Schlussbewertung
Projekt „Bugtracker“
LV SWP2 SS 2005

Betreuung: **Prof. Knabe**

Autor: Manuel Bähnisch (s716631)
Thomas Barmeyer (s708459)
Isabel Schäfer Batista (s708888)
Alexander Fohgrub (s714797)
Thorsten Herrmann (s714610)

Datum: 09.07.2005
Version: 1.0
Dateiname: TFH_SWP2_sb_BugTracker.doc

Dokumenten Änderungen:

1. Known Issues

1.1. Bekannte Fehler

Abschließend sei noch auf einen Fehler im „Bugtrackers“ hingewiesen, der bis zur Fertigstellung des Projektes nicht mehr gelöst werden konnte. Hierbei handelt es sich um einen Bug in der Maske „Fehlermeldung erfassen“, die der Customer-Support-Mitarbeiter benutzt. Bei der genaueren Beschreibung des Produktes, in dem ein gemeldeter Fehler aufgetreten ist, wird die Angabe von Hardware- oder Software-Modul erwartet. Vorgesehen ist es, dass entweder nur die hardware-bezogenen oder nur die software-bezogenen Felder ausgefüllt werden können beziehungsweise beide Feldergruppen. Der Bug besteht darin, dass es im Moment noch erforderlich ist, beide Gruppen auszufüllen, da es sonst zu einer „empty-String“-Fehlermeldung kommt. Nur die Angabe von Hardware *oder* Software führt zu der genannten Fehlermeldung des Systems. Da bisher der Ursprung des Fehlers nicht lokalisiert werden konnte, wurde das System trotz dieses Fehlers zur Abgabe freigegeben.

1.2. Änderungen gegenüber dem Pflichtenheft

- Im Pflichtenheft wird der Begriff „spätester Abarbeitungstermin“ verwendet und ist mit dem „Fälligkeitstermin“ im System selbst synonym (s. 2.1.7 im Pflichtenheft).
- Eine Protokollierung von Änderungen an der Datenbank findet nur rudimentär bei einigen wenigen Funktionen statt (s. 2.2.1 PH)
- Das Anfügen von Daten an die Fehlermeldung ist nicht vollständig implementiert. Es läßt sich lediglich der Link zu einer Datei oder Verzeichnis händisch eingeben, welcher dann zur Fehlermeldung abgespeichert wird (s. 2.2.3 PH).
- Tests unter Betriebsbedingungen konnten nur recht eingeschränkt durchgeführt werden. Hierzu zählen Mehrbenutzertests, Tests über Modemverbindungen und Tests mit großem Datenbankinhalt (s. 3. und 3.3 sowie 4.2 PH)
- Entgegen der Vorgabe des Einsatzes einer PostgreSQL-Datenbank wurde für die Entwicklung eine MySQL-Datenbank verwendet. Der Austausch sollte jedoch keine größeren Probleme darstellen (s. 4.3 PH).
- Mangels Zeit wurde die Funktionalität, dass Entwickler die Bearbeitung eines Fehlers ablehnen können nicht implementiert. Diese hätte massiven Mehraufwand bedeutet, wie z.B. Verwalten der Gründe, Verwaltung welche Entwickler abgelehnt haben (je Fehlermeldung), etc. Desweiteren haben sich hier Plausibilitätsschwierigkeiten und mögliche Inkonsistenzen abgeleitet (s. 5.1.3 PH).
- Da Fehler nicht explizit angelehnt werden können, wurde die Mächtigkeit der Zuweisung von Fehlermeldungen durch den Abteilungsleiter im Gegenzug soweit gestärkt, als das dieser einen Überblick über die aktuellen Arbeiten seiner Mitarbeiter erhält und ggf. Fehler während der Bearbeitung zuweisen kann, was so nicht vorgesehen war (s. 6.1.7 und 6.1.8 PH).
- Bzgl der Priorität werden die Begriffe „Hoch“ und „Dringend“ bzw. „Niedrig“ und „Gering“ jeweils synonym verwendet (s. 6.1.5 PH).

2. Technikbewertung

2.1. JSP/Struts

Die Struts bzw JSP Technologie hat mir persönlich von der Idee her sehr gut gefallen, doch haperte es manchmal an der Umsetzung bzw. der Dokumentation. Bei Struts hatte ich das gefühl, das diese Technologie noch nicht sehr ausgereift ist, was sich durch irritierende Namensgebungen innerhalb der Deployment Diskriptoren und der fast wöchentlich erscheinenden Releases meiner Meinung ausdrückt.

Das Arbeiten mit diesen Techniken gestaltete sich nicht immer einfach und Problemlos. Da die Dokumentationen mit der Release-flut nicht immer mit halten konnten. Ein Beispiel dazu ist die Beschreibung von Funktionalitäten die es seid der Version 1.1 nicht mehr gibt. Auch hatte ich das gefühl, dass die JSP-Technologie im Sun Dokumentationen Wust versinkt.

2.2. Validation Framework

Das Validation-Framework der Apache-Commons-Group ist ein sehr nettes Tool, welches einem das Validieren von Formulareingabefeldern sehr vereinfacht. Negativ an diesem Framework ist eindeutig das Fehlerreporting, d.h. entweder es funktioniert oder man bekommt halt keine Anzeige. In den seltensten Fällen bekommt einen Aussagekräftigen Exception-Text, meist bekommt man gar keine Fehlerausgabe.

Alles im allen ist das Validation-Framework ein nettes Tool, welches schwächen in der Dokumentation und dem Error-Handling, aber dennoch sehr empfehlenswert für Formular validierungen ist.

2.3.Tomcat

Der Tomcat ist quasi Standard unter den freien Servlet Container. Er ist fast ohne jeglichen Aufwand einzurichten und leicht zu benutzen. Für Webapplikationen mit einem nicht zu hohen Anteil der Applikationsschicht reicht er völlig aus. Für große Enterprise Applikationen mit hohen Zugriffsraten und umfangreicher Logik, sollte jedoch auf einen Web Container mit externem Applikationsserver wie dem JBoss umgestiegen werden.
Durch die in unserem Projekt verwendete J2EE Security, musste ich leider erfahren, dass auch der Tomcat noch nicht voll ausgereift ist, und es an manchen Stellen an aktueller Dokumentation mangelt. So ist beispielsweise die Form-based Authentication nicht genau nach der Servlet Spezifikation implementiert worden, wodurch sie nicht zusammen mit einem Servlet Filter benutzt werden kann. Dieser Bug ist auch bekannt und wird in der Bugzilla Datenbank dokumentiert.
Durch den deklarativen Zugriffsschutz des Web Containers besteht die Möglichkeit mit relativ wenig Arbeitsaufwand seine Webapplikation zu schützen und zu isolieren. Doch so flexibel wie von der Spezifikation angepriesen ist sie leider noch nicht. Wenn in dem System auf spezielle Anforderungen eingegangen werden muss, erhöht sich der Arbeitsaufwand für die Anpassung sehr schnell, weshalb in manchen Fällen der Ansatz eines eigenen Security Systems sicherlich die bessere Lösung darstellt. In unserem Projekt überwiegten aber klar die Vorteile der Container Managed Security, auch gerade durch das Zusammenspiel mit dem Struts Framework.

2.4.Hibernate

Das objekt-relationale Mapping mit dem Tool Hibernate ist, zumindest in der älteren Version 2.1, sehr zu empfehlen, da die Einarbeitung recht unkompliziert ist und man sich schnell auf die erweiterten Funktionen konzentrieren kann, wie zum Beispiel die Hibernate Query Language (HQL), die komplexe Abfragen auf die Datenbank ermöglicht. Die Referenz zu Hibernate ist ausserordentlich gut und hilft bei nahezu allen Fragestellen weiter. Auch die Laufzeit-Exceptions bei der Entwicklung waren sehr aussagekräftig, so dass ohne große Erfahrung in der Deutung der Fehlermeldungen klar erkannt werden konnte, woher der Fehler kommt und wie er zu beheben ist.

2.5.JUnit

Junit ist als Testwerkzeug für komplexe Regeln sicherlich hilfreich. Beim Test einfacher Geschäftsregeln die schnell zu überblicken sind, haben sie mehr eine Alibi-Funktion gegenüber dem Kunden erzeugen aber keinen echten Nutzen. Es wird Code geschrieben, um Code zu testen, der offensichtlich korrekt sein muß.

2.6.DbUnit

Als Erweiterung von JUnit und der einhergehenden gleichen Anwendung stellt DBUnit ein starkes Tool dar, Datenbanken zu validieren. Das Auszählen oder Vergleichen von Metadaten, Spalten, Zellen etc. sind mit diesem Framework kein Problem.
Leider kam es bei der Programmierung mit DBUnit zu besonders schwerwiegenden Fehlern. So wurden bspw. Zeilen oder gar ganze Tabellen gelöscht, sobald man die Metadaten und Datensätze miteinander verglichen hat.

2.7.MulTEx

MulTEx ist ein Framework für das Exceptionhandling über mehrere Schichten.
Der große Vorteil dieses Frameworks liegt in den leicht zu verwendenen Methoden. Gemäß der Unterscheidung zwischen originären und Folgefehlern, ist es möglich Fehlermeldungsketten zu erstellen.
Beides, sowohl der Beginn als auch das Hinzufügen von Meldungen werden durch das Framework zum Kinderspiel.

3. Projektbewertung

3.1. Manuel Bähnisch

Als gewählter Projektleiter stellt sich mir die Bewertung des Projekts aus zwei Gesichtspunkten. Einmal aus Sicht des Teamleiters, zum anderen aus Sicht des Programmierers selbst.

Aus Sicht eines Projektleiters wurde ich im Wesentlichen als Koordinator gefragt. Die Verteilung und Zuteilung von Aufgaben bei Leerlauf ist auch ein Umstand der gelernt sein möchte, besonders dann wenn Termine, Interessen und Ansichten untereinander konsolidiert werden müssen. Ebenso ist der Umfang der Programmiererfahrung innerhalb des Teams recht unterschiedlich, was für ein ausschlaggebender Faktor ist. Trotz des anfänglich sehr trägen Fortschritts ist es uns letztendlich noch gelungen das Projekt zeitnah zu beenden. Dabei mußten einige der gesteckten Ziele sicherlich hinten angestellt werden, wobei diese aus heutiger Sicht bereits zu hoch gesteckt waren. Der blanke Einsatz von Arbeitszeit weit über das übliche Maß heraus bis an die Grenzen des möglichen hat uns letztendlich dazu verholfen ein recht ansehnliches qualitativ gutes Produkt abzuliefern. Es läßt sich sicherlich nicht mit kommerziellen Produkten vergleichen, ist jedoch auf der anderen Seite weit über den Prototypenstatus hinaus geschossen. An dieser Stelle möchte ich meinen Teamkollegen daher für die Zusammenarbeit danken.

Neben der Projektleitung oblag es mir ebenfalls mich mit der Logik des Systems auseinander zusetzen. Dies hatte vor allem den Nebeneffekt, das ich „zwischen den Schichten sitzend“ die Anforderungen aus der GUI selbst implementieren und die sich daraus ergebenden Folgen an die Datenbankschicht als Projektleiter delegieren konnte, so dass ich stets einen Überblick über den Status des Gesamtsystems hatte.

Obwohl anfangs vermehrt der Ruf nach dem Vorgehen nach XP laut wurde (so auch von mir selbst), habe ich im Nachhinein doch die großen Vorteile des klassischen Vorgehens (Analyse und Implementierung) feststellen müssen. Ein wesentlicher Faktor dabei ist, das man anfangs vor einem Riesenprojekt steht und eine Stelle finden muß, mit der begonnen wird. Nach anfänglich massiven Problemen, haben wir doch noch den richtigen Ansatzpunkt gefunden wodurch allerdings erst zu einem recht späten Stadium die End-Implementierung begann. Desweiteren sind mir, obwohl ich im derzeitigen Berufsleben ausschließlich mit C zu tun habe, besonders durch dieses Projekt die großen Vorteile der OOP bewußt geworden. Der eigene Einarbeitung in verschiedene Design-Patterns ist ebenfalls eine lehrreiche Erfahrung gewesen.

3.2. Thomas Barmeyer

Zum Ende des Software Projekts kann ich für mich persönlich folgendes Resumé ziehen. Im ersten Semester haben wir ziemlich viel Zeit für die Analyse des Systems verwendet, welche uns im zweiten Semester eindeutig gefehlt hat. Der Arbeitsaufwand im zweiten Semester war wesentlich höher, und ich hätte mir gewünscht, schon etwas früher mit der Implementierung begonnen zu haben. In der Analyse sind einfach noch nicht die Probleme bekannt, mit denen später zu kämpfen ist. Trotz allem ist eine Analysephase sehr wichtig, nur sollte sie etwas kompakter Ausfallen, sodass am Ende des zweiten Semesters noch genügend Zeit für die Testphase und das Schreiben des Handbuchs ist.

In meiner Vorbereitungszeit in den Semesterferien habe ich mich nach dem Vorbereitungsplan in die Logging API Log4J von Apache und in die Remote Method Invocation eingearbeitet. Im laufenden Semester stellte ich jedoch schnell fest, dass ich mit diesem Wissensstand unser unter Zeitdruck stehendes Projekt wenig vorantreiben könnte. Meine neuen Kernaufgaben im Projekt konzentrierten sich ab nun an darin, ein geeignetes Loginkonzept zu entwickeln und eine dynamische Anzeige der GUI entsprechend der Rollenverteilung zu erstellen. Durch diese Arbeit habe ich mich viele Stunden mit dem Tomcat Server und dem Struts Framework beschäftigt. Ich habe das Security System der J2EE kennen gelernt und weis um die kontrovers diskutierten Vor- und Nachteile bescheid. Ich kenne die Vielfalt der Struts Tags die in den JSP’s gebraucht werden und lernte den Code so redundanzfrei und unabhängig wie möglich zu programmieren. Des Weiteren habe ich mich mit dem Tiles Framework beschäftigt, mit dem wir ursprünglich auch noch die Redundanz der JSP’s vernichten wollten. Durch das anspruchsvolle Software Projekt habe ich einen großen Überblick der Standardtechniken bekommen und das Interesse für die Java Enterprise Welt wurde in mir geweckt.

An dieser Stelle möchte ich mich noch für die gute Zusammenarbeit im Team bedanken und an alle ein großes Lob aussprechen. Wir haben alle eine gute Arbeit geleistet, was nicht zuletzt an der sehr guten Projektleitung von Manuel gelegen hat.

3.3. Isabel Schäfer Batista

Nach einer sehr umfangreichen Analysephase im ersten Semester des Software-Projektes schloss sich direkt ein äußerst kurzes und an Arbeitsaufwand umfangreiches zweites Semester an. Schon zu Beginn des zweiten Semesters war absehbar, dass viele Analyse-Dokumente nicht oder nur kaum für die Implementierungsphase genutzt werden würden und dass die zeitliche Aufteilung der Aufgaben Schwierigkeiten bereiten würde. Trotz dieser anfänglichen Schwierigkeiten und der allgemeinen Einstellung, dass die Analysephase zu ausgedehnt gewesen sein könnte, komme ich jetzt zu dem Schluss, dass wir nur durch diese ausgedehnt Analysephase ein sicheres Fundament für die Implementierungsphase errichtet haben. Wir konnten uns insbesondere an das

erstellte Pflichtenheft aus dem ersten Semester bis auf geringfügige Abweichungen halten, ohne dieses nocheinmal zu Rate gezogen zu haben. Daran ist klar geworden, dass wir „unser“ System so intensiv bearbeitet hatten, dass wir es in- und auswendig kannten. Alle Schwierigkeiten während der Implementierungsphase ergaben sich meist aus der Anwendung neuer Techniken, der Schichtentrennung und Session-/Transaktion-Handhabung, und niemals aus einem eventuell schlechten Verständnis für das eigene Konzept des Systems.

Die Einarbeitung in das O/R-Mapping-Tool Hibernate war eine äußerst knifflige, aber auch wissenschaffende Aufgabe. Das Wissen über Arbeiten mit Datenbanken wurde dadurch wesentlich transparenter. Besonders die etwas spezielleren Kenntnisse über Hibernate, zum Beispiel über die Vorgehensweise bei Relationen zwischen Tabellen oder die Benutzung der HQL waren schöne Herausforderungen. Sicher werde ich mich in Zukunft noch weiter mit dem Tool auseinandersetzen, um noch weitere Benutzungsvarianten kennenzulernen und meinen Wissensschatz in dieser Richtung zu erweitern.

Zum Schluss möchte ich noch sagen, dass die Zusammenarbeit mit den Teamkollegen Bähnisch, Herrmann, Fohgrub und Barmeyer überaus angenehm war. Wenn so der Alltag im Informatiker-Berufsleben aussieht, dann bin ich zuversichtlich, dass ich mich darin wohlfühlen werde. Unser Team „Bugtracker“ war eine intelektuelle und persönliche Bereicherung. Während meines gesamten Studiums habe ich bei weitem nicht soviel dazugelernt, wie in diesem Jahr des Software-Projektes. Sei es die Entwicklung redundanzfreien und gut lesbaren Codes oder die gemeinsame Zusammenarbeit, welche durch ausserordentliche Kommunikation zwischen den Kollegen zu den unmöglichsten Zeiten und vor allem einer gehörigen Portion Tatendrang und Motivation bestand. Es hat alles gestimmt. Immer wieder gerne.

3.4. Alexander Fohgrub

Aus meiner Sicht ist ein sehr ehrgeiziges Projekt erfolgreich zum Abschluss gekommen. Die Entscheidung einen Projektleiter zu wählen, war aus meiner Sicht eine der wichtigsten Gruppenentscheidungen. Durch diesen Umstand hatte immer einer aus dem Team die Fäden in der Hand und jeder konnte sich auf sein Spezialgebiet konzentrieren.
Des Weiteren war die Trennung der Schichten Datenbank, Logik und GUI nicht nur auf dem Papier, sondern auch in der Aufgabenverteilung ein großer Schritt in Richtung Fertigstellung.
Aus diesem Grunde bewerte ich die Teamarbeit als ganz hervorragend. Jeder hat seine Schicht im besten Maße vertreten und mit größtem Können fertiggestellt.

Für mich ziehe ich ganz besondere Erfahrungen aus dem Projekt. Aufgrund des äußerst schwierigen Starts im zweiten Projektsemester hat es sehr lange gedauert, meinen Platz in diesem Projekt zu finden.
Zu meinem Part wurde dann nach einigem Hin und Her die Zuarbeit in Datenbankschicht. MulTEx, DBUnit und die Transaktionen in Hibernate wurden mir anvertraut.
MulTEx das Framework zur Exceptionbehandlung und DBUnit zum Testen waren zwei sehr hilfreiche Frameworks für unser Projekt.
Mit MulTEx erscheint es einem Exceptions von einer Schicht in die nächste , in einer Reihe gereiht, weiterzureichen, als unkompliziert.
DBUnit habe ich verwendet, um die Methoden, mit denen wir die Datenbank verwaltet haben, zu testen. Da DBUnit auf JUnit aufbaut, hat man bei entsprechenden Erfahrungen, keine Schwierigkeiten sich hier einzuarbeiten.
Die Arbeit mit den Transaktionen in Hibernate machten einen Blick aus der DB-Schicht notwendig. Hier konnte ich ganz besondere Erfahrungen mit der Schichtentrennung machen.

3.5. Thorsten Herrmann

In den letzten drei Monaten habe ich sehr viel dazu gelernt, nicht nur im Bezug auf das Programmieren und umsetzen von Projekten, sondern auch im Umgang mit Menschen. Ein fünf Mann-Team immer unter einen Hut zu bekommen verlangt einen hohen grad an Kompromissbereitschaft und manchmal halt auch „Feuer“ vom Projektmanager. Ich glaube ohne Hr. Bähnisch und seinem positiven Auftreten wäre dieses Projekt so nicht möglich gewesen.

Im Projekt „Bugtracker“ war ich bzw. bin ich als Programmierer tätig, dem die Verantwortung über die GUI-Schicht oblag bzw. obliegt. Da ich kein Vorwissen im Bezug auf die angewandten Techniken in der GUI-Schicht hatte (Struts/JSP/Validation Framework und ExceptionHandling), war es teilweise sehr mühsam, aber auch interessant sich in diese einzuarbeiten. Da es sehr häufig zu Konflikten zwischen den Dokumentationen einer Technik und ihrer beschrieben Funktionalität und der in der Technik wirklich bereit gestellten Funktionalität

kam, war das einarbeiten nur durch Beschaffung von meist sehr teuren und aktuellen Büchern möglich. Diesen Fakt würde ich persönlich nicht als negativ betrachten, jedoch als faden Beigeschmack zu einer sonst guten Idee.

Wenn ich nun auf die letzten drei Monate zurückblicke, finde ich es Schade, dass wir in dieser form wahrscheinlich nie wieder an einem Projekt zusammen arbeiten werden. Denn die gemeinsame Zeit mit meinen Kollegen war doch eine sehr angenehme Zeit, da sich jeder eigentlich auf den anderen verlassen konnte und es keine Streitereien um Probleme und Schuldzuweisungen (es gab einfach keine) gab.

Zur Lehrveranstaltung SWP1/SWP2 kann ich sagen das sie mir durchweg durch gut gefallen hat, vor allen dingen hat mir das Systematische vorgehen in den Übungen gefallen, welches leider an der TFH nicht sehr häufig anzutreffen ist. Die Vorlesungen waren meist ein misch masch aus sehr interessanten Themen und dann wieder für mich persönlich langweiligen Themen.

Insgesamt gesehen waren die Kurse SWP1/SWP2 sehr lehrreiche und Informative Kurse, welche es verdient haben im Lehrplan zustehen und wo ich hoffe, das die Art und Weise dieser Kurse auf andere Kurse ausgeweitet wird z.B. swe und sa.

Als letzen Satz möchte ich mich bei meinen Kollegen und ihnen als Betreuer des Projektes bedanken.

Hochachtungsvoll

Thorsten Herrmann