Universität Bremen
FB 3 – Informatik
Prof. Dr. Rainer Koschke
TutorIn: Euer/Eure TutorIn

Software–Projekt 2 WiSe 2019/2020

VAK 03-BA-901.02

Architekturbeschreibung

xxxxxx xxxxxxx xxxxxxxx@tzi.de
xxxx xxxxxxxx xxxx@tzi.de

Abgabe: TT. Monat JJJJ — Version 1.0

Inhaltsverzeichnis

Version und Änderungsgeschichte

Die aktuelle Versionsnummer des Dokumentes sollte eindeutig und gut zu identifizieren sein, hier und optimalerweise auf dem Titelblatt.

Version	Datum	Änderungen
0.1	TT.MM.JJJJ	Dokumentvorlage als initiale Fassung kopiert
0.2	TT.MM.JJJJ	...
...		

1 Einführung

1.1 Zweck

Was ist der Zweck dieser Architekturbeschreibung? Wer sind die LeserInnen?

1.2 Status

1.3 Definitionen, Akronyme und Abkürzungen

1.4 Referenzen

1.5 Übersicht über das Dokument

2 Globale Analyse

Hier werden Einflussfaktoren aufgezählt und bewertet sowie Strategien zum Umgang mit interferierenden Einflussfaktoren entwickelt.

2.1 Einflussfaktoren

Hier sind Einflussfaktoren gefragt, die sich auf die Architektur beziehen. Es sind ausschließlich architekturrelevante Einflussfaktoren, und nicht z. B. solche, die lediglich einen Einfluss auf das Projektmanagement haben. Fragt Euch also bei jedem Faktor: Beeinflusst er wirklich die Architektur? Macht einen einfachen Test: Wie würde die Architektur aussehen, wenn ihr den Einflussfaktor E berücksichtigt? Wie würde sie aussehen, wenn Ihr E nicht berücksichtigt? Kommt in beiden Fällen dieselbe Architektur heraus, dann kann der Einflussfaktor nicht architekturrelevant sein.

Es geht hier um Einflussfaktoren, die

1. *sich über die Zeit ändern,*

2. *viele Komponenten betreffen,*
3. *schwer zu erfüllen sind oder*
4. *mit denen man wenig Erfahrung hat.*

Die Flexibilität und Veränderlichkeit müssen ebenfalls charakterisiert werden.

1. *Flexibilität: Könnt Ihr den Faktor zum jetzigen Zeitpunkt beeinflussen?*
2. *Veränderlichkeit: ändert der Faktor sich später durch äußere Einflüsse?*

Unter Auswirkungen sollte dann beschrieben werden, wie *der Faktor* was *beeinflusst. Das können sein:*

- *andere Faktoren*
- *Komponenten*
- *Operationsmodi*
- *Designentscheidungen (Strategien)*

Verwendet eine eindeutige Nummerierung der Faktoren, um sie auf den Problemkarten einfach referenzieren zu können.

2.2 Probleme und Strategien

Aus einer Menge von Faktoren ergeben sich Probleme, die nun in Form von Problemkarten beschrieben werden. Diese resultieren z. B. aus

- *Grenzen oder Einschränkungen durch Faktoren*
- *der Notwendigkeit, die Auswirkung eines Faktors zu begrenzen*
- *der Schwierigkeit, einen Produktfaktor zu erfüllen, oder*
- *der Notwendigkeit einer allgemeinen Lösung zu globalen Anforderungen.*

Dazu entwickelt Ihr Strategien, um mit den identifizierten Problemen umzugehen.

Achtet auch hier darauf, dass die Probleme und Strategien wirklich die Architektur betreffen und nicht etwa das Projektmanagement. Die Strategien stellen im Prinzip die Designentscheidungen dar. Sie sollten also die Erklärung für den konkreten Aufbau der verschiedenen Sichten liefern.

Beschreibt möglichst mehrere Alternativen und gebt an, für welche Ihr Euch letztlich aus welchem Grunde entschieden habt. Natürlich müssen die genannten Strategien in den folgenden Sichten auch tatsächlich umgesetzt werden!

Ein sehr häufiger Fehler ist es, dass SWP-Gruppen arbeitsteilig vorgehen: die eine Gruppe schreibt das Kapitel zur Analyse von Faktoren und zu den Strategien, die andere Gruppe beschreibt die diversen Sichten, ohne dass diese beiden Gruppen sich abstimmen. Natürlich besteht aber ein Zusammenhang zwischen den Faktoren, Strategien und Sichten. Dieser muss erkennbar sein, indem sich die verschiedenen Kapitel eindeutig

aufeinander beziehen.

3 Konzeptionelle Sicht

Diese Sicht beschreibt das System auf einer hohen Abstraktionsebene, d. h. mit sehr starkem Bezug zur Anwendungsdomäne und den geforderten Produktfunktionen und -attributen. Sie legt die Grobstruktur fest, ohne gleich in die Details von spezifischen Technologien abzugleiten. Sie wird in den nachfolgenden Sichten konkretisiert und verfeinert. Die konzeptionelle Sicht wird mit UML-Komponentendiagrammen visualisiert.

4 Modulsicht

Diese Sicht beschreibt den statischen Aufbau des Systems mit Hilfe von Modulen, Subsystemen, Schichten und Schnittstellen. Diese Sicht ist hierarchisch, d. h. Module werden in Teilmodule zerlegt. Die Zerlegung endet bei Modulen, die ein klar umrissenes Arbeitspaket für eine Person darstellen und in einer Kalenderwoche implementiert werden können. Die Modulbeschreibung der Blätter dieser Hierarchie muss genau genug und ausreichend sein, um das Modul implementieren zu können.

Die Modulsicht wird durch UML-Paket- und Klassendiagramme visualisiert.

Die Module werden durch ihre Schnittstellen beschrieben. Die Schnittstelle eines Moduls M ist die Menge aller Annahmen, die andere Module über M machen dürfen, bzw. jene Annahmen, die M über seine verwendeten Module macht (bzw. seine Umgebung, wozu auch Speicher, Laufzeit etc. gehören). Konkrete Implementierungen dieser Schnittstellen sind das Geheimnis des Moduls und können vom Programmierer festgelegt werden. Sie sollen hier dementsprechend nicht beschrieben werden.

Die Diagramme der Modulsicht sollten die zur Schnittstelle gehörenden Methoden enthalten. Die Beschreibung der einzelnen Methoden (im Sinne der Schnittstellenbeschreibung) geschieht allerdings per Javadoc im zugehörigen Quelltext. Das bedeutet, dass Ihr für alle Eure Module Klassen, Interfaces und Pakete erstellt und sie mit den Methoden der Schnittstellen verseht. Natürlich noch ohne Methodenrümpfe bzw. mit minimalen Rümpfen. Dieses Vorgehen vereinfacht den Schnittstellenentwurf und stellt Konsistenz sicher.

Jeder Schnittstelle liegt ein Protokoll zugrunde. Das Protokoll beschreibt die Vor- und Nachbedingungen der Schnittstellenelemente. Dazu gehören die erlaubten Reihenfolgen, in denen Methoden der Schnittstelle aufgerufen werden dürfen, sowie Annahmen über Eingabeparameter und Zusicherungen über Ausgabeparameter. Das Protokoll von Modulen wird in der Modulsicht beschrieben. Dort, wo es sinnvoll ist, sollte es mit Hilfe von Zustands- oder Sequenzdiagrammen spezifiziert werden. Diese sind dann einzusetzen, wenn der Text allein kein ausreichendes Verständnis vermittelt (insbesondere bei komplexen oder nicht offensichtlichen Zusammenhängen).

Der Bezug zur konzeptionellen Sicht muss klar ersichtlich sein. Im Zweifel sollte er explizit erklärt werden. Auch für diese Sicht muss die Entstehung anhand der Strategien erläutert werden.

5 Datensicht

Hier wird das der Anwendung zugrundeliegende Datenmodell beschrieben. Hierzu werden neben einem erläuternden Text auch ein oder mehrere UML-Klassendiagramme verwendet. Das hier beschriebene Datenmodell wird u. a. jenes der Anforderungsspezifikation enthalten, allerdings mit implementierungsspezifischen Änderungen und Erweiterungen. Siehe die gesonderten Hinweise.

6 Ausführungssicht

Die Ausführungssicht beschreibt das Laufzeitverhalten. Hier werden die Laufzeitelemente aufgeführt und beschrieben, welche Module sie zur Ausführung bringen. Ein Modul kann von mehreren Laufzeitelementen zur Laufzeit verwendet werden. Die Ausführungssicht beschreibt darüber hinaus, welche Laufzeitelemente spezifisch miteinander kommunizieren. Zudem wird bei verteilten Systemen (z. B. Client-Server-Systeme) dargestellt, welche Module von welchen Prozessen auf welchen Rechnern ausgeführt werden.

7 Zusammenhänge zwischen Anwendungsfällen und Architektur

In diesem Abschnitt sollen Sequenzdiagramme mit Beschreibung(!) für zwei bis drei von Euch ausgewählte Anwendungsfälle erstellt werden. Ein Sequenzdiagramm beschreibt den Nachrichtenverkehr zwischen allen Modulen, die an der Realisierung des Anwendungsfalles beteiligt sind. Wählt die Anwendungsfälle so, dass nach Möglichkeit alle Module Eures entworfenen Systems in mindestens einem Sequenzdiagramm vorkommen. Falls Euch das nicht gelingt, versucht möglichst viele und die wichtigsten Module abzudecken.

8 Evolution

Beschreibt in diesem Abschnitt, welche Änderungen Ihr vornehmen müsst, wenn sich Anforderungen oder Rahmenbedingungen ändern. Insbesondere würden hierbei die in der Anforderungsspezifikation unter „Ausblick“ genannten Punkte behandelt werden.

. . .