Team Fahrrad-Schotte **Berlin, SS/03**
Kai-Michael Bodach
Sebastian Haberey
Tatjana Manych
Matthias Nietz

Abschlußbewertung für Softwareprojekt I/ II

Projekt: Fahrrad-Schotte, ein Preisvergleichsystem

Fahrrad-Schotte ist ein Internetportal, das dem Besucher Preisvergleiche von Fahrrädern und Ersatzteilen aller Berliner Fahrradhändler bietet. Die Preisübersicht ist nach Produktkategorien geordnet.
Händler können ihre aktuellen Preislisten zur Verfügung stellen und Kunden diese dann abrufen.

Analysephase: Softwareprojekt I, bei Herrn Faustmann
Entwurfs-/
Implementierung: Softwareprojekt II, bei Prof. Knabe

Bewertung:

SWPI und SWPII sind unverzichtbare Fächer im Bereich Softwareentwicklung der Medieninformatiker. Das ist das erste Fazit, das wir nach dem Softwareprojekt ziehen können. Wir alle haben hier viele Erfahrungen und Rückschlüsse ziehen können, was die Komplexität und die Schwierigkeiten bei der Entwicklung und Implementierung eines Softwarprojektes angeht. Es bot uns die Möglichkeit das Gelernte aus den vorherigen Fächern (wie Software-Engineering, Systemanalyse, …) einmal selbst umzusetzen und anzuwenden. Es wäre wünschenswert, wenn es mehr solche Projekte in unserem Studium gegeben hätte, denn das sind Erfahrungen, die uns in unserem weiteren Werdegang von großem Nutzen sein werden.

Doch es gibt natührlich auch Dinge die man noch verbessern kann. Als erstes, ist das die Abstimmung zwischen den beiden Fächern SWPI und SWPII. Im Normalfall hat man diese Fächer in den Semestern 6 und 7 hintereinander und bei dem gleichen Lehrbeauftragten oder Professor. Bei unserem Team war das nicht der Fall. Das führte zu großen Schwierigkeiten, da die Anforderungen und Kriterien der einzelnen Lehrbeauftragten und Professoren an das gewählte Projekt sehr unterschiedlich waren.

Das führte dazu, das wir am Anfang der Entwurfs- und Implementierungsphase mit komplett neuen Techniken und Konzepten konfrontiert wurden. Beispielsweise mußten wir von einer PHP- und Perl-Lösung auf eine JSP- und Java-Lösung wechseln. Damit wurde unsere ganze Vorbereitung, welche wir im dazwischenliegenden Semester und in den Semesterferien durchgeführt hatten, auf einem Schlag hinfällig.
Wir möchten hier ein Beispiel geben in was für Schwierigkeiten uns das Ganze gebracht hat. Wir hatten am Anfang des SWPII Kurses ca. 2 Wochen Zeit uns in neue Techniken, wie JSP, Struts und Castor (OR-Mapping) einzuarbeiten. Da die Zeit nicht reichte ein umfangreichen Test durchzuführen, ob die Techniken den Anforderungen unseres Projektes genügen, hatten wir uns auf die Empfehlung unseres Professors verlassen müssen. Es stellte sich im Verlauf des Projektes (leider erst im letzten Drittel) heraus, das Castor noch so in den Kinderschuhen zu stecken scheint, daß es nicht möglich war einige Teile unseres Konzeptes umzusetzen. Beispielsweise kann Castor als OR-Mapping Tool nicht einmal das Kompositum Muster ohne Probleme umsetzen oder auch Polymorphie ist ein Punkt den Castor nicht beherrscht. Im Großen und Ganzen bleibt uns hier nur zu sagen, das man lieber auf Produkte setzen sollte die schon mindestens über das 1.0er Stadium hinaus sind.
Das ganze hat für uns natührlich auch positive Auswirkungen gehabt. Wir hatten so einen sehr guten Vergleich zwischen der PHP & PERL und der JSP & JAVA Lösung unseres Projektes. So ist es für uns viel leichter geworden Entscheidungen über die Anwendungen von bestimmten Technologien bei bestimmten Anforderungen von Projekten zu treffen.
Zweitens war das ganze Projekt ein große Bewährungsprobe für unser Team, gerade weil wir im zweiten Teil des Projektes nur noch zu Dritt waren, da ein Teammitglied sein „Final Year" in England absovierte. Hier kann man sagen, das bei uns immer ein großer Zusammenhalt vorhanden war und wir uns gegenseitig aufeinander verlassen konnten. Zusätzlich muß bei solch einem Projekt auch die Kommunikation funktionieren, bei uns reichte ein normaler E-Mail Verkehr und eine überdurchschnittliche telefonische Kommunikation aus. Aus unserer Sicht ist es von großem Vorteil, schon mit den Teammitgliedern vor dem Softwareprojekt Erfahrungen in anderen Projekten oder Fächern gesammelt zu haben.

In Bezug auf die Struktur des Faches SWPI/II hat sich das Konzept der Meilensteine gut bewährt. Denn so weiß man immer wo man steht und wo der nächste Zielpunkt liegt. Es ist außerdem positiv die Meilensteine so zu definieren, das es zu einem möglichst frühen Zeitpunkt an die Implementierung geht. Auch hier zeigt sich einmal mehr die „Pareto-Regel". 80% der Implementierung haben wir in 20% der Zeit geschafft, aber für die schwierigen Punkte (20%) der Implementierung haben wir 80% der Zeit benötigt. Darum ist es wichtig möglichst früh auf die Schwierigkeiten zu stoßen und diese auch möglichst früh zu bearbeiten. Es ist durchaus von Vorteil in der Analysephase lieber weniger Muß-Kriterien zu definieren. Auch hier ist eine gute Abstimmung zwischen den Lehrbeauftragten und Professoren (SWPI/II) nötig.
Als negativ bewerten wir, daß bei den Abgaben der Meilensteine mehr auf die Einzelheiten, als auf die Gesamtheit des Systems geachtet wurde. Dadurch kann es kommen, daß man Architekturfehler erst zu spät entdeckt und dies zu Folgefehlern führen kann. Eine Ursache dafür ist bestimmt die begrentzte Zeit bei den Abgaben.
Ebenfalls negativ ist, daß die JUnit-Testtreiber und die Dokumentation erst zu spät gefordert wurden. Vergleicht man das mit der Gewichtung in der Abgabe, hätte dies auch bei den Meilensteinen deutlicher erkenntlich sein müssen. Wir haben die Erfahrung gemacht, das die JUnit-Tests ein unverzichtbarer Teil eines Java-Softwareprojektes sind. Denn jeder muß sich darauf verlassen können, das nur „gut" getesteter Code in die Software aufgenommen wird. Ansonsten kann man viel Zeit bei der Behebung von eigentlich einfachen Fehlern verlieren.
Als Versions Managment System haben wir das CVS aus Eclipse genutzt. Nach einer gewissen Einarbeitungszeit hat es uns gute Dienste geleistet und ist auch zu einer unverzichtbaren Technik in unserem Projekt geworden. Als IDE ist Eclipse aus unserer Sicht nur zu empfehlen.
Positiv empfinden wir auch die Erstellung einer solchen Schlußbewertung, so können andere Projekte aus unseren Fehlern lernen und auch ein schnelleren und besseren Eindruck über die eingesetzten Techniken und die eingesetzte Fremd-Software erlangen. Hierzu wäre es nötig die Studenten am Anfang des Semesters auf die Schlußbewertung der anderen Projekte explizit hinzuweisen.
Zusammenfassend ist es jedem Medieninformatiker (Bereich:Softwareentwicklung) zu empfehlen diesen Kurs ernst zu nehmen und diesen mit viel Fleiß und Spaß am Entwickeln neuer Software zu absolvieren. Wir wünschen allen kommenden Teams viel Erfolg bei ihren Projekten.

Team Fahrrad-Schotte

Bewertung der eingesetzten Techiken und Fremd-Software:

Servlets/JSP

+gute Dokumentation
+sehr gut erweiterbar (über TLDs, Struts, Tiles)
+mächtiges Instrument für dynamische Webanwendungen (bietet kompletten Funktionsumfang von Java)
- kann zu einer Mischung von Präsentation und Anwendungskern einer Applikation führen (bei Modell 1 Architektur) mit allen negativen Konsequenzen

Struts

+gutes Framework zur Umsetzung der Modell 2 Architektur (MVC) in der Ui Schicht, dadurch ist die Trennung von deklarativen (HTML, TLD, etc.) und objektorientierten Sprache (Java) möglich und auch die Trennung nach Kompetenzen (HTML Design, Logik Programmierung)
+gut dokumentiert
+ab 1.1 Tiles Ersatz für HTML-Frames und einfache JSP Includes
+umfangreiche Tag-Library (sehr gut erweiterbar)
+gute Kapselung der dynamischen Werte von HTML (Form Parameter) durch Java Beans
+kompatibel zu J2EE
+/- junge Technologie mit viel Potential aber auch stetige Änderungen

Castor

+einfache Objekt-/Klassenstrukturen lassen sich leicht auf relationale Datenbanken abbilden
+/-Lazy-Loading Mechanismus (aber funktioniert noch nicht in allen Konzepten)
+lebhafte Community
- nicht gut dokumentiert
- komplizierte Strukturen (z.B. Kompositum: rekursive O-/K-Struktur) können nicht intuitiv abgebildet werden
- steckt noch in den Kinderschuhen, professioneller Einsatz erst nach Release 1.0 möglich (sonst nur mit Mut zu Überraschungen)
~Alternative ist z.B. Hibernate

Multex

+gute Fehlerbehandlung für mehrschichtige Anwendungen, erleichtert auch das Entwickeln

Eclipse

+viele PlugIns
+gutes CVS
+sehr flexibel und individuell Konfigurierbar