Softwareprojekt

TI 5

WS2008/2009

Erfahrungsbericht

Gruppe 1

Inhaltsverzeichnis

Erfahrungen mit dem Projekt....2
Erfahrungen mit unserem gewählten Prozessmodell....3
Erfahrungen mit wxWidgets....4

Knappe sieben Wochen sind nun vorbei und das Projekt neigt sich dem Ende zu. Wir alle sind uns sicher, sehr viel von dem Projekt mitgenommen zu haben, viel an Erfahrung dazu gewonnen zu haben. Aber es gab auch Punkte, die uns aufgefallen sind, die nicht so toll gelaufen sind.

Auch unsere eigenen Vorgaben, das Produkt nur mit OpenSource Komponenten zu entwickeln, hat uns um viele Erfahrungen reicher gemacht.

Im folgenden wollen wir auf die wichtigsten Punkte etwas genauer anschauen.

Erfahrungen mit dem Projekt

Das Projekt begann für uns gleich am ersten Tag mit einer sehr überraschenden und auch tiefgreifenden Veränderung: wir wurden darüber informiert, dass unser fünfter Mann, der sich eigentlich fürs Softwaremodell verantwortlich erklärt hatte, die Entscheidung getroffen hatte, nicht an unserem Projekt teilzunehmen. Somit mussten wir gleich zu Beginn feststellen, wie wichtig es ist, sich auf seine Projektmitglieder verlassen zu können, bzw. was es bedeutet, wenn genau dies nicht der Fall ist.

Unser nächstes Problem war, dass wir keinen eigenen Raum zum Arbeiten hatten. Zunächst versuchten wir es mit den Tischen auf den Gängen, mussten jedoch bald feststellen, dass dies keine dauerhafte Lösung sein konnte. Der später gefundene Raum löste das Problem auch nicht endgültig, da auch dieser mehrmals pro Woche durch andere belegt war. Wir denken, dass dieser Umstand für folgende Gruppen unbedingt beseitigt werden sollte. Es sollte pro Gruppe ein Arbeitsraum zur Verfügung stehen, um ein ungestörtes Arbeiten zu ermöglichen.

Ein weiterer, sehr großer Störungsfaktor war die parallel laufende Vorlesung in Projektmanagement, da man hierdurch immer wieder aus seiner Arbeit gezogen wurde. Besonders störend waren die in dieser Vorlesung verlangten Präsentationen und Ausarbeitungen, die gerade in der Endphase des Projekts sehr viel Zeit raubten.

Allgemein hatten wir während der sieben Wochen sehr stark mit unserer kleinen Teamgröße und den immensen Anforderungen zu kämpfen. An Freitzeit war kaum noch zu denken. Jede Woche endete bei uns mit mindestens 60 Stunden reine Arbeitszeit. Hinzu kommen noch tägliche Pendelzeiten.
Dies kam zum Einen zwar dadurch, dass wir während der vorlesungsfreien Zeit Arbeiten mussten und dadurch keine Vorbereitungen treffen konnten, zum Anderen aber auch von der komplexen Materie, die solch ein Projekt mit sich bringt. Für uns war wirklich alles Neuland. Zuvor hatte noch niemand von uns mit OpenGL oder wxWidgets gearbeitet, geschweige denn ein Prozess miterlebt. All diese Gegegebenheiten forderten einfach zu viel Einarbeitungszeit.

Ein letzter kleiner Wunsch, der während der Arbeit mit Subversion aufgetaucht ist wäre gewesen, dass wir eine kleine Einführung in die Besonderheiten eines Repositorys, wie z.B. Branch oder Rollback, bekommen hätten.

Abschließend gilt es trotzdem noch sagen, dass wir durch dieses Projekt riesige Erfahrungen gemacht haben, die uns auf jeden Fall noch zu Gute kommen werden. Wir haben auf jeden Fall vieles gelernt, wenn auch in einigen Fällen die Tatsache, dass unser Weg der Falsche war.

Erfahrungen mit unserem gewählten Prozessmodell

Das Projekt startete für uns alle am ersten Vorlesungstag ohne Vorbereitung, da wir leider alle die vorlesungsfreie Zeit nutzen mussten um Geld zu verdienen. Das bedeutete aber für uns, dass wir alle ohne Vorkenntnisse ins Projekt starteten. Hier konnten wir feststellen, dass sich unser gewähltes Vorgehensmodell – Extreme Programming (XP) – bestens dafür eignet um technisches Neuland zu betreten. Wir konnten ohne Vorplanung starten und uns in die neue Materie einarbeiten, indem wir einfach verschiedenste Sachen ausprobieren, und unsere Ideen auf Machbarkeit prüfen konnten. So konnten wir schon in den ersten beiden Wochen betrachtliche Ergebnisse erziehlen.

Im Nachhinein müssen wir feststellen, dass wir zu diesem Zeitpunkt dann einen sehr markanten und entscheidenden Fehler gemacht haben. Wir haben unsere Ergebnisse zusammengeführt, uns dann aber direkt an die nächsten Features gemacht. Genau an dieser Stelle wäre jedoch das, von XP auch geforderte, Refactoring nötig gewesen um unser Design zu verbessern und zukunftsicher zu machen.

Vielleicht waren wir an dieser Stelle auch von dem Gedanken geleitet, dass in XP keine Architekturplanung nötig ist. Dies macht jedoch Refactoring umso wichtiger, was wir an dieser Stelle total vernachlässigt haben. Das lag vielleicht auch am Zeitdruck (siehe Erfahrungen zum Projekt), da wir einfach keine Zeit für “unnötige” Dinge hatten.

Auch wenn XP, wie oben schon erwähnt, keine Architekturplanung vorsieht sind wir dennoch der Meinung, dass eine Planung der Grundarchitektur bzw. zentralen Datenstruktur wichtig ist. Wir mussten feststellen, dass ansonsten Änderungen an zentralen Punkten Änderungen in allen Bereichen mit sich bringt, was sehr lange Überarbeitungszeiten mit sich zieht.

Ein weiterer Punkt, dessen Wichtigkeit wir erst im Laufe des Projekts erkannt haben ist die Kommunikation. Es wäre wirklich sehr wichtig gewesen, gleich zu Beginn die Erwartungen aller Beteiligter auf eine gemeinsame Linie zu bekommen. Auch während des Projekts hätten mehr Statusmeetings stattfinden sollen, um ein gemeinsames Vorgehen zu planen.

Sehr wichtig in diesem Zusammenhang ist die permanente Codeintegration, damit alle Beteiligten immer mit dem aktuellen Code arbeiten und Merge-Vorgänge möglichst einfach bleiben und nicht, wie es uns passiert ist ein eigener Branch entsteht.

Die kollektive Codeownership erwies sich für uns als sehr nützlich, brachte jedoch Anfangs auch Unstimmigkeiten ins Team. Durch diese Möglichkeit war es zunächt einmal möglich, Änderungen am Code direkt durchzuführen, ohne den entsprechenden Codeowner aufzusuchen, was die Geschwindigkeit erheblich erhöht. Lediglich am Anfang traten hierbei bei uns Probleme auf, da man sich zunächst einmal daran gewöhnen musste, dass der selbst geschriebene Code von anderen verändert wird.

Große Schwierigkeiten hatten wir auch beim Erstellen der UserStories. Dies lag zum einen am nicht vorhandenen Kunden, aber auch an der Problematik, die formalen und inhaltlichen Anforderungen zu erfüllen und den notwendigen Abstraktionsgrad zu wahren. Wir sind uns auch nicht sicher, ob UserStories für ein Projekt wie unseres, mit sehr sehr vielen aber meist kleinen Funktionen, wirklich geeinget sind. Die Beschreibung einer Benutzeraktion ist hier kaum noch möglich.

Pairprogramming haben wir auch getestet, mussten jedoch sehr schnell feststellen, dass diese Art der Programmierung für unser kleines Team auf keinen Fall in Frage kommt. Allgemein sind wir der Meinung, dass sich Pairprogramming für kritische Abschnitte im Programmcode bestens eignet, für einfache, sich wiederholende Tätigkeiten jedoch eher Zeitverschwendung mit sich bringt.

Auch UnitTests blieben bei der kurzen Dauer und den hohen Anforderungen leider auf der Strecke. Wir haben es nur geschafft ein paar automatisierte Tests durchzuführen, die sich doch als sehr nützlich erwiesen haben.

Am allerwichtigsten ist bei einem Prozess wie XP jedoch die Kommunikation. Durch die Tatsache, dass es kein feststehendes Design gibt müssen die einzelnen Schritte unbedingt aufeinander abgestimmt werden und vorallem Änderungen mit den anderen abgesprochen werden.

Erfahrungen mit wxWidgets

Mit wxWidgets betraten wir alle vollkommenes Neuland. Bisher hatte noch niemand von uns wirklich Kontakt mit diesem Framework und genau diese Tatsache bekamen wir während des Projekts regelmäßig zu spüren. Im folgenden wollen wir etwas genauer auf unsere Erfahrungen mit WX eingehen.

Auf den ersten Blick sah alles super einfach und toll aus. Ein paar Klicks und wir hatten ein Frame, das sowohl auf Windows als auch unter Linux lief.
Doch genau mit der GUI kamen schon die ersten Probleme: für WX gibt es keinen vernünftigen RAD-Tool. Das GUI-Design wurde uns also stark erschwert. Teilweise ging das GUI-Design von Hand sogar einfacher.

Das nächste große Problem von WX sind Events. Hier gibt es zwei Typen von Events. Der erste Typ ist nur in der aktuellen und in abgeleiteten Klassen gültig. Dazu gehören zum Beispiel Maus- oder Keyevents. Der andere Typ von Events steigt immer zu seinem Vater auf, jedoch nur, wenn diese Klasse von wxWindow abgeleitet ist. Als letzte Station landet das Event in der App-Klasse. Leider ist das Verhalten von Events in WX auch nur sehr schlecht dokumentiert, wodurch wir auf diese Eigenschaften erst sehr spät gestoßen sind.
Durch diese Event-Struktur wird leider auch das Softwaredesign sehr stark eingeschränkt. Es wird schwer möglich zusammenhängende Klassen als Module zu kapseln.
Es gibt auch keine Möglichkeit, dass sich Klassen für bestimmte Events einschreiben können.

Wir mussten auch feststellen, dass plattformunabhängig leider nicht plattformunabhängig bedeutet. Immer wieder hatten wir Versionen, die unter Windows komplett lauffähig waren, unter Linux hingegen schon beim Start Speicherzugriffsfehler produzierten. Teilweise war OpenGL daran schuld, teilweise wxWidgets selbst.

WX hat aber auch seine guten Seiten. So liefert es zum Beispiel eine sehr ausführliche Bibliothek mit Funktionen wie ZIP oder XML mit. Auch die GUI-Funktionalität ist nicht zu verkennen.