Universität Kaiserslautern
Fachbereich Informatik
Lehrgebiet Datenverwaltungssysteme

Softwarepraktikum 2003 – GBIS

Boris Stumm

28. April 2003

Inhaltsverzeichnis

1 Allgemeine Informationen zum Praktikum 2
1.1 Aufgabenstellung . . . 2
1.1.1 Inhalt und Ziel der Aufgabe . . . 2
1.1.2 Motivation . . . 2
1.1.3 Anwendungs-Szenario . . . 3
1.1.4 Konkretisierung des Teilsystems . . . 4
1.2 Übersicht über Durchführung und Ablauf . . . 4
1.2.1 Anforderungsanalyse . . . 5
1.2.2 Systementwurf und Modellierung . . . 5
1.2.3 Implementierung . . . 5
1.2.4 Integration und Validation . . . 5
1.3 Organisatorisches . . . 5
1.3.1 Gruppenstärke . . . 5
1.3.2 Pair-Programming . . . 6
1.3.3 Testat & Kolloquium . . . 6
1.3.4 Materialien . . . 6
1.3.5 Bewertung . . . 6
1.4 Schlussbemerkung . . . 7

2 Aufgaben 7
2.1 Aufgaben in den Entwicklungsphasen . . . 8
2.1.1 Anforderungsanalyse . . . 8
2.1.2 Systementwurf und Modellierung . . . 9
2.1.3 Implementierung . . . 9
2.1.4 Integration und Validation . . . 10

3 Hinweise und Richtlinien 11
3.1 Abgaben & Besprechungen . . . 11
3.1.1 Besprechungen . . . 11
3.1.2 Abgaben . . . 11
3.2 Veröffentlichen der Abgaben im Web . . . 11
3.3 Allgemeine Entwurfshinweise . . . 12
3.4 Hinweise zu den Komponenten . . . 12
3.4.1 DB-Zugriffskomponente und Generator-Komponente . . . 12
3.4.2 Grafische Benutzeroberfläche . . . 12

3.4.3 Visualisierungs-Komponente . . . 13
3.4.4 Metadaten-Verwaltung . . . 13
3.5 Richtlinien zu Dokumentation, Entwurf und Implementierung . . . 13
3.5.1 Grundprinzipien . . . 13
3.5.2 Dokumentation des Entwurfs . . . 14
3.5.3 Dokumentation und Formatierung des Codes . . . 14
3.5.4 Änderungen in späteren Entwicklungsphasen . . . 14

4 Materialien 15
4.1 Folien und Aufgabenbeschreibungen, Material fürs Praktikum . . . 15
4.2 Tools . . . 15
4.3 Websites, Dokumentationen . . . 15
4.3.1 HTML . . . 15
4.3.2 TPC-W . . . 15
4.3.3 Java . . . 16
4.3.4 JDBC . . . 16
4.3.5 Design Patterns . . . 16
4.3.6 UML . . . 16
4.3.7 Javadoc . . . 16

5 Systemumgebung 16
5.1 Rechte . . . 16
5.2 Werkzeuge . . . 17
5.3 Web-Veröffentlichungen . . . 17

6 Terminplan SWP 2003 17
6.1 Termine . . . 17
6.2 Übersicht . . . 18
6.3 Erläuterungen . . . 18

1 Allgemeine Informationen zum Praktikum

1.1 Aufgabenstellung

Die Aufgabe des Praktikums im Vertiefungsbereich „Betriebliche Informationssysteme“ baut auf Themen des Vorlesungszyklus „Entwicklung von Softwaresystemen (I-III)“ auf und soll Fähigkeiten im Umgang mit Inhalten aus diesem Bereich vertiefen.

Die in den Vorlesungen erlernten Methoden werden bei der Entwicklung eines größeren Softwaresystems praktisch angewendet und ihre Notwendigkeit verdeutlicht. Der Bearbeitungsaufwand ist so gewählt, dass die Aufgaben innerhalb eines Vorlesungszeitraums von einer Praktikumsgruppe (6 Personen) bewältigt werden knnen. Es werden alle Phasen eines Software-Entwicklungsprozesses (Anforderungen, Entwurf, Implementierung, Integration und Validation) durchlaufen.

Inhalt und Ziel der Aufgabe

Im Praktikum soll ein Teil des durch die TPC-W-Spezifikation vorgegebenen Systems zur Leistungsbewertung von E-Commerce-Servern implementiert werden.

Die TPC-W-Spezifikation gibt ein System zum webbasierten Buchhandel vor, das dann auf verschiedenen Hardware/Software-Plattformen implementiert und getestet werden kann. TPC-W spezifiziert detailliert, welche Funktionalitäten der Buchhandel haben soll, wie das Verhalten der Besucher des Buchhandels simuliert werden muss, in welchem Rahmen sich Antwortzeiten bewegen müssen und nicht zuletzt, wie die Datenbasis aufgebaut ist.

Die Entwicklung eines Systems zur automatischen Generierung dieser Datenbasis, d.h. Tabellen mit Autoren, Büchern, Kunden, Bestellungen usw. ist das Ziel dieses Praktikums. Die Datenbasis muss bestimmten Anforderungen bezüglich Verteilung, Umfang und Struktur der Daten genügen und wird in einem relationalen Datenbanksystem verwaltet.

Motivation

Softwareentwicklung im Sinne des Software Engineerings bedeutet die Erstellung eines Software-Systems ausgehend von einer System-Spezifikation durch ein Team mittels vorgegebener Techniken, Methoden und Werkzeuge. Bei der Entwicklung großer, komplexer Systeme sind u. a. folgende Prinzipien für den Entwurf und die Programmierung elementar:

- Modularisierung (Divide and conquer)
- Explizite Schnittstellen
- Dokumentation

Reale Anwendungsszenarien kann man weiter charakteristisieren.

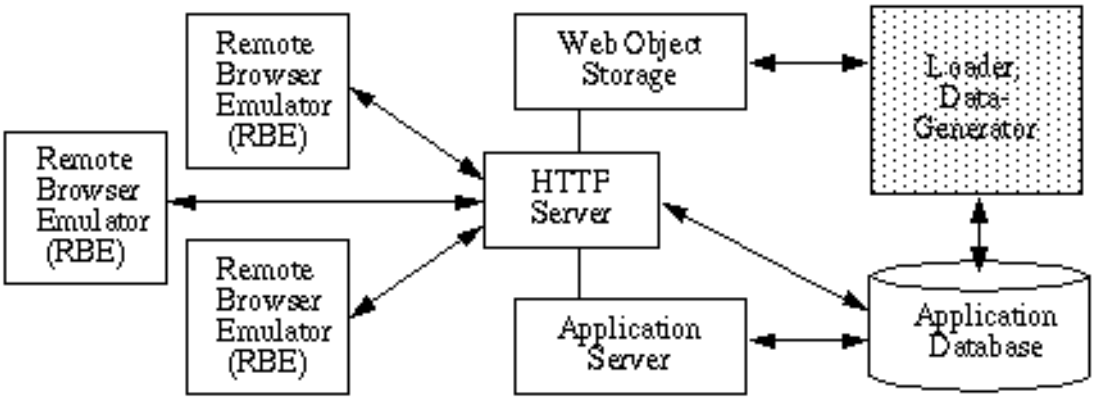
- Die Software muss Anforderungen Dritter genügen.
- Die Benutzung erfolgt nicht nur durch den jeweiligen Entwickler, sondern durch eine Vielzahl weiterer Personen.
- Die Entwicklung findet in der Regel in Teams statt, d.h., das Verhalten der Komponenten ist ausreichend zu spezifizieren und dokumentieren, um Integrations- und Koordinationsprobleme zu vermeiden.

Aufgrund dieser Charakteristika wurde der TPC-W Benchmark http://www.tpc.org als zentrales Thema dieser Aufgabe ausgesucht. Seine sehr genaue und vor allen Dingen hinreichend vollständige Spezifikation der funktionalen und nicht-funktionalen Anforderungen erlaubt eine Fokussierung auf die Anwendung von Methoden zur Softwareentwicklung und die Verfeinerung von bekannten Techniken für Modellierung, Entwurf und Programmierung, ohne durch bekannte Probleme, wie z.B. Mehrdeutigkeiten in der Problembeschreibung oder Unzulänglichkeiten der Dokumentation, zu verwirren oder aufzuhalten. Das gesamte Spektrum an Freiheitsgraden während der Realisierung bzw. der Modellierung des Softwaresystems bleibt erhalten.

Das Teilsystem kann nur durch die Leistung der gesamten Gruppe als Team realisiert werden, da der Umfang der Aufgaben kein anderes Arbeiten zulässt. Dies verdeutlicht die Notwendigkeit der Einhaltung oben genannter Prinzipien.

Anwendungs-Szenario

Im Bild ist ein Überblick der Architektur gegeben. Es zeigt die notwendigen Komponenten und deren Beziehung zueinander. Von diesen Komponenten ist im Zuge des Praktikums der Loader/Data-Generator (grau unterlegt) zu entwickeln.



Eine RBE-Instanz ist ein das Benutzerverhalten simulierender HTTP-Client (WWW-Browser-Simulation). Auf der Basis von vorgegebenen Wahrscheinlichkeiten werden Anfragen generiert, anschließend Antworten analysiert und daraus wieder entsprechende Anfragen erzeugt. Der RBE ist nicht nur für die Lastgenerierung, sondern auch für die Messung der Verarbeitungszeit einer Anfrage verantwortlich.

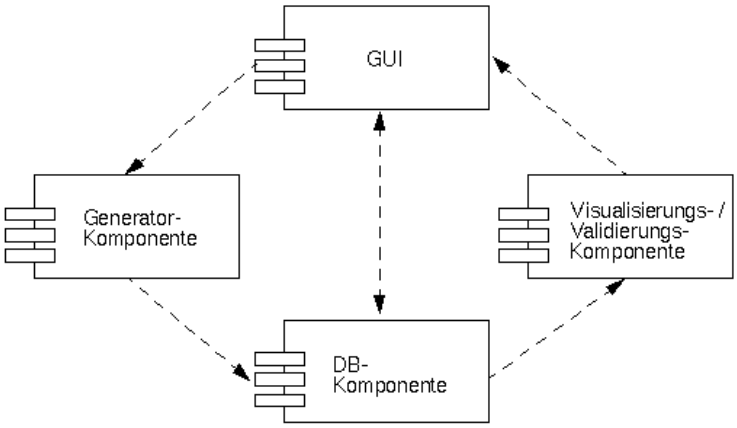
Zentrale Instanz des Systems ist der HTTP-Server, der Anfragen von RBE-Instanzen verarbeitet. Zu diesem Zweck muss er auf das Datenbanksystem zugreifen oder Verarbeitungseinheiten an den Applikationsserver delegieren. Ein Datenbankzugriff ist immer dann notwendig, wenn Bücherlisten angefordert, Suchanfragen formuliert oder Bücher geordert werden. Der Applikationsserver ist auch für die Transaktionsverarbeitung verantwortlich, die unter anderem bei verbindlichen Bestellungen notwendig wird.

Web-Objekte, wie beispielsweise Bilder, werden gesondert verwaltet. Sie können in einem Datenbanksystem verwaltet werden und müssen nicht zwingend in einem Dateisystem gehalten werden.

Das Teilsystem zur Population der Datenbank ist für die Erzeugung der Datenbasis notwendig. Zur Datenbasis gehören neben anderen Daten: Autorenverzeichnisse, Bücherkataloge, Bestellinformationen und auch Web-Objekte, wie beispielsweise Bilder. Dabei sind Vorgaben bezüglich der statistischen Verteilung der Ausprägungen einzelner Attribute zu beachten. Der TPC-W macht sehr genaue Angaben bezüglich der Häufigkeit des Vorkommens einzelner Ausprägungen.

Konkretisierung des Teilsystems

Das zu realisierende System besteht grob aus den nachfolgenden Komponenten, wobei in der Modellierungs- bzw. Entwurfsphase diese Aufteilung verfeinert oder durch eine geschicktere Aufteilung ersetzt werden kann. Sie dient hier nur der Verdeutlichung des Gesamtzusammenhangs. Wichtig bei der Modellierung ist die explizite Definition und Dokumentation der Schnittstellen, um eine parallele Entwicklung der entworfenen Komponenten zu unterstützen:



Die DB-Komponente stellt der Generator- und Visualisierung-Komponente eine Schnittstelle zur Verfügung, die den Datenbankzugriff kapselt und von dem Datenbanksystem abstrahiert. Es sind verschiedene Abstraktionsgrade denkbar: beispielsweise kann für jede DB-Operation (wie SELECT, INSERT, UPDATE, ...) eine Methode an dieser Schnittelle angeboten werden oder auch für jede Relation (Buch, Autor, Bestellung, ...) eine Methode zum Schreiben oder Lesen einer Instanz bzw. einer Menge von Instanzen definiert werden. Auch ein objektorientierter Ansatz ist vorstellbar, so dass jede Relation in einer eigenen Klasse gekapselt wird. Der Zugriff auf das Datenbanksystem selber geschieht mittels der standardisierten Anfragesprache SQL.

Die Generator-Komponente umfasst die Funktionalität zum Erzeugen von Instanzen unter Beachtung bestimmter statistischer Annahmen in Abhängigkeit von vorgegebener Anzahl und Struktur. Die benötigten statistischen Funktionen können eventuell in einer zusätzlichen Komponente modelliert werden.

Die Visualisierung-Komponente dient zur Qualitätssicherung. Sie soll den Inhalt der Datenbank visualisieren, um die Überprüfung der Einhaltung der vorgegebenen statistischen Randbedingungen zu ermöglichen (liegen die Werte in den vorgegebenen Grenzen, sind sie gleichverteilt, usw.). Die hier benutzten Funktionen zur Überprüfung der Korrektheit von Verteilungen sollten nicht auf die Funktionen der Generator-Komponente zugreifen, um nicht falsche Ergebnisse mit gleichen, falschen Annahmen zu verifizieren.

Der Zugriff auf Generator und Visualisierung soll über eine grafische Benutzeroberfläche ermöglicht werden.

Da die Programmierung in Java erfolgt, wird als Schnittstelle zum Datenbanksystem JDBC eingesetzt. Als Datenbankverwaltungssystem kommt der Informix Dynamic Server 2000 zum Einsatz.

1.2 Übersicht über Durchführung und Ablauf

Die Durchführung der Aufgabe orientiert sich am Wasserfall-Modell [B. Boehm]. Dieses Lebenszyklusmodell gliedert sich in vier Phasen, welche nach dem Modell genau einmal durchlaufen werden. Es eignet sich in diesem Fall sehr gut, da die Anforderungen zu Beginn vollständig beschrieben werden können und keine Integrationsprobleme zu erwarten sind:

- Anforderungsanalyse
- Systementwurf und Modellierung
- Implementierung
- Integration und Validation

Am Ende jeder Phase sind von den Gruppen jeweils Dokumente abzugeben, die die Ergebnisse der Phase zusammenfassen und das weitere Vorgehen beschreiben. Im Rahmen eines Kolloquiums werden diese Ergebnisse diskutiert und bewertet.

Anforderungsanalyse

Die eigentliche Anforderungsanalyse entfällt aufgrund der Vorgabe der TPC-W-Benchmark-Spezifikation. Diese Phase dient vielmehr der Findung der eigenen Rolle eines jeden Studierenden innerhalb seiner Gruppe bzw. seines Teams und der allgemeinen Planung und Einarbeitung in die Teilaufgabe des Praktikums, so dass eine Aufgabenverteilung im Team möglich wird.

Diese Phase umfasst ungefähr eine Woche.

Systementwurf und Modellierung

Neben weiterer Vertiefung des Themas und der einzusetzenden API ist ein Entwurf zu modellieren, der den Anforderungen genügt. Die Anforderungen ergeben sich aus der Spezifikation. Architektur, Schnittstellen, Klassen, interne Repräsentation von Datenbankobjekten und weitere Strukturierung durch Einsatz von Paketen (respektive Java Packages) sind zu definieren und zu dokumentieren. Weiterhin sind für jede Komponente Testfälle zu entwickeln, die eine Überprüfung derselben ermöglichen.

Diese Phase umfasst ungefähr drei Wochen.

Implementierung

Die Implementierung des vorgestellten Systementwurfs verfeinert u. a. die Technik im Umgang mit Java, JDBC und SQL durch die notwendige Auseinandersetzung mit der Thematik und deren tatsächliche Anwendung.

Diese Phase umfasst ungefähr vier Wochen.

Integration und Validation

Die Komponenten sind zu einem funktionierenden System zu integrieren und zu testen. Neben dem Funktionstest zur Überprüfung der Zuverlässigkeit ist ferner das System gegen die Anforderungen zu validieren, geeignete Testfälle können direkt aus der Spezifikation abgeleitet werden. Die Überprüfung geschieht mit Hilfe der Visualisierungskomponente. Am Ende dieser Aufgabe und zugleich am Ende des SWPs steht eine Systempräsentation im Rahmen aller Gruppen eines jeden Betreuers.

Diese Phase umfasst ungefähr drei Wochen.

Eine ausführliche Beschreibung der einzelnen Teilaufgaben befindet sich in Abschnitt 2.

1.3 Organisatorisches

Gruppenstärke

Eine Gruppe besteht aus 6 Teilnehmern.

Pair-Programming

Die Gruppe wird weiter in drei Zweiergruppen unterteilt. Jede dieser Gruppen ist für die Umsetzung eines Teils des Gesamtsystems verantwortlich (eine sinnvolle Aufteilung ist z.B. Visualisierung/GUI, DB-Komponente, Generator). Insbesondere gelten die beiden Mitglieder der Zweiergruppe als Ansprechpartner bei Rückfragen zu ihrer Komponente. Die Zweiergruppen werden von der Gruppe zu Beginn des Praktikums selbst festgelegt und bleiben bis zum Ende bestehen. Sinn des Pair-Programming ist ein gemeinsames Arbeiten, kein simples Aufteilen der Aufgaben. Beide Entwickler sollen zusammen an einem Rechner arbeiten. Bei Nachfragen müssen daher auch beide Entwickler jederzeit alle Teile der ihnen übertragenen Komponente genaustens kennen. In der Programmentwicklung erfahrenere Teilnehmer sollen darauf achten, dass ihr Partner nicht auf der Strecke bleibt. Bei Besprechungen mit den Betreuern hat jeweils ein Mitglied pro Zweierteam teilzunehmen; dabei sollen am Ende beide an der gleichen Anzahl Besprechungen teilgenommen haben.

Testat & Kolloquium

Nach der Phase des Entwurfs wird ein schriftliches Testat über die bisherigen Inhalte des Praktikums stattfinden. Nach der Systemintegration wird es weiterhin ein Kolloquium geben, in dem jeder Teilnehmer in einem kurzen Einzelgespräch von seinem Betreuer zum Praktikum befragt wird. Diese beiden Noten fließen in die Individualbewertung ein.

Materialien

Zu Beginn der Aufgabe werden folgende Dokumente ausgehändigt.

- Die Aufgabenbeschreibung (dieses Dokument).
- Die TPC-W-Spezifikation Version 1.8, Klauseln 0,1 und 4. Mehr ist zur Bearbeitung der Aufgaben nicht nötig. Um sich einen besseren Überblick zu verschaffen, können sich Interessierte die komplette Spezifikation auf der Praktikumsseite als PDF herunterladen.

Weitere Materialien finden sich auf der Praktikumsseite.

Bewertung

Die Bewertung des Praktikums setzt sich aus einer Gruppen- und einer Individualnote zusammen. Bei vollständiger Bearbeitung der gestellten Aufgaben kann eine maximale Punktzahl von 100 Punkten erreicht werden. Nach jeder Phase des Praktikums werden die Abgaben der Gruppe bewertet, die Summe der dabei erreichten Punkte ergibt die Gruppenbewertung. Diese macht insgesamt 50% der Punkte aus. Zusätzlich kann jeder Teilnehmer in den Kolloquien Punkte erringen, die in den Individualanteil einfließen.

Zur Bewertung der Gruppenleistung sind zu vorgegebenen Terminen Dokumente abzugeben, die die Ergebnisse der vorangegangenen Arbeit dokumentieren und erläutern. Ferner beinhalten sie Antworten zu Fragen, bzw. Aufgaben, die zusätzlich zur Hauptaufgabe der Realisierung eines Software-Systems vor oder während der Praktikumsaufgabe gestellt werden.

Die absolute Mindestanforderung für einen erfolgreichen Abschluss des Praktikums des Praktikums besteht darin, zum Ende ein lauffähiges System vorzustellen, das zumindest eine Relation des vorgegebenen DB-Schemas mit sinnvollen Daten füllt. Als Programmiersprache ist ausschließlich Java (JDK 1.x, JDBC 1.x) zu verwenden!

1.4 Schlussbemerkung

Das Praktikum wurde konzipiert, um die in den ersten Semestern vorgestellten Techniken und Methoden praktisch anzuwenden. Aufgrund der beschränkten Zeit von 13 Wochen kann natürlich nur ein Teil davon berücksichtigt werden. Nichtsdestotrotz ist die Aufgabe sehr gut geeignet, um ein Gefühl für die Probleme bei der Entwicklung eines großen Systems im Team zu entwickeln. Die benutzten Werkzeuge und die Programmierung in Java sollten (hoffentlich) schon aus den Übungen zu den Vorlesungen bekannt sein, so dass sich der Einarbeitungsaufwand hierfür in Grenzen hält.

Und nun: Viel Spaß beim Entwickeln :-)

2 Aufgaben

Die Hauptaufgabe besteht in der Realisierung eines Software-Systems zur Population der Datenbank laut TPCW-Spezifikation. Konkret sind die Paragraphen 1 und 4 der Spezifikation zu erfüllen. Paragraph 1 beschreibt die Anforderungen an das Datenbankschema und die Datenbank-relevanten Aspekte, während Paragraph 4 die Anforderungen an die zu generierenden Daten definiert. Eine Datenbank mit entsprechendem Schema ist vorgegeben und muss nicht modelliert werden.

Kommentar 2.1: Änderung zur TPC-W-Spezifikation

Abschnitt 4.6.2.19 der TPC-W-Spezifikation wird in diesem Praktikum durch folgenden Text ersetzt[1]:

4.6.2.19a The item title (I_TITLE) must be generated as a random a-string[14..60]. It must have an embedded substring generated as DigSyl(num, 7). The substring must start at an offset within the I_TITLE string random within [0..46].

```
if I_ID <= (NUM_ITEMS / 5)
    num = I_ID;
else
    num = random within [0..(NUM_ITEMS / 5)];
```

Example: Given an I_ID of 4628, the embedded substring is generated as DigSyl(4628, 7). Thus, the I_TITLE corresponding to an I_ID of 4628 would contain the substring BABABAREATALIN starting at some random offset between 0 and 46.

Comment: The intent of this Clause is to ensure that web interactions that execute a search based on the content of I_TITLE find a number of matching records.

4.6.2.19b The author last name (A_LNAME) must be generated as a random a-string [14..20]. It must have an embedded substring generated as DigSyl(num, 7). The substring must start at an offset within the A_LNAME string random within [0..6].

```
if A_ID <= (NUM_AUTHORS / 2.5)
    num = A_ID;
else
    num = random within [0..(NUM_AUTHORS / 2.5)];
```

[1] Der Text entspricht den Absätzen 4.6.2.17 und 4.6.2.17 des TPCW-Drafts D 5.1

Die für das zu realisierende Software-System relevanten funktionalen Anforderungen finden sich in Paragraph 4 und müssen erfüllt werden.

Alle Anforderungen in Paragraph 4, die sich nicht direkt auf das zu realisierende Software-System beziehen, beispielsweise Anforderungen an den Web-Server etc., können ignoriert werden. Ferner ist es nicht erforderlich den Punkt 4.6.2.13 zu erfüllen. Somit müssen zunächst keine Bilder (Images) oder Verkleinerungen (Thumbnails) erzeugt werden. Grafische Objekte brauchen nicht unterstützt zu werden!

2.1 Aufgaben in den Entwicklungsphasen

Das vorgegebene Lebenszyklusmodell gliedert sich in vier Phasen, welche nach dem Modell genau einmal durchlaufen werden. Die Durchführung der Aufgabe orientiert sich an diesen Phasen:

- Anforderungsanalyse
- Systementwurf und Modellierung
- Kodierung
- Integration und Validation

In jeder Phase ist die Bearbeitung zu dokumentieren.

Anforderungsanalyse

Die eigentliche Anforderungsanalyse entfällt aufgrund der Vorgabe der TPCW-Benchmark-Spezifikation. Vielmehr ist sich hier in die Spezifikation und Dokumentation zu vertiefen und eine Strategie zu entwickeln, um eine Datenbasis schnell, effizient und mit korrekten Daten aufzubauen.

Zeit

Diese Phase umfasst eine Woche.

Zweck

- Verteilung der Aufgaben innerhalb der Gruppe,
- Einarbeitung in die Thematik,
- Planung des allgemeinen Vorgehens und der grafischen Benutzerschnittstelle.

Aufgaben dieser Phase

Arbeiten Sie die Spezifikation durch und entwickeln Sie eine Strategie, die beschreibt, wie die benötigten Daten zu erzeugen sind. Erläutern Sie kurz, wie im Allgemeinen bei der Population der Datenbank vorzugehen ist und wie im Speziellen Bestellungen behandelt werden müssen. Berücksichtigen Sie dabei die Abhängigkeiten, die sich durch Fremdschlüsselbeziehungen oder andere Beziehungen aus dem DB-Schema ergeben und die eventuell lange Laufzeit eines Generierungsdurchgangs. Überlegen Sie, welche funktionalen Anforderungen die grafische Benutzerschnittstelle konkret erfüllen muss, und entwerfen Sie entsprechende Interface-Flow-Diagramme.

Entwickeln Sie jeweils eine Formel für die Abschätzung des Speicheraufwandes und für die Anzahl der zu erzeugenden Objekte in Abhängigkeit von der Anzahl der emulierten Browser (EB) und Produkte (Items).

Schätzen Sie den Aufwand für

EB = 1, Items = 1 000 und

EB = 10, Items = 100 000 ab.

Abgabe

System-Anforderungs-Beschreibung, die oben genannte Strategie, Interface-Flow-Diagramme, Abschätzungen und die interne Gruppenaufteilung.

Systementwurf und Modellierung

Im weiteren Verlauf ist ein System zu entwerfen, das den Anforderungen der Spezifikation genügt. Dabei ist den Prinzipien eines objektorientierten Entwurfs zu folgen.

Zeit

Diese Phase umfasst drei Wochen.

Zweck

- Erstellung eines objektorientierten System-Entwurfs
- Erarbeiten von Testfällen

Aufgaben dieser Phase

Architektur, Schnittstellen, Klassen, interne Repräsentation von Datenbankobjekten und weitere Strukturierung durch Einsatz von Paketen (respektive Java Packages) sind zu definieren und zu dokumentieren. Entwickeln Sie für jede Komponente Testfälle. Beschreiben Sie das dabei zu erwartende Verhalten. Bei der GUI ist z.B. ein Testfall „SCHLIESSEN Button wird gedrückt“ denkbar. Das erwartete Verhalten ist das Beenden der Anwendung und Freigabe aller Ressourcen.

Abgabe

System-Entwurfs-Beschreibung (mit Klassendiagramm nach UML)

Testplan (für jede Komponente 10 Testfälle mit Verhaltensbeschreibung)

Bitte besonders die Entwurfshinweise in See Entwurfshinweise beachten!

Implementierung

In dieser Phase ist der Entwurf zu implementieren. Es wird Java als Programmiersprache verwendet.

Zeit

Diese Phase umfasst höchstens vier Wochen.

Zweck

Implementierung der einzelnen Komponenten.

Aufgaben dieser Phase

Realisieren Sie den von Ihnen erstellten Entwurf unter besonderer Berücksichtigung der Einhaltung der Kardinalitäten der zu erzeugenden Objektmengen. Beschreiben Sie, wie die Kardinalitt der Menge von Bestellpositionen eingehalten wird. Benutzen Sie für die Datenbankanbindung die JDBC-Schnittstelle. Für Testläufe sollte die Konfiguration (EB = 1, Items = 1000) genutzt werden.

Benutzen Sie für die Kode-Dokumentation das JDK-Tool „javadoc“!

Testen Sie weiterhin Ihre Komponenten mit den von Ihnen beschriebenen Testfällen und protokollieren Sie alle Abweichungen vom erwarteten Verhalten.

Abgabe

kommentierter Code,

Code-Beschreibung (HTML, mit javadoc erzeugt),

Protokoll Komponententest

Integration und Validation

Die Komponenten sind zu einem funktionierenden System zu integrieren und erneut zu testen. Neben dem Funktionstest zur Überprüfung der Zuverlässigkeit ist ferner das System gegen die Anforderungen zu validieren. Geeignete Testfälle können aus der Spezifikation direkt abgeleitet werden. Dies geschieht mit Hilfe der Visualisierungskomponente.

Die Realisierung ist am Ende der Praktikumsaufgabe im Rahmen einer Systemvorstellung zu präsentieren.

Zeit

Diese Phase umfasst höchstens drei Wochen.

Zweck

Integration,

Test, Validation,

Ausführung: Population der Datenbank. EB=10, Item=100000

Aufgaben dieser Phase

Fügen Sie alle Komponenten zu einem vollständigen System zusammen. Testen Sie es zunächst mit der Konfiguration (EB = 1, Item = 1 000) und dann mit der Konfiguration (EB = 10, Item = 100 000). Messen Sie die Zeit, die zum Fällen der Datenbank nötig ist. Validieren Sie das System gegen die Anforderungen, indem Sie geeignete Anfragen an das DBMS stellen und diese visualisieren.

Testen Sie Ihr System mit geeigneten Testfällen. Arbeiten alle Komponenten fehlerfrei zusammen? Denken Sie über Transaktionskontrolle und Recovery nach! Was geschieht bei einem vorzeitigen Abbruch der Verarbeitung? Sind Sie gezwungen, von vorne zu beginnen, oder können Sie einzelne Schritte rückgängig machen, um in einem konsistenten Zustand die Verarbeitung fortzusetzen.

Welche Möglichkeiten bieten DBVS, um die Anforderungen an die spezifischen Eigenschaften der Attribute und ihrer Ausprgungen zu gewährleisten?

Abgabe

Abschluss-Dokumentation (Integrations- und Validations-Protokoll mit Testfällen und Ergebnis),

Dokumentation des Einsatzes der Visualisierungs-Komponente

System-Vorführung

3 Hinweise und Richtlinien

3.1 Abgaben & Besprechungen

Besprechungen

Zu den Besprechungen sollen immer drei von einer Gruppe kommen, und zwar aus jedem Zweier-Team einer. Die Zweier-Teams wechseln sich ab, so dass am Ende jeder ungefähr gleich oft bei einer Besprechung teilgenommen hat.

Abgaben

Für jede Aufgabe gibt es einen bestimmten Abgabetermin. Für manche Aufgaben auch einen Zwischenabgabetermin. Eine Übersicht findet sich in Abschnitt 6. Ungefähr zwei Tage nach jeder (Zwischen)abgabe wird diese besprochen.

Zwischenabgaben

Die Zwischenabgaben sind nicht bewertet. Mit ihnen soll festgestellt werden, wie gut das Praktikum bei den einzelnen Gruppen vorangeht. Bei Problemen kann so rechtzeitig eigegriffen werden. Der Abgabetermin soll eingehalten werden, es ist natürlich nicht notwendig, zu diesem Zeitpunkt schon vollständige Dokumente zu haben.

Abgaben

Die Abgaben fließen mit in die Gruppenbewertung ein. Bei einer verspäteten Abgabe werden 25% der Punkte abgezogen, bei mehr als einem Tag Verspätung 50%, bei mehr als zwei Tagen 100%.

Format

Alle Dokumente müssen in einem druckbaren Format (Postscript oder PDF) bis zum Abgabetermin an den jeweiligen Betreuer der Gruppe geschickt werden (Gruppennumer bei der Abgabe nicht vergessen!).

Nach der Besprechung werden die Abgaben im Web (Uni-intern) veröffentlicht. Details hierzu finden sich in Abschnitt 3.2.

3.2 Veröffentlichen der Abgaben im Web

Für jede Gruppe wird ein Verantwortlicher bestimmt, auf dessen Account ein eigenes lokales WWW-Verzeichnis für Dokumentationen angelegt wird. Die Einstiegsseite wird dann über einen Link auf der zentralen Seite zugänglich gemacht. Die Seiten können lokal von allen gelesen werden, sind aber nicht im gesamten WWW verfügbar. Besonders ist darauf zu achten, das die Dateirechte richtig gesetzt werden. Siehe auch Abschnitt5

Auch das Bereitstellen der Abgaben im Web gehört zum Praktikum!.

3.3 Allgemeine Entwurfshinweise

Sie sollten, wenn möglich, Design Patterns einsetzen. Einige Beispiele für Design Patterns sind hier genannt.

- Erzeugungsmuster
  - Abstrakte Fabrik (Abstract Factory)
  - Erbauer (Builder)
  - Singleton
- Strukturierungsmuster
  - Adapter
  - Fliegengewicht (Flyweight)
- Verhaltensmuster
  - Beobachter (Observer)
  - Besucher (Visitor)
  - Strategie (Strategy)

In Abschnitt 4 sind einige Verweise zu Webseiten über Design Patterns aufgelistet. Der Aufwand, sich einige Patterns mal anzuschauen lohnt sich. Es ist besser, schon bekannte und erfolgreiche Lösungen zu verwenden, als das Rad neu erfinden zu müssen. Ein Hinweis in der Dokumentation auf die Verwendung eines Patterns wird es anderen Entwicklern sehr erleichtern, sich in den Code einzuarbeiten.

Es müssen natürlich nicht alle der genannten Muster eingesetzt werden, und möglicherweise findet man ein nützliches, aber hier nicht aufgeführtes Muster, das man auch einsetzen kann.

3.4 Hinweise zu den Komponenten

DB-Zugriffskomponente und Generator-Komponente

Versuchen Sie, folgender Intention zu folgen:

- Trennung zwischen der Strategie zur Erzeugung und dem Vorgang der eigentlichen Erzeugung, sowie der tatschlichen Repräsentation von Objekten,
- Abstraktion von konkreten Tabellen des DB-Schemas und der Anfragesprache SQL,
- Kapselung der Datenbankschnittstelle, um später auch andere DBS unterstützen zu können.

Grafische Benutzeroberfläche

Die grafische Benutzerschnittstelle sollte nicht nur die Möglichkeit geben, die Verarbeitung zu starten, zu beenden und zu unterbrechen, sondern auch Informationen über deren Fortschritt zur Verfügung stellen. Dazu ist es notwendig, etwas über den Zustand anderer Klassen, bzw. ihrer Objekte zu erfahren. Beachten Sie aber die Kapselung der Objekte! Prüfen Sie daher auch die Einsatzmöglichkeit des Observer-Patterns. Java bietet dafür spezielle Unterstützung.

Visualisierungs-Komponente

Es ist eine Komponente zu erstellen, die den Inhalt der erzeugten Datenbank visualisiert. Sie kann eingesetzt werden, um die Datenbank gegen die Anforderungen aus der Spezifikation zu validieren. Hierbei ist zweierlei zu beachten.

Zum einen soll die Verteilung der Attribute überprüft werden. Es sollte daher möglich sein, statistische Information über das Minimum, das Maximum, den Mittelwert, Anzahl der von Null verschiedenen Tupel und Anzahl der voneinander verschiedenen Werte zu erhalten. Anhand einer entsprechenden grafischen Darstellung soll ferner gezeigt werden, dass die gewählte Zufallsfunktion wirklich eine Gleichverteilung der Attributwerte erzeugt, d. h. jede Ausprägung ist mit ungefähr der gleichen Anzahl vorhanden. Zur Darstellung sind geeignete Diagramme anzubieten (z. B. Torten-, Linien-, Balken- oder Blockdiagramme) und dem Benutzer zur Auswahl anzubieten. Da es gilt, eine sehr große Anzahl von Daten auszuwerten, sind sinnvolle Maßnahmen zur Skalierung vorzusehen. Auch die Auswahl eines Anzeigebereichs (Zoom-Funktion) erscheint hier sinnvoll. Stichproben, also die Auswahl eines zufälligen Tupels aus einer vorgegeben Tabelle, sollten ebenso unterstützt werden wie die gezielte Auswahl anhand einer ID. Die grafischen Elemente und Realisierungsaspekte sind dabei allerdings der grafischen Benutzeroberfläche zuzuordnen.

Metadaten-Verwaltung

An verschiedenen Stellen des Systems ist es notwendig, auf Informationen über das zugrundeliegende Schema zuzugreifen. Beispielsweise soll in der GUI bei der Visualisierung der Daten bei Auswahl einer Tabelle automatisch eine Auswahlbox mit den zugehörigen Attributen gefüllt werden. Es hat sich hier als äußerst praktisch erwiesen, diese Metadaten in einer eigenen Hilfskomponente bereitzuhalten, die bei Bedarf diese Informationen liefern kann. Da es sich dabei um Schemainformationen handelt, wird diese Komponente sinnvollerweise der DB-Komponente angegliedert. Die Implementierung kann in einer recht primitiven Weise erfolgen, jedoch sollte dabei darauf geachtet werden, dass Änderungen im Schema nachgezogen werden müssen.

3.5 Richtlinien zu Dokumentation, Entwurf und Implementierung

Grundprinzipien

Verständliche Darstellung Dazu gehört eine angemessene Notation und Struktur, sowie vollständige und konsistente Dokumente.

Minimale Darstellung „Minimal“ bedeutet in diesem Zusammenhang die Beschränkung auf das Wesentliche. Wenn z.B. etwas wunderbar in einer Strichliste dargestellt werden kann, muss man keinen seitenlangen Prosatext schreiben. Auf der anderen Seite muss natürlich darauf geachtet werden, das keine wichtigen Punkte ausgelassen werden!

Modulare Struktur Modularisierung sollte in ausreichendem Maße in den Vorlesungen besprochen worden sein. Zur Erinnerung, nochmal einige Punkte:

- Divide and Conquer
- Minimale Kopplung von Komponenten
- Maximale Kohäsion von Komponenten
- Beachtung des Geheimnisprinzips

Vertikale Verfolgbarkeit Verschiedene Dokumente unterschiedlichen Abstraktionsgrades müssen zueinander konsistent sein. Beziehungen zwischen Anforderungen, Entwurf und Realisierung müssen klar und explizit dokumentiert werden. Ein Beispiel sind Variablen- und Klassenbezeichner: in jeder Phase müssen die gleichen benutzt werden.

Horizontale Verfolgbarkeit Verschiedene Sichten innerhalb einer Abstraktionsebene müssen nachvollziehbar und konsistent sein. (Beispielsweise muss im Klassendiagramm und in der Beschreibung des Verhaltens Namenskonformität herrschen.)

Dokumentation der Verifikation
- Dokumentation der Verifikation des Entwurfs gegen die Anforderungen.
- Dokumentation der Verifikation des Programmkodes gegen den Entwurf.

Dokumentation der Validation Dokumentation der Validation des ausführbaren Systems gegen die Anforderungsbeschreibung (Spezifikation)

Dokumentation des Entwurfs

- Es wird ein objektorientierter Entwurf erwartet.
- Es ist ein Klassendiagramm zu erstellen und eine Darstellung nach UML zu wählen.
- Es sind die Schnittstellen der Klassen zu definieren.
- Es ist die Funktion einer Klasse und das Verhalten ihrer Methoden kurz zu erläutern.
- Die Abhängigkeiten zwischen einzelnen Komponenten bzw. Klassen sind zu dokumentieren.
- Die System-Entwurfs-Beschreibung sollte kurz, aber prägnant sein und alles wichtige enthalten. Implementierungsdetails haben im Entwurf nichts zu suchen.

Dokumentation und Formatierung des Codes

Es ist der Quelltext in einer für das Verständnis hinreichenden Art und Weise zu dokumentieren. Der Quelltext ist so mit Anmerkungen zu versehen, da mit Hilfe des JDK-Tools „javadoc“ automatisch eine Dokumentation in HTML erstellt werden kann. Diese Dokumentation ist im lokalen WWW-Verzeichnis dem Betreuer zugänglich zu machen. Die Dokumentation beschränkt sich nicht nur auf die Signatur, sondern bezieht sich auch auf Anweisungen innerhalb von Methoden. Es sollen wichtige Stellen im Quelltext erläutert werden (beispielsweise Fallunterscheidungen, Schleifen o. ä.), um die Lesbarkeit zu erhöhen.

Die Java Code Conventions (Verweis in Abschnitt 4 geben Richtlinien vor, wie bei der Formatierung und Dokumentation von Code vorgegangen werden soll. Es muss sich nicht zwingend an dieses Format gehalten werden, jedoch soll das Code-Format einer jeden Gruppe konsistent sein!

Änderungen in späteren Entwicklungsphasen

In diesem Praktikum wird nach dem Wasserfallmodell vorgegangen, d.h. Entscheidungen aus früheren Phasen können nicht mehr Rückgängig gemacht werden. Diese Handhabung ist etwas zu strikt, und kann das weiterkommen unnötig erschweren, falls ein Fehler entdeckt wird.

Wenn in einer späteren Phase (z.B. Implementierung) schwere Fehler oder Probleme auftreten, die durch Entscheidungen in einer früheren Phase (z.B. Entwurf) resultieren, ist es unter Umständen möglich, hier eine Korrektur zu machen. Diese Korrekturen müssen dokumentiert werden und sind vorher mit dem Betreuer abzusprechen.

4 Materialien

Bemerkung: Dieser Abschnitt ist auf Papier recht nutzlos. Auf der Homepage des SWP-Praktikums gibt es die HTML-Version mit Links (http://wwwdvs.informatik.uni-kl.de/courses/praktika/swp/2003/4_Materialien.html) und eine PDF-Version, auch mit Links (http://wwwdvs.informatik.uni-kl.de/courses/praktika/swp/2003/materialien/aufgabenbeschreibung-swp.pdf).

4.1 Folien und Aufgabenbeschreibungen, Material fürs Praktikum

Aufgabenbeschreibung

Folien der 1. Besprechung

HTML-Template für Online-Doku

JDBC-Treiber für Informix

Informix SQL Tutorial

Informix SQL Syntax

Informix SQL Referenz

Informix JDBC Guide

Informix DBAccess Guide

TPC-W-Schema

Daten für die `COUNTRY`-Tabelle

TPC-W Spezifikation Version 1.8 Die vollständige Spezifikation des TPC-W-Benchmarks. Zur Bearbeitung der Praktikumsaufgaben werden nur die Kapitel 1 und 4 benötigt.

Java Code Conventions Richtlinien und Tipps zur Formatierung von Java-Code, Benennung von Klassen und Variablen usw.

4.2 Tools

Dia – Diagram creation program Zum Erzeugen von UML-Diagrammen

TCM – Toolkit for Conceptual Modeling Zum Erzeugen von UML-Diagrammen

4.3 Websites, Dokumentationen

HTML

Selfhtml Ausführliches Online-Handbuch für HTML, JavaScript, CSS und mehr.

TPC-W

Homepage des TPC-W-Benchmarks

Java

Java-Homepage

Java SDK 1.4.1 Dokumentation Die Java-Dokumentation bei Sun. Zum schnelleren Nachschlagen entweder runterladen, oder auf der `roulett` lokal lesen. Dort finden sich auch verschiedene Tutorials und andere Infos.

Java API-Dokumentation Lokale (`roulett`) API-Dokumentation.

Java Programming Resources Mehr Material zu Java.

JDBC

JDBC Homepage

API Dokumentation In das J2SDK integriert.

Design Patterns

Design Patterns (V. Huston) Beschreibungen von vielen Patterns, mit C++ und Java Demos und weiterführenden Links.

Design Patterns Java Companion

Patterns Home Page (Hillside)

Patterns (Cetus)

UML

UML (Cetus)

Javadoc

Javadoc Homepage (Sun)

5 Systemumgebung

Zum Bearbeiten des Praktikums steht auf dem SCI-Cluster jedem Praktikanten ein Account zur Verfügung. Alle Mitglieder einer Praktikumsgruppe sind in der gleichen Unix-Gruppe, so dass die Zusammenarbeit problemlos verlaufen kann.

5.1 Rechte

Es wird empfohlen, sich das UNIX-Zugriffsrecht-System für Benutzer und Gruppen anzusehen. Auf die Dokumente einer Gruppe sollte nur die Gruppe selbst Zugriff haben, abgesehen von den Web-Veröffentlichungen. Vermutlich ist es jedoch sinnvoll, zumindest an einigen Punkten der Gruppe auch Schreibrechte zu geben.

5.2 Werkzeuge

Zum Zugriff auf die DB wird dbaccess verwendet.

Für die Diagrammerstellung ist tcm installiert. Vor Benutzung muss eine Umgebungsvariable gesetzt werden:

setenv TCM_HOME /opt/tcm

Am besten trägt man das in die /.cshrc-Datei ein.

5.3 Web-Veröffentlichungen

Jede Gruppe muss dem Betreuer ein Web-Verzeichnis, in dem die veröffentlichten Aufgaben liegen, mitteilen. Dazu muss im Home-Verzeichnis eines beliebigen Gruppenmitglieds ein Unterverzeichnis angelegt werden.

ACHTUNG: Um den Zugriff darauf zu erlauben, muss das Homeverzeichnis und das Web-Verzeichnis ein Durchgreifrecht für alle haben (chmod a+x /www). Die Dateien darin müssen selbstverständlich lesbar für alle sein.

Die Webseiten sind erreichbar unter:

http://roulett:8888/swp/gbis/swpgXX

6 Terminplan SWP 2003

6.1 Termine

Montag, 28. April, 15:30, 46/280	Erste Besprechung
Montag, 5. Mai, 10:00	Abgabe Aufgabe 1
~ 7. Mai	Besprechung Aufgabe 1
Mittwoch, 14. Mai, 10:00	Zwischenabgabe Aufgabe 2
~ 16. Mai	Zwischenbesprechung Aufgabe 2
Montag, 26. Mai, 10:00	Abgabe Aufgabe 2
~ 28. Mai	Besprechung Aufgabe 2
Montag, 2. Juni, 17:15, 46/220	Testat
Montag, 30.Juni, 10:00	Abgabe Aufgabe 3
~ 2. Juli	Besprechung Aufgabe 3
Mittwoch, 9. Juli, 10:00	Zwischenabgabe Aufgabe 4
~ 11. Juli	Zwischenbesprechung Aufgabe 4
Montag, 21. Juli, 10:00	Abgabe Aufgabe 4
~ 23. Juli	Besprechung Aufgabe 4
24. Juli	Einzelkolloquium
25. Juli	Vorführung

6.2 Übersicht

Woche	Phase
18	Anforderungen
19	Entwurf
20	Entwurf
21	Entwurf
22	Implementierung
23	Implementierung
24	Pfingstferien
25	Implementierung
26	Implementierung
27	Integration und Validation
28	Integration und Validation
29	Integration und Validation
30	Vorführung, Kolloquium
31	Frei

6.3 Erläuterungen

Besprechungen sind immer ca. 2 Tage nach Abgabe. Zu jeder Besprechung kommt genau einer aus jedem Team, d.h. 3 Mann pro Gruppe. Jeder soll möglichst gleich oft erscheinen.