Übungsbeispiel zur VU
Software Wartung und Evolution
SS 2008 (184.169)

Wartung und Evolution der "Java Notes" Applikation

© DI Dr. Johannes Weidl-Rektenwald

1 Einleitung

Die Firma No Real Comp Inc. (NRC) hat eine Applikation zur Informationserfassung in Verwendung („Java Notes“). Nach dem Umstieg auf neue Client PCs im Zuge eines Desktop Outsourcing Projektes wurde die Applikation von der Installations-CD des Softwarelieferanten neu installiert, funktioniert aber nicht mehr. Ihr Team wurde beauftragt, die Applikation wieder zum Laufen zu bringen. Im Gespräch mit der Firma NRC stellt sich heraus, dass auch Programmerweiterungen unbedingt nötig wären und sie erklären sich bereit, diese ebenfalls durchzuführen. Der ursprüngliche Lieferant wurde in der Zwischenzeit aus dem Firmenregister gelöscht und steht nicht mehr zur Verfügung.

Die Applikation soll wie folgt adaptiert werden:

- Die Applikation soll eine neue Datenbank unterstützen
  - Verwenden sie eine Datenbank ihrer Wahl, solange es nicht dieselbe ist, wie die ursprünglich unterstütze
  - Die verwendete Datenbank soll konfigurierbar sein (möglichst ohne den Code zu ändern).

- Die Applikation soll weitestgehend über ein Konfigurationsfile konfiguriert werden.

- Man kann einen Eintrag in der Datenbank nicht direkt ersetzen. Dieses Feature soll implementiert werden.

- Zurzeit ist es nicht möglich, alle Inhalte der Datenbank aufzulisten. Dieses Feature soll implementiert werden.

- Die Bedienung soll durchgehend über Tastatur Shortcuts ermöglicht werden, um die Eingabe zu beschleunigen.

- Es soll zusätzlich zur grafischen Oberfläche einen Command Line Client (CLI – Command Line Interace) geben, über den man – via Parameter gesteuert - die gängigsten Funktionen ausführen kann. Dieser Command Line Client kann dann zusätzlich für Batch Verarbeitungen eingesetzt werden (es ist **keine** interaktive Shell gefordert).

- Die Applikation soll einer 3-tier Architektur entsprechen

    - Alle Mitarbeiter sollen mit einer Client Applikation gleichzeitig über einen zentralen Server auf einer zentralen Datenbank arbeiten können – jeder mit seiner eigenen User Kennung und unter Verwendung eines Passwortes.

- Die Bedienung über die grafische Oberfläche ist nicht konsistent und soll verbessert bzw. vereinheitlicht werden. Auch soll das Look&Feel verbessert bzw. vereinheitlicht werden.

2 Aufgabenstellung und Deliverables

2.1 Projektplanung

Erstellen sie einen Projektplan mit folgendem Inhalt:

Dokument 1.1: Projektplan

- Enthält eine Einleitung, die das Projekt umreißt
- Enthält die Work Breakdown Structure (WBS)
    - Wer im Team macht was und bis wann?
    - Welchen Aufwand planen sie für die Tätigkeiten ein?
    - **Was ist der Gesamtaufwand des Projektes?**
- Enthält die Milestones und Fertigstellungstermine der Deliverables
- Welche Tools wollen sie verwenden und wozu?
- Enthält die Rollenvergabe im Team (Projektleiter etc.)

Beachten sie die zeitgerechte Abgabe des Projektplans (siehe „Übungsablauf und Organisation“ unten).

2.2 Redocumentation

Nachdem die ursprüngliche Dokumentation nicht mehr vorhanden ist (auf der CD ist außer dem Programm nichts zu finden), soll im **ersten Reverse Engineering Schritt** das Programm analysiert und neue Dokumentation erstellt werden. Da auch die Anforderungsspezifikation nicht mehr existiert (nirgends steht, was das Programm eigentlich macht), müssen sie die Funktionsweise der Software erst analysieren und dokumentieren.

Erstellen sie Dokumente mit dem folgenden Inhalt:

Dokument 2.1: Anforderungspezifikation / Use Case Spezifikation

- Was macht das Programm?
- Welche Funktionalitäten bietet es?
- Welche Use Cases gibt es?
    - D.h. wie interagiert der User mit dem Programm?
- Erstellen sie ein Glossar, das alle wichtigen fachlichen Begriffe auch für einen Projektfremden spezifiziert und erklärt.

- Können anhand des Source Codes auch Rückschlüsse auf ursprüngliche **nicht-funktionale** Anforderungen (z.B. Usability, Reliability, Performance, Security, Maintainability, Evolvability) gezogen werden?

Dokument 2.2: Technisches Handbuch

- Charakterisieren sie alle vom Programm verwendeten Technologien und die Systemumgebung
  - Programmiersprache
  - Unterstützte Betriebssysteme
  - Unterstütze Datenbanken
  - Externe Libraries
  - Installation und Verzeichnisstruktur
  - Start des Programmes

Javadoc Abgabe 2.3: Programmdokumentation in javadoc

- Dokumentieren sie das Programm ausführlich mit javadoc Kommentaren.
- Generieren sie mittels des Java Tools „javadoc“ die javadoc Dokumentation (Output: HTML).

2.3 Design Recovery

Im **zweiten Reverse Engineering Schritt** sollen Darstellungen des Systems auf höherer Abstraktionsstufe erzeugt werden.

Dokument 3.1: Architektur und Designspezifikation

- Beschreibung aller Komponenten bzw. der Klassenstruktur
  - Wie spielen die Komponenten zusammen?
  - Wie sind die logischen Verbindungen zwischen den Komponenten?
- Erstellen sie eine grafische Skizze aller Komponenten
- Beschreiben sie die Datenbankstruktur. Wie werden die Daten abgelegt?
- **Beurteilung des Designs und der Umsetzung**
  - Beurteilen sie die vom Programmierer gewählte Lösung. Was wurde gut und was weniger gut gelöst.
  - Gehen sie insbesondere **anhand von mehreren Beispielen** auf die Wartbarkeit und Erweiterbarkeit ein und beurteilen sie diese.
  - Welche Design-Entscheidungen können nachvollzogen werden? Welche halten sie für bedenklich?

2.4 Inbetriebnahme

Bringen sie das Programm zum Laufen.

Erstellen sie ein Dokument mit dem folgenden Inhalt:

Dokument 4.1: Dokumentation zur Inbetriebnahme

- Welche Probleme mussten gelöst werden, um das Programm zum Laufen zu bringen?
- Welche Änderungen am Code mussten durchgeführt werden?

2.5 Test

Testen sie das Programm, um mögliche funktionale Fehler zu finden.

Dokument 5.1: Funktionaler Test

- Dokumentieren sie möglichst alle funktionalen Fehler im ursprünglichen Programm.

2.6 Korrektive Wartung

Korrigieren sie die Fehler im Programmcode.

Codeabgabe 6.1: Verbessertes Programm nach korrektiver Wartung (Java Source Code)

- Geben sie das Programm und alle zur Ausführung benötigten Artefakte ab (Beachten sie für die Abgabe unbedingt die Modalitäten im Abschnitt „Übungsablauf und Organisation“)

Erweitern sie Dokument 5.1 und dokumentieren sie zu jedem Fehler die Schritte, die zur Fehlerbeseitigung durchgeführt wurden. Dokumentieren sie auch die Tests, die die Fehlerbehebung verifizieren. Markieren sie die Erweiterungen zu Dokument 5.1, das in Abschnitt 2.5 erstellt wurde, andersfarbig!

Passen sie auch unbedingt die javadoc Kommentare im Source Code an!

2.7 Perfektionierende Wartung – Restructuring

Es sollen Restructuring Maßnahmen durchgeführt werden. Das Programm soll dadurch auf die bevorstehenden Modifikationen vorbereitet werden, damit diese möglichst einfach und ökonomisch durchgeführt werden können. Dabei wird – definitionsgemäß – die Funktionalität des Programms nicht verändert. Wenden sie zum Restructuring auch unbedingt **Refactoring** an.

Erstellen sie folgende Deliverables:

Dokument 7.1: Restructuring

- Vor dem Restructuring/Refactoring
  - Beschreiben sie die Maßnahmen und die erwarteten/erreichten Verbesserungen.
  - Schätzen sie den Aufwand, um die Änderung durchzuführen
  - Ökonomische Beurteilung: Steht der erwartete Aufwand in Relation zur erreichten Verbesserung?
  - Sollen im Hinblick auf Performance Verbesserung Restructuring Maßnahmen gesetzt werden und wenn ja, welche?
- Nach dem Restrucutring
  - Welche Methoden haben sie angewandt, wie und warum?

- Was ist **Refactoring** und welche Refactorings haben sie wie angewandt?
  - Geben sie die genauen Namen der Refactorings an!
- Welche Vorteile bzw. Nachteile ergeben sich aus dem Restructuring?
- Wie hat sich das Restructuring auf die Wartbarkeit und Evolutionsmöglichkeit ausgewirkt?

Codeabgabe 7.2: Restrukturiertes Programm

- Geben sie das Programm und alle zur Ausführung benötigten Artefakte ab (Beachten sie für die Abgabe unbedingt die Modalitäten im Abschnitt „Übungsablauf und Organisation“)

Javadoc Abgabe 7.3: Programmdokumentation in javadoc

- Generieren sie die veränderte Javadoc Dokumentation

2.8 Adaptive Wartung

Nun soll das Programm entsprechend den Kundenanforderungen (siehe Einleitung) angepasst bzw. erweitert werden

Erstellen sie folgende Deliverables:

Dokument 8.1: Spezifikation und Planung der Modifikationen

- Spezifizieren sie die nötigen Schritte, um die Modifikationen durchzuführen
- Spezifizieren sie die geplanten Modifikationen im Detail
- Schätzen sie den notwendigen Aufwand
- Ist es sinnvoll, weitere Restructuring Maßnahmen vor der adaptiven Wartung durchzuführen?
  - Wenn ja, beschrieben sie diese
- Dokumentieren sie alle **Designentscheidungen**, die sie bei der Umsetzung der Modifikationen treffen

Codeabgabe 8.2: Modifiziertes Programm

Javadoc Abgabe 8.3: Javadoc zum modifizierten Programm

Dokument 8.4: Beurteilung der Wartbarkeit und Erweiterbarkeit des modifizierten Programmes

- Inwiefern wurde durch die Modifikationen die Wartbarkeit und Erweiterbarkeit beeinflusst?
- War es notwendig, neuerlich Restructuring durchzuführen (und wenn ja, welche Maßnahmen wurden gesetzt), oder konnten alle Modifikationen sauber eingepflegt werden?
- Kann man ein Programm ohne Kenntnis der Natur der anstehenden Modifikationen für eine Wartbarkeit und Erweiterbarkeit optimieren?

2.9 Erstellung der Wartungsdokumentation

Um in Zukunft die geregelte Wartung zu gewährleisten, soll Wartungsdokumentation erstellt werden. Damit soll es sowohl der internen EDV Abteilung als auch externen Firmen möglich sein, die Wartung der „Java Notes“ Applikation zu übernehmen.

Erstellen sie folgendes Deliverable:

Dokument 9.1: Erstellen eines Wartungshandbuches

- Einführung in Architektur, Design, Systemumgebung, etc. für den Wartungsingenieur
- Beschreibung der Entwicklungsumgebung – was wird benötigt, um Wartung durchzuführen?
- Beschreibung der Systemumgebung
- Was sind die relevanten Dokumente für die verschiedenen Wartungsfälle?
- Wie ist im Wartungsfall vorzugehen?
- Welche Einstiegspunkte in den Code gibt es?
- Wie ist die Produktivstellung zu planen, was ist dabei zu berücksichtigen?
- Wie wird die Wartung zentral dokumentiert, um die Nachvollziehbarkeit zu gewährleisten?
- Wie ist die Software gepackaged? Welche Verzeichnisse beinhalten welche Komponenten der Applikation?
- Wie wird die Software installiert?
    - Wie muss das Zielsystem verändert werden, um die Applikation zu installieren (Server und Client)?
    - Wie kommunizieren Client und Server? Welche Ports müssen zwischen den Rechnern geöffnet sein.
    - Wie erfolgt die Kommunikation mit der Datenbank?

2.10 Produktivstellung

Es soll auch die Ablöse der Software durch die neu erstellte genau geplant werden, da das Programm von den Mitarbeitern rund um die Uhr eingesetzt wird und Ausfallszeiten minimiert werden sollen. Außerdem existiert kein Server, der das neue Client/Server System hosten könnte.

Dokument 10.1: Roll-out Plan / Planung des Wartungsfensters (Client/Server!)

- Beschreiben sie Schritt für Schritt die Ablöse des alten durch das neue System in einem Projektplan
    - Wie wird das neue System installiert?
    - Welche Anschaffungen müssen getätigt werden?
    - Wann wird auf das neue System umgestellt?
    - Wie lange ist die dadurch bedingte Ausfallszeit? Wie wird sie minimal gehalten?
    - Schätzen sie für jeden Schritt auch den nötigen Aufwand und bezeichnen sie die Personen im Unternehmen bzw. aus ihren Team, die den Schritt durchführen
    - Wie sieht das Notfallszenario aus?

2.11 Projektabschluss

Jedes Teammitglied erstellt einen Projektabschlussbericht. Der Name des Teammitglieds muss aus dem Bericht klar erkennbar sein!

Dokumentgruppe 11.1: Lessons Learned und Conclusio

- Beurteilen sie den Ablauf des Projektes
  - Was hat gut funktioniert, was weniger gut. Begründen sie ihre Beurteilung.
  - Haben sie den ursprünglich geschätzten Aufwand überschritten? Um wieviel und warum?
  - Was würden sie anders bzw. besser machen, wenn sie mit ihrer jetzigen Erfahrung das Projekt nochmals machen müssten.
- Beurteilen sie die finale Software
  - Welche Verbesserungen sind gut geglückt, welche Teile sind aus ihrer Sicht gar nicht oder nur mangelhaft umgesetzt worden.
- Verfassen sie ihr persönliches Fazit.

3 Ihr Input

- Source Code des Softwareprogramms "Java Notes" (in Java geschrieben)
- Test Datenbank (ein einzelnes File)

4 Übungsablauf und Organisation

- Über die Gruppenanmeldung im TUWIS++ System werden Übungsgruppen zu je fünf Personen gebildet (http://tuwis.tuwien.ac.at).
- Arbeiten sie das Web Skriptum von 2007 für die für die Übung relevanten Teile durch.
- Mailen sie den Projektplan (Deliverable Dokument 1.1) **bis spätestens Sonntag, den 13.04.2008, 24:00** an swe08@xion.at
  - **Alle** Dokumente müssen im .rtf, .doc, .pdf oder .html Format abgegeben werden! Im speziellen ist für den Projektplan kein Microsoft Project® oder anderes Projektmanagementtool-spezifisches Format erlaubt! Achten sie darauf, dass keine Links auf externe Objekte (Bilder, Diagramme) in den Dokumenten vorhanden sind. Achten sie darauf, dass die Dokumente nicht exorbitant groß sind (z.B. durch Einbinden großer Bilder).
  - **Alle** abgegebenen Dokumente müssen als Dateinamen bzw. Dokumenttitel die Dokumentnummer und den Dokumenttitel (wie in der Aufgabenstellung spezifiziert) tragen! Der Titel muss auf der ersten Seite sowie über Fußzeilen im ganzen Dokument ersichtlich sein.
    - Der Dateiname für die Projektplanabgabe als pdf ist also **„SWE08Gruppe<nn>_1_1_Projektplan.pdf"**

        - <nn> entspricht ihrer Gruppennummer formatiert auf zwei Stellen
        - Der Dateiname der Projektplan Datei in der Gesamtabgabe ist also „**1_1_Projektplan.pdf**“
    - Es darf jeder Mailabgabe nur **ein** Attachement beigefügt werden! Falls sie z.B. HTML als Format wählen und das Dokument mehrere HTML Dateien umfasst, packen sie die Dateien in ein ZIP Archiv. Achten sie darauf, dass das Dokument direkt aus dem ZIP Archiv durchgebrowst werden kann, d.h. das ZIP Archiv nicht auf die Festplatte entpackt werden muss.
        - Der Dateiname für die Projektplanabgabe als zip Archiv ist also **„SWE08Gruppe<nn>_1_1_Projektplan.zip“**
- Vereinbaren sie pro Gruppe für die Kalenderwoche 23 bzw. 24 (05.06.2008 bis 13.06.2008) **bis spätestens Sonntag, 27.04.2008, 24:00** einen Präsentationstermin (Ort: Fa. Xion IT Systems, Dresdnerstrasse 81-85, 8. Stock; Zeit: von 09:00 bis 16:00 jeweils zur vollen Stunde) per email an swe08@xion.at
    - In der ersten Woche (Woche 23) stehen Donnerstag (05.06.) und Freitag (06.06.) zur Verfügung, in der zweiten Woche (Woche 24) stehen Montag (09.06.) und Dienstag (10.06.) zur Verfügung.
- Liefern sie alle Deliverables bis spätestens **eine Woche vor** der Präsentation **(ohne Ausnahme!**) per email an swe08@xion.at.
    - Alle Files der Abgabe müssen in einer sinnvollen Verzeichnisstruktur abgelegt werden, die der Nummerierung der Deliverables entspricht
        - Legen sie der Abgabe unbedingt auch wieder Deliverable 1.1 (Projektplan) bei.
    - Jede Codeabgabe muss mittels **Batch File start.bat**, **startServer.bat startClient.bat bzw. startCLI.bat** aus dem Root-Verzeichnis ihrer jeweiligen Unterverzeichnisstruktur gestartet werden können (Achtung: Der Test erfolgt unter dem Betriebssystem Windows®)
        - Sie können davon ausgehen, dass sich das Programm "java" im Pfad befindet (aktuelles Java 6 JRE).
        - Zum Test der Programme wird das von der Software ursprünglich unterstützte Datenbankmanagementsystem (DBMS) herangezogen. Stellen sie also sicher, dass ein entsprechendes Datenbankfile mitgesendet wird. Der Name des Datenbankfiles muss dabei unbedingt „**dbnotes.mdb**“ sein. Die Datenbankstrukturen können sie natürlich ändern, d.h. ***ihre angepasste Software muss nicht mit dem ursprünglichen Datenbank-Schema laufen.***
        - Zum Test ist eine „ODBC Datasource“ mit dem Namen „**myDB**“ eingerichtet, die auf die Datei „dbnotes.mdb“ verweist.
        - Ansonsten muss das Programm ohne weitere Modifikationen lauffähig sein!
        - Insbesondere sind Passwörter, die zur Inbetriebnahme der Applikation gebraucht werden, **unbedingt** in einem File „readme.txt“ auf Root-Ebene der Abgabe zu dokumentieren!
        - ***Achten sie darauf, dass etwaige neu erstellte bzw. veränderte Datenbanken zumindest einige Testdaten beinhalten!***

        - Das Batch File zum Starten des CLIs sollte so aufgebaut sein, dass jeder implementierte Befehl aufgerufen wird (eine Art Test Batch)
    - Es darf der Mail nur **ein** Attachement beigefügt werden!
    - Dieses Attachement enthält die ZIP gepackte Verzeichnisstruktur.
    - Der Dateiname der Abgabe muss **SWE08AbgabeGruppe<nn>.zip** lauten
    - Alternative zur Mail-Abgabe: Download via URL
- Ablauf der Präsentation
    - Zur Präsentation ist die Anwesenheit **aller** Teammitglieder erforderlich.
    - Erstellen sie **eine** Powerpoint Präsentation (auf Diskette, CD oder USB Stick) - wenige Folien dafür präzise Inhalte
    - Mailen sie die Powerpoint Präsentation bis spätestens einen Tag vor dem Präsentationstermin an swe08@xion.at.
        - Der Dateiname der Präsentation muss **SWE08FinalGruppe<nn>.ppt** lauten
    - Jeder Gruppenteilnehmer muss die von ihm erarbeiteten Teile und sein Schlussresumé präsentieren
    - Jeder Gruppenteilnehmer muss das Gesamtprojekt - d.h. auch die Teile, die die anderen Gruppenteilnehmer präsentieren - kennen und auch Fragen dazu beantworten können
        - Nach der Präsentation gibt es eine Fragen- bzw. Diskussionsrunde
        - "Stoff" sind alle Themen, die das Übungsbeispiel umfaßt
        - Dauer: ca. 45-60 Minuten

5 Übungsinhalte

- Projektarbeit/Projektplanung
- Reverse Engineering (Design Recovery / Redocumentation)
- Restructuring
- Beurteilung der Wartbarkeit und Erweiterbarkeit von Software
- Umsetzen von Wartbarkeit und Erweiterbarkeit als nicht-funktionale Requirements in einem Softwaresystem
- Wartung eines Softwaresystems
- Arten der Software Wartung
- Verstehen des Unterschieds Software Entwicklung zu Software Wartung
- Kenntnis des Wartungs-Life-Cylces
- Wartungshandbuch und Wartungsfenster
- Gruppenkommunikation- und Zusammenarbeit, Präsentationstechnik

6 Tipps

- **Studieren sie die Angabe genau!** Es werden immer wieder Deliverables vergessen bzw. Themen verfehlt. Für eine gute Note müssen alle Aufgaben erfüllt und alle Deliverables bereitgestellt werden. Ich behalte mir vor, formal nicht entsprechende Deliverables nicht zu akzeptieren.
- Teilen sie Rollen bzw. Zuständigkeiten innerhalb der Gruppe ein (für z.B. Projektleitung, Koordination, Technik, Tools, Dokumentation). Planen sie Projektmeetings auch während der Laufzeit, z.B. zu bestimmten Milestones. Heben sie nicht alles für den Schluss auf.
- Bei der Präsentation muss man über Gesamtprojekt informiert sein – nicht nur über seinen eigenen Teil!
- Wenn sie bei der Lösung aufgrund von unvollständigen Angaben bzw. Informationen *Annahmen* treffen, dokumentieren sie diese unbedingt! Annahmen dürfen in keinem Fall die Lösung trivialisieren.

7 Anhang

Dokumenteninhalt:

Auflistung der Deliverables bzw. Unterverzeichnisstruktur der Abgabe:

Dokument 1.1: Projektplan

Dokument 2.1: Anforderungspezifikation / Use Case Spezifikation

Dokument 2.2: Technisches Handbuch

Javadoc Abgabe 2.3: Programmdokumentation in javadoc

Dokument 3.1: Architektur und Designspezifikation

Dokument 4.1: Dokumentation zur Inbetriebnahme

Dokument 5.1: Funktionaler Test

Codeabgabe 6.1: Verbessertes Programm nach korrektiver Wartung (Java Source Code)

Dokument 7.1: Restructuring

Codeabgabe 7.2: Restrukturierter Code

Javadoc Abgabe 7.3: Programmdokumentation in javadoc

Dokument 8.1: Spezifikation und Planung der Modifikationen

Codeabgabe 8.2: Modifiziertes Programm (Java Source Code)

Javadoc Abgabe 8.3: Javadoc zum modifizierten Programm

Dokument 8.4: Beurteilung der Wartbarkeit und Erweiterbarkeit des modifizierten Programmes

Dokument 9.1: Erstellen eines Wartungshandbuches

Dokument 10.1: Roll-out Plan / Planung des Wartungsfensters (Client/Server!)

Dokumentgruppe 11.1: Lessons Learned und Conclusio

Dokumentenlenkung:

V1.0 Version vom 17.02.2008