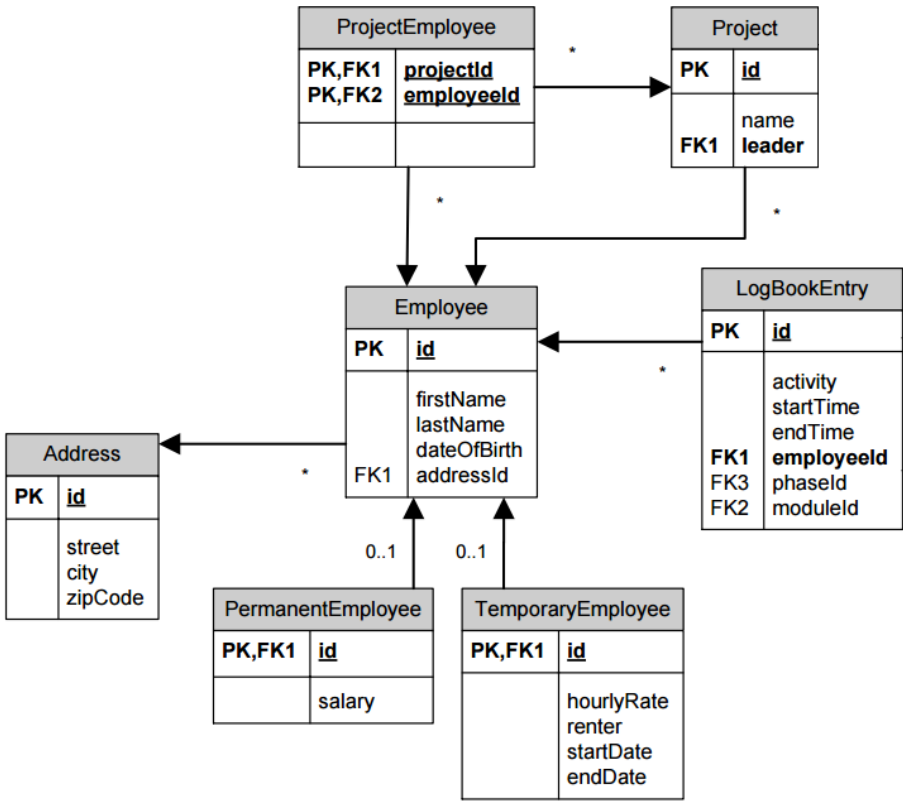
Dinu Marius-Constantin
SEv16.6.SWT UE, Sebastian Pimminger

Aufwand: 25h

Übungsaufgabe 2 (O/R-Mapping)

→ Deadline: 14. 3. 2016, 24:00 Uhr

Domänenmodell und Mapping



- Vervollständigen Sie das Domänenmodell des Zeiterfassungssystems, sodass darin Angestellte, die Adressen der Angestellten und Projekte mit deren Zuordnung zu Angestellten berücksichtigt werden. Für fix Angestellte ist zusätzlich zu erfassen, in welchem Ausmaß sie beschäftigt sind.
- Erweitern Sie das Domänenmodelle um Klassen, welche die Abwicklung von Projekten nach dem Vorgehensmodell Scrum unterstützen. Gehen Sie von folgenden Anforderungen aus:
    - Ein Projekt besteht aus mehreren Sprints.
    - Zu jedem Projekt existiert eine Liste von Anforderungen *(Requirements)*, die einem Sprint zugeordnet sein können. Für jede Anforderung sind eine menschenlesbare Kurzbezeichnung und eine Beschreibung zu speichern.
    - Jeder Anforderung wird eine Liste von Aufgaben *(Tasks)* zugeordnet.
    - Für jede Aufgabe sind eine menschenlesbare Kurzbezeichnung, eine Beschreibung und der geschätzte Aufwand zu speichern.
    - Jeder Arbeitszeiteintrag wird einer Aufgabe zugeordnet.

- Definieren Sie das Mapping des Domänenmodells auf die Datenbank mithilfe von JPA-Annotationen.
- Als O/R-Mapping-Framework können Sie Hibernate oder einen beliebigen JPA-Provider verwenden

Datenzugriffsschicht

- Kapseln der Datenbankzugriffslogik durch konsequente Anwendung des DAO-Musters.
    - Die DAOs sollten für alle Domänenklassen eine gewisse Grundfunktionalität anbieten: Einfügen, Aktualisieren und Löschen von Domänenobjekten.
    - Gewährleisten Sie auch, dass die Beziehungen zwischen den Domänenobjekten verwaltet werden können: Hinzufügen und Trennen von Beziehungen. Überlegen Sie sich sinnvolle Einstellungen für die Kaskadierungs- und die Ladestrategien.
    - Sehen Sie auf Datenzugriffsebene Funktionalität vor, mit der statistische Auswertungen durchgeführt werden können. Sie sollten damit Fragen wie die folgenden beantworten können: Wie verteilt sich die Arbeitszeit in einem Projekt auf verschiedene Mitarbeiter, Sprints und Anforderungen? Wie verteilt sich die Arbeitszeit eines Mitarbeiters auf verschiedene Projekte und Sprints.
    - Die Datenzugriffschicht muss alle Daten liefern, um für einen bestimmten Sprint den Burn-Down-Chart ermitteln zu können.
- Implementierung von Integrationstest mit JUnit, mit denen die DAOs ausführlich getestet werden.
    - Entwickeln Sie für jede DAO-Methode zumindest einen Unit-Test.
    - Achten Sie darauf, dass Ihre Tests unabhängig voneinander sind, d. h. die Ausführung eines Tests nicht das Ergebnis eines anderen beeinflusst bzw. dass die Tests in einer bestimmen Reihenfolge durchgeführt werden müssen.
    - Verwenden Sie eine In-Memory-Datenbank (H2, SQLite, Derby etc.) für Ihre Tests, damit diese möglichst effizient ausgeführt werden können.
    - Strukturieren Sie Ihre Tests so, dass sie mit Maven ausgeführt werden können (auch von der Kommandozeile).
    - *Optional, für Beurteilung mit „Sehr gut“ erforderlich:*Verwenden Sie zur Initialisierung der Test-Datenbank *DbUnit* (http://dbunit.sourceforge.net), *DBSetup*(http://dbsetup.ninja-squad.com) oder ein vergleichbares Werkzeug. Sie können dieses Werkzeug auch zur Überprüfung der Korrektheit der Tests verwenden.

Konsolenanwendung

- Entwickeln Sie eine einfache Konsolen-Anwendung, in der Sie
    - in der Sie Ihre Entitäten und die Beziehungen dazwischen verwalten können,
    - die geforderten statistischen Auswertungen durchführen können,
    - den Burn-Down-Chart in tabellarischer Form ausgeben können und

- die bisher angefallenen Personalkosten für ein Projekt ermitteln können (rechnen Sie mit 220 Arbeitstagen pro Jahr).
- Die Konsolenanwendung soll auf eine Datenbank zugreifen, die mit ausreichend Daten befüllt ist.

Sie dürfen für diese Aufgabenstellung keine Funktionen des Spring Frameworks verwenden.

Lösungsidee:

Für die Instanziierung der DAOs wird die DaoFactory verwendet. Die DaoFactory beinhaltet eine generische Lösung für die Objekterstellung. Die DaoFactory besitzt auch Hilfsmethoden für die Transaktionsverwaltung (commit und rollback).

Um die Beziehungen zwischen den Entitäten abbilden zu können, wurden die JPA Annotationen herangezogen. Diese deklarieren auch das Cascading-Verhalten und Initialisierungsverhalten (lazy loading).

Diese Methoden werden von den Service Klassen der Business-Logic verwendet.

Die Service-Klassen dienen größtenteils als Delegationen der vorhandenen DAO-Klassen, können jedoch Prüfungen und weitere Methoden, wie statistische Berechnungsmethoden für den Client bereitstellen.

Die UnitTests leiten von einer gemeinsamen Basisklasse ab und initialisieren die DbSetup Objekte bei jedem Test neu. Somit wird Unabhängigkeit zwischen den Tests gewährleistet.