Freie Universität Berlin

Fachbereich Informatik/Mathematik

Replikation eines Kontrollierten Experiments zu Entwurfsmustern

Bachelorarbeit

im Studiengang Informatik

eingereicht von: Martin Liesenberg martin.liesenberg@googlemail.com

eingereicht am: 25. November 2010

Betreuer: Herr Prof. Dr. Lutz Prechelt

Ich versichere eidesstattlich, die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen benutzt zu haben. Alle wörtlichen und sinngemäßen Entlehnungen sind unter genauer Angabe der Quelle kenntlich gemacht.

Glienicke, den 25.11.2010

Martin Liesenberg

Contents

List of Figures

List of Tables

1 Einführung

Entwurfsmuster erfreuen sich seit Mitte der 90er Jahre des letzten Jahrhunderts großer Beliebtheit in der Softwareentwicklung. Dabei ist nicht eindeutig geklärt, ob und wenn ja wie stark, sie die Güte der entwickelten Programme positiv beeinflussen. Das gilt besonders für kleinere Programme, die sich leicht ohne ein Entwurfsmuster programmieren ließen.

Aus diesem Grund führten Prechelt et al.[1] 1997 ein kontrolliertes Experiment durch, um die Auswirkungen von Entwurfsmustern auf die Erweiterbarkeit und Wartbarkeit von kleinen Programmen empirisch zu überprüfen. Die Ergebnisse werden in Abschnitt 3.1.1 kurz dargestellt.

Die Notwendigkeit der Durchführung Experimente und Replikationen in der Softwaretechnik wurde wiederholt diskutiert und begründet. ([4], [9] und Kapitel 14 aus [3]) Allerdings gibt es bis heute nur wenige Replikationen des o.a. Experiments. Die Schwierigkeiten bei der Durchführung eines Experiments im Allgemeinen und einer Replikation im Speziellen liegen vor allem darin genügend Teilnehmer für aussagekräftige Resultate zu finden und die Experimentumgebung aufzusetzen.

Aus diesem Grund kam die Idee einer sogenannten Joint Replication, einer verbundenen Replikation, auf. Dabei tragen mehrere Replikationen zu einem Pool von Ergebnissen bei. Die Infrastruktur für die Durchführung dieser Replikationen wird an einer zentralen Stelle bereitgestellt. Dadurch wird den Experimentatoren ein großer Teil der Arbeit abgenommen und damit die Hemmschwelle für die Teilnahme möglichst gering gehalten. Idealerweise muss der Experimentator nur eine Gruppe von Teilnehmern finden, ihnen Anweisungen zur Durchführung geben und kann danach seine separat auswerten. Ziel der Joint – Replication ist es u.a. die Ergebnisse einer Anzahl gleichartiger Replikationen zusammen auszuwerten und damit Populationsgrößen zu erreichen, die sonst nur schwer zu realisieren sind.

Im Rahmen der vorliegenden Arbeit wurde die Infrastruktur zur Durchführung einer Joint – Replikationen entwickelt und eine erste Replikation auf der Basis dieser Infrastruktur mit Studenten aus dem Softwaretechnikprojekt des Fachbereichs Informatik der Freien Universität Berlin durchgeführt. Der erste Teil der Arbeit befasst sich mit der Entwicklung der Infrastruktur - ein Internetportal, der zweite Teil befasst sich mit der Durchführung der Replikation und der Analyse der Ergebnisse.

2 Internetportal

In diesem Abschnitt gebe ich einen kurzen Überblick zur Entwicklung des Internetportals. Der Fokus liegt auf den Anforderungen und deren Analyse. Außerdem gehe ich kurz auf nachträgliche Änderungen ein.

2.1 Anforderungsbestimmung

Die Anforderungen an das Portal wurden basierend auf dem Originalexperiment und denjenigen an eine Joint – Replication erhoben. Eine ausführliche Beschreibung der Anforderungen durch Anwendungsfälle kann unter [6] gefunden werden, die Wichtigsten sind hier kurz zusammengefasst.

- Die Teilnehmer werden vom Portal durch das Experiment geführt. Das beinhaltet:
  - Vor Beginn der Teilnahme findet eine Authentifikation statt.
  - Ein vorgegebener Ablauf für jeden Teilnehmer, damit keinerlei Verwirrung entstehen kann
  - Alle Informationen für die Bearbeitung der Aufgaben und die Durchführung werden den Teilnehmern vom Portal bereitgestellt
  - Das Ziel einer möglichst geringen Abbrecherquote unter den Teilnehmern
  - Die notwendigen Daten zum Hintergrund der Teilnehmer werden durch das Portal erhoben, gespeichert und verfügbar gemacht.
- Antworten der bearbeiteten Aufgaben werden vom Portal gespeichert und verfügbar gemacht. Das umfasst:
  - Bearbeitungszeit für jeden Arbeitsschritt
  - Bearbeitete Programmversionen
  - Antworten zu den Fragebögen nach jedem Programm
- Jeder Experimentator kann nur die für seine Experimente gespeicherten Daten herunterladen.
- Das Portal leistet einem Experimentator Hilfestellung in folgenden Bereichen:
  - Anlegen von Replikationen
  - Aufteilung der Teilnehmer in Gruppen
  - Organisation des Experimentablaufs, d.h. Aufgaben und Reihenfolge der Bearbeitung müssen nicht vom Experimentator geklärt werden
  - Herunterladen der Ergebnisse
  - Der Experimentator erhält außerdem eine Übersicht zu den von ihm betreuten Experimenten
- große Stabilität vor allem im Teilnehmerteil, um die Mortalität so gering wie möglich zu halten

2.2 Entwurfsentscheidungen

Die Analyse der Anforderungen brachte eine Reihe von Entwurfsentscheidungen hervor von denen ich auf einige hier näher eingehen möchte.

Um Gruppenzugehörigkeit und Authentifizierung zu gewährleisten wird jedem der Teilnehmer der Replikationen eine 5 - stellige Zahl zugewiesen mit der sie sich im Internetportal anmelden. Die Zahlen werden vom Portal zufällig generiert. Falscheingaben werden nicht akzeptiert. Der große Zahlenbereich macht das zufällige Eingeben einer vorhandenen Zahl hinreichend unwahrscheinlich.

Die Gruppeneinteilung wird implizit durch die den modulo 4 - Wert der Zahl vorgenommen. Dadurch kann es unter Umständen bezüglich der Fähigkeiten der Teilnehmer zu unbalancierten Gruppen kommen.

Damit den Teilnehmern möglichst viele Informationen kompakt zur Verfügung gestellt werden können, gibt es für jedes Programm einen Block im Portal. Dieser Block setzt sich zusammen aus:

- einer Einführungsseite auf der Informationen zum Programm gewährt werden und der zu bearbeitenden Quelltext heruntergeladen werden muss
- einer Seite mit der Aufgabenstellung
- einer Seite zum Hochladen des bearbeiteten Quelltexts und
- einem Fragebogen zur Selbsteinschätzung der erbrachten Leistung

Diese Modularisierung erlaubt eine leichte Erweiterbarkeit des Portals und während der Durchführung entsteht Kontinuität durch sich wiederholende Strukturen.

Experimentatoren melden sich, nach einer Registrierung, beim Portal an und haben dann Zugriff auf die komplette Funktionalität. Durch die Anmeldung wird gewährleistet, dass jeder Experimentator nur die von ihm angelegten Replikationen sieht und manipulieren kann.

Unabhängig vom Portal wurde außerdem beschlossen mehrere mögliche Modi für eine Replikation anzubieten. Das Originalexperiment arbeitete mit vier Programmen und erstreckte sich über zwei Tage. Das erschien

2.3 Umsetzung

Bei der Implementierung bediente ich mich im wesentlichen zweier Werkzeuge:

- Apache Tapestry
- Hibernate

Tapestry ist ein Open – Source – Framework für die Programmierung von Webanwendungen mit JAVA. Hauptbestandteile einer Tapestry – Anwendung sind JAVA - Klassen und korrespondierende XML – Vorlagen. In den JAVA – Klassen wird die Applikationslogik bereitgestellt, die XML – Vorlagen bestimmen das Aussehen der entsprechenden Seiten. [7] Hibernate ist ein Open – Source – ORM – und Persistenz – Framework, übernimmt also die Kommunikation mit der Datenbank der Applikation. [8]

Im Hintergrund der Anwendung arbeitet eine Datenbank, die die Antworten der Teilnehmer zu den verschiedenen Fragebögen, die Zeit, die jeder Teilnehmer auf jeder Seite verbringt, die bearbeiteten Programme und Antworten zu den Aufgaben sowie die Daten der Experimentatoren. Es handelt sich hierbei um eine MySQL – Datenbank. Abbildung 1 zeigt ein Schema der Datenbank.

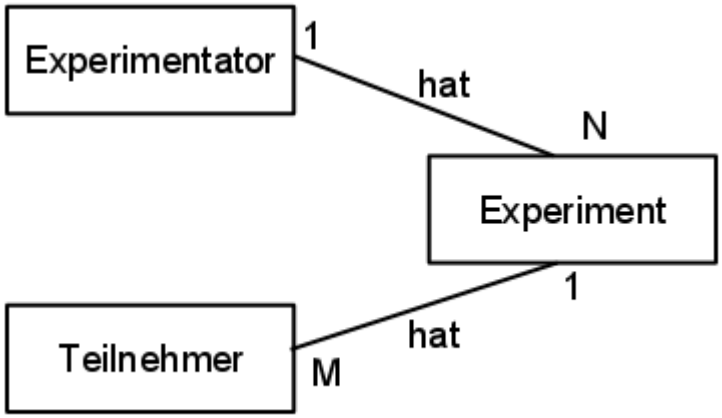

Figure 1: Schema der Datenbank

2.4 Feedback und nachträgliche Erweiterungen

Während der Entwicklungsphase und nach Fertigstellung des Portals erhielt ich ausführliches Feedback zur Stabilität und Funktionalität des Portals. Die wichtigsten Punkte und ihre Auswirkungen sollen hier noch einmal genannt werden.

- **Hibernate – Timeout:** Durch eine unerwarteten Absturz während des Versuchs einen neuen Experimentator zu registrieren, stellte sich heraus, dass das in Hibernate integrierte Werkzeug zur Verwaltung der Verbindungen zur Datenbank nicht automatisch eine neue Verbindung aufbaut nachdem es zu einem Timeout gekommen ist. Das führt dazu, dass Anfragen an die Datenbank nicht mehr bearbeitet werden können.

Deshalb überschrieb ich die Verbindungseinstellungen dahingehend das Programm c3p0 zu benutzen, ein Hibernate Werkezug, dass automatisch eine neue Verbindung aufbaut. Die E – Mail – Korrespondenz befindet sich im Anhang unter Hibernate – Timeout.

- **Unterstützung für mehrere Sprachen:** Ursprünglich war das Portal nur für Replikationen mit JAVA – Programmen konzipiert. Da von verschiedenen Seiten Interesse angemeldet wurde auch Replikationen mit C++ – Programmen und C# – Programmen durchzuführen, entschieden wir uns für die Aufnahme dieser Funktionalität in das Portal. Damit erweitert sich der Kreis der potentiellen Teilnehmer für Replikationen, was ganz im Sinne der ursprünglichen Idee ist.

- **Robustheit:** Der Fokus der Tests im Verlauf der Entwicklung lag vor allem auf der Robustheit der Teile des Portals, die durch Replikationsteilnehmer benutzt werden. Der Grund dafür liegt in der Bedeutung dieses Bereichs für das gesamte Vorhaben. Je größer die Anzahl der Fehler, mit denen ein Teilnehmer konfrontiert wird, desto höher die Wahrscheinlichkeit, dass er die Replikation vorzeitig abbricht.

3 Replikation

Dieser Abschnitt beschäftigt sich mir der Durchführung und Auswertung der Replikation. Außerdem stelle ich kurz das Origianlexperiment und die erste Replikation vor.

3.1 Verwandte Arbeiten

3.1.1 Originalexperiemt

Prechelt et al.[1] führten das Originalexperiment im November 1997 mit 29 professionellen Softwareentwicklern über 2 Tage hinweg in München durch. Das Experiment befasste sich mit der Fragestellung, ob Entwurfsmuster in kleineren Programmen die Wartbarkeit positiv oder negativ beeinflussen. Zu diesem Zweck wurden 4 C++ – Programme in je 2 Versionen geschrieben - eine Version mit Entwurfsmuster (Pat) eine Version ohne Entwurfsmuster (Alt). Zu jedem Programm wurden Wartungsaufgaben gestellt, die für beide Programmversionen identisch waren. Die Teilnehmer wurden in 4 Gruppen eingeteilt und bearbeiteten die Aufgaben mit je einer Version der Programme. Die Aufgaben wurden mit Stift und Papier gelöst. Nach 2 Programmen besuchten die Teilnehmer einen Kurs mit dem Thema Entwurfsmuster. Abbildung 2 veranschaulicht diesen Ablauf. Rechtecke stellen dabei Pat – Versionen der Programme dar, Kreise Alt – Versionen. Die Abbildung entstand in Anlehnung an eine ähnliche Abbildung in [2], jedoch wurden an Stelle der Programmnamen Zahlen in die Rechtecke und Kreise geschrieben.

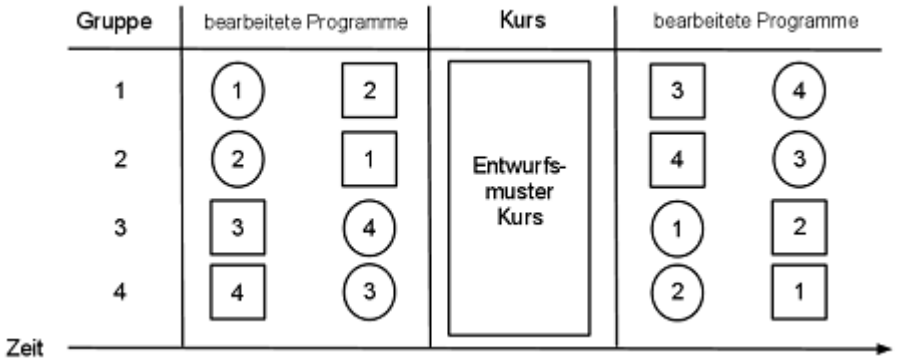

Figure 2: Ablauf des Originalexperiments

Die Ergebnisse der Auswertung bestätigten weitgehend die aufgestellten Hypothesen. Das Observer – Muster hat in kleinen Programmen negative Auswirkungen auf die Wartbarkeit, das Visitor – Muster hat weder positive noch negative Effekte, das Decorator – Muster hat einen positiven Einfluss auf die Wartbarkeit und das Abstract Factory - Muster brachte nur kleine Auswirkungen mit sich.

3.1.2 Replikation aus Norwegen

Vokac et al.[2] führten 2002 die erste Replikation des Originalexperiments durch. Die 44 Teilnehmer waren ebenfalls professionelle Softwareentwickler, wurden für ihre Teilnahme bezahlt und bearbeiteten die Aufgaben an Computern, um realistischere Experimentumstände zu simulieren. Die Ergebnisse decken sich nur teilweise mit denen des Origianlexperiments. Für das Observer – Muster wurden positive Auswirkungen gefunden und das Visitor – Muster hatte starke negative Auswirkungen. Decorator und Abstract Factory – Muster hatten ähnliche Auswirkungen.

3.2 Aktuelle Replikation

Die aktuelle Replikation wurde im Herbst 2010 an der Freien Universität zu Berlin im Rahmen der vorliegenden Bachelorarbeit durchgeführt. Nach [4] handelt es sich dabei um eine 'abhängige Replikation', d.h. es wurde z.B. die Population geändert, aber im Gegensatz zu einer 'unabhängigen Replikation' der Entwurf des Experiments zum größten Teil beibehalten.

3.2.1 Unterschiede zum Originalexperiment

Im Gegensatz zur Vorlage werden JAVA statt C++ – Programme verwendet. Außerdem werden nur zwei der vier Programme verwendet, die Graphics

Library und die Communication Library. Tabelle 1 zeigt eine Übersicht der verwendeten Programmen. Die Entwurfsmuster, die in diesen Programmen zum Einsatz kommen, zeigten sich in den obengenannten Experimenten als die interessantesten. Des Weiteren sahen wir die Notwendigkeit den Umfang zu kürzen, um die Schwelle zur Teilnahme und die Mortalität möglichst gering zu halten.

Außerdem wurden die Aufgaben am Computer bearbeitet, d.h. zur Lösung der Aufgaben reichte es nicht aus zu skizzieren was geändert werden muss, die Änderungen mussten auch umgesetzt werden.

Die Teilnehmer waren keine professionellen Softwareentwickler mit mehrjähriger Berufserfahrung, sondern Studenten, die meisten von ihnen im Bachelorstudium.

3.2.2 Teilnehmer und Gruppen

Insgesamt nahmen 13 Studenten aus dem Softwaretechnik – Softwareprojekt teil. Zwei Teilnehmer führten das Experiment nicht komplett durch. Ihre Datensätze wurden daher bei der Auswertung der Ergebnisse nicht berücksichtigt. Informationen über den fachlichen Hintergrund der Teilnehmer wurden mit Hilfe zweier im Portal integrierter Fragebögen vor Beginn der Bearbeitung der Aufgaben gesammelt.

Alle Teilnehmer studieren Informatik als Hauptfach und haben Erfahrung in der Programmierung mit JAVA und Entwurfsmustern. Acht der Teilnehmer haben bereits Berufserfahrung, im Durchschnitt anderthalb Jahre.

Die Zuteilung zu den zwei Gruppen erfolgte wie folgt: Je vier zufällig ausgewählten Teilnehmern wurde ein Block mit vier IDs zugeteilt. Die IDs waren dabei bezüglich des modulo vier Ergebnisses paarweise verschieden und je zwei Ergebnisse sind einer Gruppe zugehörig.

In Tabelle 2 sieht man einen Vergleich der Gruppen bez"uglich des abgefragten fachlichen Hintergrunds, aufgeführt sind die Gruppen - Mittelwerte. Der große Unterschied beim Umfang des bisher geschriebenen Quelltexts, ist auf einen extremen Wert in Gruppe 1 zurückzuführen. Lässt man diesen Wert weg, verschwindet der unverhältnismäßig große Unterschied weg. Anders hingegen verhält es sich bei den geschriebenen Zeilen JAVA – Quelltext, hier gibt es keine extremen Datenpunkte, die die Werte verzerren, allerdings halte ich 1000 Zeilen Quelltext Unterschied für keine versuchsverfälschende Größe.

Tabelle 3 vergleicht die Erfahrung der Gruppen in Bezug auf die in den verwendeten Programmen vorkommenen Entwurfsmuster. B und V stehen für 'Muster bekannt' und 'Muster verwendet'. Die Zahl dahinter gibt an auf wie viele der Gruppenmitglieder diese Eigenschaft zutrifft. Ein deutlicher Unterschied ist hier vor allem beim Muster Abstract Factory zu erkennnen. In Gruppe 1 gab keiner der Teilnehmer an, das Muster gut zu kennen,

Programm	Beschreibung und Komplexität	Aufgabenstellung	Entwurfsmuster
Graphics Library	Stellt Methoden zum Zeichnen von primitiven Grafikelementen bereit und eine Wahlmöglichkeit zwischen verschiedenen Ausgabegeäten LOC Pat: 614 LOC Alt: 561	**1.** Hinzufügen eines neuen Ausgabegeräts **2.** Überprüfen der Funktionalität einer vorhandenen Methode	**Pat:** Abstract Factory und Composite **Alt:** Abstract Factory
Comm. Library	Stellt Methoden eines simplen Kommunikationsprotokolls bereit. LOC Pat: 436 LOC Alt: 384	**1.** Hinzufügen eines Befehls zur Hamming Kodierung **2.** Mögliche Zustände des Programms überprüfen **3.** Neues Kommunikationsobjekt mit bestimmter Funktionalität erstellen	**Pat:** Decorator **Alt:** kein Entwurfsmuster

Table 1: Programmübersicht

Variabe	Gruppe 1	Gruppe 2
Zeilen Quelltext gesamt	38800	24300
Zeilen Quelltext JAVA	9100	10200
Programmierstunden / Woche	21	16
Benutzte Entwurfsmuster	6	6

Table 2: Fachlicher Hintergrund der Gruppen

Entwurfsmuster	Gruppe 1	Gruppe 2
Abstract Factory	B5 V0	B3 V3
Composite	B0 V5	B2 V4
Decorator	B3 V2	B3 V3

Table 3: Erfahrung mit den verwendeten Entwurfsmustern

wohingegen die Hälfte der Mitglieder von Gruppe 2 schon mit dem Muster gearbeitet hat. Für die Muster Composite und Decorator sind keine nennenswerten Unterschiede aufgefallen.

3.2.3 Durchführung

Ein erster Versuch das Experiment durchzuführen fand Ende September an der Freien Universität zu Berlin statt. Auf Grund technischer Probleme schlug dieser Versuch jedoch fehl. Daher wurde nach ausführlicher Überarbeitungsphase Mitte Oktober 2010 ein weiterer Versuch gestartet. Zur Vorbereitung auf die neuen Umstände - die Teilnehmer arbeiteten nun von zu Hause aus und nicht in der Universität - sandte ich den Teilnehmern detailierte Anweisungen zum Procedere via E – Mail zu.

Um Anonymisierung zu gewährleisten, habe ich die E - Mails danach wieder gelöscht. Im Anhang findet sich unter dem Punkt 'Ankündigungsemail' sich eine Vorlage der versendeten E – Mails.

Die Aufgabenstellungen wurden vom Portal bereitgestellt, die Ergebnisse im Portal hochgeladen und gespeichert. Empfohlen war die Entwicklungsumgebung Eclipse zur Lösung der Programmieraufgaben zu benutzen.

3.2.4 Hypothese

In den Abschnitten 3.4 und 3.5 werden Erwartungen an die Ergebnisse der Messungen aufgestellt. Implizit liegt diesen Erwartungen folgende Hypothese zu Grunde: enthält ein Programm P das Entwurfsmuster E wird Wartungsaufgabe A schneller durchgeführt, als wenn sie im Programm P ohne E durchgefüht wird.

3.2.5 Entwurf

Da es sich um eine abhängige Replikation handelt, orientiert sich der Entwurf in großen Teilen am Originalexperiment.

Variablen:
Im Experiment gibt es 2 unabhängige und 2 abhängige Variablen.

1. Unabhängige Variablen

   - **Aufgabe:** Für beide Programme gibt es jeweils eine Programmieraufgabe und eine - für Communication Library zwei - Verständnisaufgaben. Es handelt sich dabei um Aufgaben, die bei der Wartung der Programme anfallen könnten.
   - **Programmversion:** Jedes Programm liegt in 2 Versionen vor: eine Version stützt sich auf Entwurfsmuster, die andere realisiert die gleiche Funktionalität ohne Entwurfsmuster.

2. Abhängige Variablen

   - **Zeit:** Die Zeit benötigt für eine einzelne Aufgabe, gemessen in Sekunden.
   - **Korrektheit der Lösung:** Bis zu welchem Grad erfüllt die Lösung die Aufgabenstellung.

Die Teilnehmer wurden in 2 Gruppen eingeteilt, jeder Teilnehmer bearbeitete ein Programm in der Version mit Entwurfsmuster und das andere Programm ohne Entwurfsmuster.

Performance–Modell:
Analog zum Originalexperiment wird davon ausgegangen, dass sich die Zeit für eine Aufgabe aus folgenden 5 Faktoren zusammensetzt:

1. Verstehen der Aufgabe
2. Finden der anzupassenden Programmteile
3. Verstehen der betroffenen Programmteile
4. Verstehen wie die Änderung durchgeführt wird
5. Durchführen der Änderung

Abgesehen von Punkt 1 hängt der tatsächliche Aufwand für die einzelnen Punkte von der Programmversion und dem Aufbau des Programms ab. Teile der Überlegungen angestellt in Punkt 2 und 3 werden auch für die 2. (und 3.) Aufgabe nützlich sein.

3.2.6 Interne und Externe Gültigkeit

Intern Die interne Gültigkeit ist der Grad zu dem die Beobachtungen und Schlussfolgerungen nur auf die unabhängigen Variablen des Experiments zurückführen sind. Je höher die interne Gültigkeit, desto wahrscheinlicher ist es, dass durch das Experiment ein kausaler Zusammenhang zwischen Variablen und Beobachtung gezeigt wird. Für dieses Experiment sind dabei folgende Einschränkungen zu beachten:

- Die Einteilung der Gruppen erfolgte rein zufällig. Dadurch können die Gruppen unbalanciert sein. Ich habe dies in Abschnitt 3.2.2 erläutert und bin in der Auswertung darauf eingegangen.
- Die Teilnehmer arbeiteten von zu Hause aus und waren damit prinzipiell mehr störenden Einflüssen ausgesetzt. Vor der Durchführung wies ich die Teilnehmer auf dieses Problem hin und bat sie auftretende Störungen zu protokollieren und mir mitzuteilen. Ich habe keinen Grund anzunehmer, dass das nicht geschehen ist.
- Durch die Benutzung des 'Zurück' – Knopfs im Browser kann die Zeitmessung 'überlistet' werden und damit insgesamt verfälscht werden. Ich wies die Teilnehmer auf diese Tatsache hin und bat sie von der Benutzung abzusehen. Da es keine großen Ausreißer nach oben oder nach unten gibt, gehe ich davon aus, dass dieser Bitte entsprochen wurde.

Extern Durch die externe Gültigkeit wird beschrieben inwieweit die Ergebnisse des Experiments auf andere Situationen anwendbar sind. Für dieses Experiment sind dabei folgende Einschränkungen zu beachten:

- Die Teilnehmer des Experiments sind Studenten, im Gegensatz zu 'Profis' mit abgeschlossenem Studium und größerer Erfahrung, die im Regelfall in einer Firma diese Aufgaben übernähmen. Es ist nicht klar wie sich dieser Umstand auswirkt. Denkbar ist, dass die hier gemessenen Ergebnisse eine untere Schranke für professionelle Probanden darstellen, da diese möglicherweise über mehr Erfahrung und größeres Wissen verfügen.
- Die Wartungsaufgaben werden normalerweise von den Autoren der Programme, bei denen die Einarbeitungszeit zu einem großen Teil entfiele, übernommen. Auf die Teilnehmer traf dies nicht zu, daher kann man aus den Ergebnissen keine Schlüsse über die genannte Konstellation ziehen.
- Der hohe Grad an Dokumentation wird in nicht - künstlichen Programmen nicht anzufinden sein. Aus den Fragebögen zu den einzelnen

Variable	F	Signifikanz
Aufgabe	30.840	< 0.001
LOC	0.523	0.831
Version	0.299	0.587
bekannte Muster	0.263	0.951

Table 4: Ergebnisse der einfaktoriellen Varianzanalyse

Aufgaben wird ersichtlich, dass die Teilnehmer die Dokumentation nie als störend, häufig aber als hilfreich empfanden. Das mag eine sehr subjektive Einschätzung sein, sollte aber bedacht werden, da in [10] gezeigt wurde, dass entwurfsmusterspezifische Dokumentation hilfreich ist.

3.3 Analyse

Nun folgt die Analyse der erhaltenen Daten. In Anlehnung an das Originalexperiment bezeichnet 'Pat - Gruppe' hier die Gruppe aus der Gesamtheit der Teilnehmer, die für ein gegebenes Programm die Programmversion mit Entwurfsmuster bearbeitet hat. Entsprechend ist 'Alt - Gruppe' die Gruppe, die die Programmversion ohne Entwurfsmuster bearbeitet hat.

3.3.1 Nachbearbeitung der Daten

Vor der Auswertung der Daten änderte ich bei einem Teilnehmer die Zeit für die Bearbeitung der ersten Aufgabe von Graphics Library und für einen anderen Teilnehmer den Eintrag für die Zeitmessung der ersten Aufgabe für Communication Library. Die Änderungen erfolgten auf Hinweise im Schlusskommentar, dass die Teilnehmer für x Minuten unterbrochen worden waren. Außerdem entfernte ich die zwei unvollständigen Datensätze der Teilnehmer, die das Experiment nicht komplett duchgeführten.

3.3.2 Methoden

Zuerst führte ich eine einfaktorielle Varianzanalyse durch, um zu bestimmen welche Variablen den größten Einfluss auf die Bearbeitungszeit hatten. Die Ergebnisse sind in Tabelle 4 dargestellt. Den größten Einfluss hat die Art der Aufgabe, die restlichen Variablen sind nicht signifikant. Für die einzelnen Aufgaben habe ich die Mittelwerte der 2 Versionen der Programme gegenübergestellt. Die Werte habe ich mit Hilfe des Bootstrapverfahren berechnet.

3.4 Programm Communication Library

Wie bereits in Tabelle 1 beschrieben, handelt es sich bei diesem Programm um eine Bibliothek, die Datenübertragung realisieren soll und neben einem Standardkanal zusätzliche Funktionen wie Verschlüsselung und Protokollierung bereitstellt. Die zwei Programmversionen unterscheiden sich vor allem hinsichtlich der Struktur. In der Alt – Version gibt es, neben den Dummyklassen, eine Hauptklasse, die den Standardkanal samt zuschaltbaren Zusatzfunktionen enthält. In der Pat – Version kommt hingegen das Decorator - Muster zum tragen. Für den Standardkanal und jede Zusatzoption existieren Klassen, die die Funktionalität realisieren. Kanäle mit Zusatzfunktionen werden durch geschachtelte Objektaufrufe erstellt.
Für Aufgabe 2 und 3 wurde keine separate Zeitmessung durchgeführt, insofern kann über den genauen Zeitbedarf pro Aufgabe nur spekuliert werden. Daher habe ich für diese beiden Aufgaben keine Erwartungen an die Dauer der Bearbeitung formuliert. Allerdings sind die Unterschiede in den Bearbeitungszeiten der Gruppen keineswegs groß, die markanten Quantile dargestellt in den Box - Plots in Abbildung 3 sind nahezu identisch. Angesicht des geringen Aufwands der Aufgaben ist nicht davon auszugehen, dass große Unterschiede vorhanden sind.

3.4.1 Aufgabe 1

Die Bibliothek soll zusätzlich eine Funktion zur Bitsicherung durch Hamming Codes bereitstellen. Diese Funktion soll genauso bei Bedarf zuschaltbar sein, wie die Protokollierung, Verschlüsselung und Kompression.
Erwartung:
Die größere Modularisierung des Quelltexts in der Pat – Version kann sowohl hilfreich als auch hinderlich sein. Hilfreich, da man bei der Bearbeitung keine Rücksicht auf andere Programmteile nehmen muss, hinderlich, da das Verständnis des Programms eventuell erschwert wird. In der Alt – Version muss außerdem noch auf die Reihenfolge, in der die Zusatzfunktionen bearbeitet werden, geachtet werden. Daher erwarte ich, dass die Korrektheit in der Pat – Gruppe, bei längerer Bearbeitungszeit, höher als in der Alt – Gruppe ist.

Ergebnis:
Die Aufgaben wurden von fast allen Teilnehmern korrekt bearbeitet. Eine Antwort in der Alt - Gruppe war bezüglich der Reihenfolge nicht korrekt. Die Bearbeitungszeit war für die Gruppen fast identisch - im Durchschnitt 52.5 Minuten für die Gruppe mit Entwurfsmuster sowie 52.6 Minuten für die Gruppe ohne Entwurfsmsuter. Daraus schließe ich, dass Decorator – Muster keine negativen Auswirkungen bei kleineren Programmen hat. Einen Hinweis darauf, dass das Muster die bessere Wahl ist, gibt die minimal größere Korrektheit.

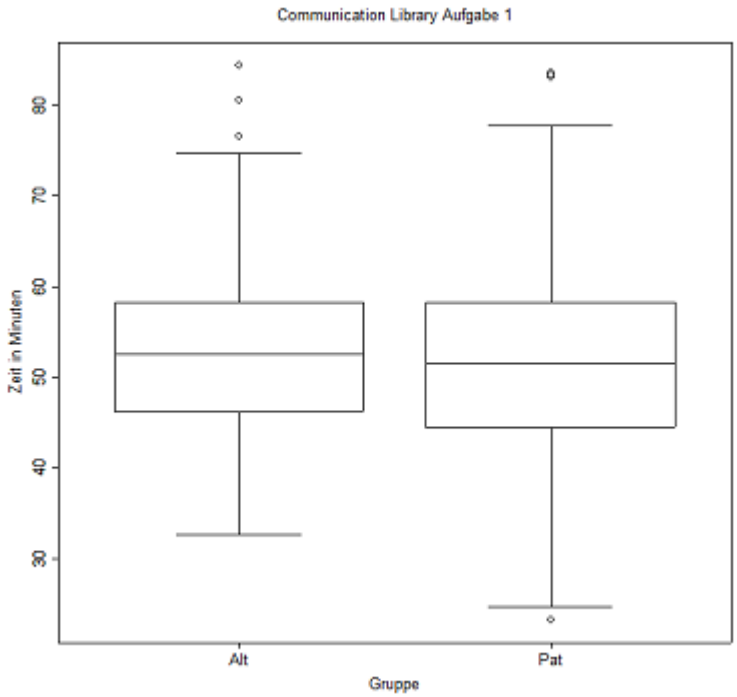

Figure 3: Communication Library Aufgabe 1 Bearbeitungszeiten

3.4.2 Aufgabe 2

Die Fragestellung ist, unter welchen Umständen der Aufruf channel.reset() für ein gegebenes Objekt CommChannel channel das Resultat CommChannel.IMPOSSIBLE liefert.

Erwartung:

Die Alt – Gruppe sollte keinerlei Probleme mit der Beantwortung der Frage haben, da sie nur die reset() – Funktion der Hauptklasse analysieren muss. Für de Pat – Gruppe, stellt sich die Aufgabe ein wenig komplizierter dar, da sie in jeder Klasse nach der reset() – Funktion suchen muss. In Aufgabe 1 sollten beide Gruppen allerdings genug Programmverständnis gesammelt haben, um die Aufgabe ohne Probleme korrekt zu lösen.

Ergebnis:

In beiden Gruppen gab es eine Antwort, die nicht alle möglcihen Zustände aufgezählt hat, der Rest war korrekt. Von den meisten Teilnehmern, wurden jedoch alle Möglichkeiten den Zustand IMPOSSIBLE zu erreichen angegeben. Meine Erwartungen bestätigten sich.

3.4.3 Aufgabe 3

Es war gefordert einen new Ausdruck für einen Kanal, der komprimiert und verschlüsselt, aber nicht protokolliert, anzugeben.
Erwartung:
Der Ausdruck ist für die Pat – Version ungleich schwieriger zu konstruieren, da die Teilnehmer auf korrekte Schachtelung der Objekte achten müssen. Das setzt sehr genaues Programmverständnis voraus. Die Alt – Gruppe muss hingegen nur einen simplen Konstruktoraufruf formulieren. Ich erwarte daher, dass die Anzahl der korrekten Antworten in der Alt – Gruppe höher ausfallen wird.
Ergebnis:
Alle Teilnehmer beantworteten diese Frage korrekt, damit hat sich meine Erwartung nicht erfüllt.

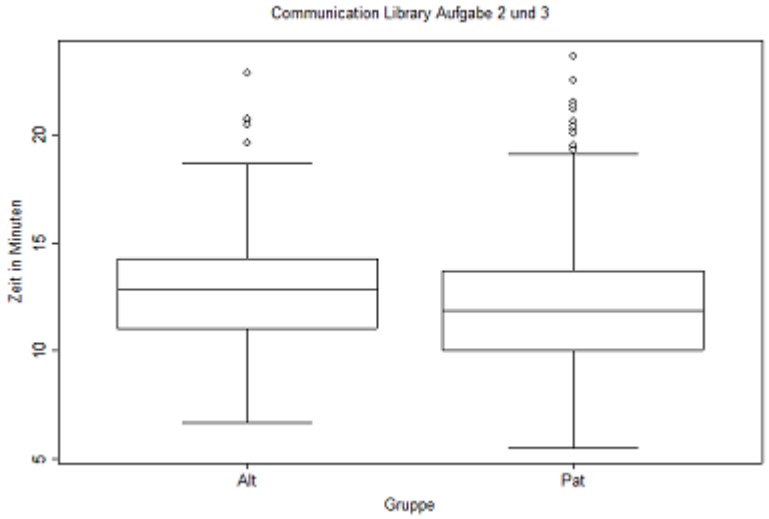

Figure 4: Communication Library Aufgabe 2 und 3 Bearbeitungszeiten

3.5 Programm Graphics Library

Wie bereits in Tabelle 1 beschrieben, handelt es sich bei diesem Programm um eine Bibliothek, die Funktionen bereitstellt, um einfache graphische Objekte mit Hilfe verschiedener Ausgabegeräte darzustellen. In der Klasse Generator wird dieses Ausgabegerät ausgewählt. Die einfacheren Objekte wie Punkt und Linie sind für die verschiedenen Ausgabegeräte identisch implementiert. Nur bei komplexeren Objekten wie einem Kreis unterscheidet sich die Implementierung. Im Gegensatz zum Communication Library - Programm besitzt Graphics Library eine ausführbare main Methode.

Die Pat – Version des Programms verwendet die Muster Abstract – Factory und Composite. Durch das Composite – Muster können Gruppen von Objekten hierarchisch geordnet werden.

Die Alt – Version des Programms verwendet nur das Muster Abstract – Factory und eine Emulation des Composite – Musters. Damit sind allerdings keine hierarchischen Ordungen in Objekgruppen möglich.

Wie man der Beschreibung bereits entnehmen kann, sind sich die Versionen sehr viel ähnlicher als die des Communication Library – Programms. Die Ergebnisse lassen darauf auf den ersten Blick allerdings nicht schließen, die Unterschiede in der Bearbeitungszeit sind enorm.

3.5.1 Aufgabe 1

Die Teilnehmer sollten die Bibliothek um eine dritte Art von Ausgabegerät, einen Plotter, erweitern.
Erwartung:
Durch die starke Ähnlichkeit der Versionen erwarte ich keine großen Unterschiede bei Korrektheit und Bearbeitungszeit zwischen den Gruppen.
Ergebnis:
Die Gruppen unterscheiden sich bezüglich der Korrektheit der Lösungen nicht. Allerdings gibt es erhebliche Unterschiede in der Bearbeiungszeit. Während die Pat – Gruppe im Durchschnitt 77.6 Minuten benötigt, sind es für die Alt - Gruppe im Durchschnitt nur 55.4 Minuten. Meine Erwartung wurde damit nicht bestätigt. Eine Erklärung für den grosse n Unterschied, der sowohl in [1] als auch in [2] nicht auftrat, mag am unterschiedlichen Wissen über das Muster Abstract – Factory liegen. Wie in Tabelle 3 beschrieben, hat keiner der Teilnehmer aus der Pat – Gruppe das Muster schon einmal angewandt, es war nur theoretisches Verständnis vorhanden. Auf Grund der Angaben der Teilnehmer zu ihrem Wissen über das Muster Composite schließe ich aus, dass dies der bestimmende Faktor war.

3.5.2 Aufgabe 2

Die Aufgabe war zu bewerten, ob eine gegebene X - förmige Abbildung mit einer ebenfalls gegebenen Reihe von Befehlen erzeugt werden kann.

Erwartung:
Die Schwierigkeit dieser Aufgabe liegt darin das Composite – Muster zu verstehen. Die Auflistung des Entwurfsmusterwissens in Tabelle 3 zeigt, dass die Teilnehmer der Pat – Gruppe praktische Erfahrung mit diesem Muster haben. Daher erwarte ich, dass die Aufgabe die Teilnehmer vor keine großen Probleme stellen sollte. Da die Alt – Version des Programms der Pat – Version sehr ähnlich ist, erwarte ich auch für die Alt – Gruppe eine größtenteils richtige Antworten.

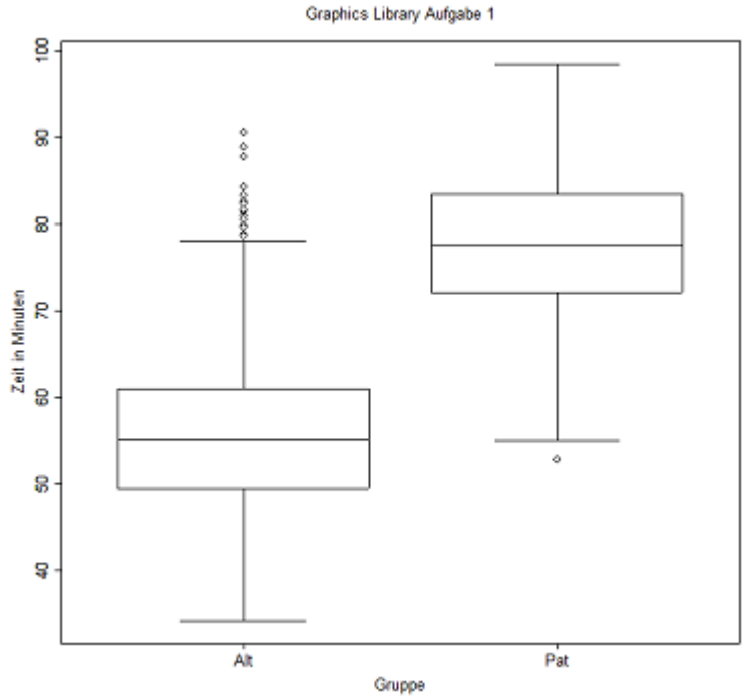

Figure 5: Graphics Library Aufgabe 1 Bearbeitungszeiten

Ergebnis:
In beiden Gruppen gab es nur eine richtige Antwort. Die gemessenen Zeiten legen besonders in der Alt – Gruppe nahe, dass sich die Teilnehmer nicht eingehend mit der Aufgabe beschäftigt haben, während die durchnittliche Bearbeitungszeit bei der Replikation in Norwegen [2] bei über 30 Minuten lag und im Originalexperiment [1] bei etwa 15 Minuten liegt, liegt sie in diesem Experiment nur bei etwa neun Minuten, der Median bei fünf Minuten.

Meine Erwartungen an die Korrektheit wurde damit nicht bestätigt, die Erwartung zu den Unterschieden zwischen den Gruppen bestätigte sich.

3.6 Fazit

Das Ziel dieser Replikation war eine Überprüfung der Forschungsergebnisse aus [1] und [2] und einen Beitrag zur geplanten Joint – Replication zu leisten.

Für das Decorator – Muster widersprechen die Ergebnisse Veröffentlichung [1], die festgestellt, dass Wartung von Programmen mit Decorator – Muster zwar Geschwindigkeitsvorteile bietet, aber unter erhöhter Fehleranfälligkeit leidet. Die Ergebnisse dieser Replikation legen nahe, dass die Unterschiede in der Wartungszeit praktisch vernachlässigbar sind, aber ein höherer Grad

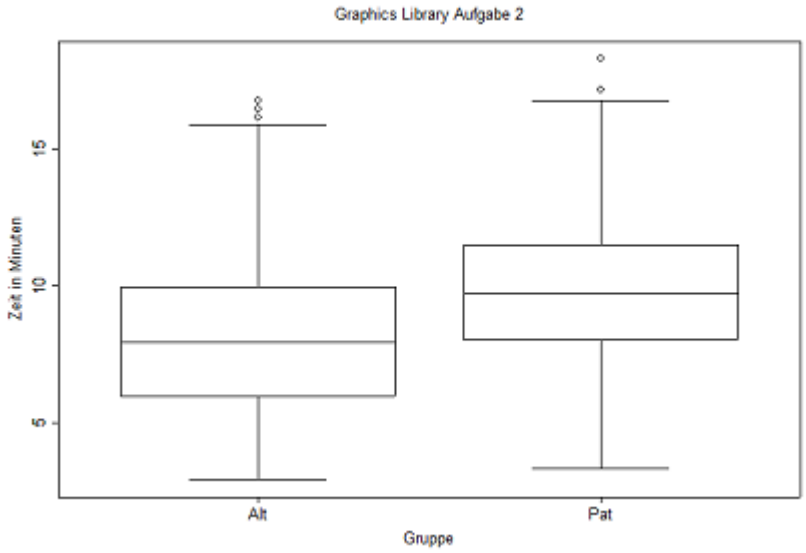

Figure 6: Graphics Library Aufgabe 2 Bearbeitungszeiten

an Korrektheit erreicht wird.

Das Muster Abstract – Factory erschwert die Wartung von Programmen, wenn den Wartenden das Muster nur in der Theorie bekannt ist. Dieses Ergebnis widerspricht demjenigen aus [2], das dem Muster kaum Einfluss auf die Bearbeitungszeit zuspricht. Es unterstützt das Resultat aus Veri¿$\frac{1}{2}$ffentlichung [1], die ebenfalls feststellt, dass die Wartung eines Programms mit dem Muster Abstract –Factory Geschwindigkeitsvorteile bietet, wenn den Wartenden das Muster bekannt ist.

Das Composite – Muster, obwohl allen Teilnehmern bekannt, hatte keinen Einfluss auf die Resultate. Diese Beobachtung bestätigt sowohl [1] als auch [2].

Das Augenmerk weiterer Replikationen sollte zum einen darauf liegen wie sich die Ergebnisse der Aufgabe 2 der Graphcs – Library ändern, wenn mit einer lauffähigen Version des Programms gearbeitet wird und zum anderen, ob die neuen Erkenntnisse vestätigt oder widerlegt werden können.

3.7 Danksagungen

An dieser Stelle sei ganz herzlich gedankt den Mitgliedern der Replication – Mailingliste für ihr ausführliches Feedback, Ulrich Stärk für seine Hilfe im Umgang mit Tapestry, Julia Schenk und Stefan Salinger für ihr Feedback zur Benutzbarkeit des Portals und Lutz Prechelt für die Betreuung dieser Arbeit.

3.8 Literaturverzeichnis

[1] Prechelt L., B. Unger, W. Tichy, P. Brössler, L. Votta: *A Controlled Experiment in Maintenance Comparing Design Patterns to Simpler Solutions*

[2] Vokac M., W. Tichy, D. Sjoberg, E. Arisholm, M. Aldrin: *A controlled experiment comparing the maintainability of programs designed with and without Design Patterns | a replication in a real programming environment*

[3] Shull F., J. Singer, D. Sjoberg (Editors): *Guide to Advanced Empirical Software Engineering*, Springer 2008

[4] Shull F., J. Carver, S. Vegas, N. Juristo: *The Role of Replications in Empirical Software Engineering*

[5] Basili V., F. Lanubile: *Building Knowledge through Families of Experiments*

[6] https://wiki.mi.fu-berlin.de/w/SE/ThesisPatmainReplication

[7] http://tapestry.apache.org/

[8] http://www.hibernate.org/

[9] Basili V.: *The Experimental Paradigm in Software Engineering*

[10] Prechelt L., B. Unger, W. Tichy, M. Philippsen: *Two Controlled Experiments Assessing the Usefulness of Design Pattern Documentation in Program Maintenance*

4 Anhang

4.1 Hibernate - Timeout

Der Fehler wurde durch Natalia Juristo entdeckt und Lutz Prechelt gemeldet. Durch ihn wurde ich auf den Fehler aufmerksam gemacht. Unten die ursprüngliche Nachricht von Natalia Juristo.

––Ursprüngliche Nachricht––
Von: replication-bounces@lists.spline.de
mailto:replication-bounces@lists.spline.de Im Auftrag von Natalia Juristo
Gesendet: Donnerstag, 14. Oktober 2010 12:58
An: discussion of software engineering experiment replications
Betreff: Re: [Replication] What will we do, really?

Hi Lutz!

...
I have tried to enter the replication portal and it failed.
I have tried to register and the attached error has occurred.

As soon as the portal is working we could start.

Thanks!

Natalia

Hier meine Antwort an die Replication - Mailingliste:

Dear PIJORs,
earlier today some of you encountered the error mentioned in the subject line when trying to register at the website set up for the experiment replication. My sincere apologies for that. As I found out, it seems that the standard hibernate connection mechanism is not able to reconnect automatically after a MySQL timeout, which occurs after 8 hours of inactivity. I was not aware of that fact, as the basic hibernate tutorials did not bring that topic up. However, using a different connection mechanism, namely H3C0, an automatic reconnect is possible. Thus, I changed the hibernate configuration to specifically use c3p0.
Regards,
Martin Liesenberg

4.2 Ankündigungsemail

Das ist die Vorlage der E - Mail mit der die Teilnehmer in die Durchführung des Experiments eingeführt wurden.

Hallo ****,
zwecks Teilnahme am angekündigten Experiment über Designmuster hier nun der Link und die ID.
Das Experiment heißt *****.
ID: *****
link: replication.inf.fu-berlin.de/replication/userlogin

Es ist wichtig, dass das Experiment nach Möglichkeit ohne große Unterbrechungen durchgeführen wird, damit die gemessenen Zeiten akkurat sind. Sollten doch Unterbrechungen auftreten, dann bitte ich dich die Unterbrechung im Kommentarfeld am Ende einzutragen. Benutze außerdem bitte nicht den Zurück - Knopf deines Browsers, da dadurch die Zeitmessung verfälscht wird. Vielen Dank dein Verständnis und deine Teilnahme,
Gruß, Martin Liesenberg