



DIAGRAMMI DEI PACKAGE

INGEGNERIA DEL SOFTWARE

Università degli Studi di Padova

Dipartimento di Matematica

Corso di Laurea in Informatica, A.A. 2017 – 2019

rcardin@math.unipd.it

SOMMARIO

- Cos'è un package
- Diagrammi dei package

Ingegneria del software

Riccardo Cardin

2

SOMMARIO

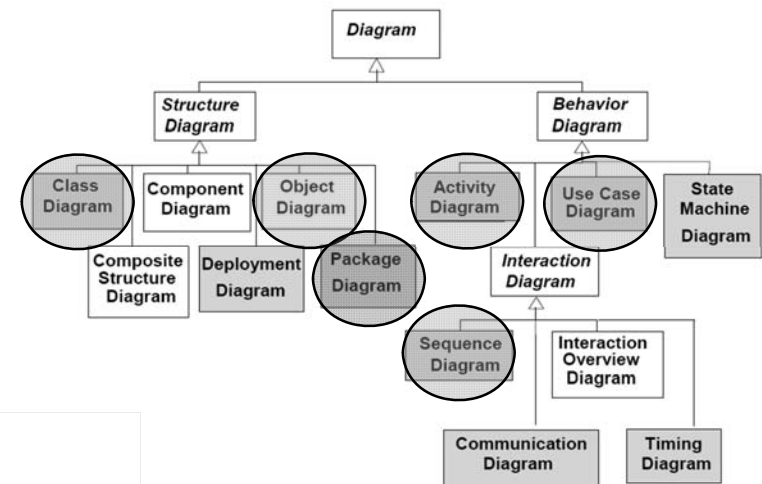
- Cos'è un package
- Diagrammi dei package

Ingegneria del software

Riccardo Cardin

3

DIAGRAMMI DEI PACKAGE



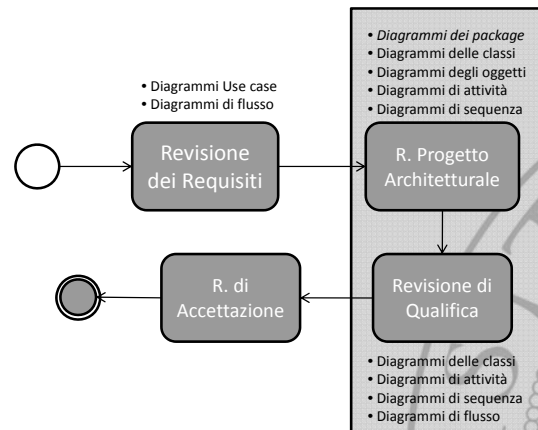
Ingegneria del software

Riccardo Cardin

4

DIAGRAMMI DEI PACKAGE

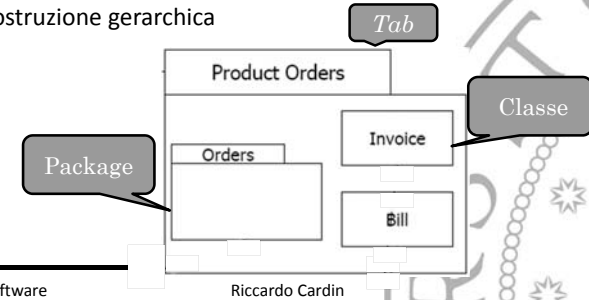
o Specifica Tecnica



COS'È UN PACKAGE

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

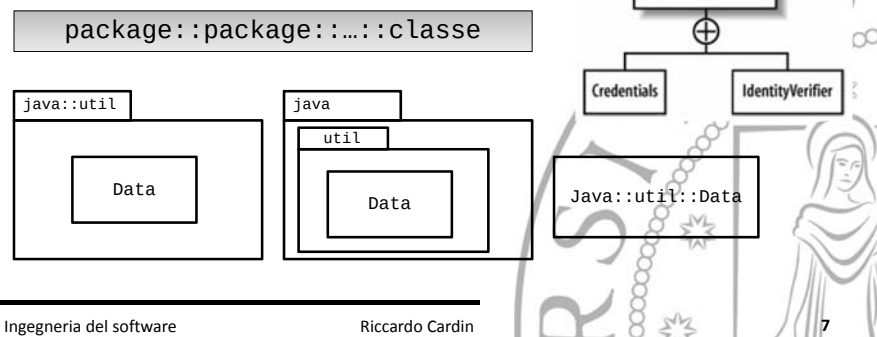
- Praticamente si raggruppano solo classi
- Ogni classe appartiene ad un solo *package*
- Un *package* può contenere un altro *package*
 - o Costruzione gerarchica



COS'È UN PACKAGE

o Il *package* individua un *namespace*

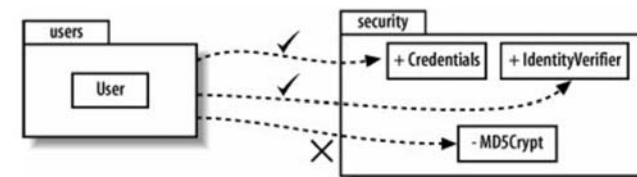
- Ogni elemento deve avere un nome distinto all'interno dello "spazio di nomi"
- Nome completamente qualificato



COS'È UN PACKAGE

o Visibilità

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

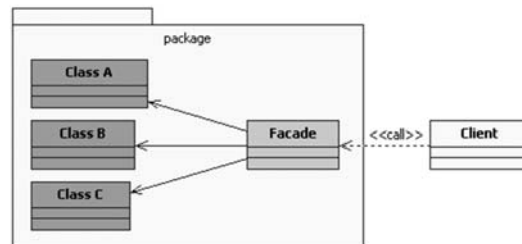


- Visibilità pubblica
 - o public class Credential {}
- Visibilità privata
 - o class MD5Crypt {}

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.

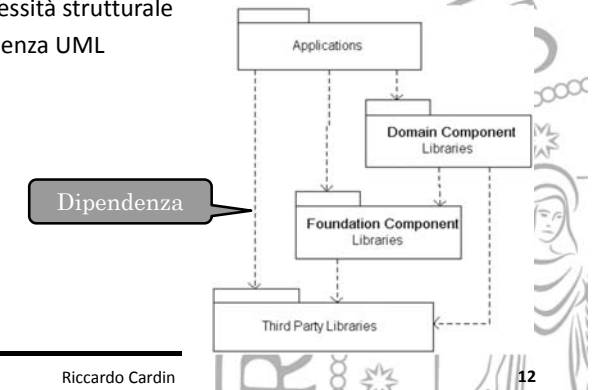
SOMMARIO

- Cos'è un package
- Diagrammi dei package

PACKAGE E DIPENDENZE

◦ Diagramma dei *package*

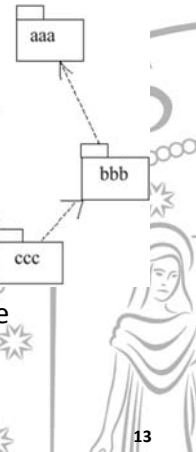
- Documenta le dipendenze fra le classi
 - In sistemi medio-grandi è utilissimo per tenere sotto controllo la complessità strutturale
 - ... qualsiasi dipendenza UML



PACKAGE E DIPENDENZE

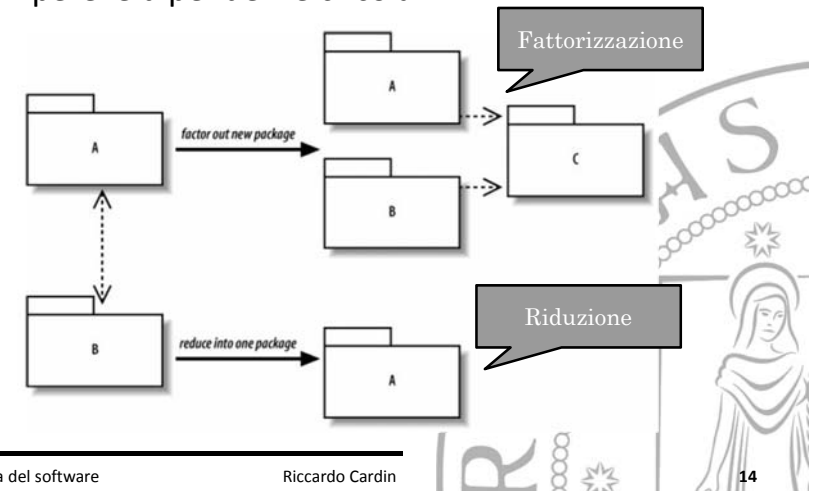
o Caratteristiche

- 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

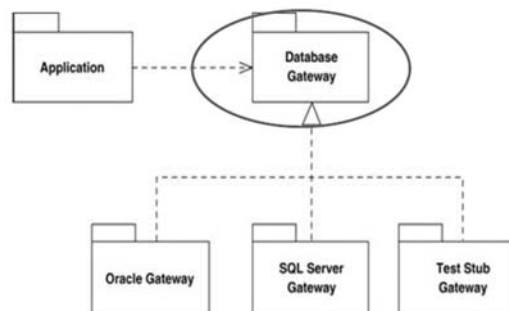
o Rompere le dipendenze circolari



PACKAGE E DIPENDENZE

o *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>

