

Studenten Entwickeln Universelles Hochschulinformationssystem: Erfahrungen aus einem Softwaretechnik-Praktikum

Wolfgang Bartsch, Klaus Bergner,
Rudolf Hettler, Barbara Paech

Zusammenfassung

Im Sommersemester 1994 wurde an der Technischen Universität München erstmals das Praktikum „Entwurf großer Systeme: Softwaretechnik“ durchgeführt. Ziel der Veranstaltung war, die teilnehmenden Studierenden anhand eines echten Projekts in die ingenieurmäßige Software-Entwicklung im Team einzuführen und mit objektorientierten Analyse- und Designtechniken vertraut zu machen. Im folgenden beschreiben wir unsere Erfahrungen mit der objektorientierten Entwicklungsmethode „Object Modeling Technique“ von J. Rumbaugh et al. [Rum91] und gehen auf die Organisation der Teamarbeit, die Motivierung der Teilnehmer und den Werkzeugeinsatz ein.

1 Das Projekt

In dem Praktikum wurde ein Informationssystem für die TU München erstellt, das den Zugriff auf Informationen etwa zu Personen (z.B. Telefon- und Raumnummer), Vorlesungen (z.B. Zeit und Ort) und angebotenen Diplomarbeiten (z.B. Thema und nötige Vorkenntnisse) ermöglicht. Viele dieser Daten lagen zwar bereits in elektronischer Form vor, ihre Eingabe und Verwaltung war allerdings nicht automatisiert und die Ausgabe- und Suchmöglichkeiten waren unzureichend.

Die hauptsächlichen Schwierigkeiten bei dem Projekt bestanden nicht so sehr in der Überwindung technischer Probleme (diese traten meist dort auf, wo die Beschränkungen und Eigenheiten der verwendeten Basissoftware überwunden werden mußten), sondern vor allem in der Beherrschung der Komplexität eines großen Systems und in der Organisation der Teamarbeit.

Die acht Praktikumsteilnehmer (es hatte sich keine Frau gemeldet) kamen überwiegend aus dem fünften Semester und hatten größtenteils keine speziellen Vorkenntnisse im Bereich Softwaretechnik. Als Kunden für das System traten drei Parteien auf, die jeweils durch einen Ansprechpartner vertreten wurden: die Studierenden (als Informationsnachfrager), die wissenschaftlichen Angestellten (als Anbieter) und die lokale Rechnerbetriebsgruppe, die das System nach seiner Fertigstellung übernehmen wird. Außerdem war am Rande die zentrale Verwaltung der TU beteiligt, die die Informationen zu Räumen und Zeiten der Veranstaltungen liefert.

Das System ist in C++ auf der Grundlage des verteilten Hypermedia-Systems „World Wide Web“ [WWW] implementiert. Mit dessen Dokumentenbeschreibungssprache *HTML* [HTML] läßt sich leicht eine einfache graphische Oberfläche mit Hyperlinks und verschiedenen Dialogelementen aufbauen, die mit Hilfe von Programmen wie *Mosaic* oder *Netscape* genutzt werden kann. Für die Datenhaltungskomponente gab es keine strikten Vorgaben: Hier mußte im Verlauf des Projekts ein geeignetes Datenbanksystem ausgewählt werden.

Bei der Entwicklung wurde neben üblichen Werkzeugen (GNU-C++, \LaTeX , Versionsverwaltung RCS) das CASE-Werkzeug „Software through Pictures“ [StP93] zur Verfügung gestellt, das die im Projekt verwendete Methode „Object Modeling Technique“ [Rum91] unterstützt. Als Dokumentationsmedium diente das WWW, in das die Teilnehmer ihre Dokumente einhängen konnten.

Am Ende des Semesters war neben umfangreichen Analyse- und Designdokumenten ein Prototyp fertig, mit dem über eine graphische Oberfläche personenbezogene Informationen aus der Datenbank abgefragt werden können (eine Dialogseite daraus zeigt Abbildung 1). Er dient als Grundgerüst für die weitere Implementierung, die von den Teilnehmern des Praktikums stufenweise im Rahmen von Fortgeschrittenenpraktika in den Semesterferien durchgeführt wird.

2 Prozeß und Methode

Als Entwicklungsmethode im Praktikum wurde die „Object Modeling Technique“ von J. Rumbaugh et al. [Rum91] ausgewählt, die sich in einem ähnlichen Projekt schon bewährt hatte [Bru92] und mit der wir bereits (überwiegend positive) Erfahrungen aus Diplomarbeiten und Fortgeschrittenenpraktika hatten. Am Lehrstuhl existiert außerdem mit „Software through Pictures“ [StP93] ein CASE-Werkzeug, das OMT unterstützt und für die verwendete Entwicklungsplattform angeboten wird.

Machbarkeitsstudie Um Erfahrungen mit dem Werkzeug und mit den Mechanismen und Eigenheiten der Basissoftware zu gewinnen, wurde vor Praktikumsbeginn ein Fortgeschrittenenpraktikum durchgeführt, in dem die Bearbeiterin eine einfache

TIS **Filter für die Personensuche**

Benutzer: infosys, Status: ---

[Einstiegsseite](#) [Hilfe](#)

Vorname: Nachname:

e-mail:

Beziehung zur Hochschule: Student
 wissenschaftlicher Mitarbeiter
 Professor
 Angestellter
 Externer

Geschlecht: männlich weiblich

Abbildung 1: Dialogseite aus dem Prototyp

Terminatenbank mit wenigen Klassen auf der Grundlage des WWW erstellt. Dabei konnten einige technische Detailprobleme aufgedeckt werden und es ergab sich ein Konzept für eine Systemarchitektur, das die Vorgabe zur Aufteilung der einzelnen Entwicklungsteams lieferte (s. Abschnitt 3).

Projekt- und Zeitplanung Vor dem Praktikum wurde von den Betreuenden ein Projektplan mit einer groben Zeiteinteilung erstellt, in dem für die Analyse etwa vier Wochen und für das Design etwa zwei Wochen veranschlagt wurden; die restliche Zeit war für Implementierung und Test reserviert. Diese Zeitplanung konnte bei der Analyse in etwa eingehalten werden, beim Design und bei der Implementierung ergaben sich Verzögerungen, die zum größten Teil auf Schwierigkeiten der Entwickler mit der Basissoftware und mit den Entwicklungswerkzeugen zurückzuführen waren. Am Ende der Analyse wurde ein Review mit den Kundenvertretern durchgeführt, nach dem Design eines mit den Vertretern der Rechnerbetriebsgruppe.

Erweiterungen von OMT OMT unterstützt die Geschäftsvorfallmodellierung, die Benutzungsschnittstellenentwicklung und die Testvorbereitung nur unzureichend. Wir haben deshalb die folgenden zusätzlichen Techniken eingeführt:

- Textuelle *Aufgabenbeschreibungen* dienen zu Beginn der Analyse dazu, die einzelnen Aufgaben, die das System unterstützen soll (wie z.B. das Anbieten einer Diplomarbeit oder die Abfrage von Zeitplänen zu Vorlesungen) im Hinblick auf nötige Eingaben, erwünschte Ergebnisse, Zeitbedingungen und Priorität zu beschreiben. Sie wurden von den Entwicklern dazu verwendet, die Organisationsabläufe innerhalb der TU festzuhalten und die Kundengespräche zu strukturieren. Eine typische Aufgabenbeschreibung zeigt Abbildung 2.
- Das *Prototyping der graphischen Oberfläche* sollte den Kunden schon früh einen Eindruck von dem System vermitteln und half den Entwicklern, sich mit der Hypertext- und Dialogbeschreibungssprache *HTML* vertraut zu machen. Das dynamische Verhalten der Oberfläche und die Navigationsmöglichkeiten zwischen den unterschiedlichen Dialogfenstern wurde in *Interaktionsdiagrammen* [Den91] spezifiziert. Sie können mit der Notation des dynamischen Modells von OMT notiert werden, beschreiben aber nicht das Verhalten einer einzelnen Klasse, sondern der gesamten Oberfläche.
- Die (textuelle) Spezifikation von *Testfällen* mit konkreten Ein- und Ausgabedaten half den Entwicklern, schon früh eventuelle Probleme mit Datenformaten zu erkennen und Konsistenzkriterien aufzustellen.

Im wesentlichen haben sich sowohl die von OMT bereitgestellten Techniken als auch unsere Erweiterungen bewährt. Sie sind genügend mächtig, um wichtige Aspekte eines Systems anschaulich zu beschreiben und konnten im Praktikum bereits nach sehr kurzer Zeit sinnvoll von den Teilnehmern eingesetzt werden, da sie leicht verständlich und somit gut vermittelbar waren. Dennoch ließ die eingesetzte Methode eine Reihe von Wünschen offen:

- OMT besitzt nur ein *schwach ausgeprägtes Vorgehensmodell*. Das Fehlen von klaren Vorgaben bezüglich der Vorgehensweise führte besonders im Design zu Unsicherheiten im Entwicklungsteam. Vielleicht ist eine Methode mit einem strengen Vorgehensmodell einem Praktikum eher angemessen; es besteht jedoch die Gefahr, daß sie die Flexibilität der Teilnehmer zu stark einschränkt und den Einarbeitungsaufwand unverhältnismäßig erhöht.
- Bei der *Beschreibung der Architektur eines Systems* während der Designphase ist es vorteilhaft, sich das System in Schichten aufgeteilt vorzustellen (etwa Datenhaltung, Anwendungskern und Benutzungsschnittstelle). OMT enthält

Diplomarbeit vorbereiten

Aufgabe: Der Student sucht Informationen über interessante Diplomarbeitsthemen der Lehrstühle oder erstellt selbst eine Themenvorgabe.

Ist-Zustand

- **Bearbeiter:** Student
- **Ressourcen:** Aushänge bei den Lehrstühlen und anderen schwarzen Brettern. Initiativen von Privatfirmen, d.h. persönliche Kontakte.
- **Zeiten, Termine:** jederzeit abrufbar, jederzeit aktualisierbar
- **Bisheriges Vorgehen:** Der Student erfaßt das Angebot anhand der Aushänge bei den Lehrstühlen. Er betrachtet dabei hauptsächlich die Themen der Lehrstühle, die die ihn interessierende Fachgebiete behandeln.
Angebote von Firmen sind stets einem Lehrstuhl zugeordnet und finden sich dort wieder, wenn die Firma selbst den Kontakt zur Uni bereits hergestellt hat. Aufgabensteller ist stets ein Professor. Bei einem vorgegebenen Thema muß der Student sein Interesse dem Lehrstuhl erst noch kundtun und nach Absprache mit dem Professor und dem Betreuer das Thema bei Herrn Lahner (Funktionsbezeichnung?) anmelden.
Alternativ kann der Student auch auf Initiative von Seiten einer Firma den Kontakt zu den Lehrstühlen herstellen und das Thema formulieren. Er muß in diesem Fall einen Professor finden, der die Arbeit bewerten möchte. Dazu nutzt er die Sprechstunden der Professoren.
- **Ergebnisse:** Der Student erhält eine Menge von interessanten Diplomarbeitsthemen der einzelnen Lehrstühle oder bereitet selbst ein Thema vor, zu dem er einen Aufgabensteller hat.
- **Priorität:** Kernsystem [x], Erweiterung [], Anregung []

Mögliche Lösungen: Sämtliche Diplomarbeitsthemen werden zentral im System gespeichert und enthalten folgende Angaben:

- Titel
- Lehrstuhl und Dozent
- Inhaltsangabe
- Zielgruppe und Voraussetzungen, z.B. Vorlesungen

Sein Interesse kann der Student analog wie bei einer Anmeldung dem Lehrstuhl innerhalb des Systems vermitteln. Vorschläge seitens der Studenten bedürfen eines intensiveren, persönlicheren Kontakts zum Professor, so daß diese besser nicht in das System integriert werden.

Abbildung 2: Beispiel für Aufgabenbeschreibung

jedoch außer einigen sehr allgemeinen Hinweisen keine Konzepte und Notationsmittel, die diese übergeordnete Sichtweise unterstützen. Für die Teilnehmer war es deshalb beim Entwurf des Systems schwierig, die einzelnen Klassen aus dem Objektmodell in die Systemarchitektur einzuordnen.

- OMT besitzt im dynamischen Modell keine Möglichkeit, *objektübergreifende Vorgänge* angemessen zu modellieren. Zwar gehen die Lebenszyklen der einzelnen Objekte aus den Zustandsdiagrammen deutlich hervor, jedoch existiert kein Formalismus, mit Hilfe dessen sich umfassend und übersichtlich beschreiben ließe, in welcher Reihenfolge bestimmte Operationen aus *verschiedenen* Klassen aufgerufen werden können. (Das funktionale Modell ist dazu nicht geeignet, da hier nur Datenflüsse ohne jede zeitliche Reihenfolge dargestellt werden können. Insgesamt war die funktionale Modellierung in unserem Anwendungsbeispiel von sehr geringem Nutzen und wurde deshalb mit fortschreitender Projektdauer mehr und mehr vernachlässigt.)
- Bei der Objektmodellierung wäre es vorteilhaft, allgemeine *statische Integritätsbedingungen für die Objektklassen* aufschreiben zu können. OMT sieht jedoch keine Bedingungen vor, die über die üblichen Kardinalitäten von Beziehungen (1:1, 1:n, . . .) hinausgehen.

3 Organisation und Betreuung

Organisationsstruktur und Selbstorganisation Die Betreuenden strebten an, daß sich die Entwickler selbständig Gedanken über die Arbeitsteilung machten und sich weitgehend selbst organisierten. Von Anfang an wurde deshalb klar gemacht, daß die Organisationsstruktur im Verlauf des Praktikums jederzeit an die jeweiligen Erfordernisse angepaßt werden könne. Abbildung 3 zeigt die zu Beginn von den Betreuenden eingeführte Organisationsstruktur. Ihr zufolge war jeder Entwickler einem *Team* und (mindestens) einer *Gruppe* zugeordnet. Die Teams waren jeweils für die vollständige Entwicklung einer Systemkomponente (durchgehend von der Anforderungsanalyse bis zur Fertigstellung) verantwortlich. Die Gruppen dienten zur teamübergreifenden Kommunikation: Hier wurden Konzepte (z.B. Programmierrichtlinien, Standards für die Dokumentation, . . .) entwickelt, die in den Teams einheitlich umgesetzt werden sollten (vgl. auch [Gro94]).

Im Verlauf des Praktikums zeigte es sich, daß die Einteilung in Teams (also in die eigentlich produktiven Einheiten) deutlich beständiger war als die Gruppeneinteilung. Wir vermuten, daß unser Projekt mit acht Entwicklern so klein war, daß teamübergreifende Koordinationsaufgaben nicht den Stellenwert erlangten, den sie in größeren Projekten besitzen. Darüberhinaus hatten wir den Eindruck, daß die Stu-

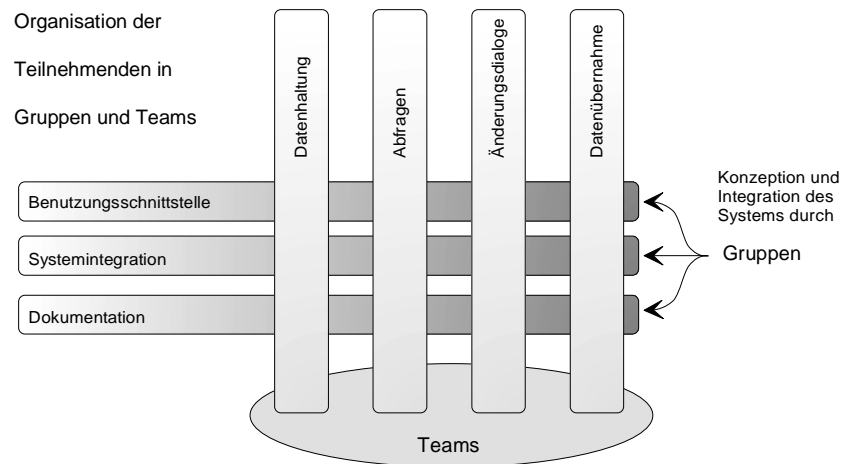


Abbildung 3: Anfängliche Organisationsstruktur im Projekt

dierenden gerade mit den Koordinationstätigkeiten Schwierigkeiten hatten und sich davor scheuten, die Verantwortung für konkrete Vorgaben an die übrigen Entwickler zu übernehmen. Im Gegensatz dazu identifizierten sich die Teilnehmer bereits nach kurzer Zeit stark mit ihren jeweiligen Teamaufgaben, was bei der Aushandlung der Schnittstellen zwischen den Systemteilen im Objektdesign und in der Implementierung manchmal zu kleinen Reibereien führte.

Kommunikation Während des Ablaufs des Praktikums wurde wöchentlich montags ein Projekttreffen durchgeführt, an dem jeweils alle Entwickler teilnehmen mußten. Moderator und Protokollführer dieser Meetings waren jeweils zwei (wöchentlich wechselnde) Entwickler.

Alle derartigen Treffen fanden in einem eigenen Projektraum statt, der während des Semesters ausschließlich für das Praktikum reserviert war und sich in unmittelbarer Nähe der Entwicklungsrechner befand. Der Projektraum wurde auch für die Arbeitstreffen der einzelnen Gruppen und Teams benutzt und ermöglichte es, daß der Informationsfluß innerhalb des Projekts zum größten Teil auf der direkten, mündlichen Kommunikation basierte.

Als asynchrones Kommunikationsmittel zwischen den Projektbeteiligten haben sich im WWW archivierte Mailing-Listen bewährt. Installiert wurde je eine pro Team und

Gruppe, eine, die nur die Betreuenden erreichte, sowie eine, die alle Projektbeteiligten enthielt.

Vorbildung, Wissenserwerb Begleitend fand zum einen die Vorlesung „Entwicklung großer Programmsysteme“ mit Übungen statt und zum anderen im Rahmen des Praktikums eine sogenannte „Informationsveranstaltung“, in der die Betreuenden jeweils die für das Praktikum gerade erforderlichen Techniken und Konzepte vermittelten (vgl. [Sch93]). Diese Veranstaltungen dienten auch als Forum für Kurzvorträge der Teilnehmer, in denen sie beispielsweise erarbeitete Lösungsansätze vorstellten oder das Arbeiten mit bestimmten Werkzeugen beschrieben.

Unsere Erfahrung hat gezeigt, daß ein Mindestmaß an Vorbildung im Bereich der Softwaretechnik dringend nötig ist: In Zukunft werden wir Vorkenntnisse in diesem Bereich als „Einstellungsvoraussetzung“ fordern. Ideal wäre, wenn alle Teilnehmer eine besonders auf das Praktikum abgestimmte, einsemestrige Softwaretechnikvorlesung mit intensiven Übungen besucht hätten.

Der Versuch, während des laufenden Praktikums innerhalb der „Informationsveranstaltungen“ Wissen zu vermitteln, ist am Zeitdruck gescheitert. Die Informationsveranstaltungen wurden deshalb in der zweiten Hälfte des Praktikums zugunsten zusätzlicher Projektmeetings aufgegeben.

Betreuung Es stellte sich heraus, daß die Betreuung im Vergleich zu anderen Lehraufgaben wie etwa Tutorübungen sehr zeitaufwendig war: Besonders in der Vorbereitungsphase und während der ersten Hälfte des Semesters waren meist mehr als 25 Wochenstunden Arbeit pro Person erforderlich.

Die Betreuenden hatten in dem Praktikum zwar begleitende Funktion, jedoch nie die Funktion von *Projektmitarbeitern*. Das lag zum einen in den zeitlichen Möglichkeiten der einzelnen Betreuenden begründet, zum anderen sollten so die Praktikusteilnehmer zur Übernahme von möglichst viel Eigenverantwortung bewegt werden. Dieses Konzept war jedoch nur eingeschränkt erfolgreich: An kritischen Stellen (insbesondere während der Designphase) mußten die Betreuenden mit „Entscheidungen von oben“ das Projekt vorantreiben. Als sehr nützlich stellte sich dabei heraus, daß ein Betreuer bereits Projekterfahrung in der freien Wirtschaft hatte.

Eine sinnvolle Alternative bei der Rollenverteilung scheint zu sein, ein oder zwei Betreuende in leitender Funktion vollzeit am Projekt mitarbeiten zu lassen (in der Rolle eines Projektleiters oder „erfahrenen Kollegen“). Allerdings dürfte es im täglichen Betrieb einer Universität nicht einfach sein, wissenschaftliches Personal für eine derartige Aufgabe komplett freizustellen.

4 Motivation der Teilnehmer

Die Entwicklung eines „echten“ Systems im Rahmen eines Praktikums bringt eine Reihe von Problemen mit sich, die typisch für die Durchführung von Software-Projekten sind. Dazu zählen insbesondere harte Termine, schwierige Kunden, mangelhafte Werkzeuge und vorhandene Basis-Software. Wesentlich erschwert wird die Projektdurchführung darüberhinaus auch noch dadurch, daß die studentischen Entwickler meist keinerlei praktische Erfahrungen im Bereich der Softwaretechnik haben und der Zeitrahmen extrem knapp bemessen ist (ein Praktikum an der TU München darf beispielsweise nicht länger als ein Semester dauern). Als stärkste Motoren für den Projektfortschritt verbleiben die hohe Motivation und ein guter Teamgeist der Projektbeteiligten. Sie wurden durch folgende Einflußfaktoren und Maßnahmen gefördert:

Vorauswahl der Teilnehmenden Die Ankündigung des Praktikums war im Stil einer Stellenanzeige gehalten (s. Abbildung 4). Sie erwähnte bewußt, daß „Zeit- und Arbeitsaufwand höher als beim durchschnittlichen Praktikum liegen“ würden (vgl. [Par66]) und verwies auf ein elektronisches Bewerbungsformular im WWW. Hier mußten die Bewerber unter anderem eingeben, warum sie sich gerade für dieses Praktikum interessierten. Wenn die Interessenten diese Hürde genommen hatten, wurden sie in Gruppen von jeweils drei bis vier Personen zu einem Vorstellungsgespräch eingeladen und dort nochmals auf die Besonderheiten des Praktikums aufmerksam gemacht. Insgesamt hat sich das geschilderte, „harte“ Vorgehen sehr bewährt: Zwar belegten letztlich nur acht Teilnehmer das Praktikum (vier weniger als eigentlich vorgesehen), diese waren aber sehr engagiert.

Interessante und praxisnahe Themenstellung Der maßgebliche Anreiz zur Teilnahme an dem Praktikum war sowohl für Studierende wie für die Betreuenden, daß keine „Wegwerf-Software“ zu erstellen war, sondern ein Informationssystem, das tatsächlich zum Einsatz kommen wird (siehe auch [Bru92], [Bru94], [Gro94]), und das die studentischen Entwickler selbst vorteilhaft nutzen können. Der Einsatz von „State-of-the-Art“-Methoden (OMT im Verbund mit einem CASE-Werkzeug) und Sprachen (C++, ESQ, HTML), deren Beherrschung Vorteile für die spätere berufliche Karriere verspricht, wirkte zusätzlich motivierend.

Gutes Betriebsklima Vor dem Projektstart (d.h. noch in den Semesterferien) wurde ein *Kickoff-Meeting* durchgeführt, in dem den studentischen Entwicklern und den veranstaltenden Professoren die grobe Organisation und das Projektziel vorgestellt wurden, und in dem das Projektteam zusammengeschmiedet werden sollte. Beim darauffolgenden gemeinsamen Abendessen lernten sich Studenten und Betreuende persönlich näher kennen. Auf diese Weise wurde erreicht, daß die Stimmung im Team

Die Lehrstühle Informatik IV (Prof. Broy) und Informatik XII (Prof. Endres) suchen im Sommersemester 1994 für das

Wahlpflichtpraktikum „Softwaretechnik“ hochmotivierte Interessentinnen und Interessenten.

Es ist geplant, im Rahmen des Praktikums ein Informationssystem für die TUM zu erstellen, mit dessen Hilfe Informationen über Vorlesungen, Diplomarbeiten, Praktika, Angestellte und Studierende komfortabel eingegeben und nach unterschiedlichen Kriterien abgerufen werden können. Die Implementierungssprache wird voraussichtlich C++ sein, außerdem werden spezielle Sprachen für Datenbank- und Benutzungsoberflächenprogrammierung verwendet.

Wir veranstalten das Praktikum Softwaretechnik in dieser Form zum ersten Mal. Unser Ziel dabei ist, die etwa fünfzehn teilnehmenden Studierenden anhand eines echten Projekts in die ingenieurmäßige Software-Entwicklung im Team einzuführen. Das Projekt gilt dann als erfolgreich abgeschlossen, wenn ein funktionierendes, dokumentiertes System rechtzeitig an den Kunden ausgeliefert werden konnte.

Wir bieten Ihnen damit die Möglichkeit, an der Universität **Projekterfahrung** zu sammeln, die für Ihre berufliche Qualifikation sehr wertvoll sein wird. Wir fordern aber hohes Engagement, da **Zeit- und Arbeitsaufwand höher als beim durchschnittlichen Praktikum** liegen werden: Alle Mitwirkenden haben jeweils die Verantwortung für einen Teil des Projekts und sind verpflichtet, an wöchentlich zwei dreiviertelstündigen Zentralübungen sowie an weiteren regelmäßigen Treffen der einzelnen Entwicklungsgruppen teilzunehmen.

Wenn Sie diese Aufgabe reizt, bewerben Sie sich bitte bis zum **18. Februar** mit Hilfe des angefügten Eingabeformulars. Wir werden Sie dann per e-mail oder telefonisch zu einem Vorstellungsgespräch einladen.

Abbildung 4: Die „Stellenausschreibung“

gleich zu Beginn hervorragend war. Auch im Projektverlauf trafen sich Betreuende und Entwickler bei privaten Gelegenheiten.

Realitätsnahe Praktikumsdurchführung

- Die Ansprechpartner der Entwickler waren „echte“ Kunden. Ihre Wünsche waren oft schwer umsetzbar und die Kommunikation mit ihnen gestaltete sich manchmal schwierig, da es Termin- und Verständnisprobleme gab. Obwohl die Kunden ihre Rolle gut ausfüllten, hätte sich die Motivation der Entwickler

bei den Kundenkontakten wohl noch steigern lassen, wenn außeruniversitäre Kunden beteiligt gewesen wären.

- *Projektpläne* mußten erstellt und vor allem eingehalten werden, um den Projekterfolg nicht zu gefährden.
- Durch *Reviews* wurde der Kontakt zwischen Entwicklern und Kunden gefestigt und die Entwickler erhielten durch das Feedback die Sicherheit, daß die Entwicklung den richtigen Lauf nahm. Zusätzlich führte der Abgabedruck der Meilensteinergebnisse jeweils zu einer spürbar höheren Produktivität des Teams.
- Zur Entlastung der Entwickler wurde ein Werkstudent als *Projektassistent* eingestellt, dem die Praktikumssteilnehmer selbständig Arbeiten übergeben konnten. Er war während des Praktikums im wesentlichen mit Routinearbeiten wie Literaturrecherchen, Kopierarbeiten und der Pflege eines Teils der Projektdokumente beschäftigt.

„Belohnungen“ Das Praktikum verlangte von den Teilnehmern einen im Verhältnis zu vergleichbaren Veranstaltungen sehr hohen Zeitaufwand. Deshalb wurden den Studierenden neben den üblichen Praktikumscheinen *Empfehlungsschreiben der veranstaltenden Professoren* für eine erfolgreiche Beendigung des Projekts in Aussicht gestellt. Diese enthalten — ähnlich einem Arbeitszeugnis — eine individuelle Beurteilung eines jeden Teilnehmers und wurden von den Betreuenden gemeinsam erstellt. Dabei wurden sowohl die fachlichen Fertigkeiten (z.B. Verständnis und Anwendung der eingesetzten Methode, Einsatz der verwendeten Sprachen) als auch die Fähigkeit zur Teamarbeit berücksichtigt. Obwohl die Betreuenden nicht an der eigentlichen Projektarbeit teilnahmen, waren sie soweit mit dem Projekt vertraut, daß die Erstellung individueller Bewertungen problemlos möglich war.

Außerdem wurde den Teilnehmern angeboten, im Rahmen von *Fortgeschrittenenpraktika* weitere Stufen des Projekts zu realisieren. Für die Studierenden ergibt sich dadurch der Vorteil, die sonst nötige Einarbeitung zu sparen. Im Vergleich zu anderen Studierenden bekommen sie also die Gelegenheit, einen Pflichtenchein mit erheblich weniger Aufwand zu erwerben. Alle Projektteilnehmer haben dieses Angebot angenommen.

5 Werkzeugeinsatz

Zur Unterstützung der Arbeit mit der Methode OMT war das CASE-Werkzeug „Software through Pictures“ [StP93] vorgesehen. Dieses Werkzeug stellt während der Ana-

lyse- und Designphase Editoren für die verschiedenen OMT-Techniken (*Objektmodell, dynamisches und funktionales Modell*) sowie ein Data Dictionary zur Verfügung. In der Codierungsphase generiert StP aus den erstellten Modellen C++-Codestücke, die dann den Entwicklern als Basis für die Implementierung dienen.

Das Werkzeug wurde erst kurz vor Praktikumsbeginn beschafft, eine gründliche Einarbeitung in die Möglichkeiten von „Software through Pictures“ war somit weder für die Betreuenden noch für die Teilnehmer möglich. Unter diesen Rahmenbedingungen stellte sich das Werkzeug als schwer handhabbar heraus und war somit keine Arbeitserleichterung. Die dadurch entstandene Frustration im Entwicklerteam führte dazu, daß ab dem Ende der Analysephase auf den Einsatz von StP vollständig verzichtet und die gesamte Dokumentation nach \LaTeX übertragen wurde. Es ist eine interessante Frage, ob der Mißerfolg beim Einsatz von „Software through Pictures“ nur auf die mangelnde Erfahrung der Benutzer (bzw. auf tatsächlich vorhandene Schwächen des Werkzeugs) zurückzuführen ist, oder ob CASE-Werkzeuge dieser Art ihre Produktivität erst bei Projekten einer gewissen Größe entfalten können, die deutlich über der unseres Projekts liegt.

Insgesamt hat sich, auch bei den anderen im Praktikum eingesetzten Werkzeugen, wieder einmal die Binsenweisheit bestätigt, daß man nur gut erprobte und lange bekannte Programme einsetzen sollte, wenn man unliebsame Überraschungen vermeiden möchte.

Danksagung

Allen voran danken wir den Teilnehmern des Praktikums, Peter Braun, Dirk Hansel, Sascha Molterer, Carsten Sapia, Dirk Steinkopf, Günter Teubner, Ralf Ulbig und Markus Zobel. Ohne ihre engagierte Mitarbeit und den hohen Einsatz wäre weder ein Projekt- noch ein Praktikumserfolg möglich gewesen. Besonderer Dank für die aufgewendete Zeit gebührt auch unseren „Kunden“, Ralf Steinbrüggen und Tobias Mayr. Den Professoren Manfred Broy, Ernst Denert und Albert Endres danken wir dafür, daß sie das Praktikum an der TU möglich machten und uns bei seiner Konzeption und Durchführung auf vielfältige Weise unterstützten. Danken möchten wir auch Prof. Bernd Bruegge für seine zahlreichen Anregungen und Anmerkungen im Vorfeld der Veranstaltung. Wesentlich an der Praktikumsvorbereitung mitgewirkt hat unser Kollege Bernhard Rumpel. Für die stets prompte Bereitstellung und Instandhaltung der Infrastruktur und als kompetente Ansprechpartner bei technischen Fragen gilt unser besonderer Dank Manuela Scherer und den Mitgliedern der RBG, vor allem Elmar Bartel, Achim Jung und Markus Stumpf.

Literatur

- [Bru92] B. Bruegge, J. Blythe, J. Jackson, J. Shufelt: *Object-Oriented System Modeling with OMT*, in: *Tagungsband zur OOPSLA 1992*, S. 359–376.
- [Bru94] B. Bruegge: *From Toy System to Real System Development: Improvements in Software Engineering Education*, in: H. Hußmann, B. Paech (Hrsg.): *SEUH'94*, German Chapter of the ACM, Bericht 43, Teubner-Verlag, 1994.
- [Den91] E. Denert: *Software-Engineering*, Springer-Verlag, 1991.
- [Gro94] B. Groth: *Erfahrungen aus einem objektorientiert durchgeführten Software-Engineering-Lehrprojekt*, in H. Hußmann, B. Paech (Hrsg.): *SEUH'94*, German Chapter of the ACM, Bericht 43, Teubner-Verlag, 1994.
- [HTML] D. Connolly (Ed.): *Hypertext Markup Language (HTML)*, http://www.hal.com/users/connolly/html-spec/HTML_TOC.html.
- [Par66] C. N. Parkinson: *Die engere Wahl oder Prinzipien der Begabtenauslese*, in: *Parkinsons Gesetz*, Rowohlt rororo 873, 1966.
- [Rum91] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, W. Lorensen: *Object-Oriented Modeling and Design*, Prentice Hall, 1991.
- [Sch93] K. Schneider: *Auf der Suche nach maßgeschneiderten Unterrichtsformen — das angeleitete Praktikum*, in J. Raasch, T. Bassler (Hrsg.): *SEUH'93*, German Chapter of the ACM, Bericht 38, Teubner-Verlag, 1993.
- [StP93] *Interactive Development Environments: Handbücher zu Software through Pictures*, 1993.
- [WWW] T. Berners-Lee: *World Wide Web Initiative*, <http://info.cern.ch/hypertext/WWW/TheProject.html>.

Wolfgang Bartsch (bartsch@informatik.tu-muenchen.de)

Klaus Bergner (bergner@informatik.tu-muenchen.de)

Rudi Hettler (hettler@informatik.tu-muenchen.de)

Barbara Paech (paech@informatik.tu-muenchen.de)

Institut für Informatik

Technische Universität München

80290 München