Università di Bologna
CdS Laurea Magistrale in Ingegneria Informatica
I Ciclo - A.A. 2013/2014

Sistemi Distribuiti M

Proposta di esercizio: Mapping Object-Relational attraverso Hibernate e JPA

A cura di:

Paolo Bellavista paolo.bellavista@unibo.it
Stefano Monti stefano.monti@epocaricerca.it

Obiettivo 1 – Mapping OR

Data la modellazione del dominio dei dati per l'applicazione Web di esempio, relativa alla gestione di *"Libri"*, *"Autori"* e *"Editori"* in una ipotetica biblioteca…

- Utilizzare le funzionalità di ***Hibernate*** per ***mapping object-relational automatico*** tra oggetti Java e tabelle di database
  - per mezzo di ***descrittori XML*** specifici di Hibernate
    oppure, a scelta
  - per mezzo di ***annotazioni Java*** conformi allo standard JPA

- Utilizzare le funzionalità di Hibernate per ***implementare gli oggetti DAO***
  - per mezzo di API specifiche di Hibernate (= basate sul concetto di `Session`)
    oppure, a scelta
  - per mezzo di API compatibili con lo standard JPA (= basate sul concetto di `EntityManager`)

Obiettivo 2: Semantica Transazionale

Supponendo che ad ogni richiesta HTTP ricevuta dal Web server debba corrispondere ***un insieme atomico di operazioni su database***, che inizia con l'ottenimento di una nuova istanza di factory DAO e termina con la restituzione della risposta al client…

Realizzare una ***implementazione DAO alternativa alla precedente***, in cui

- i singoli metodi degli oggetti DAO sono sollevati dalla responsabilità di ***dichiarare l'inizio e la fine delle transazioni al proprio interno*** (semantica = "*una transazione per ogni richiesta di operazione su database ai DAO*")
- tale ***responsabilità è assegnata all'istanza della factory* che li ha generati*** (semantica = "*una transazione per ogni richiesta formulata da un client*")

* Si introduce a tal fine, per praticità, una versione estesa delle specifiche della DAOFactory, che prevede anche l'operazione di *release()* della factory stessa, oltre a quella di *get()*

Obiettivo 3 – Simulazione dell'esistenza di un "container"

Supponendo che ad ogni richiesta HTTP ricevuta dal Web server debba corrispondere ***un insieme atomico di operazioni su database***, che inizia con l'ottenimento di una nuova istanza di factory DAO e termina con la restituzione della risposta al client…

Attraverso ***intercettazione delle richieste HTTP***, rendere trasparente agli oggetti della logica di business come pagine JSP o altri componenti, i meccanismi di

- inizio della transazione (= ottenimento di una nuova istanza della factory)
- conclusione della transazione (= restituzione della factory)

Dettagli…

Il progetto contenente il codice su cui basare lo sviluppo…

- Contiene una ***applicazione Web minimale***, una ***suite di test*** e un insieme di classi per l'inizializzazione della base di conoscenza
- Permette, attraverso ANT, di eseguire il deployment della applicazione su un'installazione del Web Server Tomcat
- Contiene una versione estesa delle specifiche della factory DAO
  - Metodi per ***l'ottenimento e la restituzione delle factory*** concrete
  - Obbligo per le factory concrete di ***implementare un metodo per la terminazione delle transazioni***
- Contiene versioni modificate dei test, per ***gestire le semantiche transazionali***

Riferimenti e Approfondimenti

Hibernate Reference
http://www.hibernate.org/hib_docs/reference/en/html/

Hibernate Mapping Cheat Sheet
http://ndpsoftware.com/HibernateMappingCheatSheet.html

JPA Implementation patterns: Saving (detached) entities
http://blog.xebia.com/2009/03/23/jpa-implementation-patterns-saving-detached-entities/