Padova 7 Maggio 2004

Introduzione alla programmazione AOP con AspectJ

by Matteo Baccan
http://www.baccan.it

Introduzione alla programmazione AOP con AspectJ

- Relatore: Matteo Baccan matteo@baccan.it

- Data: Padova 07/05/2004 13:30
- Area: Programming & development
- Target: Tecnici
- Difficoltà: Facile

Nel corso degli anni, nuove correnti di pensiero hanno fatto nascere e sviluppare la programmazione strutturata, la programmazione ad oggetti ed ora la programmazione ad aspetti (Aspect Oriented Programming). Vediamo di cosa si tratta, andando ad utilizzarla tramite AspectJ, un compilatore portabile in grado di gestire questi nuovi costrutti

Obiettivi

Dopo aver completato questo seminario sarai in grado di:

- Comprendere le basi di AOP
- Conoscere il concetto AOP di AspectJ
- Scrivere i primi programmi AOP
- Sfruttare la programmazione AOP nei tuoi programmi

AOP - prima

Sviluppo tradizionale

- Lo sviluppo di applicazioni è faticoso
- La progettazione di un buon framework richiede tempo e progettazione
- Riuscire a soddisfare delle richieste orizzontali all'intero progetto rischia di stravolgere l'architettura progettata
- Dietro l'angolo c'è sempre la possibilità di avere del codice ridondante e copiato fra un componente e l'altro

AOP – perché

AOP permette di astrarre funzionalità orizzontali alle applicazioni, definendole all'interno di aspect

AOP – evoluzione

- Programmazione destrutturata/batch
  una sola procedura

- Programmazione procedurale
  a funzioni

- Programmazione ad oggetti
  tramite classi

- Programmazione ad aspetti
  Aspetti (crosscutting concerns) – problematiche comuni a più classi

AOP – crosscutting concerns

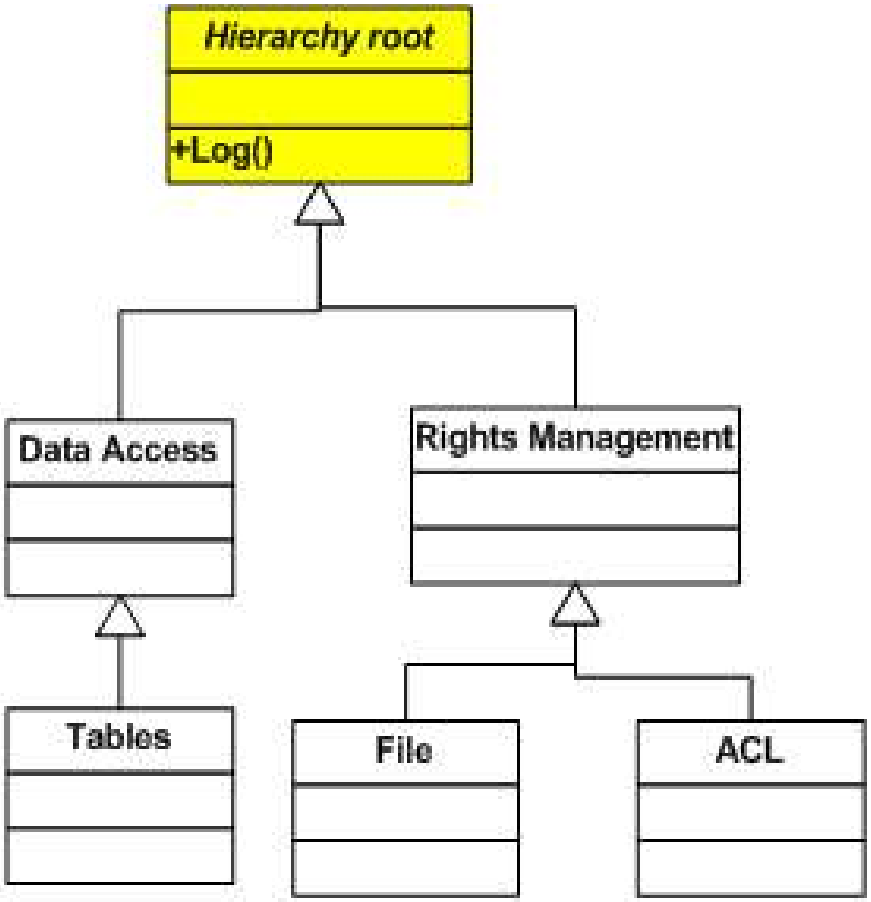
Come direbbe google: “preoccupazioni trasversali di taglio”...

...o per meglio dire: problematiche trasversali a tutto un framework o ad un'applicazione

Esempi

- Logging
- Exception
- Accesso controllato
- Monitoring
- Profiling

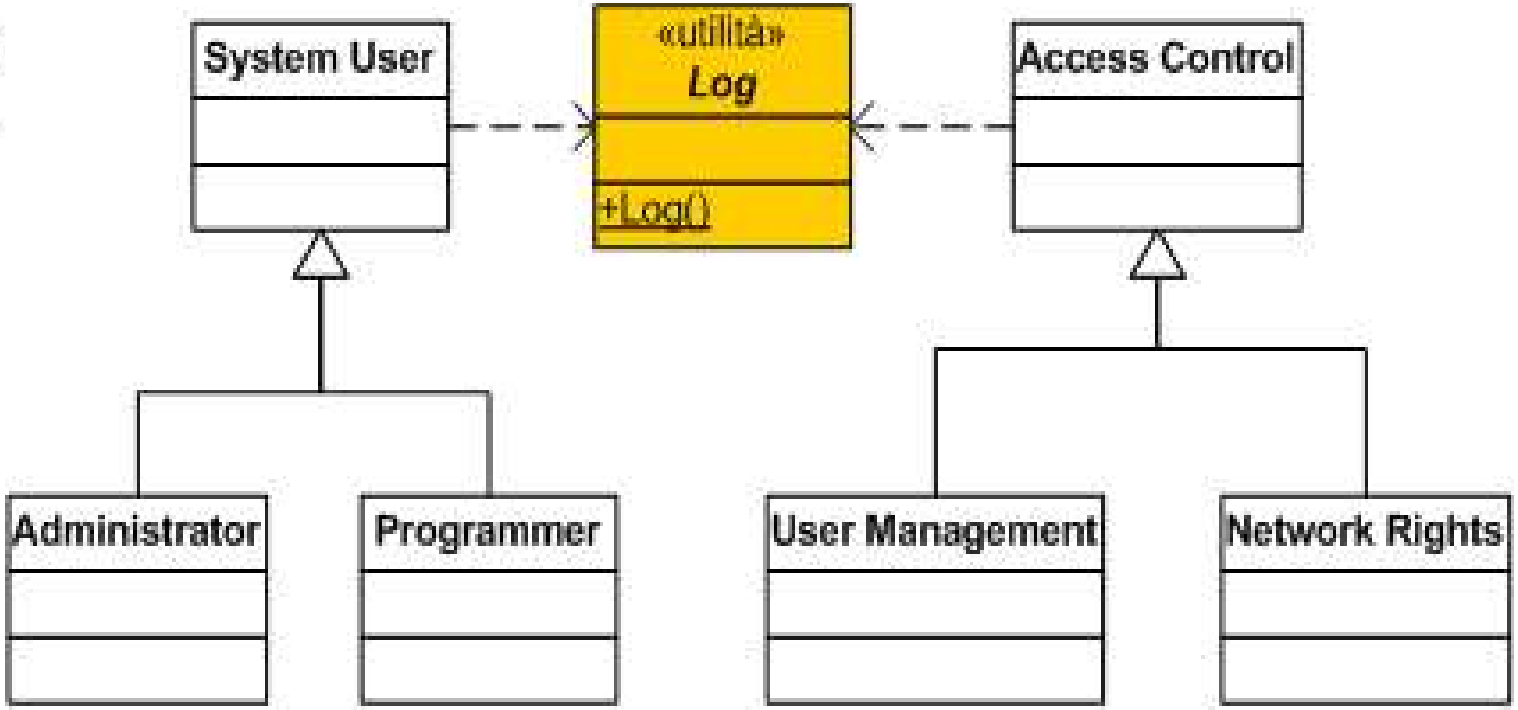
AOP - prima


Hierarchy root
+Log()
Data Access
Rights Management
Tables
File
ACL

AOP – la soluzione

- Permette di definire un concetto trasversale
- Evita la modifica delle classi in caso di crosscutting
- Separa i framework dalle modifiche “cross”
- Programmazione simile a un “reverse engineering” autorizzato

AOP – dopo


System User
«utilità»
Log
+Log()
Access Control
Administrator
Programmer
User Management
Network Rights

AspectJ

Per programmare in AOP serve un compilatore

- AspectJ è uno dei compilatori in grado di utilizzare AOP
- Non è l'unico. Alternative:
  Hyper/J, JbossAOP, JAC, AspectWerkz, Spring etc.

AspectJ – crosscutting

Dinamico

- Viene applicato a tutto il framework/programma o ad un sottoinsieme di codice e ne modifica l'esecuzione. Il codice cambia la modalità di funzionamento

Statico

- Viene applicato a tutto il framework/programma o ad un sottoinsieme di codice e ne modifica la struttura. Le classi hanno nuovi metodi o variabili d'istanza

AspectJ – elementi

Join point:

un punto di esecuzione del programma: il richiamo di un metodo o l'assegnamento di una variabile d'istanza

Pointcut:

è il costrutto che permette di selezionare i join point dove andrà ad agire un advice

Advice:

è il codice che deve essere eseguito nel join point selezionato dal pointcut

Aspect:

è l'insieme di advice e pointcut

AspectJ – join point

Method execution `execution(MethodSignature)`

Method call `call(MethodSignature)`

Constructor execution `execution(ConstructorSignature)`

Constructor call `call(ConstructorSignature)`

Class initialization `staticinitialization(TypeSignature)`

Field read access `get(FieldSignature)`

Field write access `set(FieldSignature)`

Exception handler execution `handler(TypeSignature)`

Object initialization `initialization(ConstructorSignature)`

Object pre-initialization `preinitialization(ConstructorSignature)`

Advice execution `adviceexecution()`

Join point – execution call

- call(...)
  Viene modificato il programma chiamante

- execution(...)
  Viene modificato il metodo

Esempio...

Join point – staticinitialization

- staticinitialization(...)
  permette di intercettare le inizializzazioni statiche delle classi

```
static{
   // faccio qualcosa
}
```

Esempio...

Join point – get/set

- set(...)
  Intercetta gli assegnamenti ad una variabile

- get(...)
  Intercetta le letture di una variabile

Esempio...

Pointcut - this/target

- this(...)
  Condizione sull'oggetto corrente

- target(...)
  Condizione sull'oggetto chiamato

Esempio...

Pointcut - args

- args(...)
  condiziona un pointcut a certi parametri

Esempio...

Advice- before/after/around

- before
  Richiamo del codice prima di un pointcut
- after
  Richiamo del codice dopo un pointcut
- around
  Richiamo del codice prima e dopo un pointcut

Esempio...

Advice/metodi

Come un metodo

- Segue delle regole di accesso per essere eseguito
- Può generare un'eccezione
- Può fare riferimento all'aspect corrente tramite *this*

Al contrario dei metodi

- Non ha un nome
- Non può essere chiamato direttamente, è il sistema che lo esegue
- Ha accesso ad alcune variabili valorizzate all'interno di *this* che possono servire a prendere informazioni sui join point catturati: `thisJoinPoint`

Advice - after

- returning
  Permette di porre del codice dopo il ritorno di un valore

- throwing
  Permette di porre del codice al verificarsi di un'eccezione

Esempio...

Crosscutting statico

- Permette di aggiungere variabili e metodi alla classe
- Permette di modificarne l'ereditarietà

Esempio...

Crosscutting compilazione

È possibile condizionare la compilazione con:

- warning
  avvisa l'utente di una certa condizione

- error
  blocca la compilazione

Esempio...

Riferimenti

- AspectJ
  http://www.aspectj.org
- Il progetto Eclipse
  http://www.eclipse.org
- Pagina di download di AspectJ
  http://dev.eclipse.org/viewcvs/indextech.cgi/~checkout~/aspectj-home/downloads.html
- Dove poter scaricare questo materiale
  http://www.baccan.it
  http://www.javaportal.it

and now ...

Q&A

Ringraziamenti

GRAZIE

da

Matteo Baccan

Ringraziamenti…

- Le aziende grazie alle quali posso essere qui oggi

http://www.grupposisge.it/