Hinweise zur Abgabe der Architekturbeschreibung für Software–Projekt 2

Prof. Dr. Rainer Koschke

7. Mai 2014

Ihr sollt eine Software-Architektur für das durch Eure Anforderungsspezifikation beschriebene System entwerfen und beschreiben. Hierzu führt Ihr die Faktorenanalyse und Strategieentwicklung – wie in der Vorlesung beschrieben – durch.

Die Ergebnisse der globalen Analyse sollen durch eine Faktortabelle und sogenannte Problemkarten (siehe Vorlesung) beschrieben werden. Die Problemkarten beschreiben alternative Entwurfsstrategien, um mit Problemen umzugehen.

Außerdem sollt Ihr den Aufbau des Systems aus den folgenden Blickwinkeln beschreiben:

- Konzeptioneller Aufbau (konzeptioneller Blickwinkel nach Hofmeister et al.)
- Statischer Aufbau (Modulblickwinkel nach Hofmeister et al.); zum statischen Aufbau in Form von Modulen
- Abweichend zu den Sichten von Hofmeister et al. fordern wir auch eine Beschreibung des Datenmodells ein. Dies wird u.a. jenes der Anforderungsspezifikation enthalten, allerdings mit implementierungsspezifischen Änderungen und Erweiterungen:
  - Wie kann zwischen den verschiedenen Objekten navigiert werden?
  - Welche Datentypen werden verwendet?
  - Welche Collections-Typen werden wo verwendet, um die Multiplizitäten von Assoziationen zu repräsentieren?
  - Wie wird das Modell auf die zugrundeliegende Datenbank abgebildet?
  - ...

  Die Datensicht wird in der Architekturspezifikation im Kapitel *Datensicht* beschrieben.

- Dynamischer/physikalischer Aufbau (Ausführungsblickwinkel nach Hofmeister et al.): **Muss in SWP-2 ausgefüllt werden**

Das Zusammenspiel der verschiedenen Module aus der Modulsicht wird anhand von ausgewählten Anwendungsfällen per UML-Sequenzdiagramm erläutert. Dazu könnt ihr Euch 2-3 Anwendungsfälle heraussuchen, die möglichst viele (am besten alle) Module der Architektur abdecken.

Sequenzdiagramme können u.U. sehr groß werden. In diesem Fall solltet Ihr sie sinnvoll weiter unterteilen, um das Ganze lesbar zu gestalten. Sequenzdiagramme könnt ihr zum Beispiel mit dem Tool *Visual Paradigm* erzeugen oder auch mit *pic2plot* oder *UMLGraph* erstellen.

Ihr könnt die Blickwinkel und deren Konzepte benutzen, die in der Vorlesung vorgestellt wurden. Solltet Ihr weitere Blickwinkel heranziehen oder die existierenden Blickwinkel erweitern, dann müsst Ihr diese neuen Blickwinkel bzw. Erweiterungen beschreiben (in Analogie zu den Beschreibungen aus der Vorlesung). Die Bedeutung jedes Diagramms und jedes Begriffs muss klar sein.

Diagramme könnt Ihr in der grafischen Notation, die in der Vorlesung eingeführt wurde, und/oder mit anderen Mitteln der UML erstellen. Solltet Ihr eine eigene/andere Notation verwenden, dann müsst Ihr die grafischen Elemente genau erläutern (durch Zuordnung eines grafischen Elements zu einem Konzept eines Blickwinkels). Solltet Ihr andere UML-Stereotypen benutzen als jene, die in der Vorlesung eingeführt wurden, so müsst Ihr diese erläutern.

Die Verbindung von Faktortabellen, Problemkarten und Architektursichten muss einfach nachzuvollziehen sein. Insbesondere muss klar erkennbar sein, wie sich die Strategien in den verschiedenen Sichten niedergeschlagen haben. Beschreibt daher explizit, wie die Strategien zu den verschiedenen Sichten führen. Beschreibt weiterhin die Abbildung der verschiedenen Sichten aufeinander, sofern dies nicht offensichtlich ist.

Für alle veränderlichen Einflussfaktoren soll beschrieben werden, wie das System entsprechend angepasst werden kann.

Eine LaTeX-Vorlage für die Struktur des Dokuments findet Ihr in *Stud.IP*.

Hinweise

Testet Euren Entwurf vor der Abgabe mit den Anwendungsfällen, indem Ihr für jeden relevanten Anwendungsfall (und seine Varianten) das Verhalten aller beteiligten Module/Komponenten gedanklich durchspielt.

Die Modulsicht könnt Ihr mit UML-Klassendiagrammen beschreiben. Ihr müsst also nicht unbedingt den Stereotyp *Module* benutzen.

Die Schnittstellenbeschreibung der Module erfolgt in Form von Javadoc-Kommentaren. Erzeugt dazu zu den Modulen die entsprechenden Klassen und Interfaces direkt als

Java-Quellcode. Verseht diese mit den Methoden, die sich in der von Euch entworfenen Schnittstelle befinden. Dokumentiert die Module und Methoden mit Javadoc-Kommentaren. Hier soll nicht dokumentiert werden, WIE die Methoden ihre Aufgaben erfüllen (sollen), sondern WAS die Methoden tun und WAS ihre Parameter und evtl. Rückgabewerte bedeuten. Gebt den Quellcode mit ab und verweist von Eurem Dokument an geeigneter Stelle darauf.

Beschreibt die Schnittstellen der Module (Auflistung aller Methoden mit den Signaturen sowie das zugrunde liegende Protokoll, d.h. in welcher Reihenfolge die Methoden aufgerufen werden können, und Vor- und Nachbedingungen). Ihr könnt hierzu UML-Sequenzdiagramme benutzen, um das Verhalten exemplarisch zu illustrieren; bedenkt dabei jedoch, dass UML-Sequenzdiagramme nur EINE mögliche Ausführung beschreiben, Ihr aber natürlich ALLE möglichen Ausführungen beschreiben sollt. Evtl. sind hier auch Zustandsübergangsdiagramme geeignet.

Protokolle sollen an einer Stelle beschrieben werden; hierfür eignet sich vermutlich am besten die Modulsicht.

Wenn Ihr es für notwendig haltet, könnt Ihr Blickwinkel auf nahezu beliebige Weise erweitern, indem Ihr Konzepte und Relationen hinzufügt oder spezialisiert. Die Änderung muss jedoch genau beschrieben werden und für uns nachvollziehbar sein.

Ebenso dürft Ihr die Notation erweitern oder gänzlich ändern, solange Ihr uns genau erklärt, wie sie zu interpretieren ist.

Es muss wiederum eindeutig kenntlich gemacht werden, welche Teile von welchem Autor geschrieben wurden.

Bewertungskriterien

Wir werden Eure Architekturbeschreibung aus Sicht eines Projektcontrollers des Kunden beurteilen. Hierfür sind die folgenden Aspekte relevant:

- Termintreue: Wurde die Architekturbeschreibung (AB) fristgerecht abgegeben?
- Äußere Form: Ist die AB frei von Rechtschreibfehlern? Sind Tabellen und Grafiken lesbar und verständlich?
- Äußere Vollständigkeit: Werden Aussagen zu allen geforderten Aspekten gemacht (siehe oben)?
- Inhaltliche Vollständigkeit: Erfüllt das System die Anforderungen?
- Präzision: Sind die Angaben präzise und verständlich?
- Realisierbarkeit: Lässt sich der Entwurf implementieren?
- Änderbarkeit: Ist beschrieben, welche Anpassungen vorzunehmen sind, wenn sich Einflussfaktoren ändern?
- Granularität: Erfolgen die Angaben in ausreichendem Detail, so dass die AB von

Entwicklern implementiert werden kann?

- Nachvollziehbarkeit: Sind die Entwurfsentscheidungen leicht zurückzuführen auf Kundenanforderungen und Einflussfaktoren? Sind die Teile der AB untereinander verbunden? Ist nachvollziehbar, welche Strategien wie und warum umgesetzt wurden? Ist dokumentiert, welche Alternativen prinzipiell in Frage kamen und warum die ausgewählten den nicht ausgewählten vorgezogen wurden?
- Innere Konsistenz: Sind die gemachten Aussagen konsistent zueinander?
- Äußere Konsistenz: Sind die gemachten Aussagen konsistent mit dem, was der Kunde wünscht? Ist die AB konsistent mit dem Projektplan, d.h. einigermaßen realistisch in der gegebenen Zeit realisierbar?

Dies sind die wichtigstens Teile der Architekturbeschreibung, auf die wir besonderen Wert legen:

- Konzeptionelle Sicht
- Modulsicht
- Problemkarten
- Zusammenhang Architektur+Einflussfaktoren/Anwendungsfälle