



Uma Visão Geral de UML

Prof. Fabio Kon

IME/USP

Apresentação baseada em slides de Kendall V. Scott

Linguagem de Modelagem Unificada

UML é uma linguagem padrão da OMG para

- visualização,
- especificação,
- construção e
- documentação

de software orientado a objetos.

Visualização

- A existência de um modelo visual facilita a comunicação e faz com que os membros de um grupo tenham a mesma idéia do sistema.
- Cada símbolo gráfico tem uma semântica bem definida.

Especificação



É uma ferramenta poderosa para a especificação de diferentes aspectos arquiteturais e de uso de um sistema.

Construção

- Geração automática de código a partir do modelo visual
- Geração do modelo visual a partir do código
- Ambientes de desenvolvimento de software atuais permitem:
 - movimentações em ambos sentidos e
 - manutenção da consistência entre as duas visões.

Documentação

Pode incluir artefatos como:

- *Deliverables* (documentos como especificação de requisitos, especificações funcionais, planos de teste, etc.).
- Materiais que são importantes para controlar, medir, e refletir sobre um sistema durante o seu desenvolvimento e implantação.

Descrição Arquitetônica

UML oferece uma forma padrão de se desenhar as “plantas” (como em arquitetura) de um sistema de forma a incluir

- aspectos abstratos (processos de negócio, funcionalidades do sistema)
- aspectos concretos (classes C++/Java esquemas de bancos de dados, componentes de software reutilizáveis)

Razões para Modelar

- Comunicar a estrutura e o comportamento desejado de um sistema.
- Visualizar e controlar a arquitetura de um sistema.
- Para melhorar o nosso entendimento de um sistema e, assim, expor oportunidades para melhorias e reutilização.
- Para administrar os riscos e *trade-offs*.

Diagramas Estruturais

Usados para visualizar, especificar, construir e documentar aspectos estáticos de um sistema

- diagrama de classes
- diagrama de pacotes
- diagrama de objetos
- diagrama de componentes
- diagrama de implantação

Usos Comuns para Diagramas de Classes

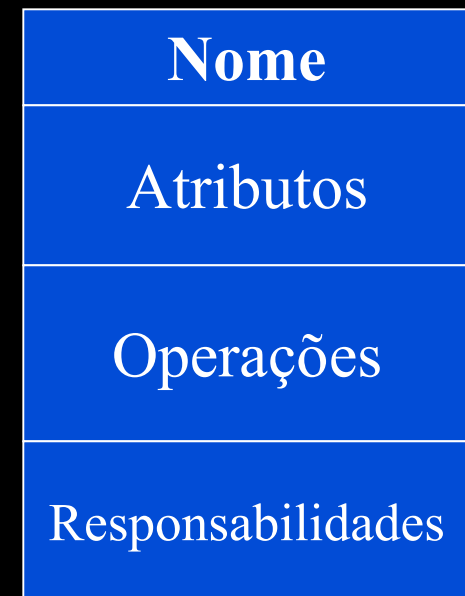
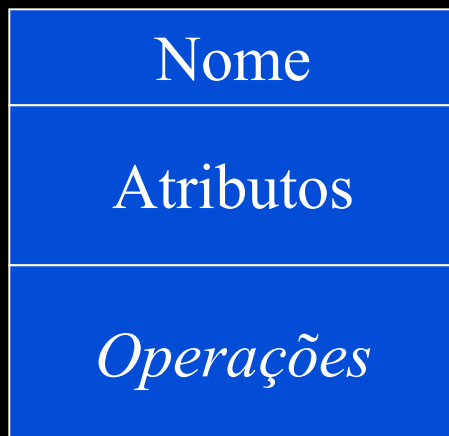
- Modelar o vocabulário do sistema, em termos de quais abstrações fazem parte do sistema e quais caem fora de seus domínios.
- Modelar as colaborações/interações (sociedades de elementos que trabalham em conjunto oferecendo algum comportamento cooperativo).
- Modelagem lógica dos dados manipulados pelo sistema (servindo de base para a definição formal do modelo da base de dados).

Notação para Classes



Nome
Atributos
Operações

Notações Alternativas



itálico → abstrata

Especificação do Acesso

Nome
+ atrib1 - atrib2
+ op1 - op2 # op3

+ public

- private

protected

Relacionamentos



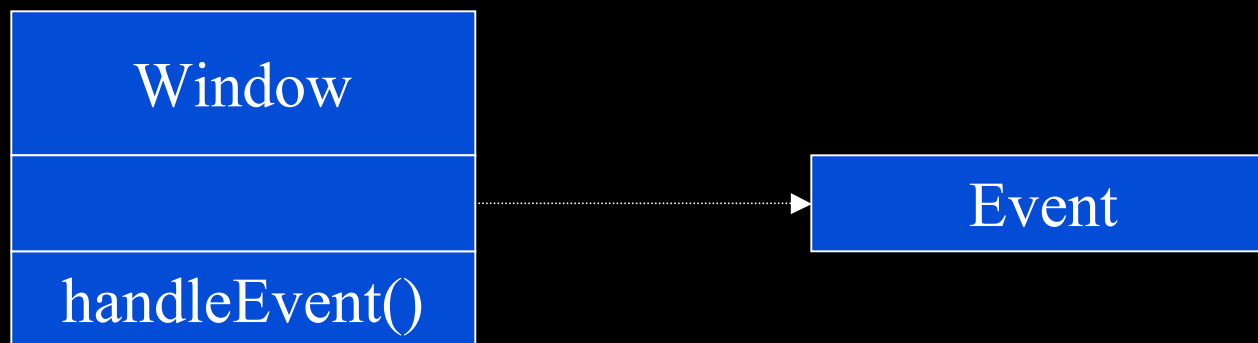
São conexões entre classes:

1. dependência
2. generalização
3. associação

Dependência

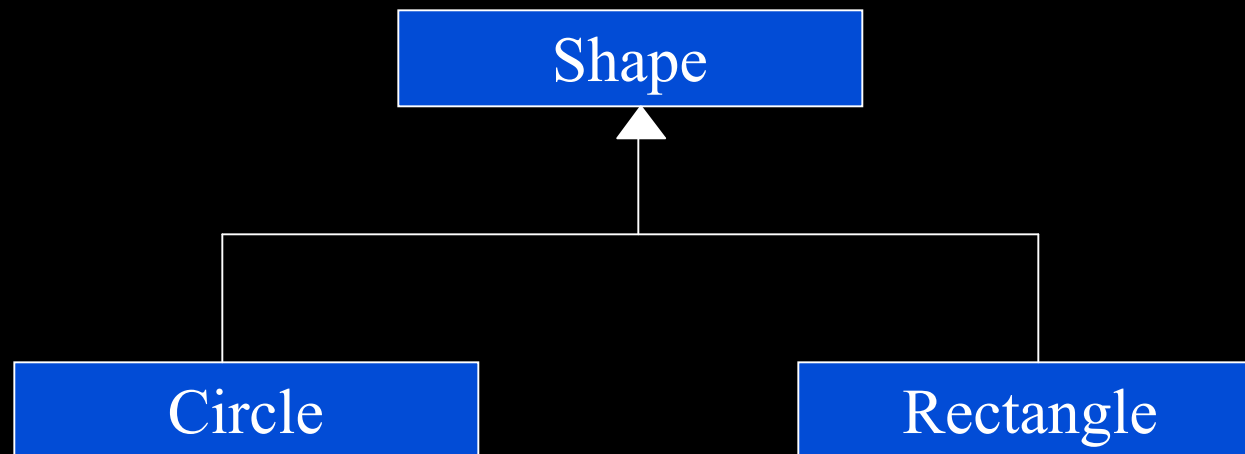
É uma relação do tipo “usa” na qual mudanças na implementação de uma classe podem causar efeitos em outra classe que a usa.

Exemplo: uma classe usa a outra.



Generalização

É uma relação do tipo “é um” entre uma coisa geral (superclasse) e uma coisa mais específica (subclasse).



Associação

É uma relação estrutural na qual classes ou objetos estão interconectados.

Uma associação entre objetos é chamada de uma ligação (*link*).



Ornamentos para Associações

- nome
- papel
- multiplicidade
- agregação
- composição

Nome da Associação

descreve a natureza da relação:



pode indicar a direção:



Papéis



- Classes e objetos podem assumir papéis diferentes em diferentes momentos.



Multiplicidade

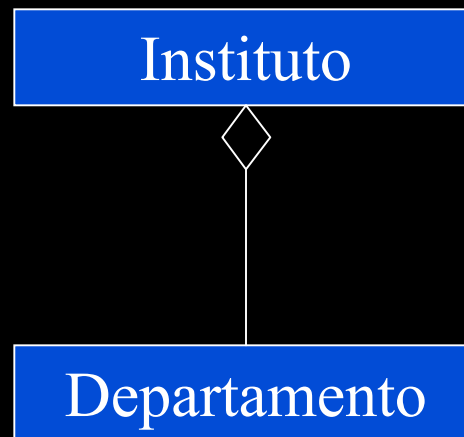
- Valores possíveis: valor exato, intervalo, ou * para “muitos”.

- Exemplo:



Agregação

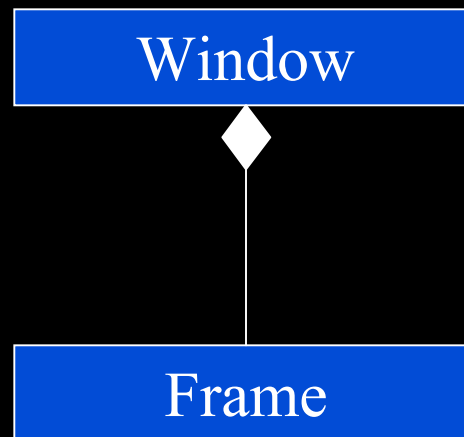
É uma relação do tipo “todo/parte” ou “possui um” na qual uma classe representa uma coisa grande que é composta de coisas menores.



(diamante vazio)

Composição

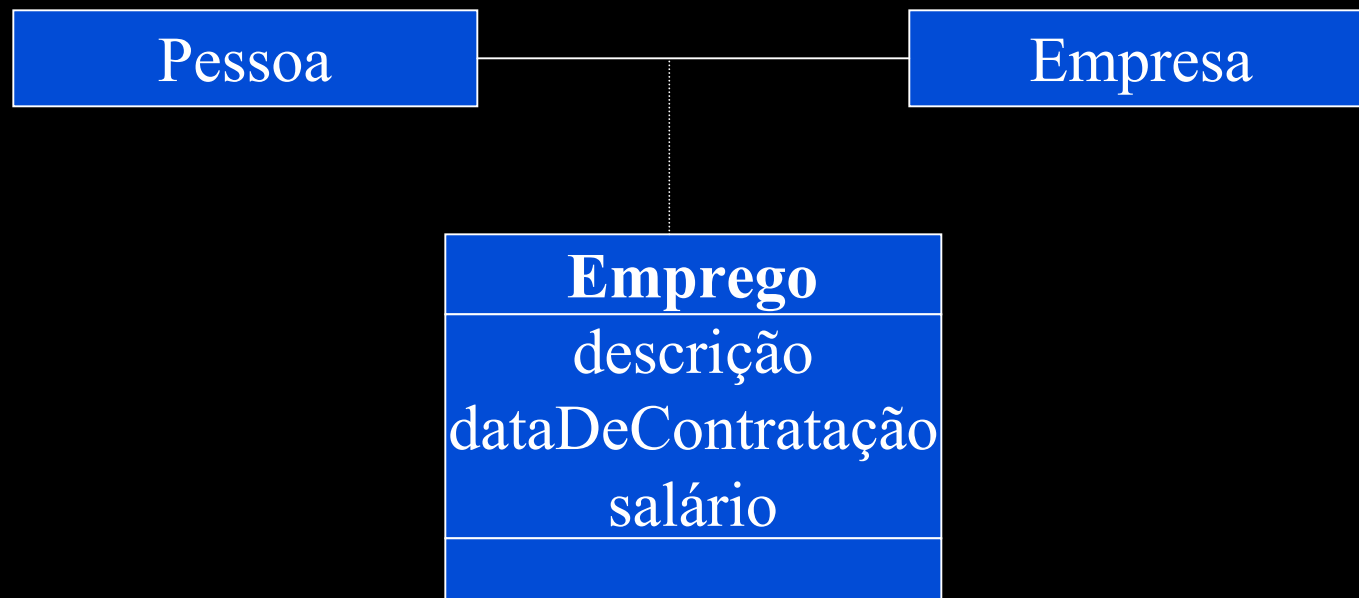
É um tipo especial de agregação na qual as partes são inseparáveis do todo.



(diamante cheio)

Classes de Associação

Uma classe de associação possui as propriedades de classes e de associações:



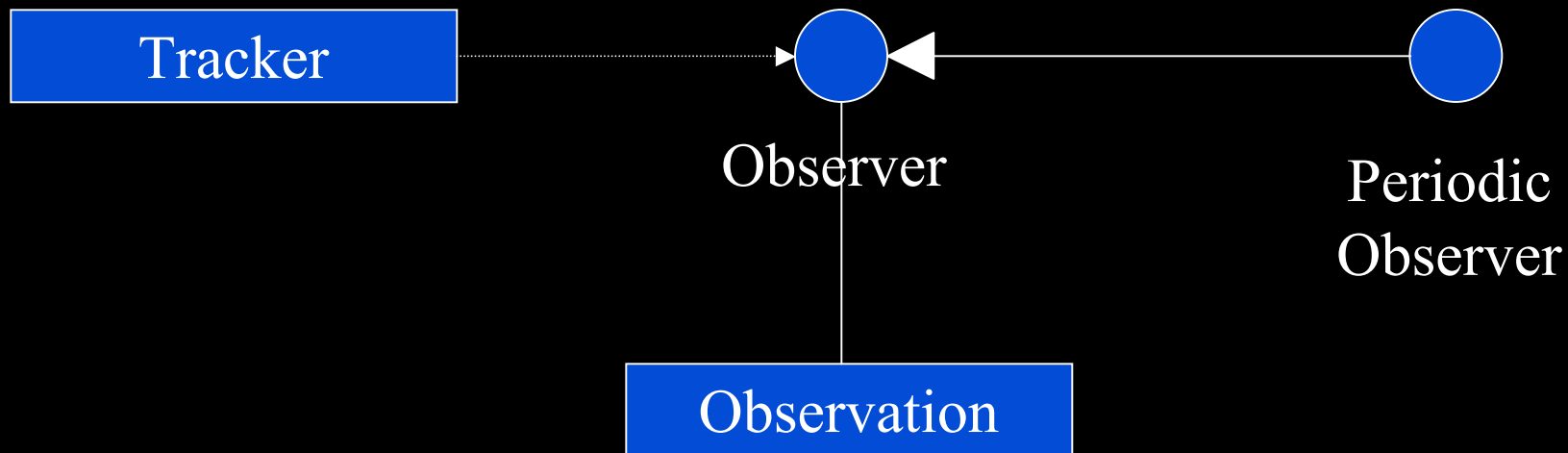
Interfaces

É uma coleção de operações que possui um nome. É usada para especificar um tipo de serviço sem ditar a sua implementação.



Interfaces e Relacionamentos

Uma interface pode participar de generalizações, associações e dependências.

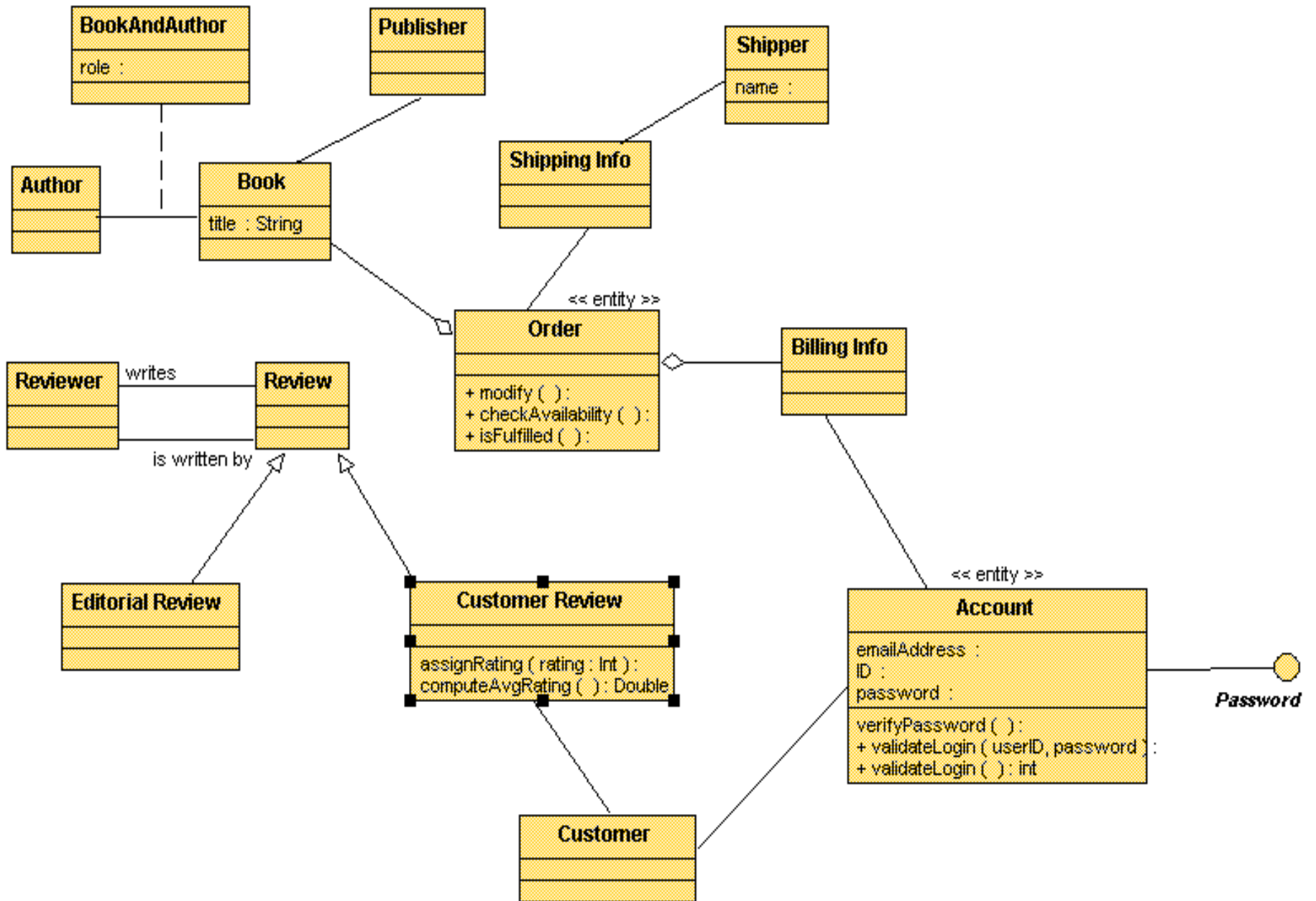


Realização

É uma relação entre uma interface e a classe que a implementa, i.e., que provê o serviço definido pela interface.



Uma classe pode realizar (implementar) várias interfaces.



Ornamentos e Extensibilidade

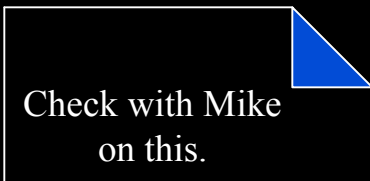
Um ornamento é algo como uma nota que adiciona texto ou algum elemento gráfico ao modelo.

UML oferece vários mecanismos que podem ser utilizados para estender a linguagem “oficial”.

- estereótipos
- valores rotulados (*tagged values*)
- restrições

Notas

É um símbolo gráfico contendo texto ou figuras oferecendo algum comentário ou detalhes sobre um elemento de um modelo.



Check with Mike
on this.



See [http://www.
softdocwiz.com](http://www.softdocwiz.com)



See encrypt.ps

Estereótipos

É uma extensão do vocabulário de UML que permite a criação de um tipo básico novo que é específico ao problema que está sendo resolvido.



Estereótipos Padrão em UML

cerca de 50, incluindo:

- *become* (indica uma dependência na qual um objeto se torna outro)
- *enumeration* (especifica um tipo enumerado incluindo seus possíveis valores)
- *utility* (uma classe na qual todos os valores e atributos pertencem à classe (e não às suas instâncias))

Valores Rotulados

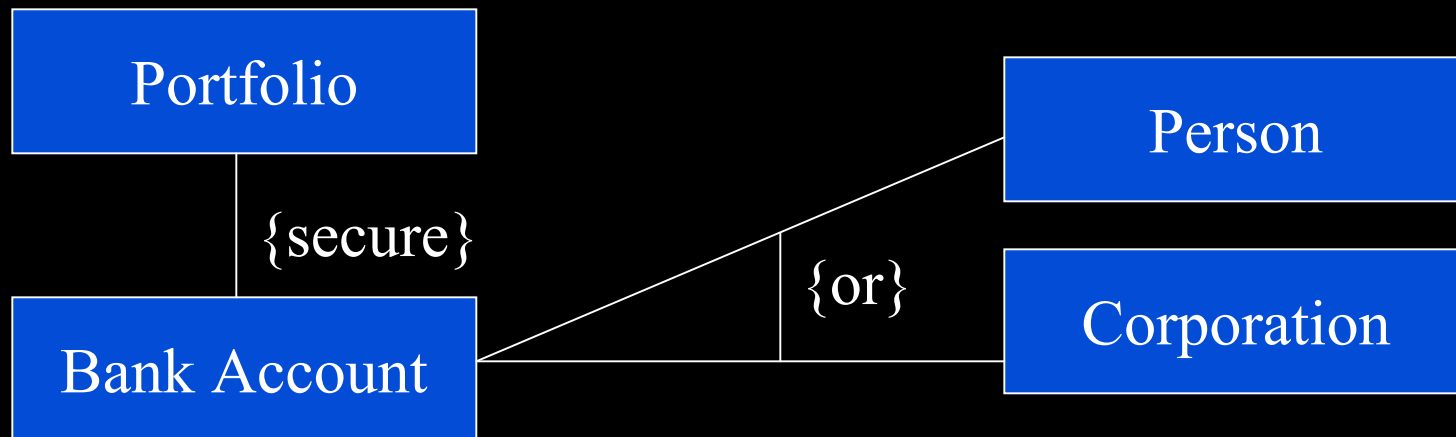
Permite a especificação de propriedades de elementos de um modelo:

```
GL Account  
{persistent}
```

```
TargetTracker  
{release = 2.0}
```

Restrições

Especifica uma condição que deve ser satisfeita pelo sistema.



Pacotes

- Um mecanismo para organizar elementos de um modelo (classes, diagramas, etc.) em grupos.
- Cada elemento de um modelo pertence a um único pacote. O seu nome dentro do pacote deve ser único.

Um Diagrama de Pacotes

- Arcabouço para construção de sistemas distribuídos adaptativos (de Francisco Silva²).

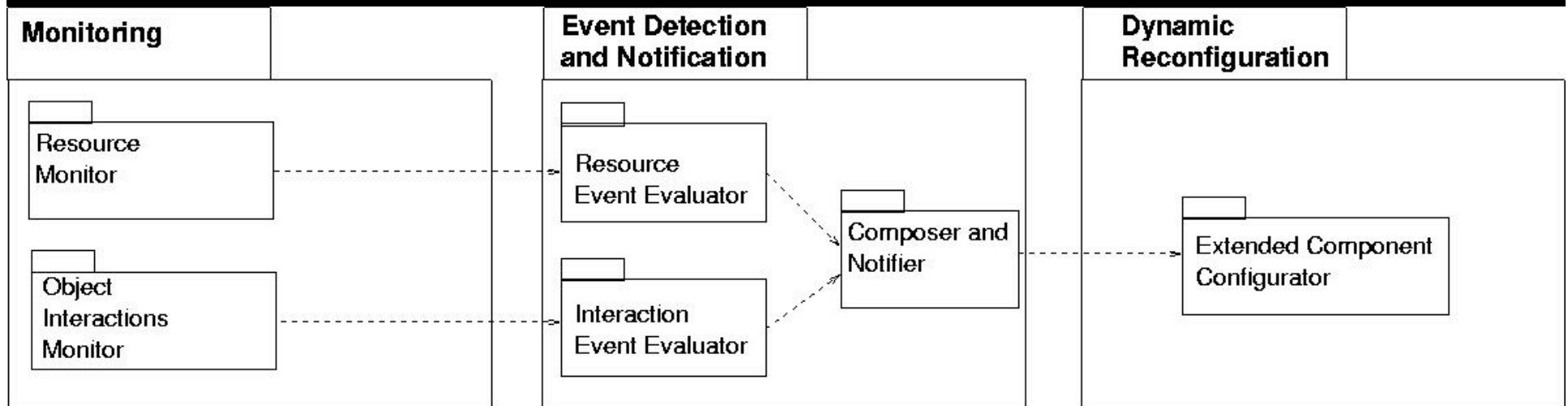
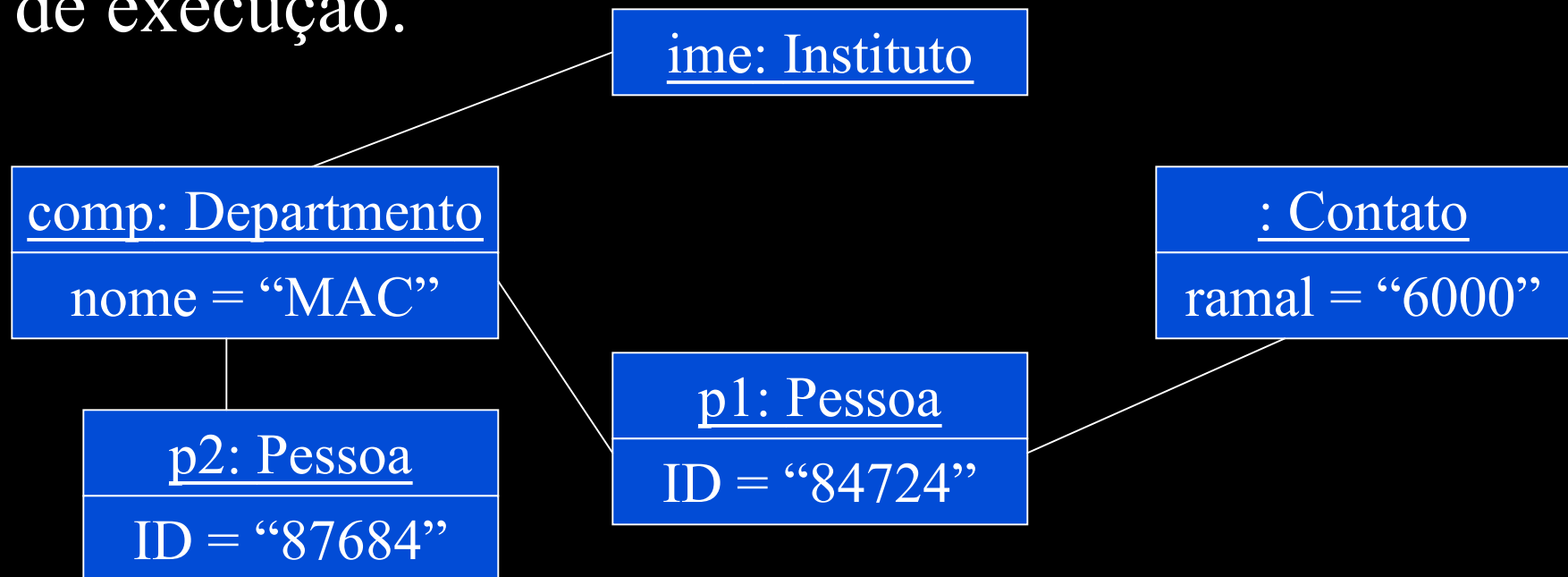


Diagrama de Objetos

Mostra um conjunto de objetos e seus relacionamentos em um certo instante em tempo de execução.



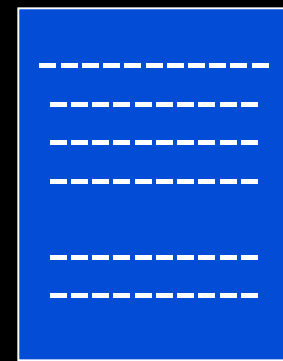
Componente

É uma parte de um sistema que pode ser substituída e que oferece uma implementação de um conjunto de interfaces.

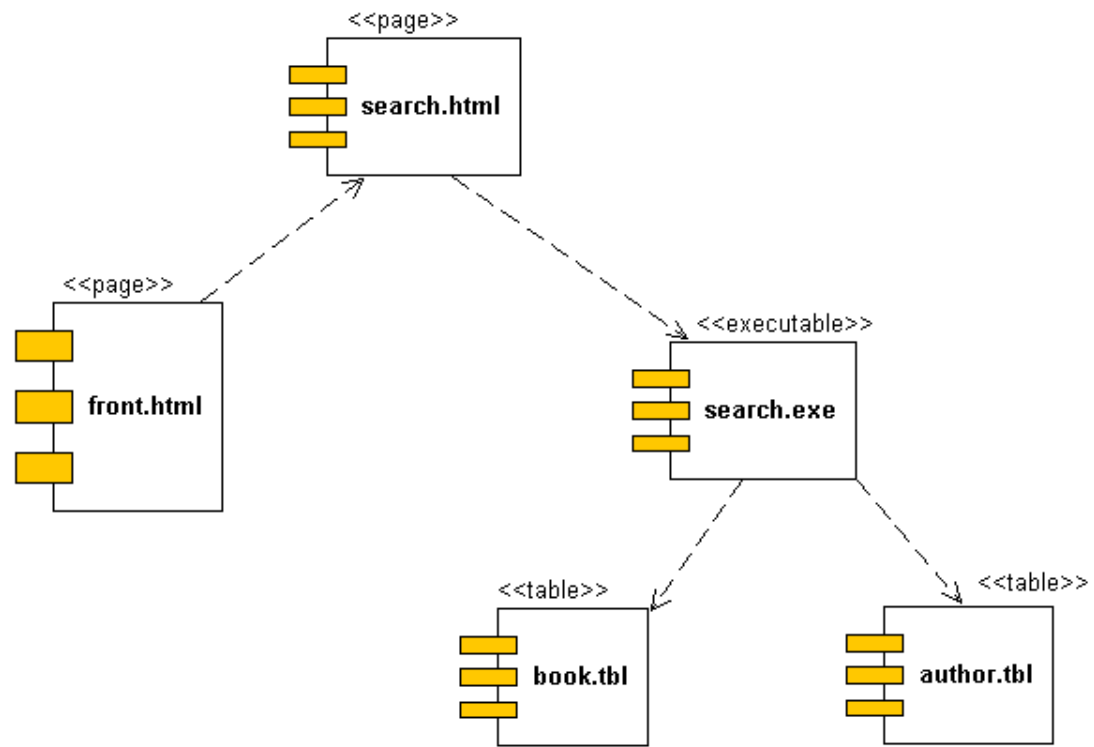
Exemplos práticos:

- Biblioteca de carga dinâmica (DLL)
- Componente CORBA
- Enterprise Java Bean (EJB)

Notação para Componentes



signal.cpp



Nó



Representa um elemento físico capaz de oferecer recursos computacionais.

Em geral, possui pelo menos memória e processador.

Diagrama de Implantação

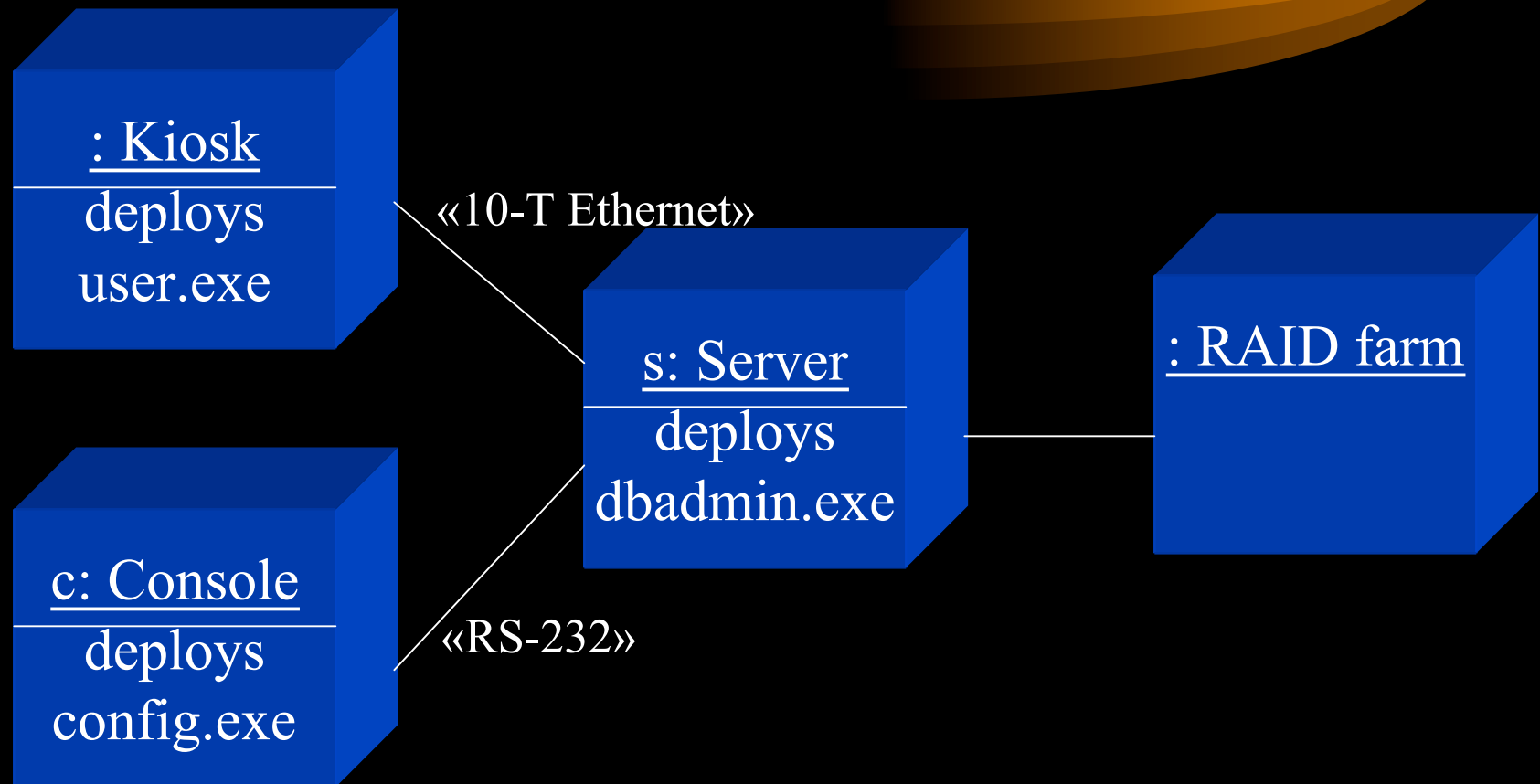
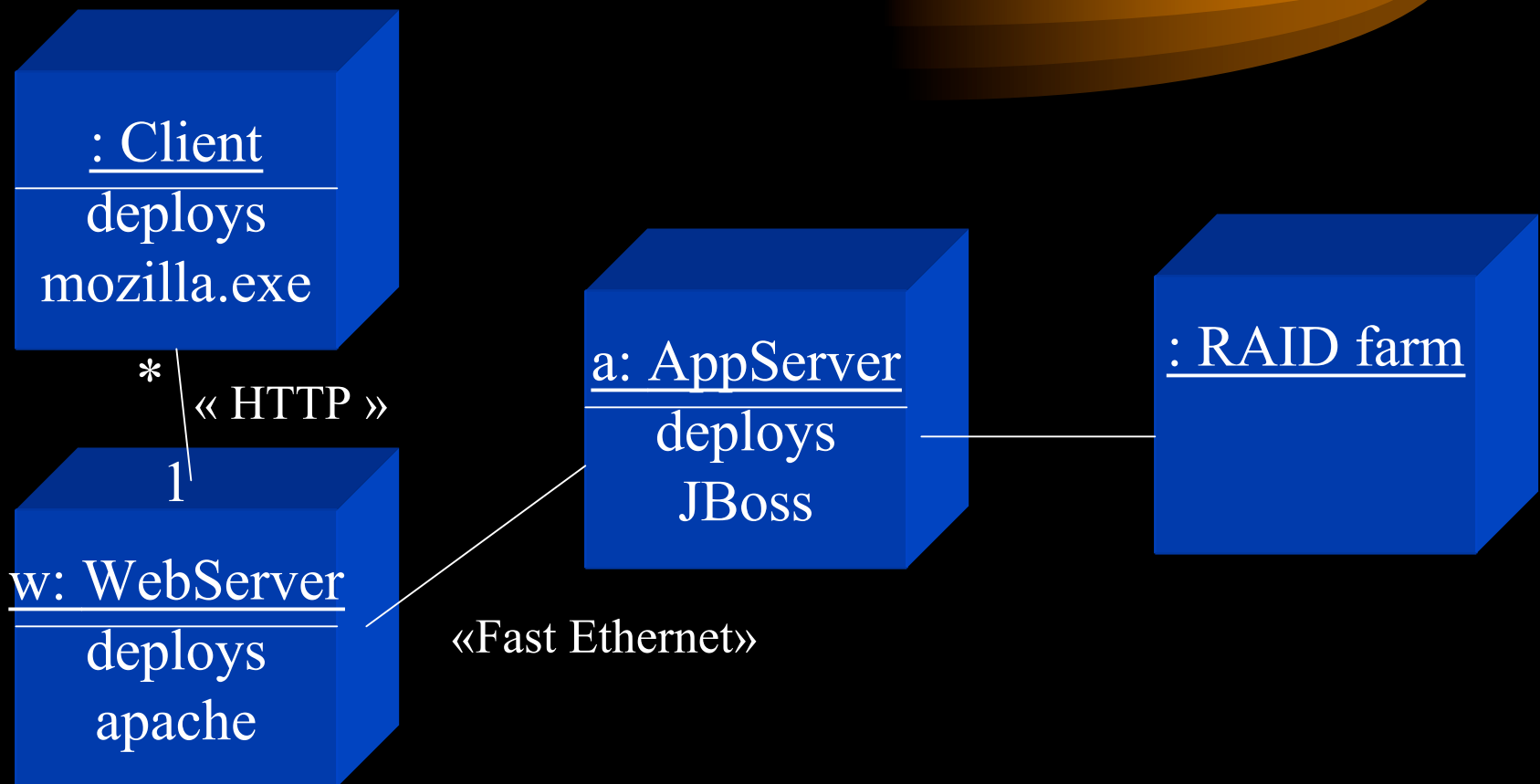


Diagrama de Implantação



Diagramas Comportamentais

Usados para visualizar, especificar, construir e documentar aspectos **dinâmicos** de um sistema

- diagrama de casos de uso
- diagrama de seqüência
- diagrama de colaboração
- diagrama de estados
- diagrama de atividades

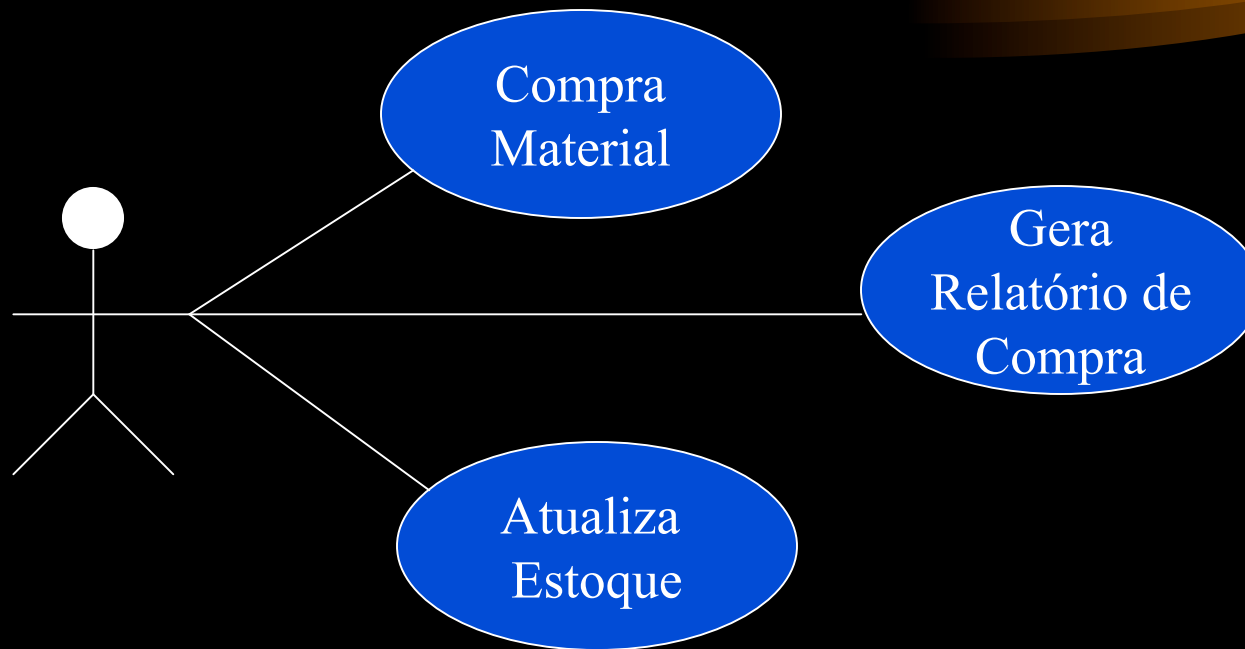
Casos de Uso e Atores

- Um caso de uso é uma seqüência de ações, incluindo variantes, que um sistema realiza a fim de gerar um resultado observável de interesse para um ator.
- Um ator é um papel (ou conjunto de papéis) que um usuário desempenha quando participa de um caso de uso.

Fluxos de Eventos

- O fluxo de eventos principal descreve o caso em que tudo corre bem.
- Fluxos de eventos excepcionais cobrem as variações que podem ocorrer quando diferentes coisas dão errado ou quando algo pouco comum acontece.

Um Diagrama de Caso de Uso



Organização de Casos de Uso



- pacotes
- generalização
- inclusão
- extensão

Pacotes de Casos de Uso

Pode ser útil para distribuir trabalho para sub-grupos de trabalho.

Estoque

Cria
Novo
Estoque

Visualiza
Estoque

Agrega
Estoques

Gera
Relatório
Mensal

Generalização



- Análoga à generalização/especialização de classes.



Inclusão

- O estereótipo «include» indica que um caso inclui o outro.
- Permite fatorar comportamento comum a vários casos.

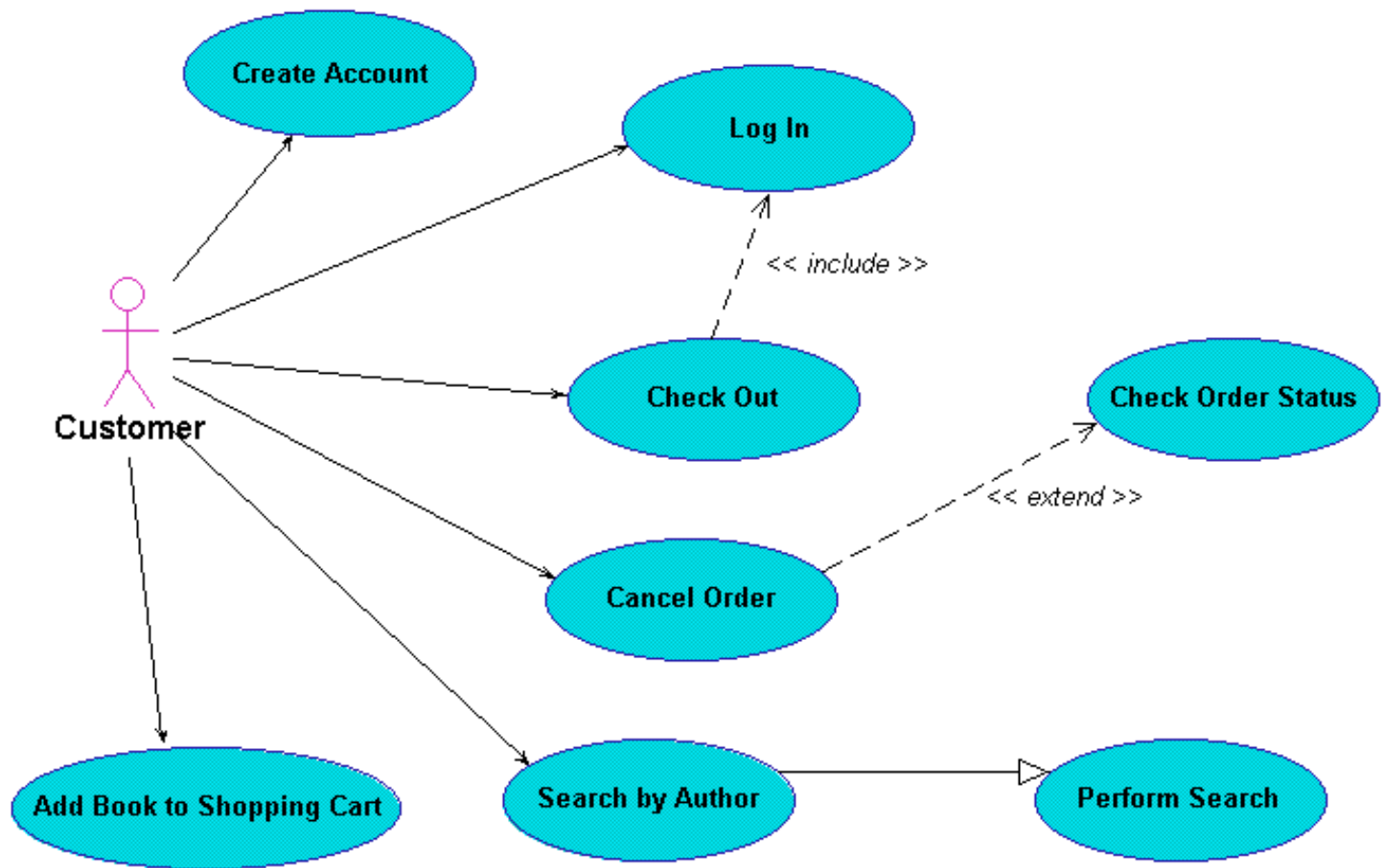


Extensão

Pode-se usar o estereótipo «extend» para indicar que um caso estende o outro.

Útil para fatorar comportamento incomum/não-padrão.





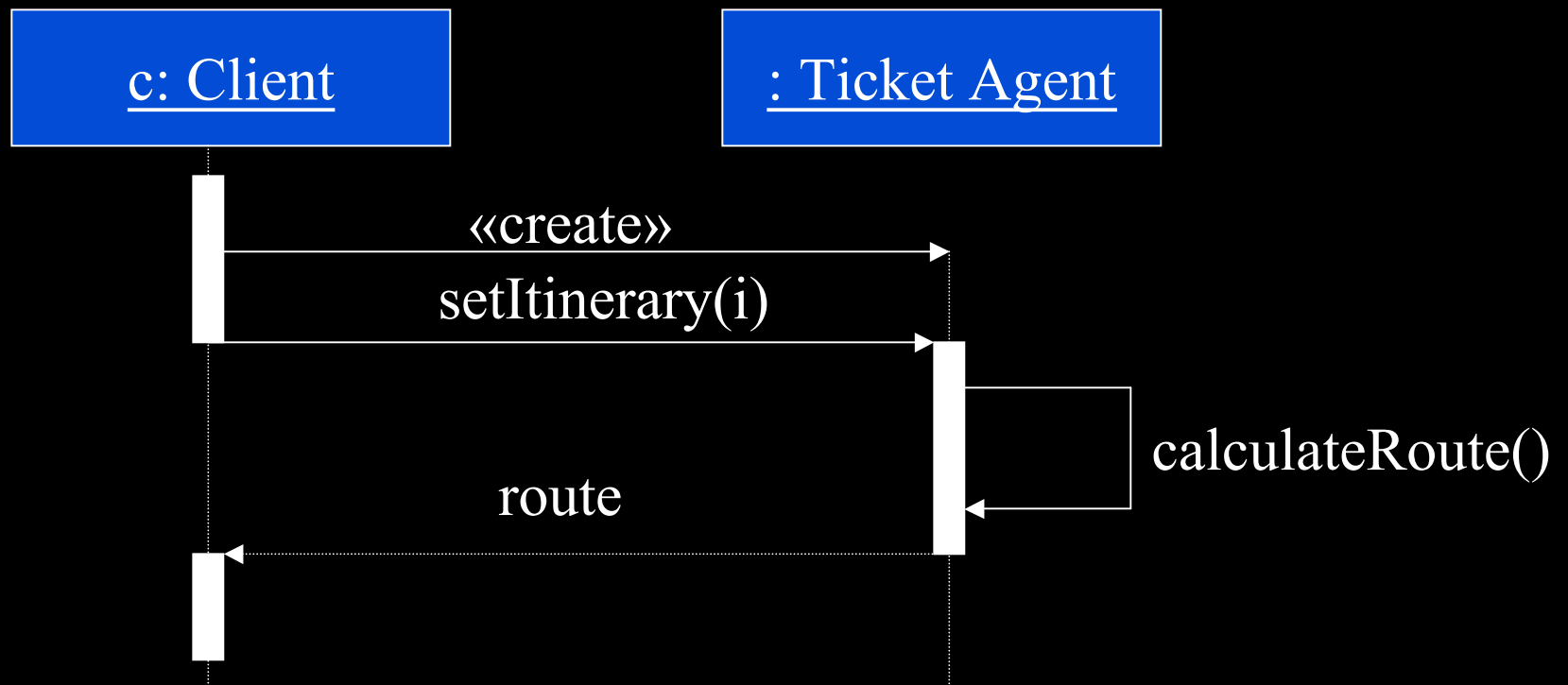
Interações e Mensagens

- Uma interação é um comportamento composto da troca de um conjunto de mensagens entre um grupo de objetos a fim de atingir um determinado objetivo.
- Uma mensagem é uma comunicação entre objetos que resulta na transmissão de informação com o intuito de que alguma atividade será realizada.

Diagrama de Seqüência

- É um *diagrama de interações* que enfatiza a ordem temporal das mensagens.
- Uma *linha de vida* é uma linha tracejada vertical que representa o tempo de vida de um objeto.
- Um *foco de controle* é um retângulo fino vertical sobreposto à linha de vida que mostra o período durante o qual um objeto está realizando uma ação.

Diagrama de Seqüência



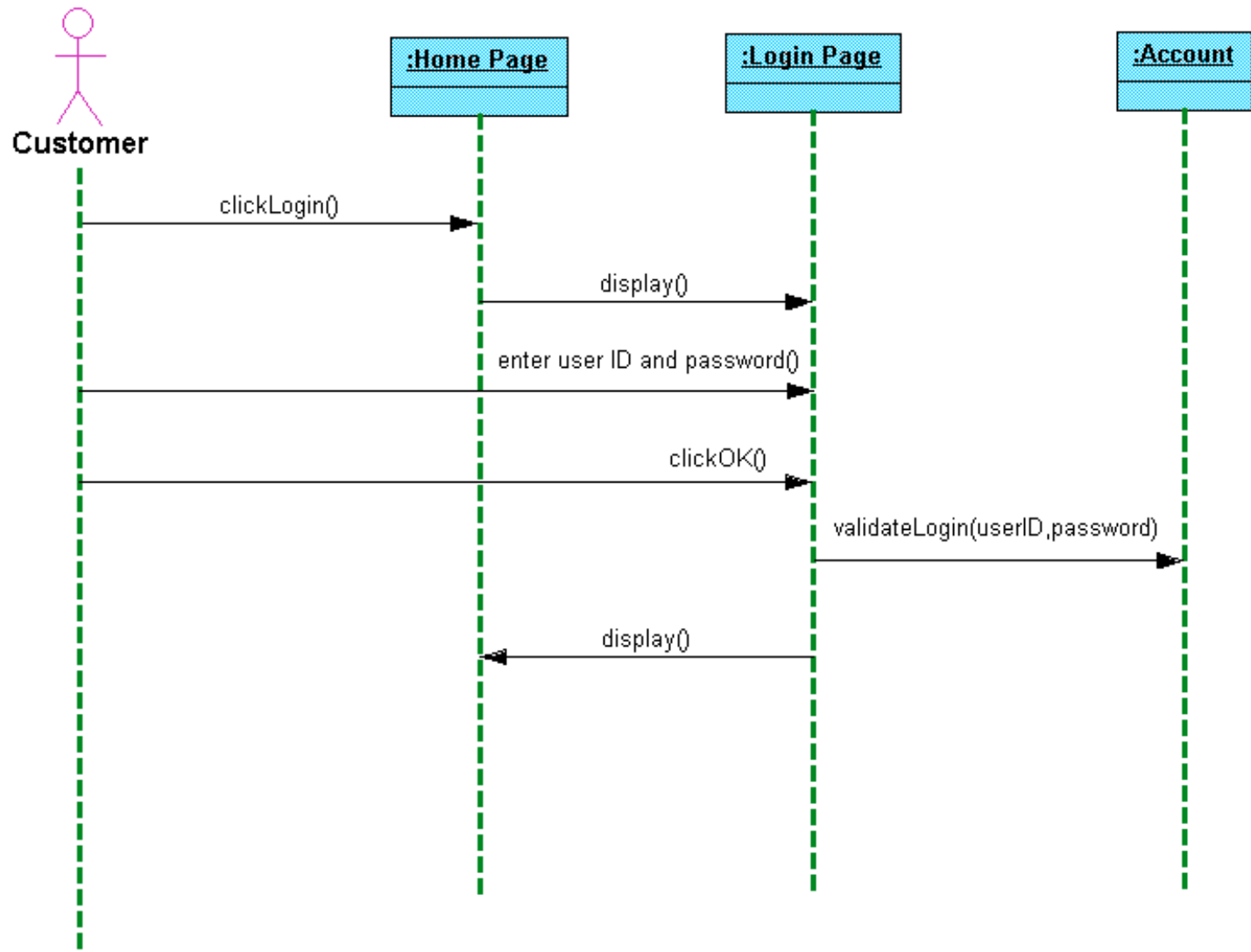
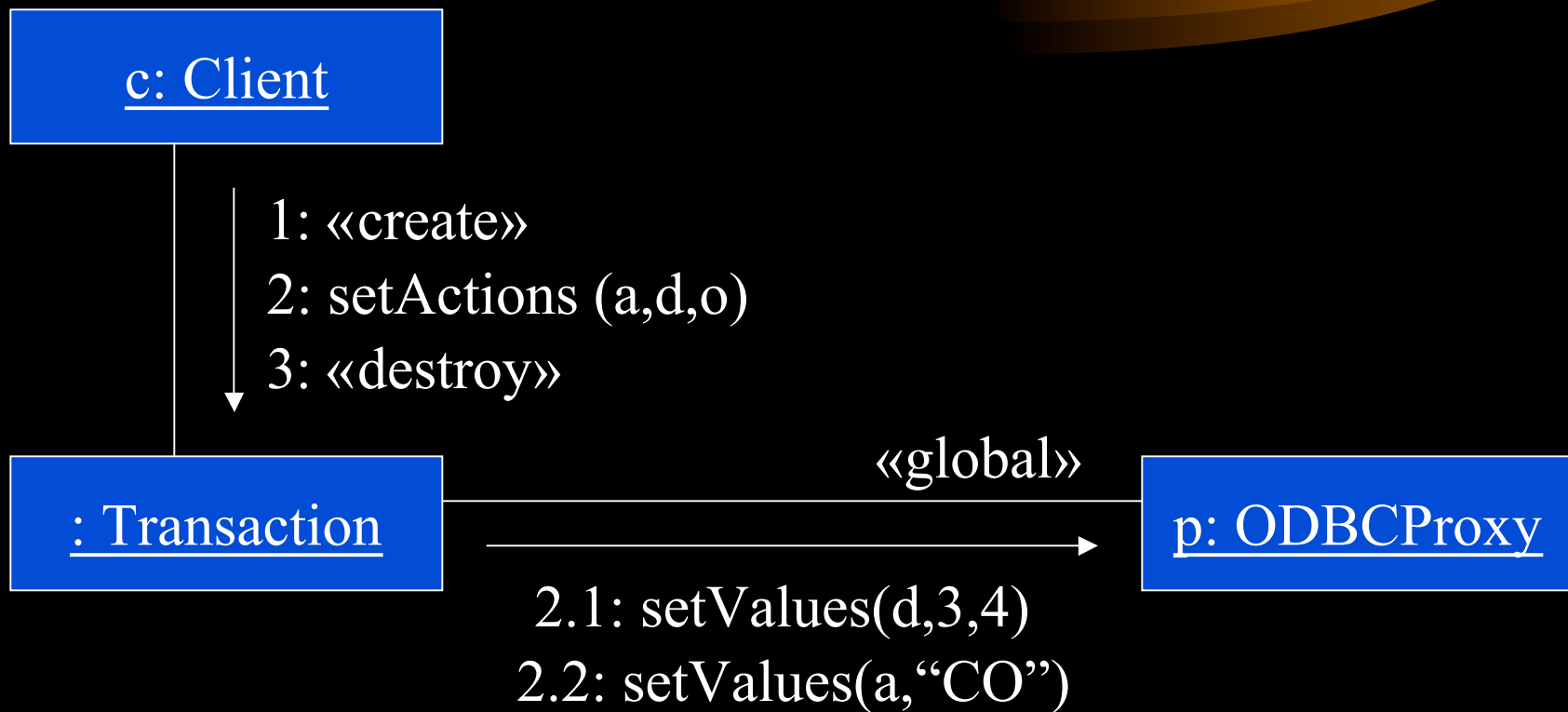


Diagrama de Colaboração

- É um diagrama de interação que enfatiza a organização dos objetos que participam da interação.
- Um *caminho* é uma ligação entre objetos, possivelmente com um estereótipo «local».
- Números de seqüência indicam a ordem temporal das mensagens em um ou mais níveis.

Diagrama de Colaboração



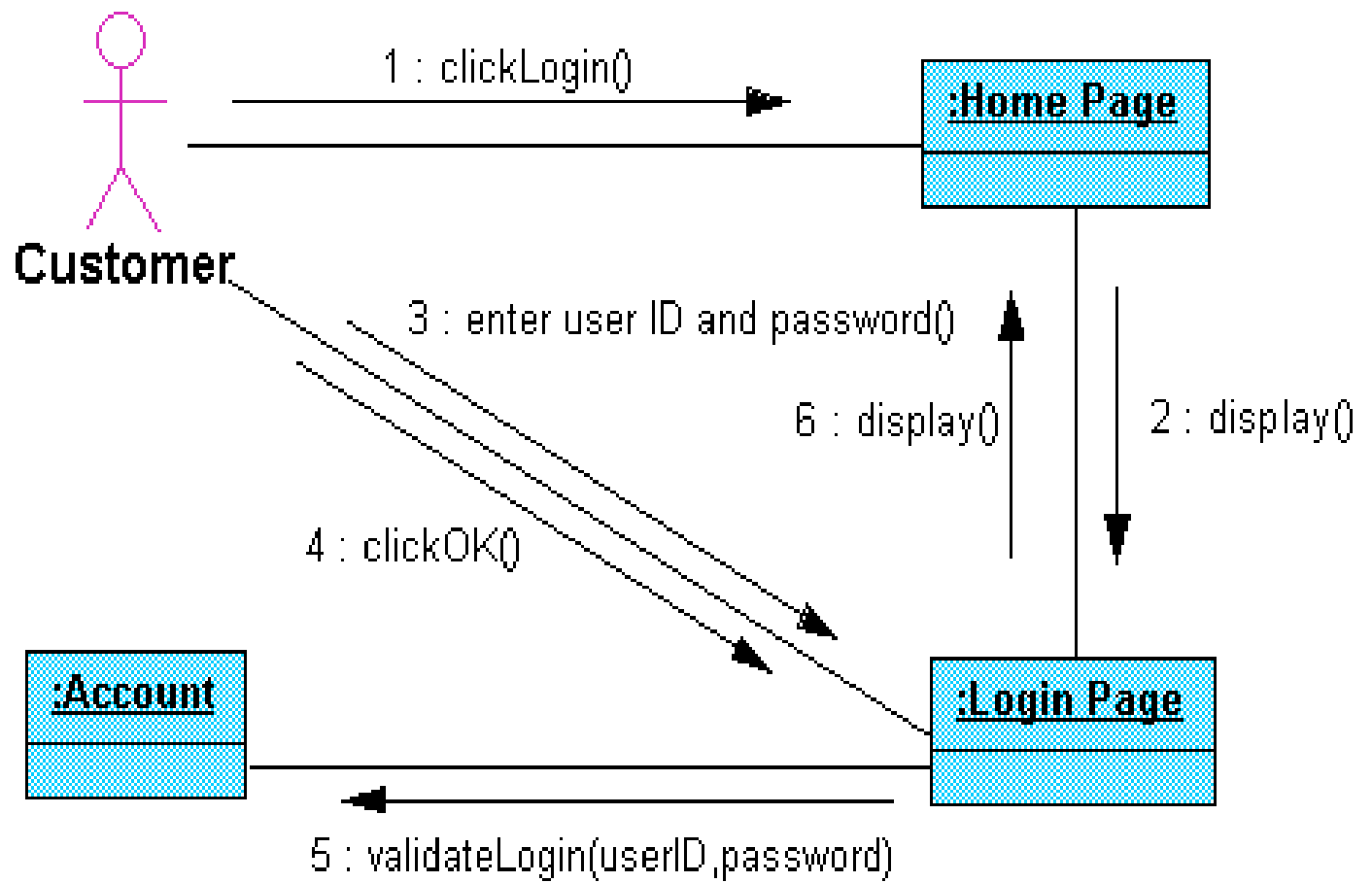


Diagrama de Estados



- Diagrama de Estados
- Representa
 - os possíveis estados que um sistema pode assumir e
 - as possíveis transições entre os estados

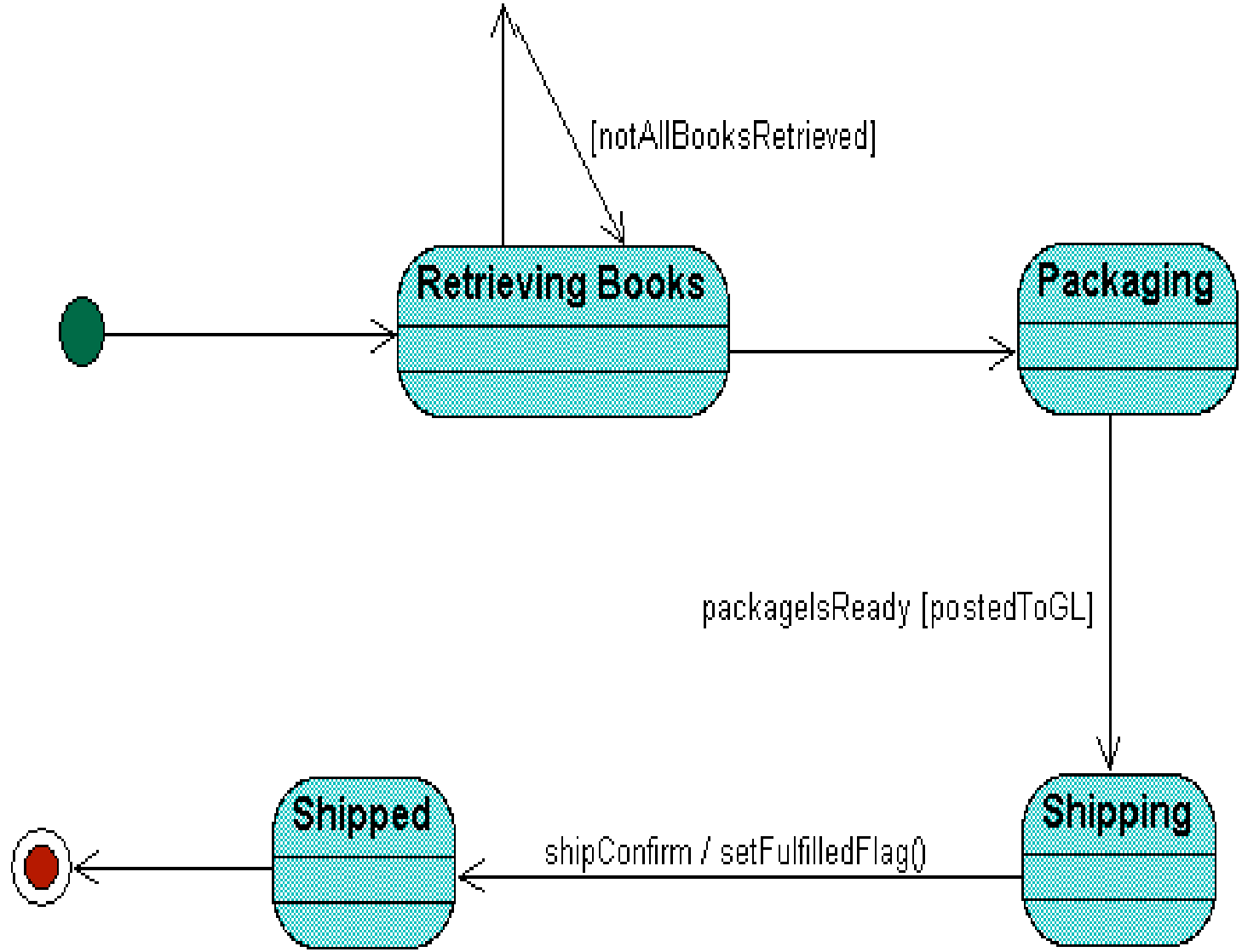
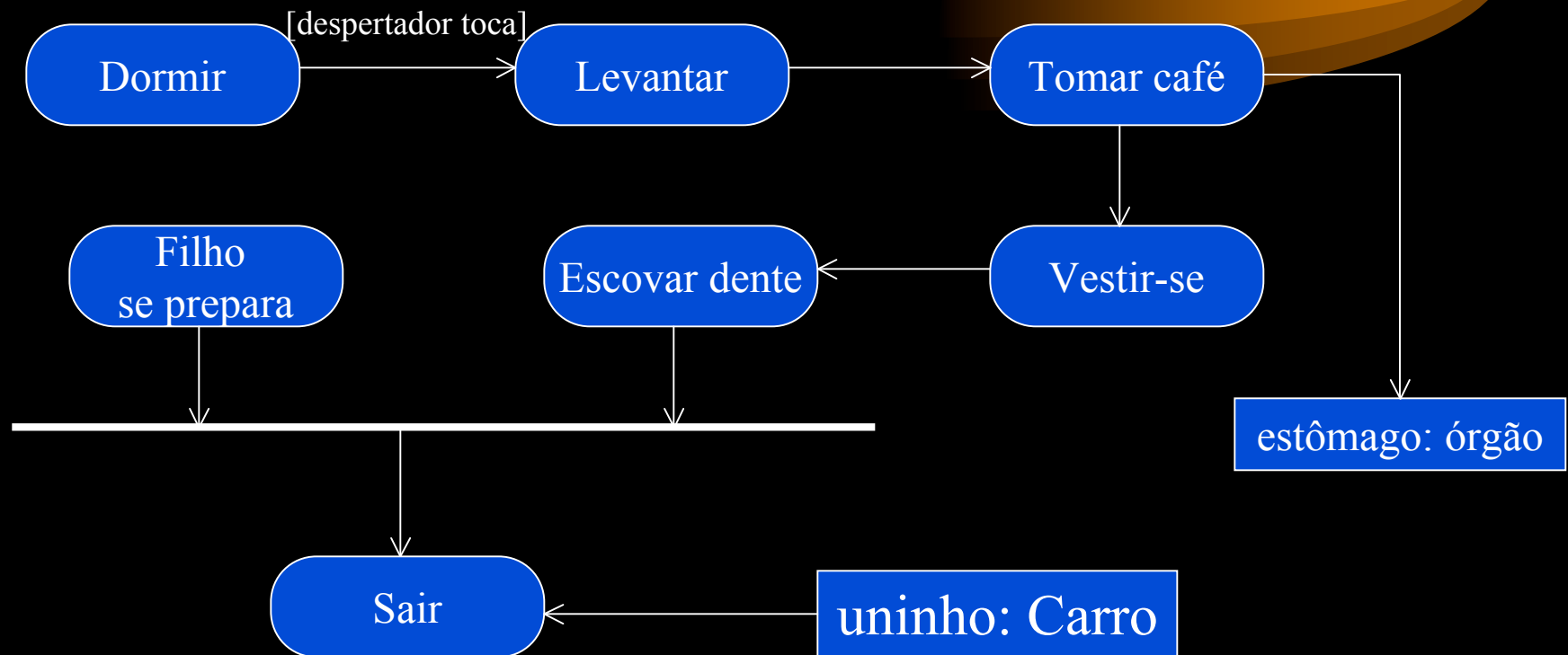


Diagrama de Atividades

- É um tipo específico de Diagrama de Estados
- Útil para modelar fluxo de trabalho (*workflow*)
- Representa as atividades que afetam o estado do sistema e os fluxos que levam de uma atividade a outra.
- É usado para modelar processos de negócio e workflows

Exemplo de Diagrama de Atividade



That's All Folks!



- Por hoje é só pessoal!