Zürcher Hochschule
für Angewandte Wissenschaften

School of Engineering

InIT Institut für Angewandte Informations-technologie

Auftrag Projektresultate

1 Einleitung

Dieses Dokument beschreibt abschliessend die Resultate, die an den vorgegebenen Meilen-steinen (M1-3) abgegeben werden müssen. Laden Sie die Artefakte vor der Präsentation auf Moodle in den entsprechenden Team-Ordner hoch. Die genauen Abgabemodalitäten sind auf Moodle beim jeweiligen Meilenstein beschrieben.

2 Projektskizze (M1)

2.1 Resultate

Projektskizze gemäss Vorgaben im Dokument «Auftrag Projektskizze».

3 Lösungsarchitektur (M2)

3.1 Inhaltsstruktur Technischer Bericht I

- Use-Case-Modell
- Zusätzliche Anforderungen
- Domänenmodell
- Softwarearchitektur
- Design-Artefakte
- Implementation
- Projektmanagement
- Glossar

3.1 Use-Case-Modell

Alle identifizierten Use Cases werden in einer Übersicht in einem UML-Use-Case-Diagramm mit den dazugehörigen Akteuren dargestellt. Dabei soll das UML-Use-Case-Diagramm auch die Abgrenzung gegenüber externen Systemen darstellen (Systemkontext).

Der oder die fachlich wichtigsten Use Cases müssen ausführlich («fully dressed») ausformuliert werden.

Weitere wichtige Use Cases werden normal («casual») ausformuliert, während der Rest der Use Cases noch kurz («brief») beschrieben wird.

Für das Standardszenario des fachlich wichtigsten und vollständig ausformulierten Use-Case wird ein System-Sequenzdiagramm (SSD) erstellt.

Für die wichtigsten Use-Cases werden UI-Sketches erstellt und allenfalls bei einem komplexe-ren UI die Navigationsmöglichkeiten (Dialogablauf) in einem Diagramm dargestellt.

3.2 Zusätzliche Anforderungen

Hier werden weitere funktionale und vor allem die nicht funktionalen Anforderungen (Randbedingungen und Qualitätsanforderungen) angegeben. Zusätzlich werden wichtige Regeln des Problemgebietes (z.B. Geschäfts- oder Spielregeln) und sonstige wichtige Informationen (z.B. weitere Vorgaben und Randbedingungen für die Entwicklung der Applikation) aufgelistet.

3.3 Domänenmodell

Es ist ein Domänenmodell zu entwickeln, das die (fachlichen) Konzepte der wichtigsten Use-Cases darstellt. Das Domänenmodell wird mit einem (konzeptuellen) UML-Klassendiagramm dargestellt.

3.4 Softwarearchitektur

Darstellung und Beschreibung der gewählten Softwarearchitektur mit Begründung des Entscheides.

Die Softwarearchitektur ist mindestens aus der Sicht der Schichtung (logische Architektur) und Subsysteme/Module mit einem UML-Paketdiagramm darzustellen. Spezieller Fokus hat dabei die Trennung von UI und Domänenlogik sowie der Einsatz von Frameworks.

Falls es eine verteilte Anwendung ist, müssen das Verteilungsmodell (Client/Server, Peer-to-Peer etc.) und die gewählte Kommunikationsart beschrieben und begründet werden.

3.5 Design-Artefakte

Dokumentieren Sie das bestehende Design mit einem Design-Klassendiagramm (DCD) und geeigneten Interaktionsdiagrammen.

3.6 Implementation

Beschreiben Sie hier kurz, wie die gewählte Softwarearchitektur verifiziert wurde. Die gewählte Softwarearchitektur muss mit einer partiellen Implementation (Prototyp) der wichtigsten Use Cases verifiziert sein. Der vollständige Quellcode (inkl. Unit-Tests) wird in elektronischer Form auf Moodle im Team-Ordner abgelegt.

3.7 Projektmanagement

Die Planung der bisherigen Iterationen und der zugehörige Aufwand sind erfasst und dargestellt. Pro Iteration wird der tatsächliche Aufwand und die erreichten Resultate mit dem geplanten Aufwand und den gesteckten Zielen verglichen und die allfällig getroffenen Massnahmen festgehalten. Zum Schluss gehört auch die detaillierte Planung der nächsten Iterationen inklusive Aufwandsschätzung zur Abgabe, wie auch die aufdatierte Risikoliste.

3.8 Glossar

Ein Glossar, wo die wichtigsten Begriffe des Problemgebiets (s. Domänenmodell) erklärt und definiert werden, ist zu erstellen. Dieses Glossar wird im Laufe des Projektes laufend ergänzt und aktualisiert.

4 Beta-Release (M3)

In den nachfolgenden zwei Abschnitten werden die Beurteilungsaspekte für das Beta-Release kurz erläutert.

4.1 Implementation

Folgende Aspekte werden bei der Implementation bewertet:

- Übereinstimmung mit dem Design
- Ist der Code lauffähig
- Code-Qualität (Clean Code, Namensgebung, Einhalten von Coding Conventions)
- Code-Dokumentation
  - Kommentierte Verantwortlichkeiten im Klassenkommentar
  - Kommentierte Öffentliche Methoden
- Testing
  - Testkonzept (im technischen Bericht beschrieben)
  - Abdeckung und Qualität der Unit-, Integrations- und Systemtests

4.2 Resultat

- Erreichte Ziele (im Vergleich zur Projektskizze bzw. den vereinbarten Features)
- Umfang der implementierten Funktionalität
- Schwierigkeitsgrad
- Investierter Aufwand

5 Technischer Bericht II (M3)

Es ist ein technischer Projektbericht (ca. 30 Seiten) mit den wichtigsten Resultaten des Projekts zu erstellen. Der Bericht soll sprachlich korrekt und logisch verfasst sein und zweckdienliche Bilder mit korrekter Nummerierung, Legenden und Referenzierung enthalten. Die im Bericht verwendeten Artefakte aus Projektskizze und Lösungsarchitektur sollen angemessen gemäss Feedback der Betreuer aufdatiert werden. Diagramme sind ausreichend zu erläutern.

5.1 Inhaltsstruktur

- Projektidee
- Analyse
- Design
- Implementation
- Resultate
- Anhang

5.2 Projektidee

Die wichtigsten Punkte (alle ausser 2.2.6-2.2.9) der aufdatierten Projektskizze sollen hier nochmals zusammenfasst werden.

5.3 Analyse

Die wichtigsten Resultate der Problemanalyse sind hier zu dokumentieren:

- Use-Case-Modell
  - UML-Use-Case-Diagramm zur Übersicht
  - Wichtigste Use Cases fully dressed, Rest casual oder brief
- Zusätzliche Anforderungen
- Domänenmodell

5.4 Design

Die wichtigsten Aspekte der gewählten Lösung sind zu dokumentieren:

- Softwarearchitektur
- Design-Klassendiagramm (DCD)
- Ausgewählte Interaktionsdiagramme
- Dokumentation wichtiger Architekturaspekte, Designentscheide und angewendeter Design Patterns

Dokumentieren Sie das bestehende Design (primär Domänenschicht, ohne UI) mit einem Design-Klassendiagramm (DCD) und geeigneten Interaktionsdiagrammen. Das UML-Klassendiagramm kann logisch auch auf mehrere Seiten aufgeteilt werden. UI-Klassen gehören nicht in dieses Diagramm.

Die angewendeten Prinzipien und Design Patterns sollen angemessen dokumentiert sein.

5.5 Implementation

Die wichtigsten Informationen zur Implementation der Lösung (Packages, etc.) sowie die durchgeführten Tests sind hier zu dokumentieren:

- Testkonzept: Zusammenfassung des gemachten Tests (Unit-, Integrations- und Systemtests) und Testresultate
- Installationsanleitung
- Code-Dokumentation
- Der vollständige Quellcode (inkl. Unit-Tests) wird in elektronischer Form auf Moodle im Team-Ordner abgelegt (ZIP-Datei).
- Jede Klasse und alle öffentlichen Methoden und Attribute müssen (kurz) mittels JavaDoc dokumentiert werden. Der fertige JavaDoc-Kommentar muss ebenfalls (elektronisch) abgegeben werden.

5.6 Resultate

Die Projektresultate sind kurz zusammenzufassen und in Bezug auf die ursprüngliche Projektidee zu stellen:

- Zusammenfassung der erreichten Ziele
- Offene Punkte
- Ausblick auf mögliche Weiterentwicklungen

5.7 Anhang

- Projektmanagement
  Beschreibung des Vorgehens/Methodik (Meilenstein-/Iterationsplan). Zusammenfassung der tatsächlichen Aufwände mit den geplanten Aufwänden pro Iteration und insgesamt. Begründung von signifikanten Abweichungen.
- Verzeichnisse
  - Literatur-, Abbildungs-, Tabellenverzeichnis
  - Glossar