Diagrammi Dei Package

Ingegneria Del Software

Università degli Studi di Padova
Dipartimento di Matematica
Corso di Laurea in Informatica

rcardin@math.unipd.it

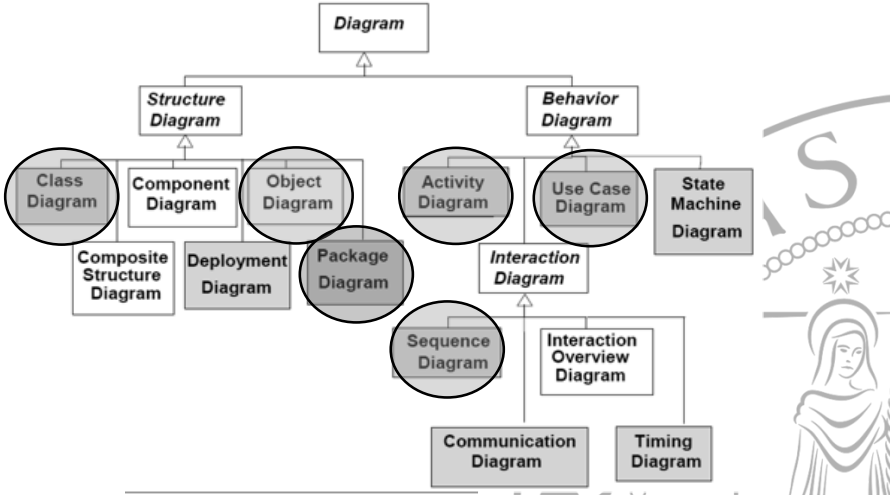
Sommario

- Cos'è un package
- Diagrammi dei package

Sommario

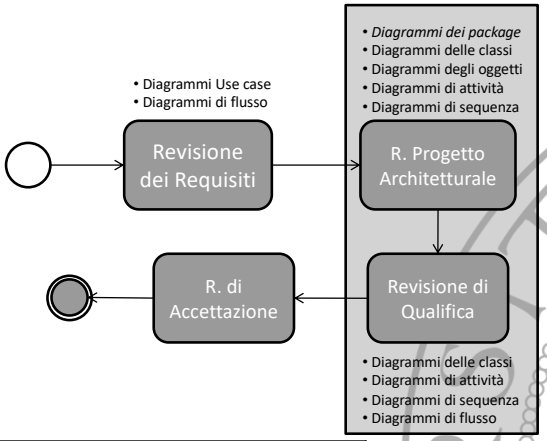
- Cos'è un package
- Diagrammi dei package

Diagrammi dei Package



Diagrammi dei Package

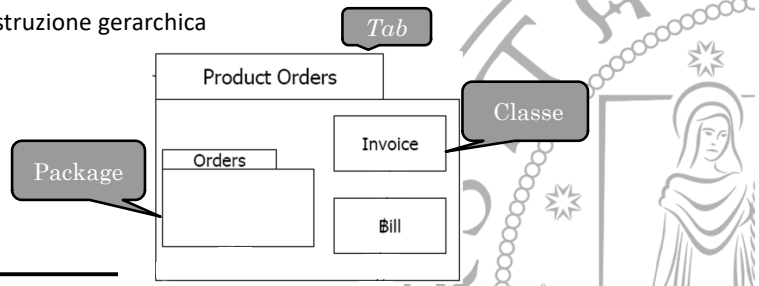
- Specifica Tecnica



Cos'è un Package

Raggruppamento di un numero arbitrario di elementi UML in una unità di livello più alto

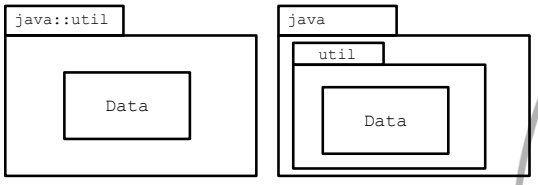
- Praticamente si raggruppano solo classi
- Ogni classe appartiene ad un solo *package*
- Un *package* può contenere un altro *package*
  - Costruzione gerarchica

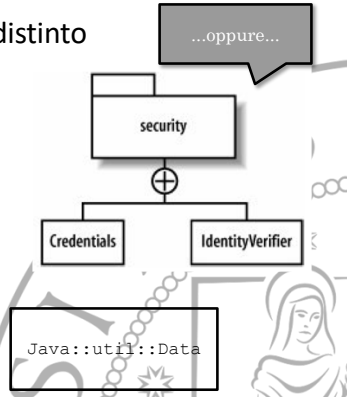


Cos'è un Package

- Il *package* individua un *namespace*
  - Ogni elemento deve avere un nome distinto all'interno dello "spazio di nomi"
  - Nome completamente qualificato

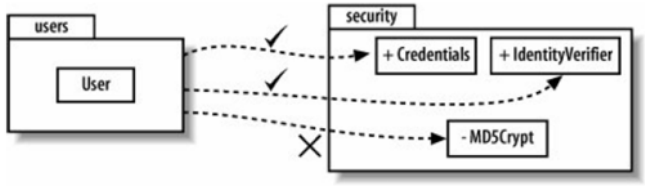
```
package::package::…::classe
```





Cos'è un Package

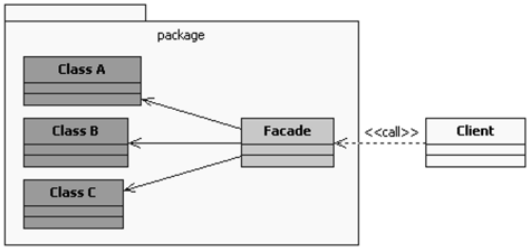
- Visibilità
  - Gli elementi del *package* possono avere visibilità pubblica (+) o privata (-)



  - Visibilità pubblica
    - `public class Credential {}`
  - Visibilità privata
    - `class MD5Crypt {}`

Cos'è un Package

- Interfaccia del *package*
  - Insieme dei tipi pubblici in un *package*
    - Possono essere definite anche classi private
  - Design pattern Façade



Cos'è un Package

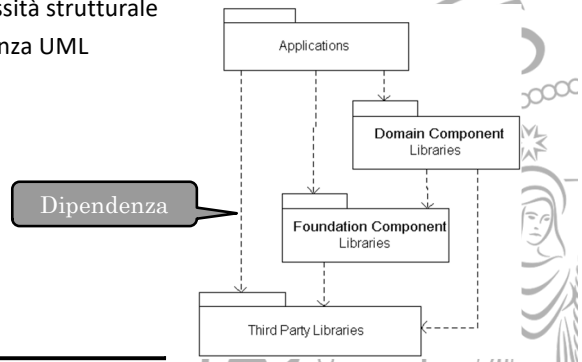
- Principi di progettazione
  - *Common Closure Principle*
    - Classi dello stesso *package* condividono la stessa causa di cambiamento.
  - *Common Reuse Principle*
    - Classi dello stesso *package* dovrebbero essere sempre riusate insieme.

Sommario

- Cos'è un package
- Diagrammi dei package

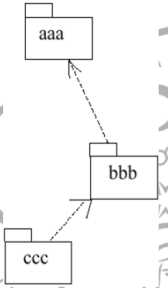
Package e Dipendenze

- Diagramma dei *package*
  - Documenta le dipendenze fra le classi
    - In sistemi medio-grandi è utilissimo per tenere sotto controllo la complessità strutturale
    - … qualsiasi dipendenza UML



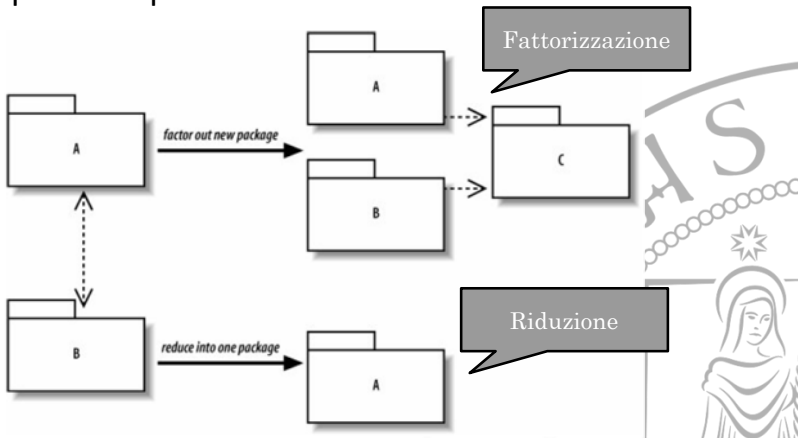
Package e Dipendenze

- Caratteristiche
  - Tutte le dipendenze dovrebbero seguire la stessa direzione
    - A meno di isolamento voluto di sottostrutture
  - Evitare le dipendenze circolari
    - *Acyclic Dependency Principle*
  - Relazioni di dipendenza non sono transitive
    - Se modifico aaa, non necessariamente devo modificare ccc
  - Più dipendenze entranti, più il *package* dovrebbe essere stabile
    - *Package* «global»



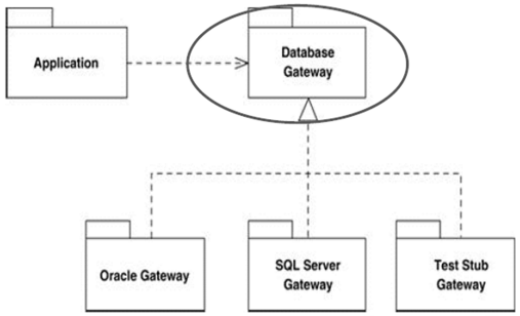
Package e Dipendenze

- Rompere le dipendenze circolari



Package e Dipendenze

- *Package* di interfaccia
  - Contiene interfacce e classi astratte
  - Design Pattern “Interfaccia separata”



Riferimenti

- OMG Homepage – www.omg.org
- UML Homepage – www.uml.org
- UML Distilled, Martin Fowler, 2004, Pearson (Addison Wesley)
- Learning UML 2.0, Kim Hamilton, Russell Miles, O’Reilly, 2006

GITHUB REPOSITORY

https://github.com/rcardin/swe