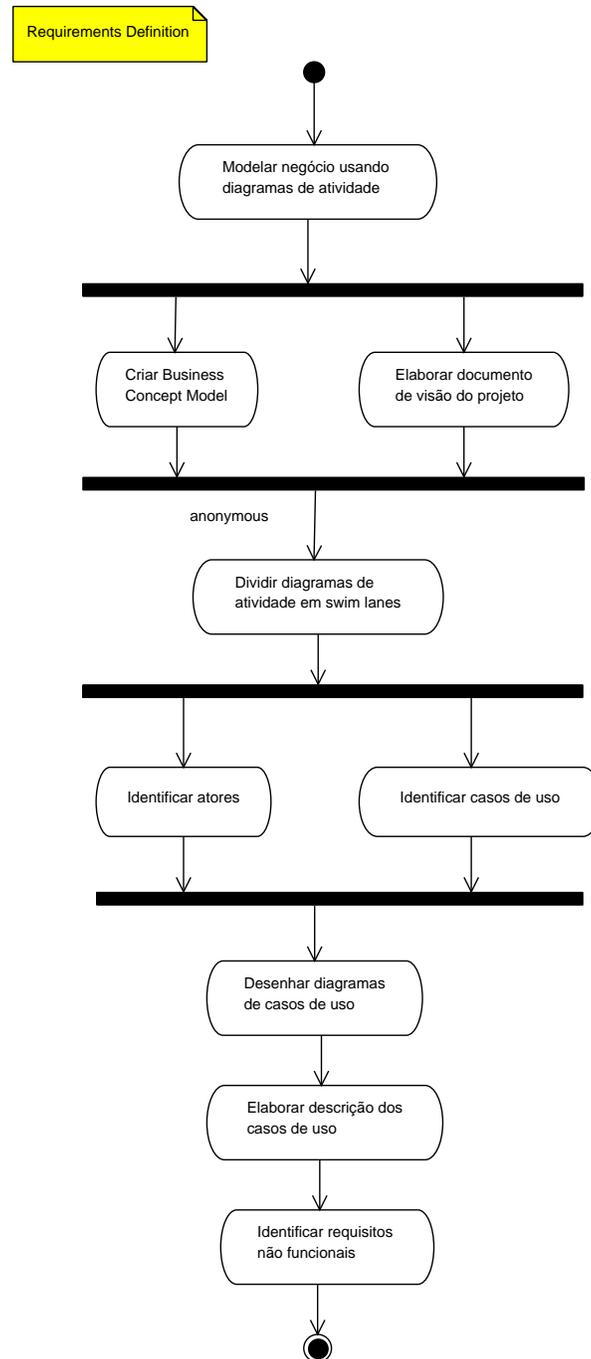


Figura 107: Requirements Definition

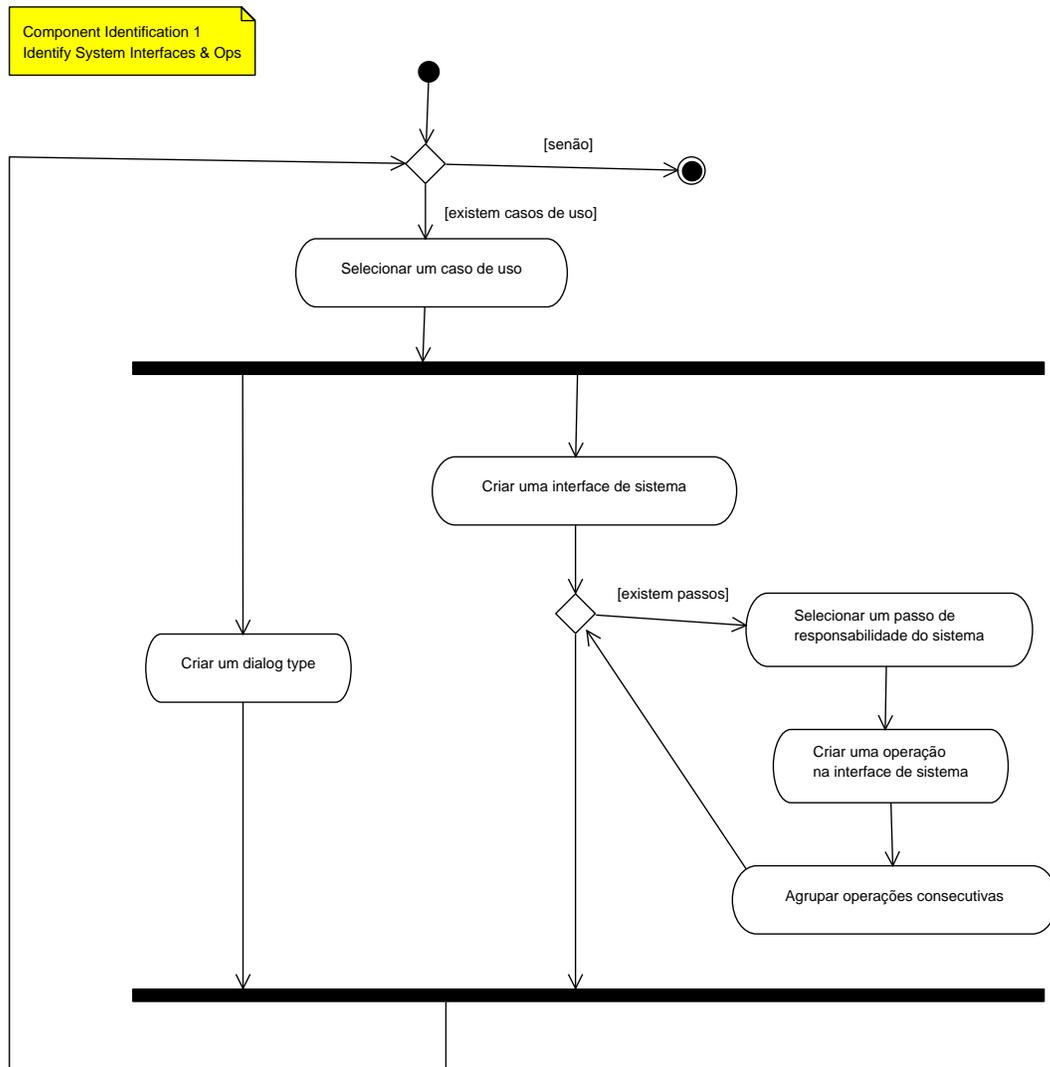


Figura 108: Especificação, Identificação de Componentes (1)

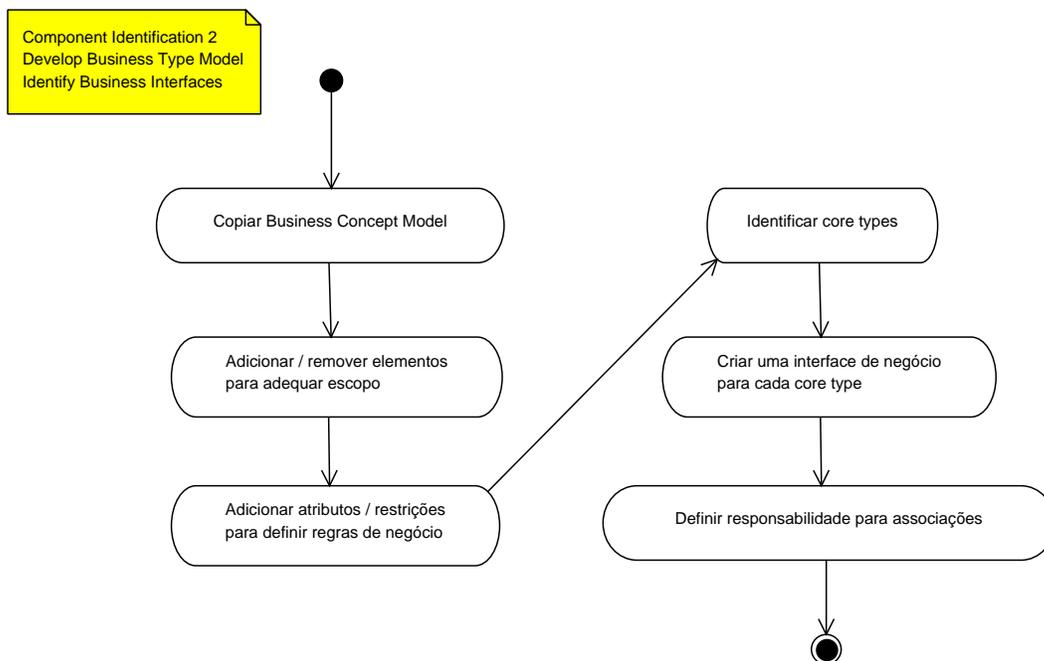


Figura 109: Especificação, Identificação de Componentes (2)

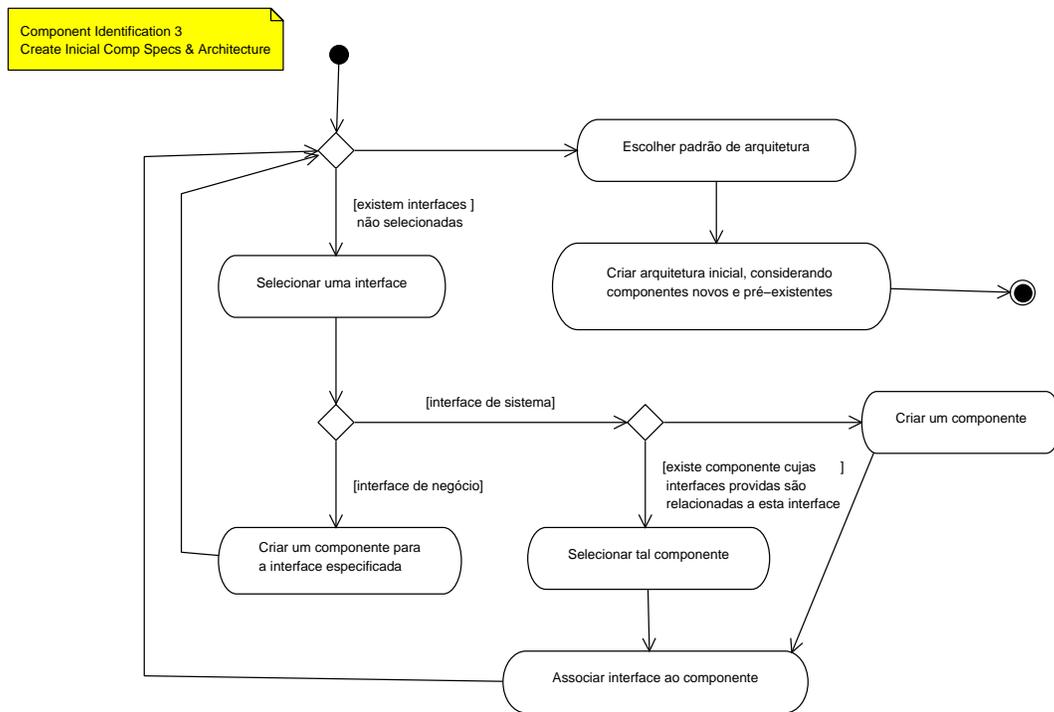


Figura 110: Especificação, Identificação de Componentes (3)

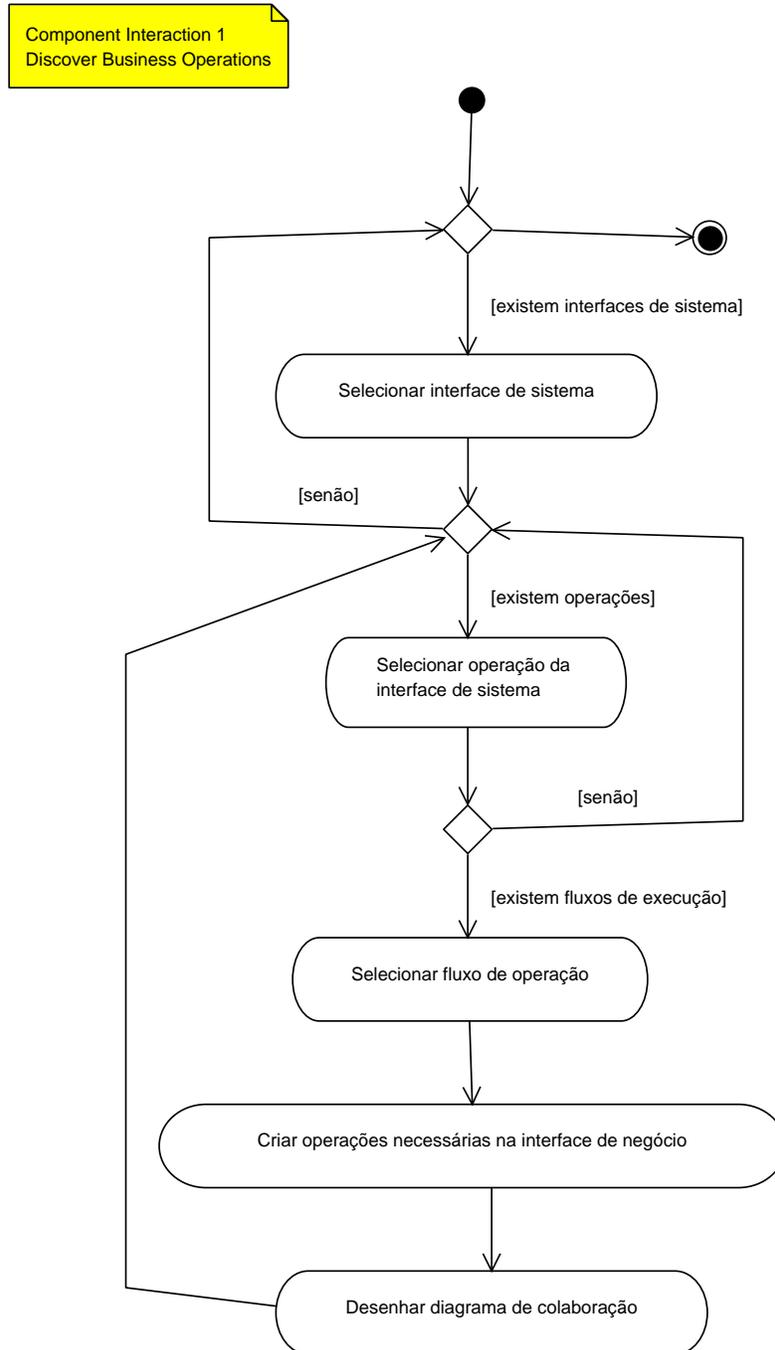


Figura 111: Especificação, Interação de Componentes (1)

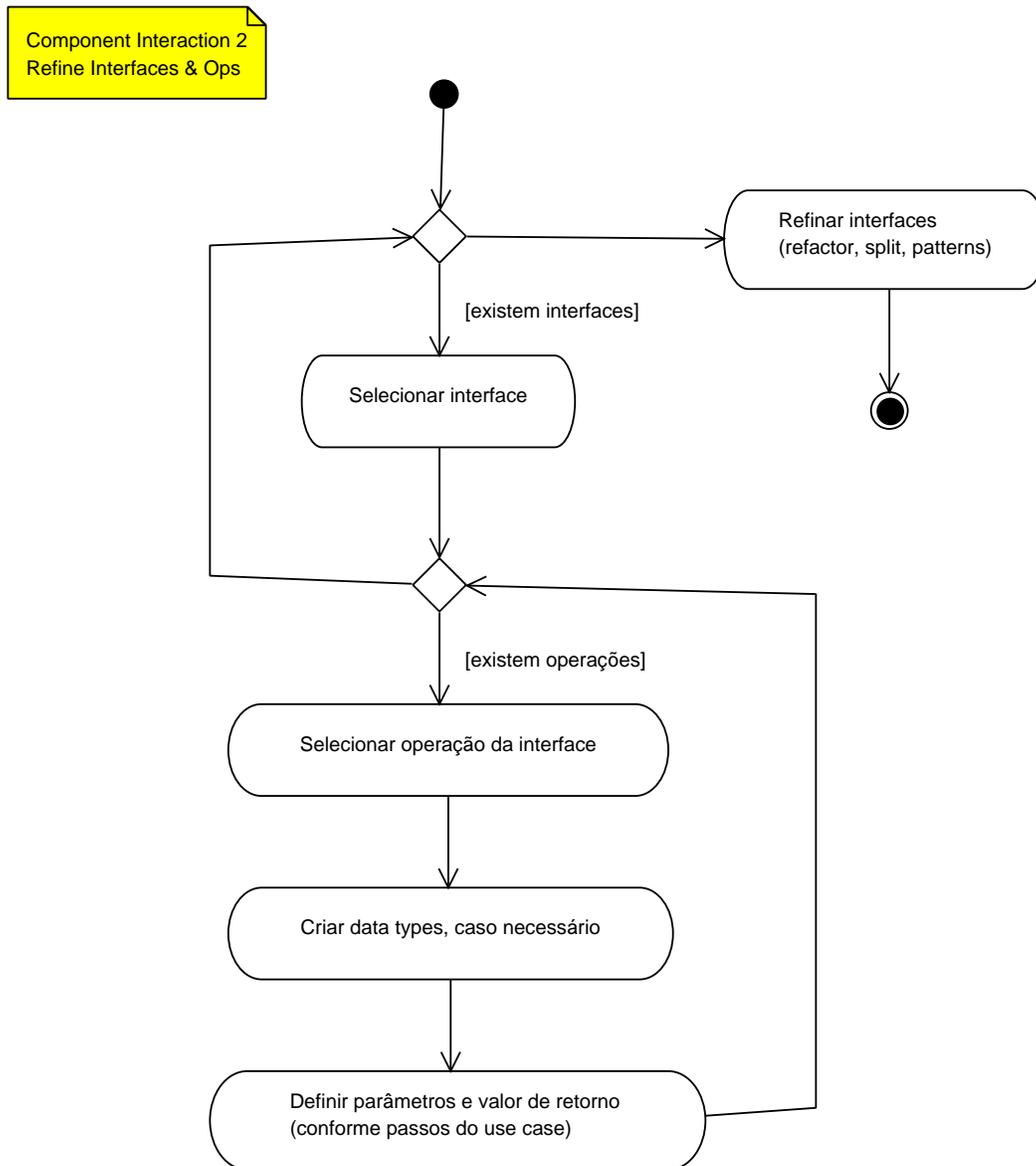


Figura 112: Especificação, Interação de Componentes (2)

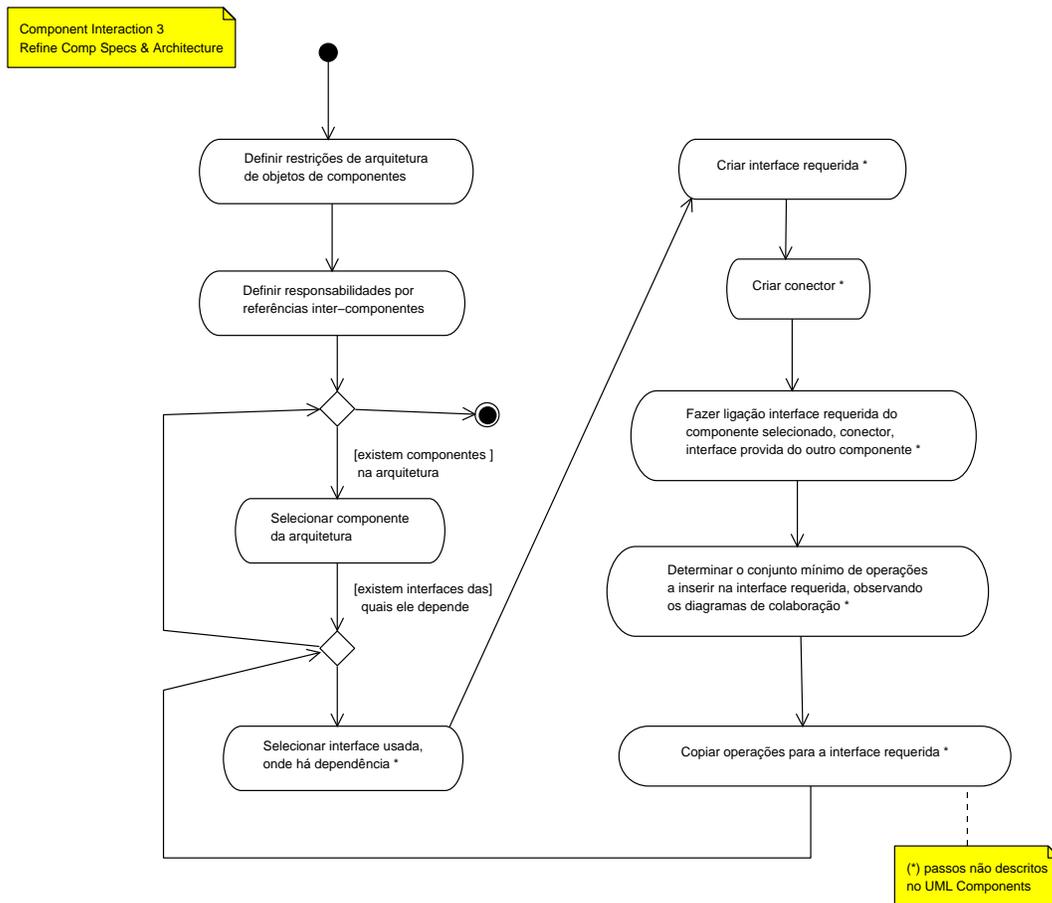


Figura 113: Especificação, Interação de Componentes (3)

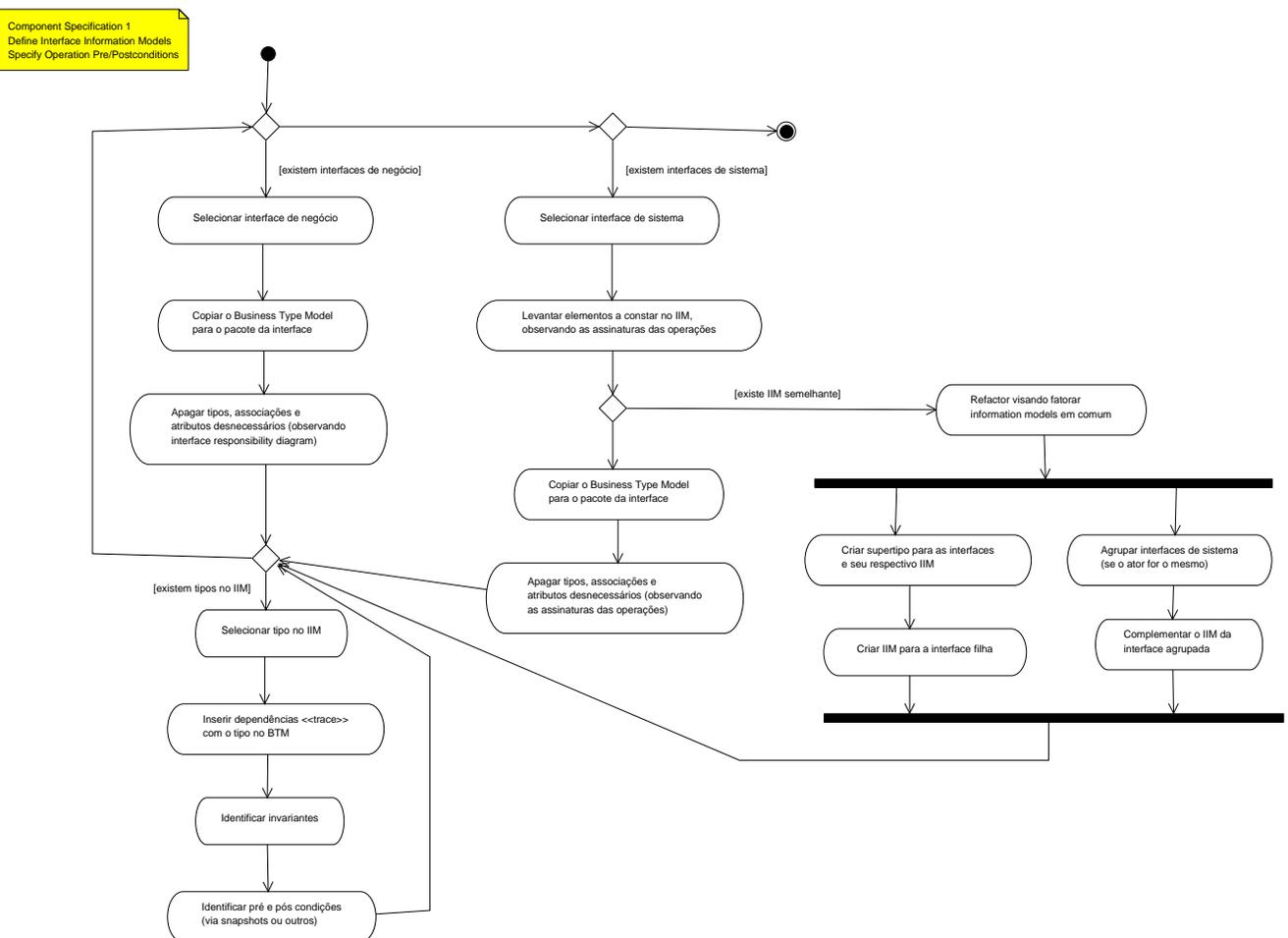


Figura 114: Especificação, Especificação de Componentes (1)

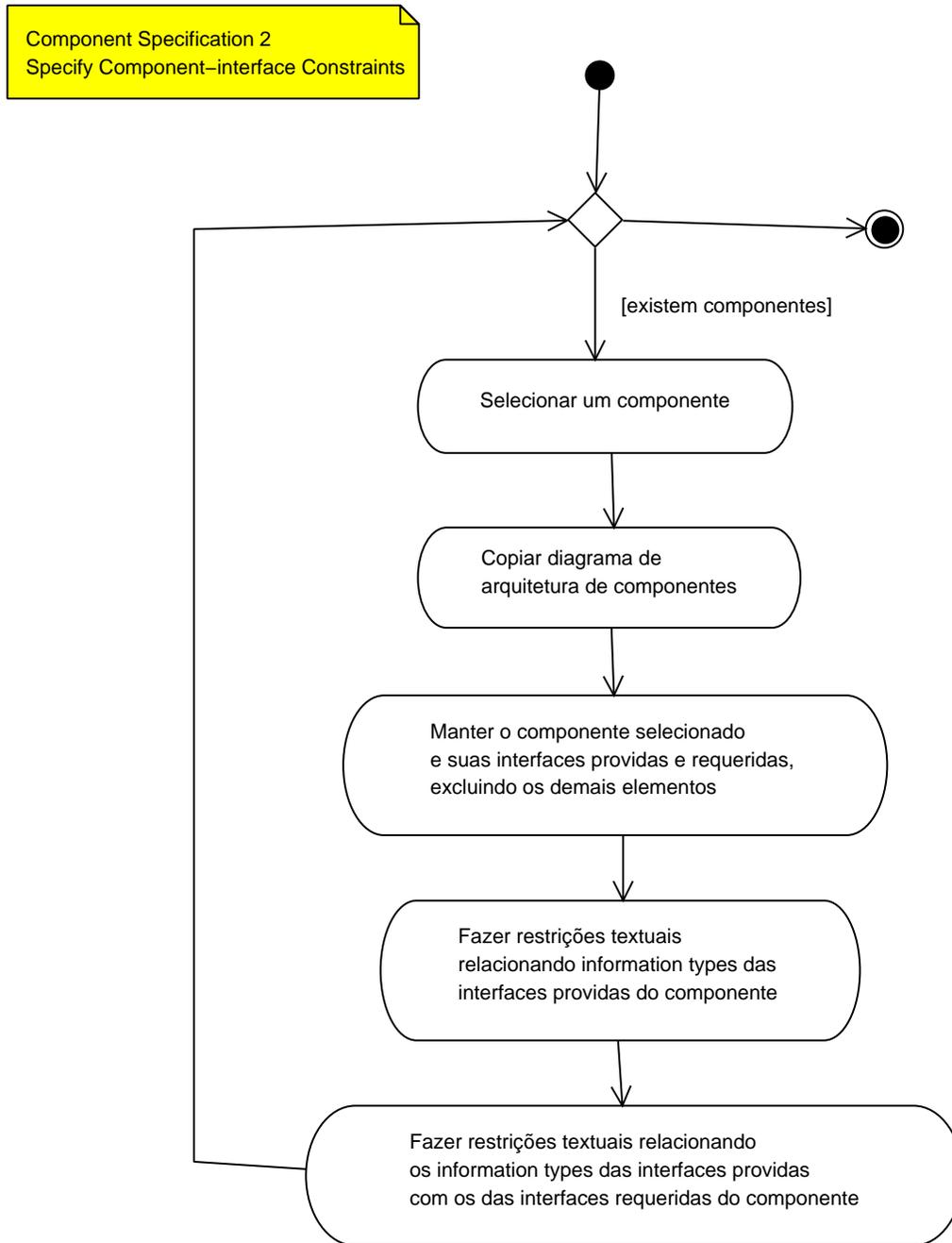


Figura 115: Especificação, Especificação de Componentes (2)

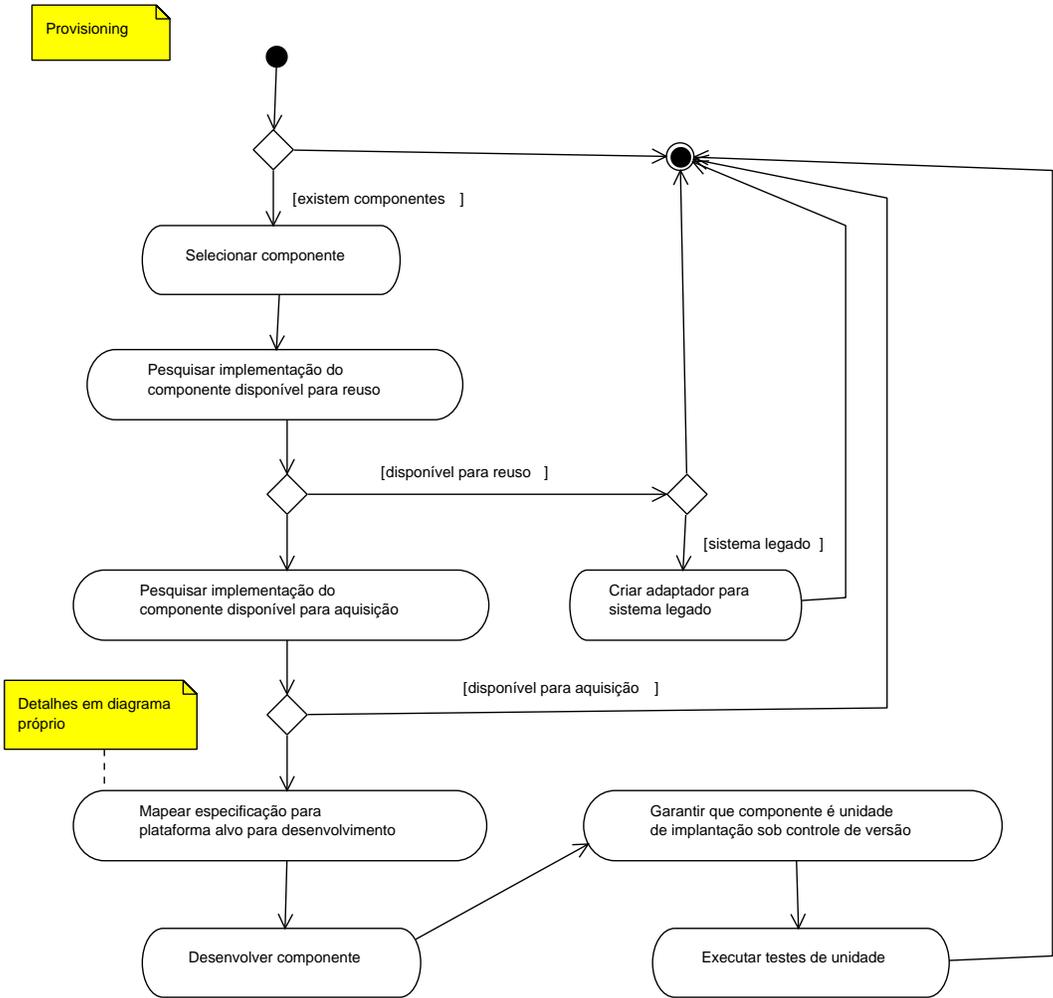


Figura 116: Provisionamento (1)

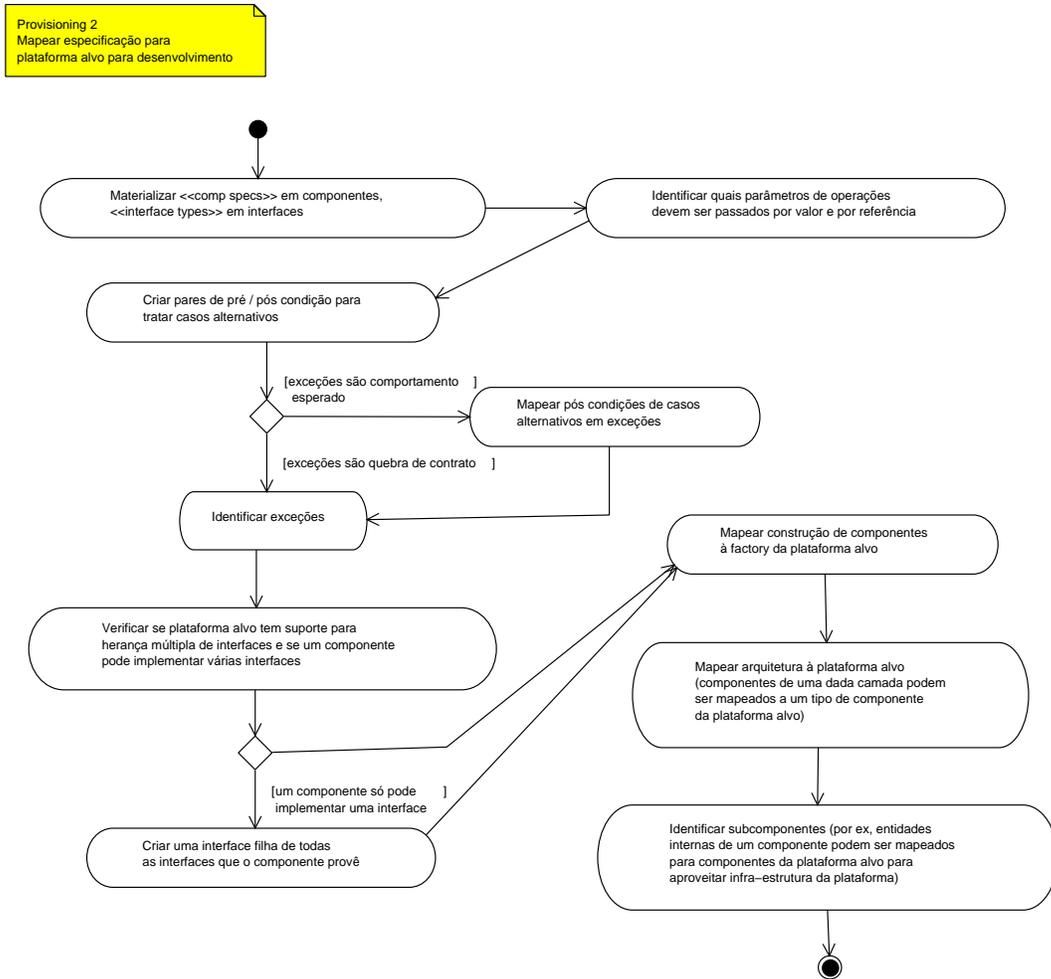


Figura 117: Provisionamento (2)

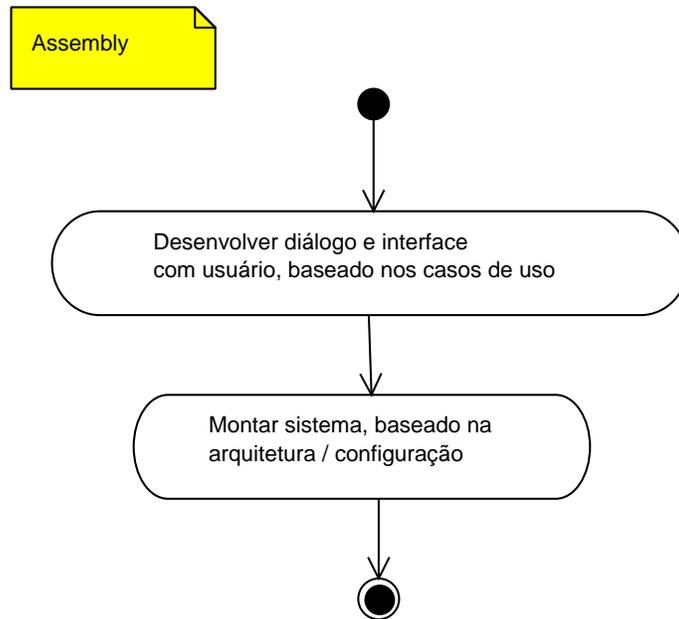


Figura 118: Montagem

## Referências

- [1] C. Atkinson et al. *Component-Based Product Line Engineering with UML*. Addison-Wesley, 2001.
- [2] Len Bass, Paul Clements, and Rick Kazman. *Software Architecture in Practice*. SEI Series in Software Engineering. Addison-Wesley, 2nd edition, 2003.
- [3] John Cheesman and John Daniels. *UML Components*. Addison-Wesley, 2000.
- [4] Eclipse Consortium. Eclipse.org main page. <http://www.eclipse.org/>, 2003.
- [5] Paulo Asterio de Castro Guerra. *Uma Abordagem Arquitetural Para Tolerância a Falhas Em Sistemas de Software Baseados Em Componentes*. PhD thesis, Instituto de Computação — Universidade Estadual de Campinas, 2004.
- [6] D. D'Souza and A. C. Wills. *Components and Frameworks with UML: The Catalysis Approach*. Addison-Wesley, 1999.
- [7] D. Lucrédio, E. S. Almeida, C. P. Bianchini, A. F. Prado, and L. C. Trevelin. Orion - a component-based software engineering environment. *Journal of Object Technology*, 3(4):51 – 74, April 2004. Special issue: TOOLS USA 2003.
- [8] Clemens Szyperski. *Component Software: Beyond Object-Oriented Programming*. ACM Press and Addison-Wesley, New York, NY, 2nd edition, 2002.
- [9] Rodrigo T. Tomita, Fernando Castor, Paulo A. C. Guerra, and Cecília M. F. Rubira. Bellatrix: Um ambiente para suporte arquitetural ao desenvolvimento baseado em componentes. In *Anais do 4o Workshop de Desenvolvimento Baseado em Componentes*, pages 43 – 48, 2004.