Diagrammi dei package

Package

- Meccanismo di raggruppamento di più classi (riferito al momento della compilazione) in unità di livello più alto, al fine di dominare la complessità strutturale di un sistema sw
- Supportato dai linguaggi di programmazione (come *namespace* in C++ e .NET)

Dipendenza

- Fra due elementi esiste una (relazione di) dipendenza se i cambiamenti apportati alla definizione dell'uno si possono potenzialmente ripercuotere sull'altro
- Idealmente solo i cambiamenti dell'interfaccia di una classe dovrebbero ripercuotersi sulle altre
- Il principio del progetto per il cambiamento richiede di minimizzare le dipendenze, cosicché gli effetti dei cambiamenti siano ridotti e il sistema si possa modificare con minore sforzo
- UML prevede molti tipi di dipendenza, ognuno con la propria semantica e parola chiave

Suddivisione delle classi fra package diversi

- La dipendenza è il metodo euristico maggiormente usato per raggruppare le classi in package
- Un altro criterio è il riuso comune, secondo cui le classi di un package dovrebbero essere usate insieme
- Un terzo criterio è la chiusura comune, secondo cui le classi di un package dovrebbero condividere le cause di un eventuale cambiamento

Dipendenza fra classi e fra package

- Ogni package può raggruppare classi pubbliche e/o private e/o protette (*protected)*
- Fra due package esiste dipendenza se c'è dipendenza fra (almeno) una classe del primo e un'altra (necessariamente pubblica) del secondo
- Le dipendenze fra package non sono transitive
- Un flusso di dipendenze unidirezionale è generalmente indice di un sistema ben strutturato

Interfaccia di un package

- È l'insieme dei metodi pubblici delle classi pubbliche del package
- Per ridurre tale insieme si può dare a tutte le classi del package visibilità privata, aggiungendo poi altre classi pubbliche, denominate facciate (mediante la parola chiave «`facade`»), per implementare il comportamento esterno
- Le facciate delegano l'esecuzione delle proprie operazioni alle classi nascoste
- Man mano che il n° di dipendenze entranti in un package aumenta, la sua interfaccia deve diventare sempre più stabile

Diagramma dei package

- Mostra più package di classi con le loro dipendenze
- Aiuta a individuare le dipendenze (al fine di ridurle)
- È uno strumento fondamentale per mantenere il controllo sulla struttura globale del sistema sw
- Nella prospettiva software, può essere generato a partire dal codice stesso

Diagramma dei package (cont.)

Elementi	Sintassi	Semantica
Package	Scatola con linguetta (*tab*) in alto a sx. La linguetta può: ▪ Essere vuota, nel qual caso il nome del package (in grassetto) è contenuto nella scatola ▪ Contenere il nome del package (in grassetto), nel qual caso la scatola contiene un elenco di classi, oppure un diagramma dei package e/o delle classi ▪ Contenere la parola chiave «`global`», che significa che tutti (o quasi) i package del sistema sono dipendenti dal package considerato Ogni classe può essere dotata dell'indicatore di visibilità Il nome di ciascun package/classe può essere semplice o completamente qualificato; quest'ultimo mostra la struttura dei package, dal più esterno al più interno, che contengono l'elemento corrente, fino a tale elemento incluso, dove due nomi consecutivi sono separati da :: Es. `System::Data`	L'aggregazione di più package entro un package consente la modellazione gerarchica di sistemi complessi Ogni classe fa parte di un solo package Ogni classe deve aver un nome distinto all'interno del package che la racchiude

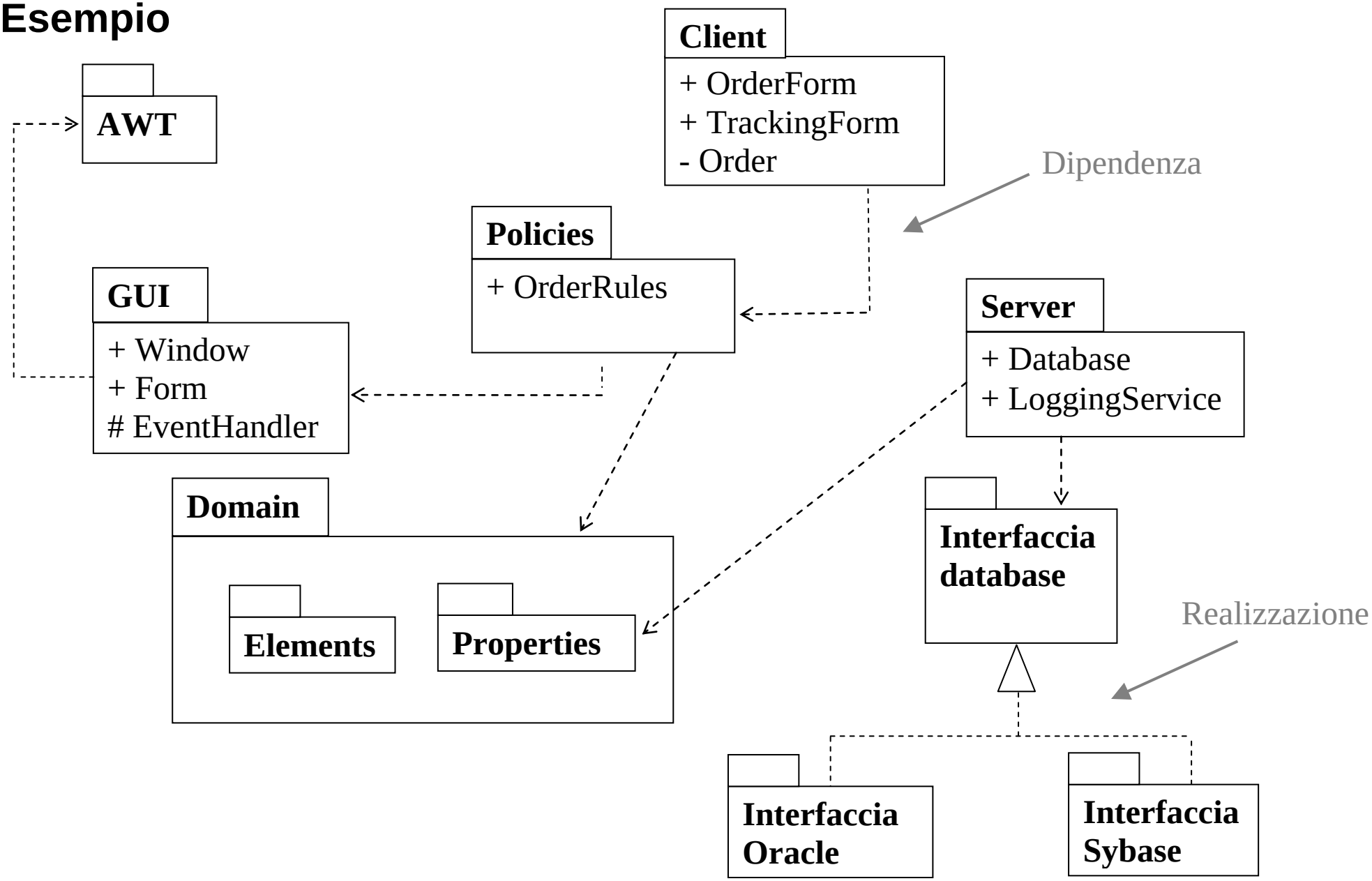
Diagramma dei package (cont.)

Elementi	Sintassi	Semantica
Dipendenza	Freccia con linea tratteggiata e punta biforcuta, uscente dal dipendente e terminante sul riquadro di un altro package o su quello di un elemento in esso contenuto	Una dipendenza verso un package che a sua volta ne contiene altri è la somma di più dipendenze di livello inferiore, cioè denota che ci sono dipendenze verso qualche elemento interno
Realizzazione (o implementazione)	Freccia, con linea tratteggiata e punta triangolare vuota, uscente dal package più specifico e terminante sul riquadro di un altro package o su quello di un elemento in esso contenuto	Il package più generale contiene (non esclusivamente) interface e classi astratte che sono implementate dal package più specifico

Suggerimenti

- Interpretare gli indicatori di visibilità delle classi in base alle regole del linguaggio di programmazione adottato dal sistema
- Minimizzare i cicli nella struttura delle dipendenze
- Se ci sono dei cicli, cercare di contenerli in un package più grande
- Sforzarsi di eliminare cicli nelle dipendenze tra i package del dominio e le interfacce esterne
- Generare automaticamente il diagramma dei package di un sistema già implementato come punto di partenza per migliorare la struttura del sistema attraverso una riduzione delle dipendenze (refactoring)
- Considerare il package come unità di base per il testing (può essere utile aggiungere delle classi che servono solo in supporto al testing delle altre classi nel package)

Esempio


Client
+ OrderForm
+ TrackingForm
- Order
AWT
Dipendenza
Policies
+ OrderRules
GUI
+ Window
+ Form
# EventHandler
Server
+ Database
+ LoggingService
Domain
Elements
Properties
Interfaccia database
Realizzazione
Interfaccia Oracle
Interfaccia Sybase