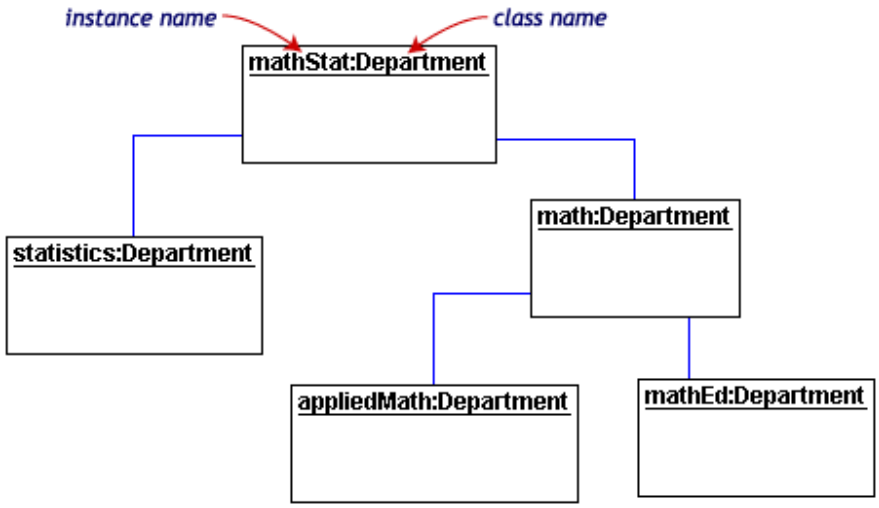
Diagramas de Objectos

Diagramas de Objectos

- Diagrama de objectos descreve a estrutura estática do sistema num instante particular
  - Diagrama de classes descreve todas as situações possíveis
- Elementos do diagrama de objectos:
  - **Objectos**, que representam instâncias particulares de classes
  - **Links,** que representam relações específicas entre objectos. São instâncias de associações

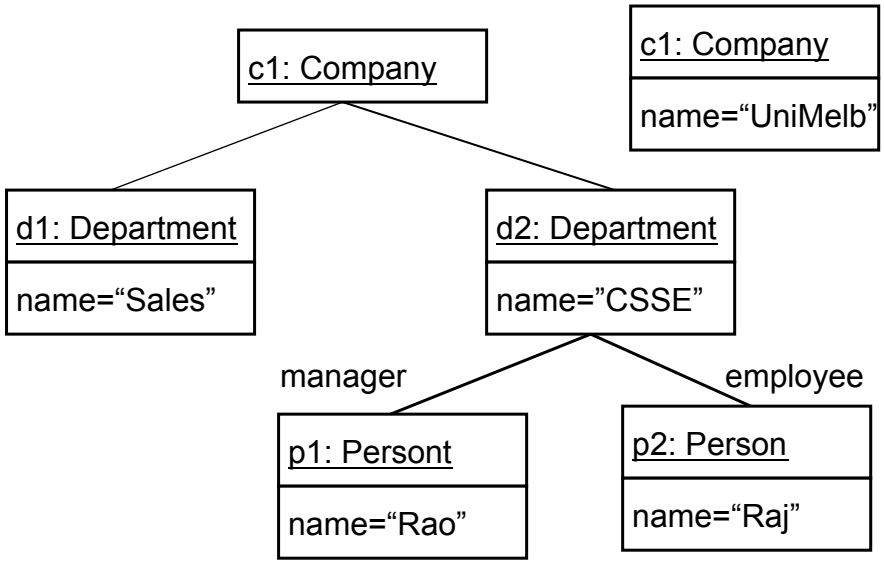
2

Diagramas de objectos



3

Diagrama de objectos: Exemplo



Qual o diagrama de classes de que deriva?

4

Diagramas de Pacotes

Diagrama de pacotes: objectivo

- Um diagrama de pacotes serve para:
  - gerir a complexidade dos diagramas, agrupando elementos desses diagramas em **packages**.
  - criar diagramas de alto nível de abstracção (de uma coleção de casos de uso, de classes, etc)
- Um pacote (package) é uma coleção de elementos UML logicamente relacionados.
- Um diagrama de pacotes é composto apenas por pacotes e dependências entre eles.

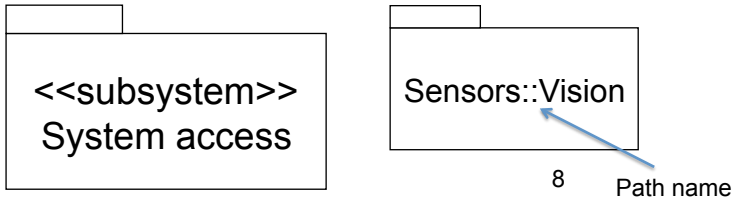
Diagrama de pacotes: introdução

- Um pacote é um mecanismo de propósito geral para organizar elementos em grupos
  - Imprescindível em grandes sistemas
- Um pacote pode conter outros elementos incluindo classes, interfaces, diagramas, componentes, nós, casos de uso e outros pacotes
  - Subsistemas agrupam objectos (e outros subsistemas), reduzindo a complexidade de um sistema
- Deve-se evitar aninhamento excessivo
- A utilização de pacotes:
  - Facilita a gestão e procura de elementos do modelo
  - Evita os conflitos de nomes
  - Possui um mecanismo de visibilidade

7

Pacotes: notação

- É representado graficamente por uma pasta (*tabbed folder*)
- Todo *package* tem um nome que o distingue de outros *packages*
- Cada elemento só pertence a um *package*
- Um *path name* é o nome de um *package* prefixado pelo nome do *package* no qual o *package* se encontra



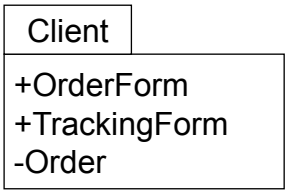
8

Visibilidade

+ (public): um elemento de um pacote *X* é público se é visível para os elementos de um pacote *Y* que importa o pacote *X*

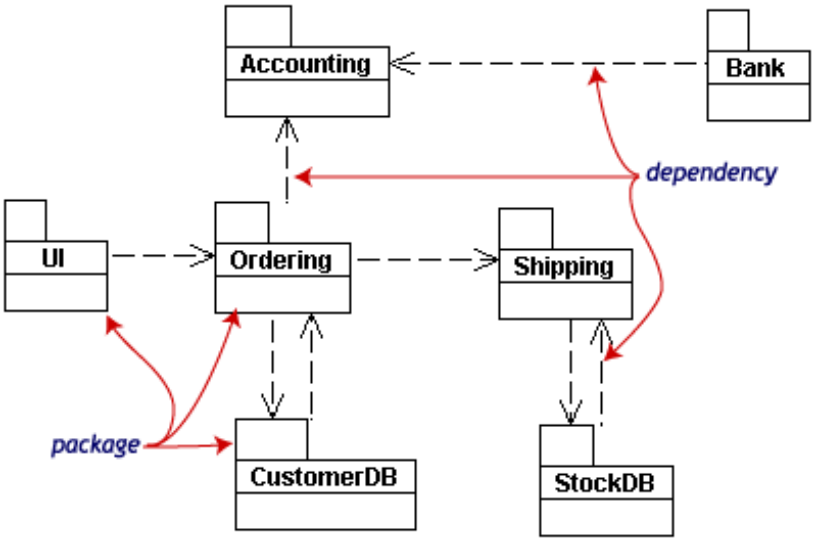
(protected): elementos protegidos só podem ser vistos pelos pacotes filhos

- (private): Elementos privados não podem ser vistos fora do pacote em que está declarado



9

Dependências entre pacotes



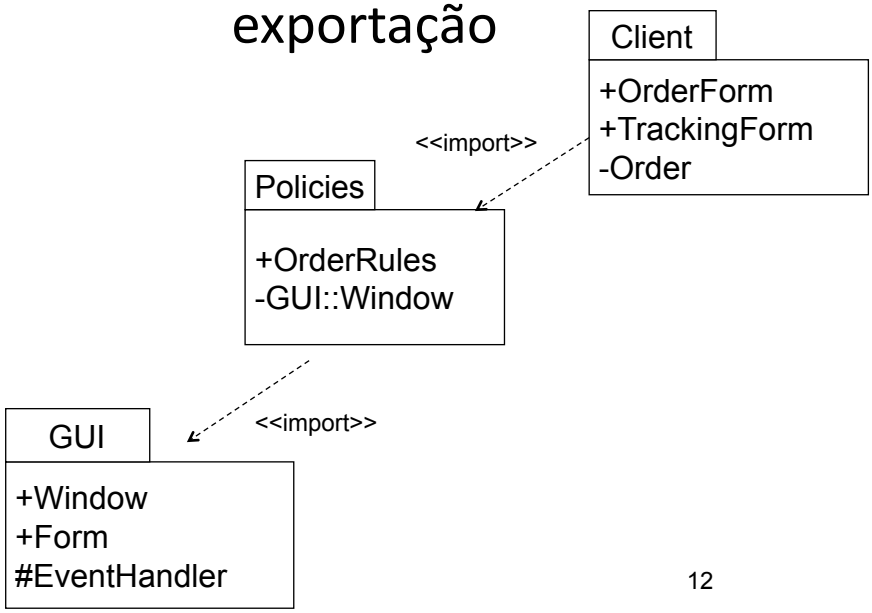
10

Exportação e Importação

- Os elementos públicos de um pacote são chamados seus *exports*
- Se X importa Y, um elemento de X pode ver os elementos públicos de Y, mas Y não pode ver os elementos de X.
- Importação é representada por uma dependência com o estereótipo <<import>>
- Se um elemento é visível num pacote X, é visível dentro de todos os pacotes aninhados de X

11

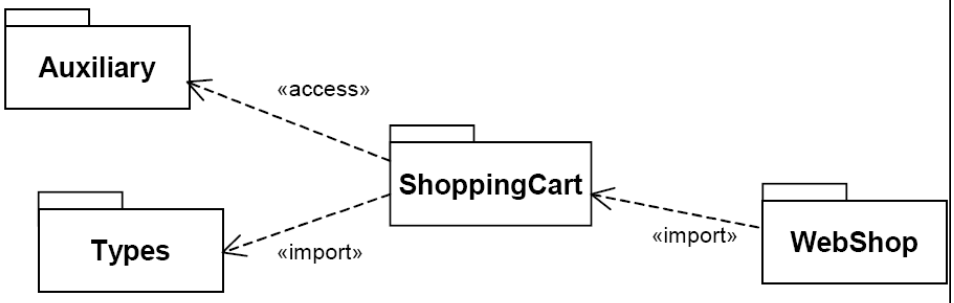
Visibilidade e importação/ exportação



12

<<access>>

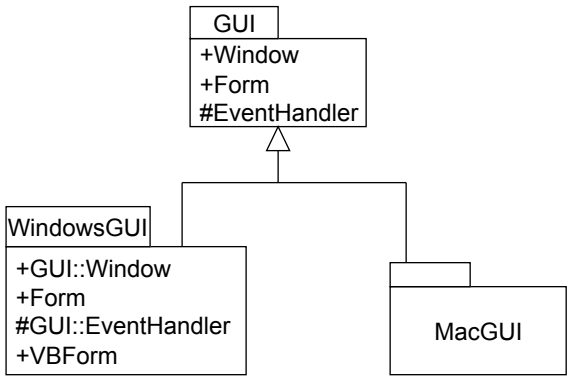
- Os elementos de *Auxiliary* só são acedidos a partir de *ShoppingCart* (é uma importação que torna os elementos importados privados)



13

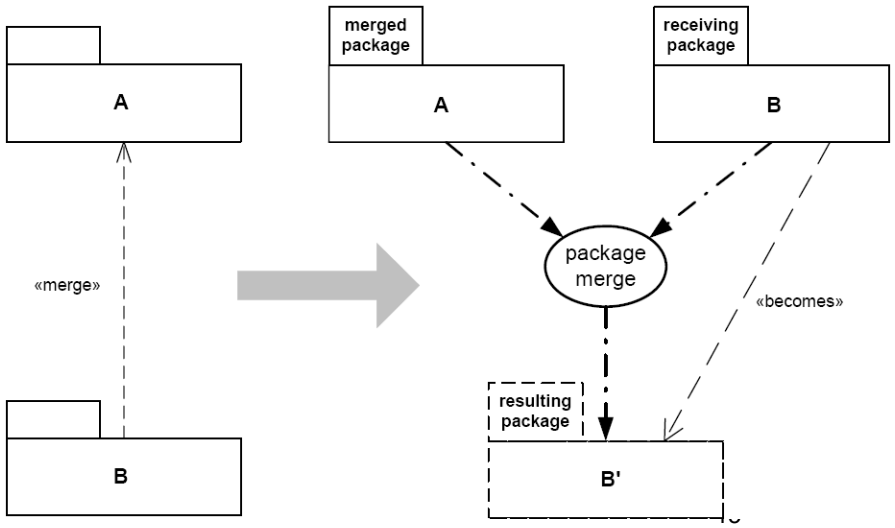
Generalização

- Especifica famílias de pacotes

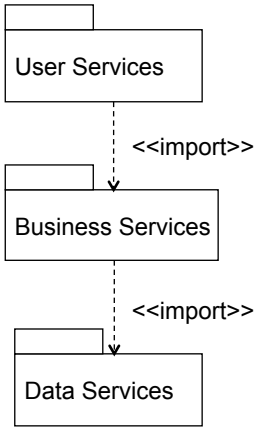


14

Merge de pacotes



Three-tier architecture



16

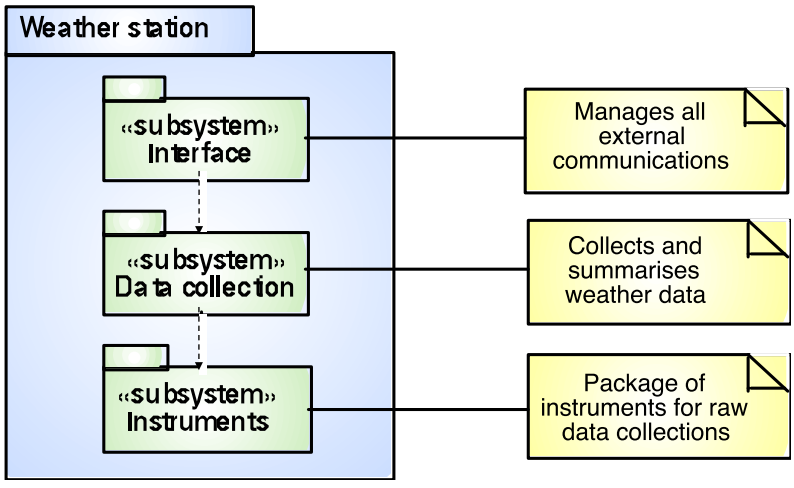
Descrição de um sistema meteorológico

Um sistema de recolha de dados meteorológicos deve gerar mapas meteorológicos regularmente utilizando dados recolhidos a partir de estações de tempo remotas, observadores de tempo, balões e satélites. As estações de tempo transmitem seus dados para o computador da área em resposta a uma requisição desta máquina.

O computador da área valida os dados recolhidos e integra-os com com os dados das outras fontes (balões, etc.). Os dados integrados são guardados e utilizados juntamente com uma base de dados de mapas digitalizados para criar um conjunto de mapas locais de tempo. Os mapas podem ser impressos numa impressora especial em diferentes formatos.

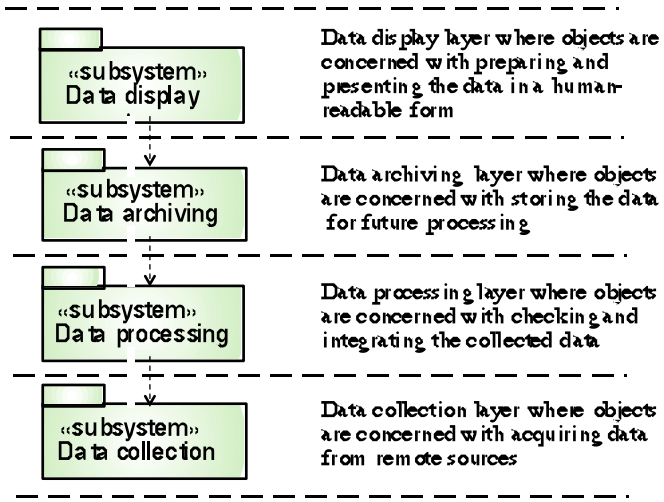
17

Arquitectura da estação meteorológica



18

Arquitectura em camadas



19

Modelar grupos de elementos

- Critério para o particionamento:
  - Coesão funcional, coesão de informação, controlo de acesso, distribuição, ...
- Guidelines:
  - Procurar elementos que são semanticamente próximos
  - Envolver estes elementos num pacote
  - Determinar a visibilidade dos elementos de cada pacote
  - Conectar os pacotes via dependências
  - No caso de famílias de pacotes, conectar via generalização

20