

DIAGRAMAS UML

COMPORTAMENTAIS

Diagrama de Casos de Uso

Usado geralmente na fase de especificação dos requisitos funcionais em sistemas de informação. Mostra quais usuários realizam cada funcionalidade do sistema, e alguns relacionamentos entre estas funcionalidades.

Figura 3: Diagrama de Casos de Uso

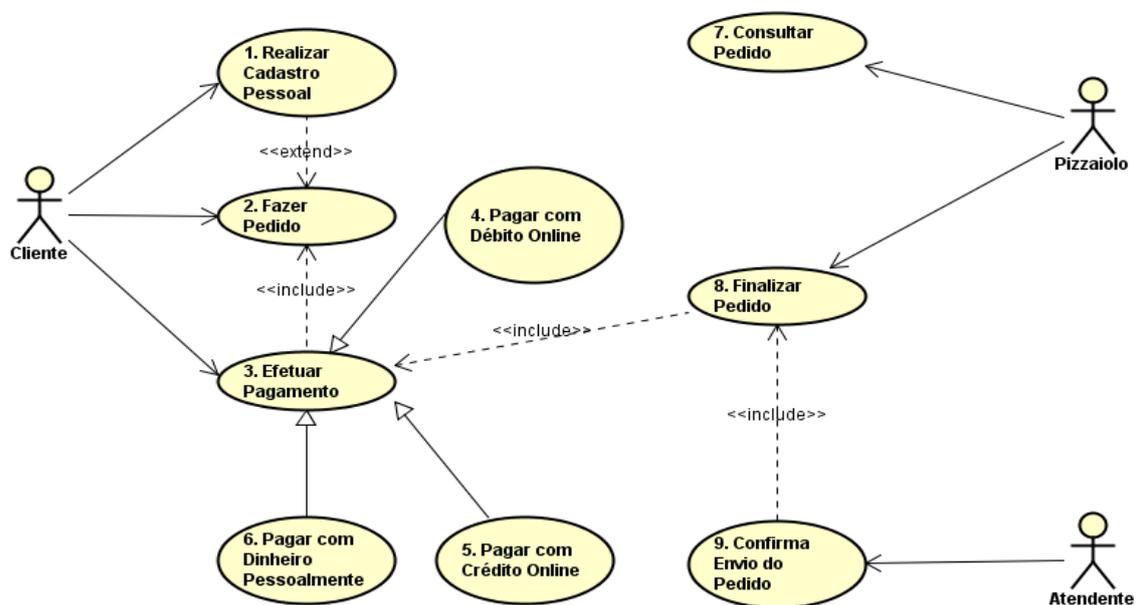
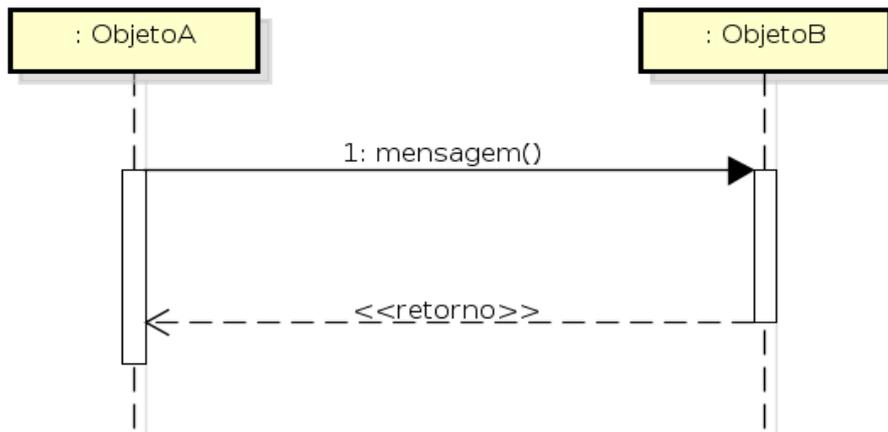


Diagrama de Sequência (Interação)

Preocupa-se com a ordem temporal em que as mensagens são trocadas, e pode se basear em um Caso de Uso. Identifica os eventos associados à funcionalidade modelada, e o ator responsável por este evento

Figura 4: Conceito de Diagrama de Sequência



Os objetos (Figura 4) podem ser, por exemplo numa arquitetura MVC:

- Ator (Figura Humana)
- Front End, Tela, Interface (View)
- Back-End, Objeto Controlador (Controller)
- Database, Banco de Dados (Model)

Figura 5: Diagrama de Sequência - Exemplo

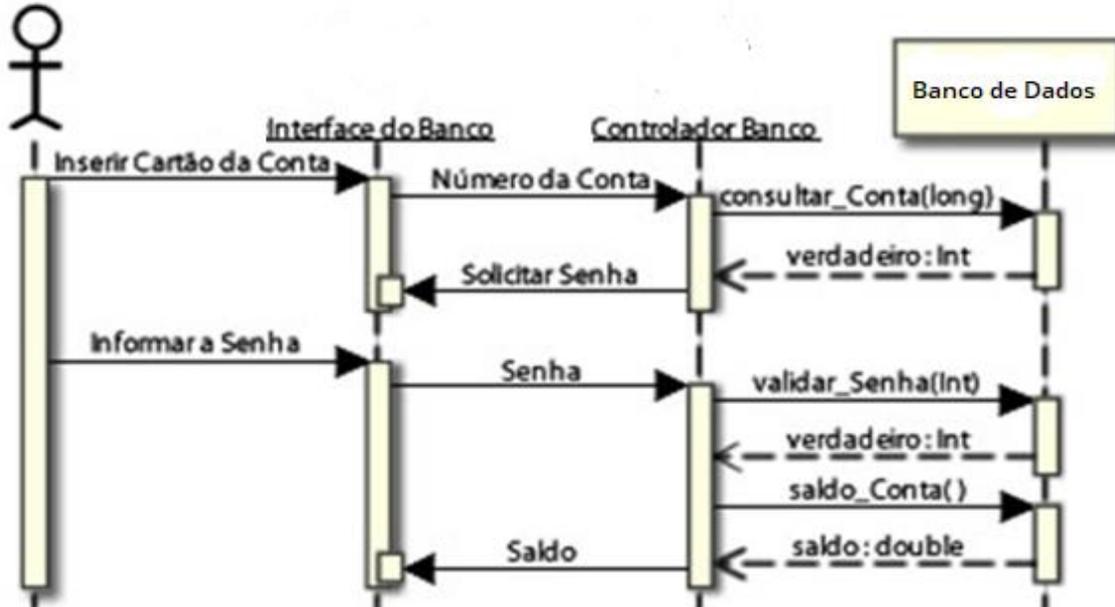


Diagrama de Colaboração (Comunicação)

Amplamente associado ao Diagrama de Sequência, são complementares, não se preocupa com a temporalidade, define como os objetos estão vinculados e quais mensagens são trocadas entre objetos

Figura 6: Diagrama de Colaboração

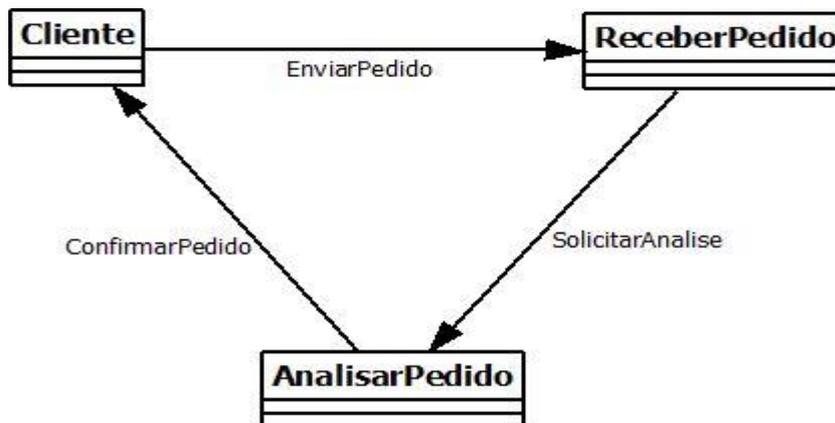


Diagrama de Atividades

Descreve as atividades a serem executadas para a conclusão de um processo. Concentra-se na representação do fluxo de controle de um processo.

Figura 7: Diagrama de Atividade

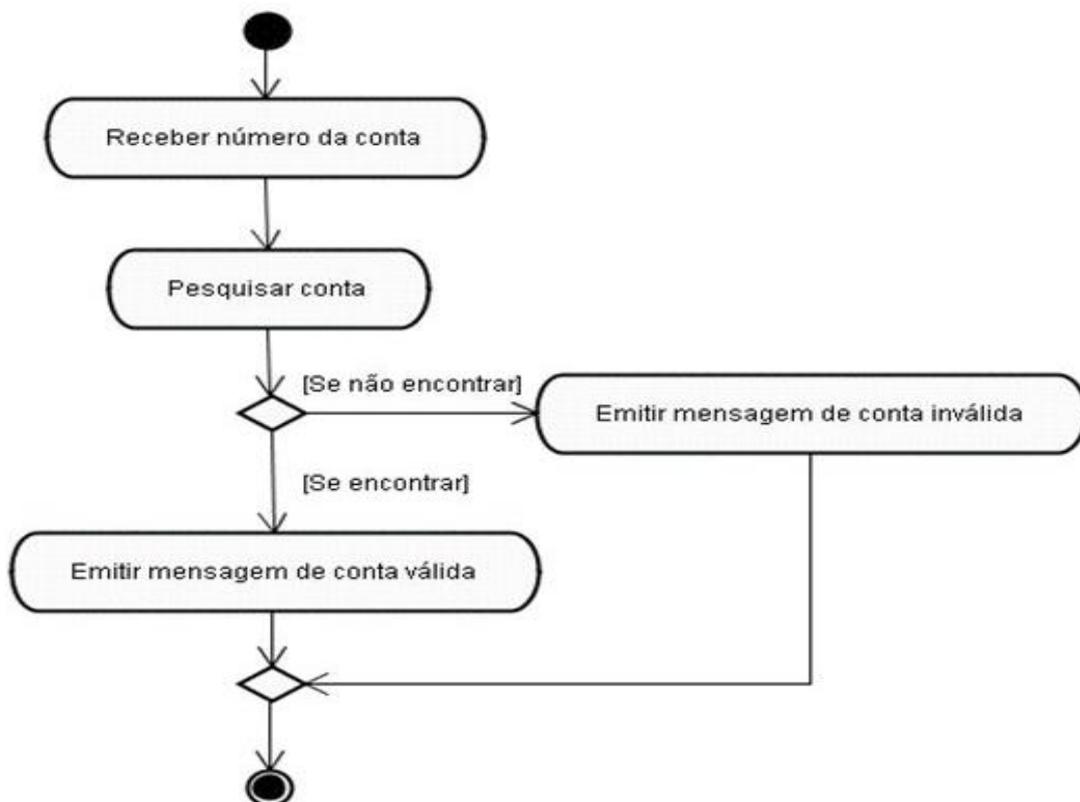
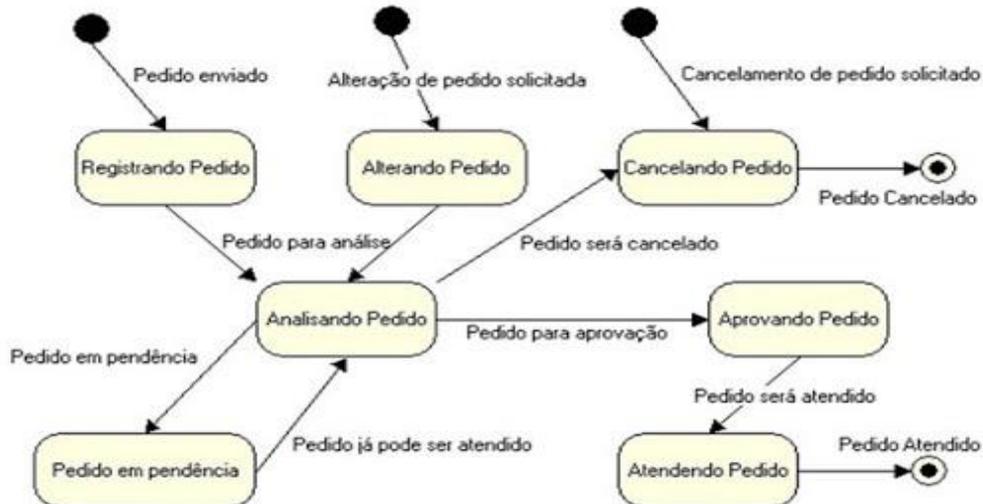


Diagrama de Estados

Modela as mudanças sofridas por um objeto dentro de um determinado processo. Pode ser utilizado para acompanhar os estados pelo qual passa uma instância de uma classe.

Figura 8: Diagrama de Estados



ESTRUTURAIS

Diagrama de Classes

Serve de apoio para a maioria dos outros diagramas, define a estrutura das classes do sistema, e estabelece como as classes se relacionam.

Figura 9: Diagrama de Classes

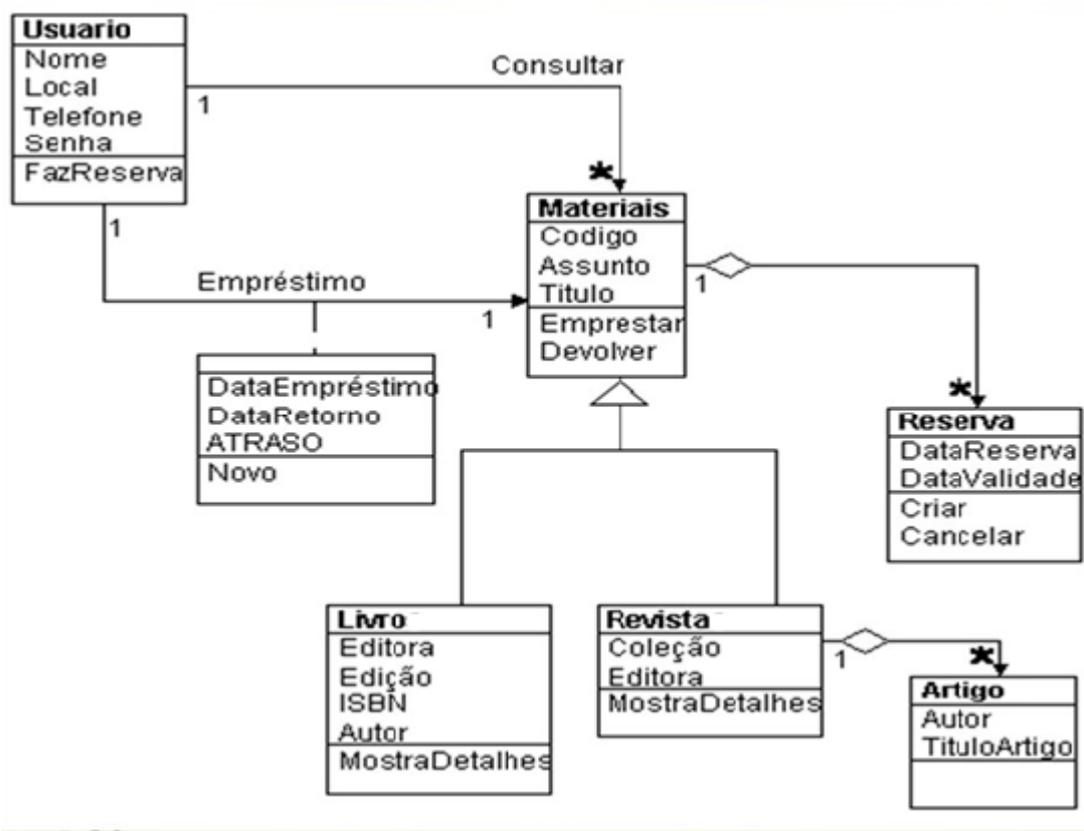


Diagrama de Objetos

Complemento do Diagrama de Classes, exibe os valores armazenados pelos objetos de um Diagrama de Classes.

Figura 10: Diagrama de Objetos

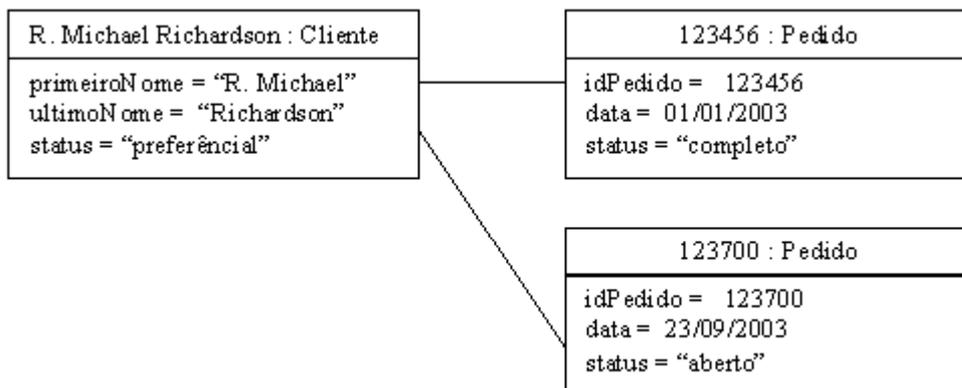


Diagrama de Componentes

Representa os componentes do sistema. Um componente é uma parte lógica e substituível do sistema. Os componentes serão implementados como: Classes de código-fonte, Bibliotecas, Arquivos de ajuda, etc.

Figura 11: Diagrama de Componentes - Notação

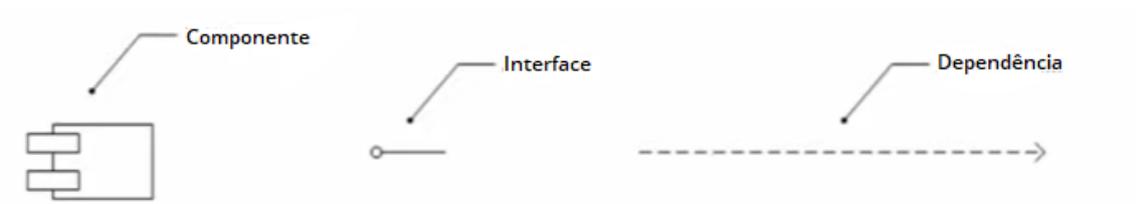


Figura 12: Diagrama de Componentes - Exemplos

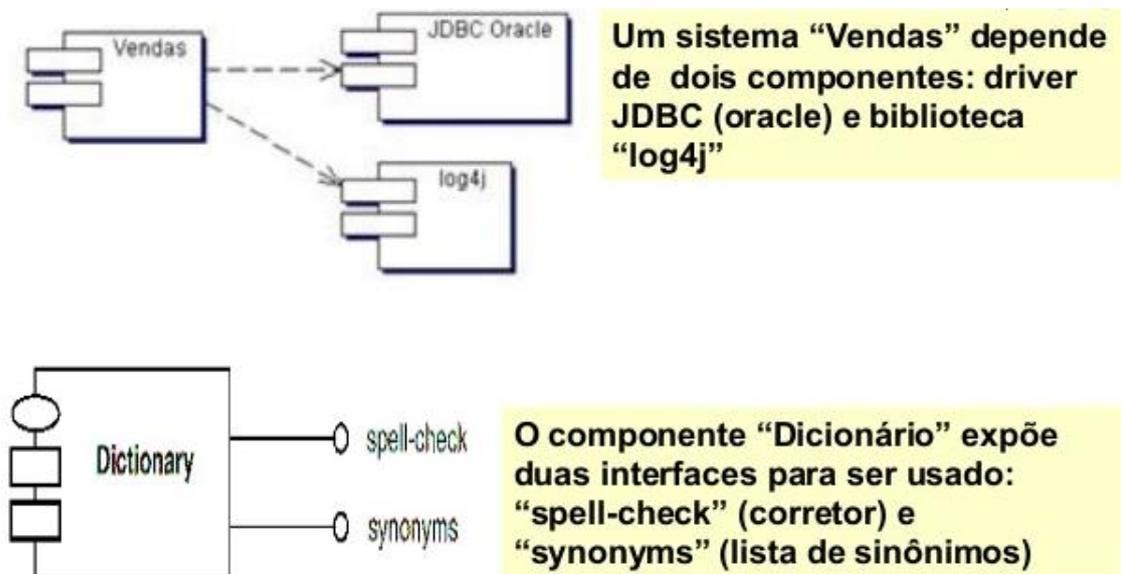


Diagrama de Pacotes (Módulos)

Descreve os pacotes ou pedaços do sistema (módulos) divididos em agrupamentos lógicos mostrando as dependências entre eles. Este diagrama é muito utilizado para ilustrar a arquitetura de um sistema mostrando o agrupamento de suas classes.

Figura 13: Diagrama de Pacotes - Notação

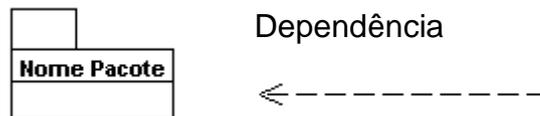


Figura 14: Diagrama de Pacotes - Exemplo

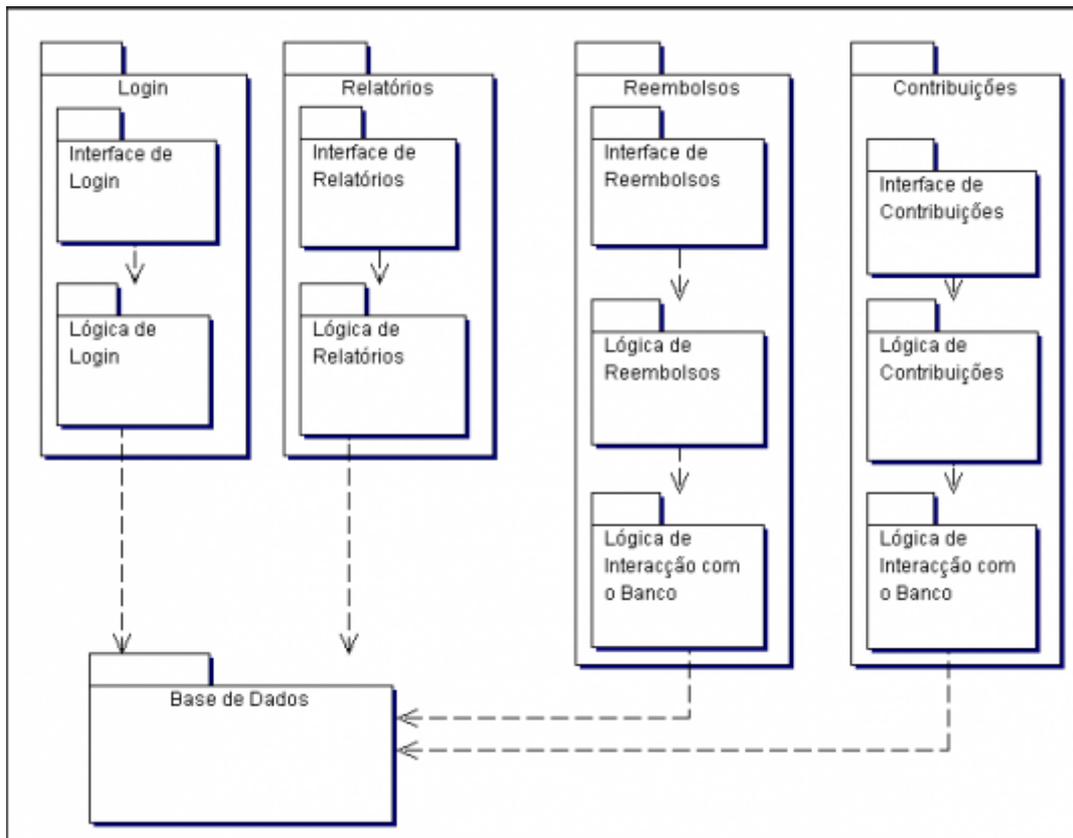


Diagrama de Implantação

Determina as necessidades de hardware e características físicas do sistema: Servidores, Estações, Topologias de Rede, Protocolos de Comunicação, etc.

Os Diagramas de implantação costumam conter:

- Nós (Recurso computacional que existe em tempo de Execução)
- Relacionamentos de Dependência (Linha pontilhada com uma seta)
- Relacionamentos de Associação (Linha que indica comunicação entre Nós)

Figura 15: Diagrama de Implantação

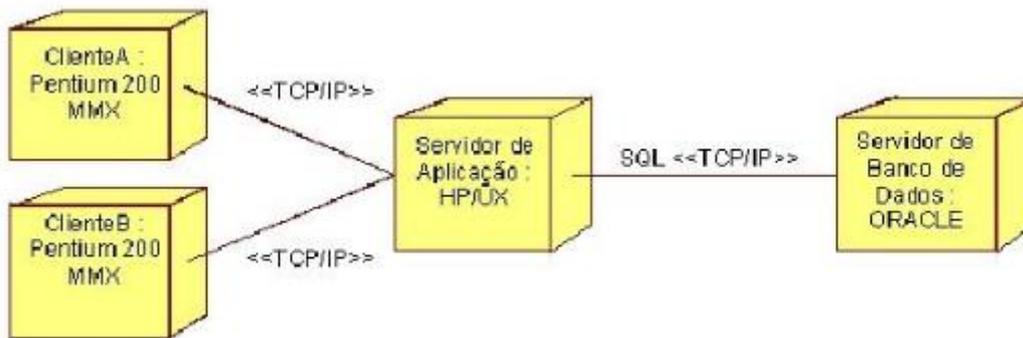


Diagrama de Estrutura Composta

Utilizado para modelar colaborações. Sua estrutura é de elementos conectados que representam instâncias de tempo de execução, colaborando por meio de vínculos de comunicação para atingir um objetivo comum.

- Mostra uma colaboração que existe entre instâncias de classes, objetos ou interfaces.
- São usadas notações disponíveis para as classes, objetos ou interfaces.
- É um diagrama opcional, sendo usado apenas em situações muito complicadas
- Colaboração é o Local onde se trocam Mensagens.

Figura 16: Diagrama de Estrutura Composta - Exemplo 1

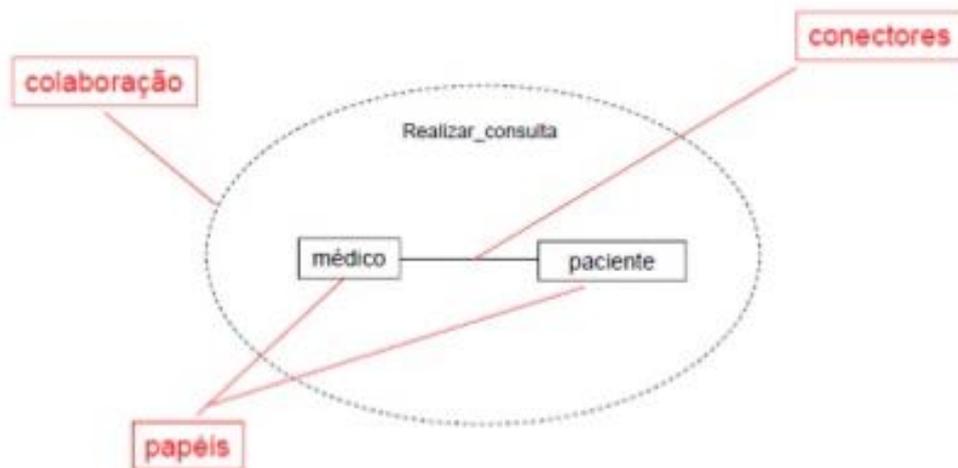


Figura 17: Diagrama de Estrutura Composta - Exemplo 2

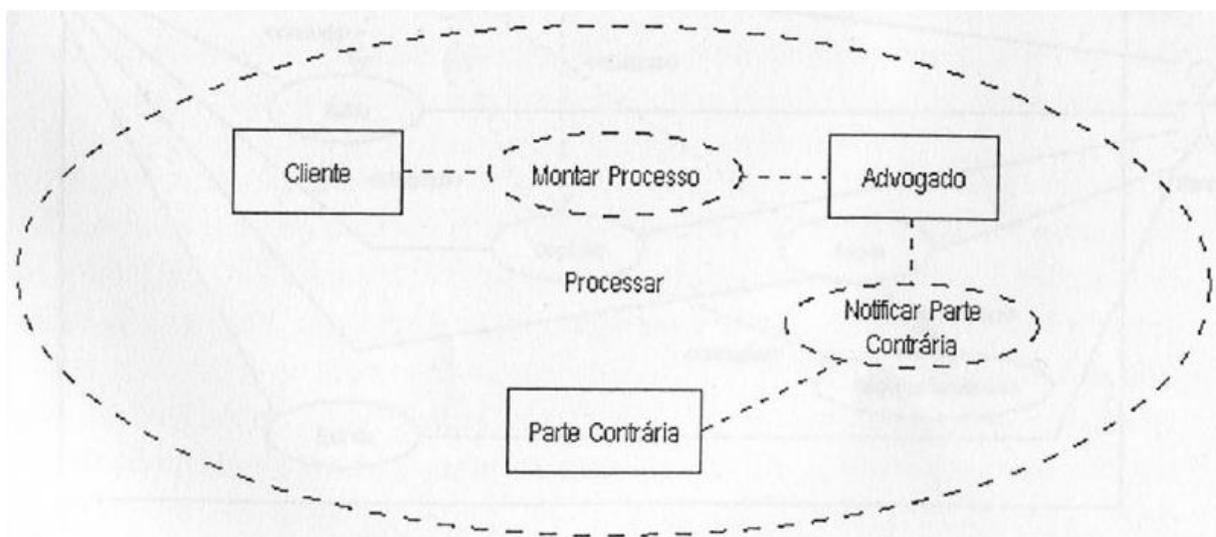


Diagrama de Perfil

Conjunto de estereótipos predefinidos, valores atribuídos, restrições e classes de base. Apresenta representações de metamodelo, permitindo definir estereótipos personalizados, valores etiquetados e restrições.

Estereótipos constituem um recurso para destacar ou diferenciar componentes ou relacionamentos, atribuindo a eles características específicas ou modificando-os de alguma forma.

A UML oferece diversos estereótipos padronizados. Podemos citar <<interface>>, <<extends>> e <<include>>; No entanto, vamos supor que haja um relacionamento “Pessoa saca Dinheiro”. Não há um estereótipo <<sacar>>, todavia, ele pode ser criado e representado usando Diagramas de Perfil por meio da expressão <<estereótipo>>.

Figura 18: Diagrama de Perfil

