Universität Zürich[UZH]

Institut für Informatik

RERG

Software Engineering Übung 3

Softwarearchitektur

1 Informationen

1.1 Daten

- Ausgabe Di 15.10.2013
- Abgabe Mo 21.10.2013 bis 23:59 Uhr
- Besprechung am Di 29.10.2013 um 12:15 Uhr

1.2 Formales

Die Lösungen sollen als PDF Datei mit dem Namen Ex[**n**] [**NameA Matrikelnummer**].pdf abgegeben werden, wobei [**n**] die Nummer der Übung ist und [**NameA Matrikelnummer**] der Name und Matrikelnummer des Studenten sind. Die PDF Datei sollte außerdem ebenfalls Ihren Namen und Matrikelnummer beinhalten.
Bitte senden Sie Ihre Lösungen via OLAT. Der Betreff der Lösung sollte mit [SE EX HS13] beginnen. Falls Sie zusätzliche Abgabematerialien (z.B. Source Code) haben, senden Sie diese als Archiv (.zip-File), welches alle Dateien, einschliesslich dem PDF, enthält. Benennen Sie das Archiv anhand der oben erwähnten Konventionen.
Die Übungen sollen in Gruppen gelöst werden. Jedes Gruppenmitglied muss über alle Teile der Lösungen Auskunft geben können. Verspätete Abgaben werden korrigiert, aber nicht bewertet.

2 Überblick und Ziele

Im Teil 1 der Aufgabenstellung erarbeiten Sie die Architektur Ihrer App zur Visualisierung von Abstimmungen im Rahmen des Übungsprojekts. Sie lernen dabei den Entwurf einer Architektur im Rahmen eines zu entwickelnden Systems. In Teil 2 identifizieren Sie in der StockWatcher App des GWT Tutorials einige Architektur- und Entwurfsmuster. Sie lernen dabei das Erkennen von Mustern in bestehendem Code. Hierzu steht Ihnen ein Hilfsblatt mit einer Auflistung von Architektur- und Entwurfsmustern zur Verfügung. Alle Teilaufgaben sind als Gruppenarbeit in Ihrer Übungsgruppe zu bearbeiten.

3 Aufgabenstellung Teil 1: Architektur

Die folgenden Teilaufgaben basieren auf der Anforderungsspezifikation einer App zur Visualisierung von Abstimmungen, welche Sie in Übung 2 erstellt haben. Wahrscheinlich werden Sie beim Entwurf

der Architektur dieser App auf Lücken oder Unklarheiten in der Anforderungsspezifikation stossen, welche Sie in einem realen Projekt später mit den beteiligten Interesseneignern diskutieren und klären würden. Da dieses Vorgehen im Rahmen des Übungsprojekts nicht möglich ist, treffen Sie in solchen Fällen Annahmen und dokumentieren diese in einer Liste (vgl. Teilaufgabe 3.5).

Teilaufgabe 3.1: Grobarchitektur (4 Punkte)

a) Prozessarchitektur (1 Punkt)

Wie viele parallel laufende Prozesse brauchen Sie in der Implementierung Ihrer App? Begründen Sie Ihre Entscheidung. Wenn Sie mehr als einen Prozess brauchen, geben sie für jeden Prozess an, was seine Hauptaufgabe ist.

b) Identifikation von Komponenten (2 Punkte)

Identifizieren und benennen Sie die Komponenten, aus denen Ihre App bestehen soll. Charakterisieren Sie jede Komponente in 1-2 Sätzen.

c) Vorläufiges Komponentendiagramm (1 Punkt)

Zeichnen Sie ein vorläufiges UML-Komponentendiagramm.

Teilaufgabe 3.2: Klassen und Schnittstellen (5 Punkte)

In dieser Teilaufgabe geht es darum, dass Sie die wichtigsten Objekte, Attribute (in UML auch features genannt) und Operationen Ihrer Architektur identifizieren. Diese abstrahieren Sie zu Klassen in UML. Charakterisieren Sie jede Klasse mit einem Satz. Es bewährt sich, die Klassen zunächst tabellarisch zu beschreiben (siehe Beispiel unten).
Danach müssen Sie entscheiden, welche Klassen in der Implementierung zu Java-Klassen werden (Operationen als Methoden implementiert) und welche zu Java-Schnittstellen (Interfaces). Dokumentieren Sie Ihre Entscheidungen in der Klassenliste, indem Sie bei Schnittstellen dem Namen die Kennzeichnung «Interface» voranstellen.
Hier ist ein Beispiel, wie Sie vorgehen sollten: Angenommen, Sie erstellen die Architektur für ein System, für das es folgende Anforderungen gibt: Das System soll auftretende Ereignisse mit einem Zeitstempel versehen in einem Logbuch eintragen. Ein Benutzer soll Ereignisse innerhalb eines Zeitintervalls im Log suchen können und die gefundenen Ereignisse eines um das andere abrufen können. Ferner sollen autorisierte Benutzer ein gefundenes Ereignis im Log löschen können.
Sie entscheiden, eine Klasse EreignisLog zu entwerfen, welche die erforderlichen Operationen eintragErzeugen, eintragLöschen, eintragFinden und nächstenFinden realisiert. Sie entscheiden ferner, dass diese Klasse keine öffentlichen Attribute hat.
Der entsprechende Tabelleneintrag würde in UML Notation dann wie folgt aussehen:

Tabelle 1. Identifizierte Klassen, Attribute und Operationen

Name	Attribute	Operationen
«interface» EreignisLog Stellt ein Logbuch für Ereignisse bereit		+eintragErzeugen (neuerText: String, neueSorte: Ereignisart) +eintragLöschen (autorisiert: Rechte) +<u>eintragFinden</u> (von: Zeitstempel, bis: Zeitstempel): Logeintrag +nächstenFinden (dieses: Logeintrag): Logeintrag

Hinweis: eintragFinden ist eine Klassenoperation, d.h. sie wirkt auf die Klasse und nicht auf ein einzelnes Objekt. Klassenoperationen werden in UML durch Unterstreichung gekennzeichnet.

Teilaufgabe 3.3: Klassendiagramm (3 Punkte)

Erstellen Sie an Hand der Ergebnisse von Teilaufgabe 3.2 ein Klassendiagramm Ihrer App. Dieses soll sowohl Klassen als auch Schnittstellen enthalten.

Teilaufgabe 3.4: Komponentenmodell (4 Punkte)

a) Zuordnung von Klassen und Schnittstellen zu Komponenten (2 Punkte)

Ordnen Sie die Klassen und Schnittstellen Ihrer Architektur den in Teilaufgabe 3.1 festgelegten Komponenten zu.
Hinweis 1: Es kann Situationen geben, in denen eine Klasse nicht genau einer Komponente zugeordnet werden kann, weil verschiedene Objekte der gleichen Klasse zu verschiedenen Komponenten gehören. In diesem Fall müssen Sie statt der Klasse die entsprechenden Objekte den Komponenten zuordnen.
Hinweis 2: Sie müssen auf der Kohäsion der Klassen und auf die Kopplung der Komponente achten. Ein gut entworfenes Komponentendiagramm besteht aus Komponenten die eine hohe Kohäsion der beinhaltenen Klassen haben und eine niedrige Kopplung zwischen den einzelnen Komponenten.

b) Erweitertes Komponentendiagramm (2 Punkte)

Erweitern Sie Ihr Komponentendiagramm aus Teilaufgabe 3.1, indem Sie die Schnittstellen zwischen den Komponenten explizit modellieren.
Hier ist ein Beispiel, wie Sie vorgehen sollten: Angenommen, Sie erstellen die Architektur für Ihr System und Sie haben die Folgende Anforderungen: Das System soll eine Logging-Funktionalität anbieten und ein Log haben zum speichern von Nachrichten. Sie entscheiden sich die Logging-Funktionalität in einer *Archiv*-Komponente zu realisiert und das der Log in der Komponente *Systemsteuerung* implementiert werden soll. Aus diesem Grund entscheiden Sie, dass die Klasse EreignisLog die Sie in der Teilaufgabe 3.2 identifiziert haben eine Schnittstelle sein soll, welche innerhalb von Archiv zu implementieren ist.

Das erweiterte Komponentenmodell in UML Notation sähe dann so aus:

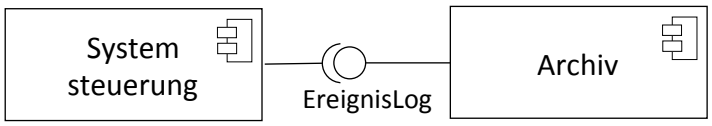

Bild 1. UML-Komponentendiagram. Die Komponente Archiv bietet die Logging-Funktionalität der Komponente Systemsteuerung mit Hilfe der EreignisLog Schnittstelle.

Teilaufgabe 3.5: Dokumentation der Annahmen (1 Punkt)

Dokumentieren Sie alle getroffenen Annahmen explizit in einer Liste. (In einem realen Projekt würden diese Annahmen fortlaufend mit den Interesseneignern geklärt.)

4 Aufgabenstellung Teil 2: Architektur- und Entwurfsmuster

Teilaufgabe 3.6: Entwurfsmuster in GWT Tutorial (3 Punkte)

Durchsuchen Sie das StockWatcher GWT Tutorial und finden Sie eine Stelle, an den ein Architekturmuster oder Entwurfsmuster verwendet wird. Für das gefundene Muster nennen Sie:
- Den Namen des Musters und seine Kategorie,
- Die Liste der Beteiligten in diesem Muster,

- Die Java Elemente (Klassen, Schnittstellen, Methoden), welche im Rahmen dieses Musters zusammenarbeiten und welche Rolle sie in diesem Muster einnehmen.

Literatur

[1] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. Design patterns: elements of reusable object oriented software. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1995.

Nach Abgabe der Übung muss jeder Mitglied der Gruppe in der Lage sein den Tutoren Auskunft über alle Teile der Übung zu geben und eventuelle Fragen beantworten.