Diagrama de Componentes

- Os diagramas de componentes capturam a estrutura física da implementação

- Têm como objectivo
  - Organizar o código fonte (ambiente de desenvolvimento)
  - Construir uma release executável (ambiente de produção)
  - Especificar componentes como base de dados, etc.

- Contém componentes, interfaces e relações entre componentes

- Os pacotes de componentes podem ser utilizados para modelar a arquitectura física
  - Identificar as principais *peças* do sistema

Diagrama de Componentes

- O que é um componente?
  - Um pedaço de software reutilizável, bem encapsulado e "*facilmente*" substituível.
  - São blocos (peças) que combinados constroem o sistema pretendido.
  - A dimensão dos componentes não é homogénea, existindo num mesmo sistema, componentes de diferentes dimensões.
- Quais são os bons candidatos a serem componentes do sistema?
  - Items que desempenham uma funcionalidade que é utilizada recorrentemente no sistema
    - Exemplos: componentes de *logging*, parsers de XML, componentes de gestão de carrinhos de compra (*shopping carts*), etc.
- Em UML um componente pode efectuar as mesmas funcionalidades que uma classe faz
  - Generalização
  - Associação com outros componentes ou classes
  - Implementação de interfaces
- Um componente representa um empacotamento físico de elementos relacionados logicamente (normalmente classes)

Diagrama de Componentes

- Um componente é representado em UML como uma caixa com o estereótipo **<<component>>** e com um ícone no canto superior direito (opcional).

- Em notações anteriores do UML o símbolo representativo de um componente era ligeramente diferente, apresentado os tabs de forma mais evidente.

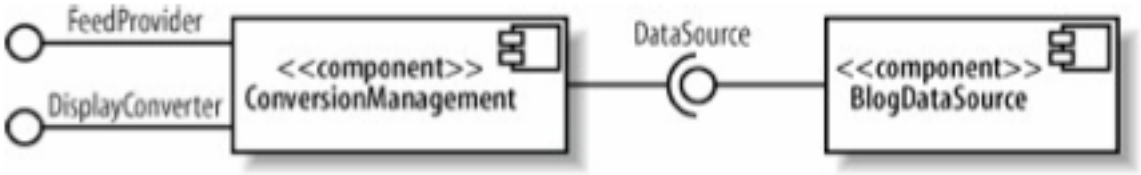
Diagrama de Componentes

- Os componentes por forma a serem facilmente trocados devem ser independentes uns dos outros
  - Comprometem-se com uma API que devem implementar
  - Em termos de desenvolvimento, significa que os componentes implementam interfaces, o que possibilita que do ponto de vista do sistema as trocas de componentes sejam pacíficas.
  - Classes e outros componentes dialogam com um determinado componente através da(s) interface(s) implementada(s)

- **Relação de concretização** (*realization*): um componente pode concretizar (implementar os serviços de) uma ou mais interfaces
  - Normalmente quer dizer que tem classes que implementam esses interfaces
  - Diz-se que as interfaces são exportadas
  - Um componente poder ser substituído por outro componente que implementa as mesmas interfaces
- **Relação de dependência**: um componente pode usar uma ou mais interfaces
  - Diz-se que essas interfaces são importadas
  - Um componente que usa outro componente através de uma interface bem definida, não deve depender da implementação (do componente em si), mas apenas da interface

Diagrama de Componentes

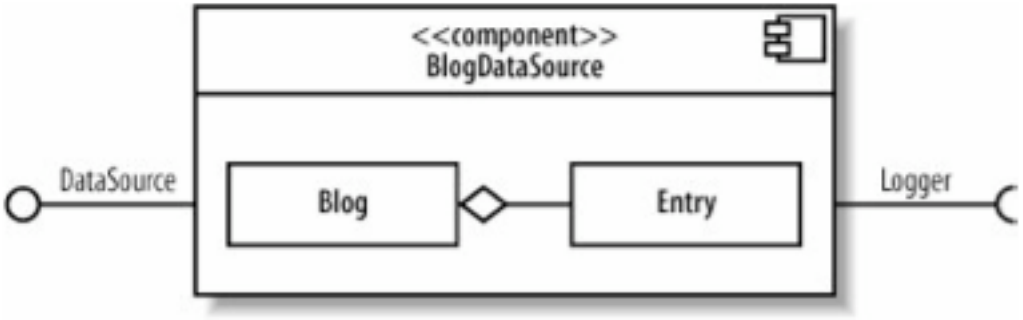
- Notação para representar a implementação de uma interface



- É possível encontrar o mesmo pedaço de diagrama com a notação alternativa

Diagrama de Componentes

- Os componentes usualmente contém classes e estas podem ser representadas no diagrama.



- Ou, em notação alternativa

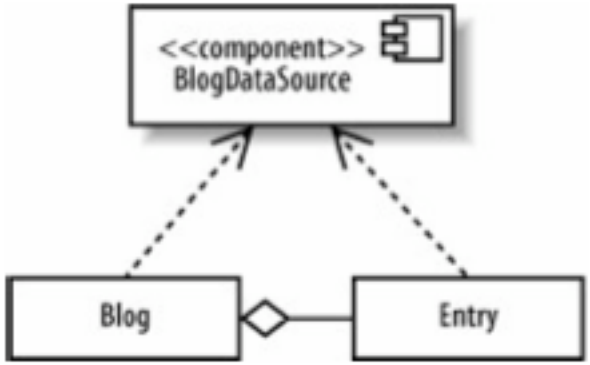
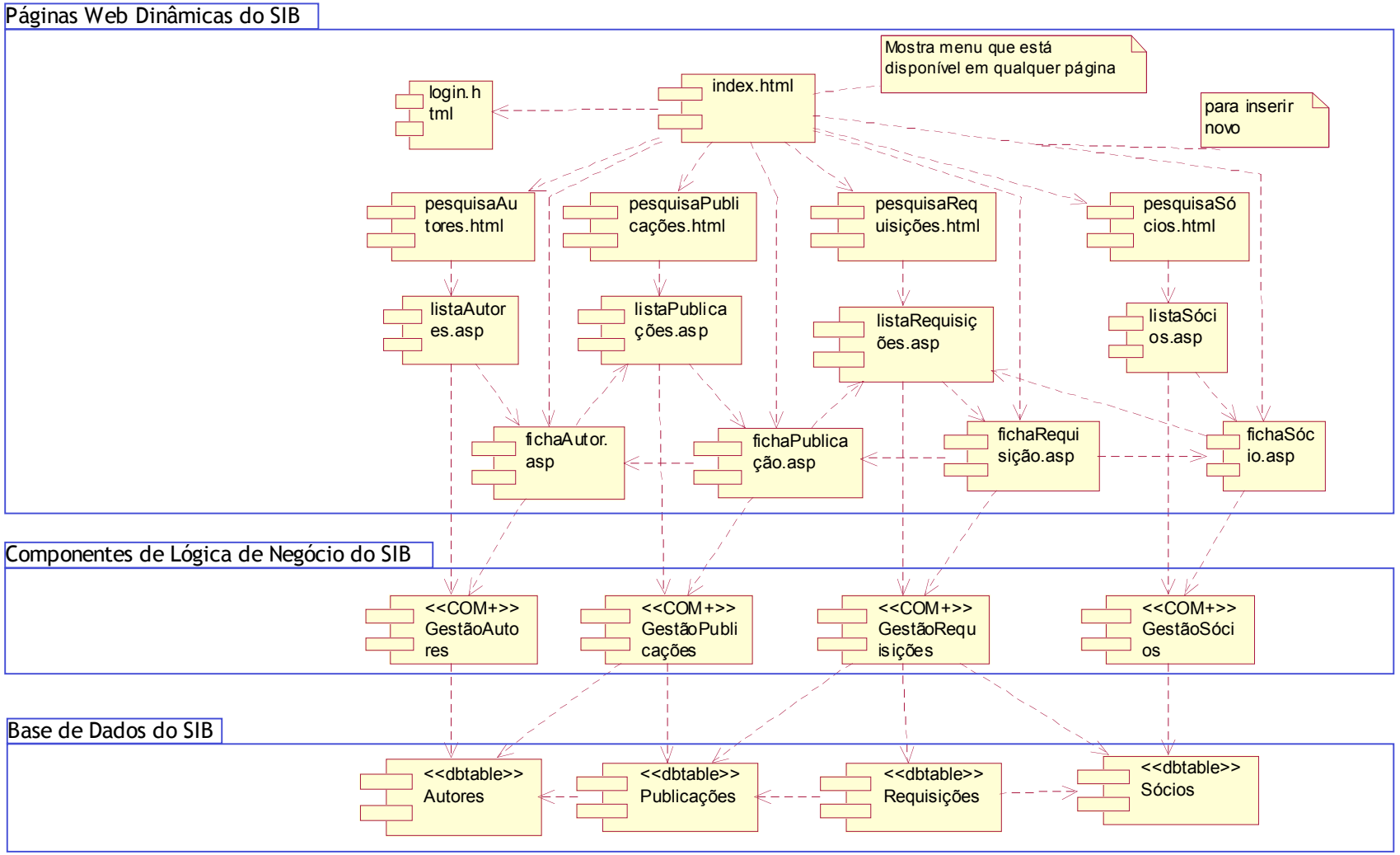


Diagrama de Componentes

Caso de estudo de um sistema de gestão da biblioteca (notação UML 1.x)



Diagramas de Deployment

- Captura a topologia (ambiente) de hardware de um sistema sobre a qual são executados os componentes de software

- Construído como parte da especificação da arquitectura física

- Objectivo:
  - Especificar a distribuição de componentes
  - Identificar estrangulamentos de desempenho

- Permitem que a equipa de engenheiros especifique a disposição física dos elementos que constituem o sistema
  - Acrescenta detalhe que tem a ver com a configuração do sistema em tempo de execução
  - Permite cruzar competências de engenharia de software com redes de comunicações, sistemas operativos e bases de dados

Diagramas de Deployment

- Elementos de um diagrama de deployment
  - Nós
  - Ligações

- **Nós:**
  - Computadores ou outros dispositivos
  - Existem nós que são nós de hardware (server, desktop, disk drives) ou nós de ambiente de execução (sistema operativo, web server, application server, etc.)
  - Os componentes localizados (deployed) em cada nó são representados explicitamente
  - É possível agrupar nós em pacotes (packages)

- **Ligações:**
  - Entre nós (podem ser decoradas com multiplicidades)
  - Podem ter estereótipos que indicam o tipo de ligação. Exemplo: <<TCP/IP>> ou <<RMI>>

Diagramas de Deployment

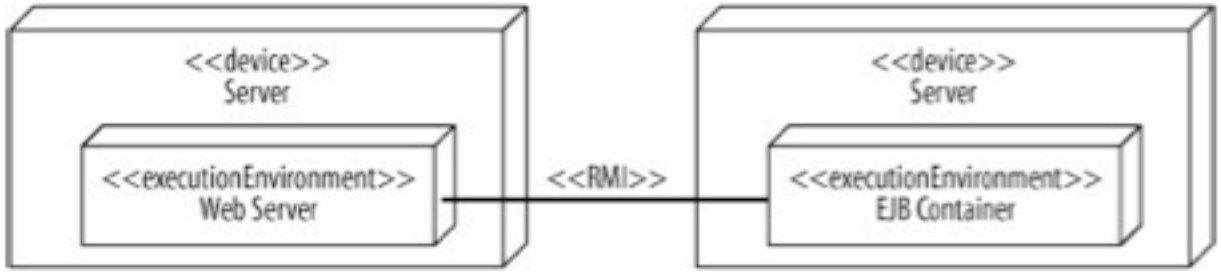
- Por vezes utiliza-se o estereótipo <<device>> para identificar os nós de hardware

- Para identificar os ambientes de execução utiliza-se o estereótipo <<executionEnvironment>>

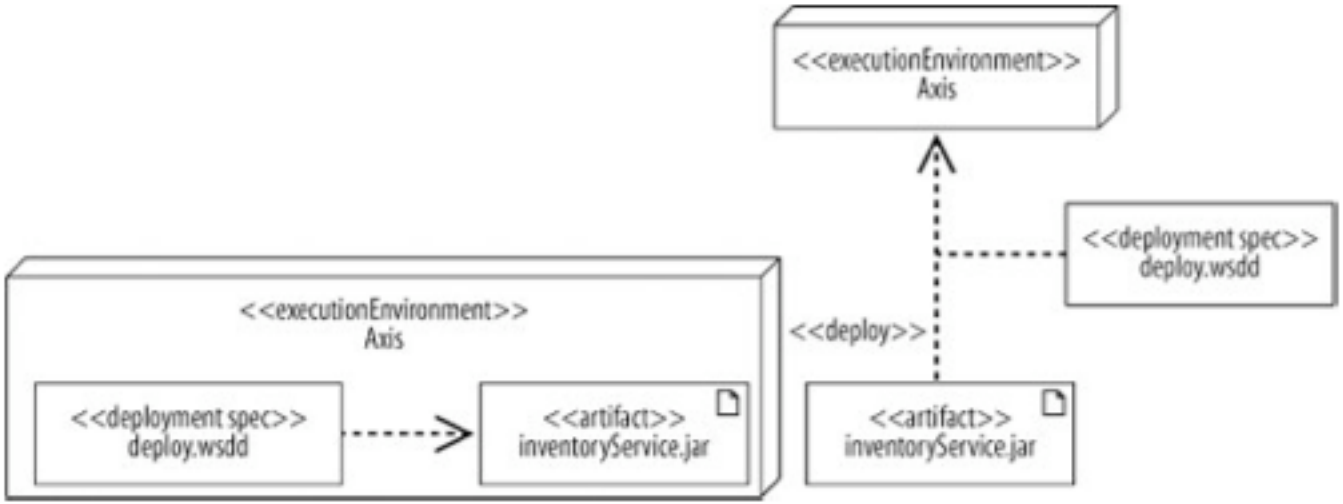
- Comunicação entre dois nós

Diagramas de Deployment

- A descrição pode ser refinada para detalhar os ambientes de execução em cada nó



- Especificação de dependências em tempo de execução



Diagramas de Deployment

- Um diagrama mais completo