Eine vergleichende Fallstudie mit CASE-Werkzeugen für objektorientierte und funktionale Modellierungstechniken

Erik Kamsties[1], Antje von Knethen[2], Jan Philipps[3], Bernhard Schätz[3]

[1]Fraunhofer Einrichtung für Experimentelles Software Engineering,
Sauerwiesen 6, D-67661 Kaiserslautern
[2]Fachbereich Informatik, Universität Kaiserslautern,
D-67653 Kaiserslautern
[3]Fakultät für Informatik, Technische Universität München,
Arcisstraße 21, D-80290 München

- Positionspapier -

1 Einleitung

Software wird zunehmend in Produkten des Alltags wie Videorecordern und Waschmaschinen eingesetzt, aber auch im Automobilbereich, in der Luft- und Raumfahrt, in der Telekommunikation und in Medizintechnik. Mehr und mehr Funktionalität solcher Systeme wird nicht mehr durch Hardware, sondern durch Software realisiert.
Objektorientierte und funktionale Modellierungstechniken ermöglichen es, Systeme präzise zu beschreiben und helfen somit Mehrdeutigkeiten der natürlichen Sprache einzuschränken. Objektorientierte Modellierungstechniken wie ROOM oder UML/RT lösen dabei langsam einige funktionale Modellierungstechniken wie SA/RT ab. Für den industriellen Einsatz von Modellierungstechniken sind CASE-Werkzeuge unerläßlich. Das herausragende Merkmal von CASE-Werkzeugen ist die Unterstützung von Aktivitäten, die ohne Werkzeug nicht in dem Umfang möglich wären, z.B. Konsistenzüberprüfung, oder Simulation. Außerdem bieten diese Werkzeuge oft partielle oder vollständige Kodegenerierung, was Entwicklungszyklen verkürzt, und unterstützen die Verwaltung von Spezifikationen.
Ein breites Spektrum von Modellierungstechniken steht dem Entwickler zur Verfügung. Dieses Positionspapier beschreibt eine Fallstudie mit acht Modellierungstechniken und den dazugehörigen CASE-Werkzeugen. Konkret wurden als objektorientierte Modellierungstechniken OCTOPUS, ROOM, SDL und UML, sowie als funktionale Modellierungstechniken Focus [BDD+92], Petrinetze, SCR und Statecharts gewählt. Es wurden die folgenden CASE-Werkzeuge benutzt: Software through Pictures/OMT (Aonix), ObjecTime Developer (ObjecTime Systems), ObjectGEODE (Verilog), Rhapsody (I-Logix), AutoFocus [HMR+98], PEP [Gra97], SCR* [HBGL95], und Statemate MAGNUM (I-Logix).
Ziel der Fallstudie war es, den Grad der Unterstützung von Aktivitäten der Anforderungsspezifikationsphase vergleichend[1] zu bewerten. Die beiden betrachten Aktivitäten waren die Erstellung einer Spezifikation und deren Überprüfung; die Anforderungen selbst waren bereits vorgegeben. Außerdem wurde auch noch der Aufwand für die Einarbeitung in die Modellierungstechnik und das entsprechende Werkzeug bewertet.
Die Fallstudie wurde im Rahmen eines Seminars durchgeführt, daß zeitgleich an der Universität Kaiserslautern und der Technischen Universität München stattfand. Insgesamt nahmen 22 Studenten an der Fallstudie teil, die über 13 Wochen lief. Als Spezifikationsaufgabe wurden textu-

1. Vergleichend heißt, daß keine absolute quantitative Bewertung (z.B. “sehr gut” bis “mangelhaft”), sondern eine Bewertung der Modellierungstechniken und Werkzeuge relativ zueinander in qualitativer Weise erfolgt.

elle Anforderungen an das Tamagotchi vorgegeben (das Tamagotchi ist ein eingebettetes System, daß ein virtuelles Lebewesen simuliert).

2 Aufbau der Fallstudie

In der Fallstudie wurden die folgenden Fragestellungen konkret untersucht:

- Einarbeitung
  - Wie leicht kann sich ein Entwickler in die Modellierungstechnik und das Werkzeug einarbeiten? Hierzu wurde der Aufwand und Qualität der Dokumentation bewertet.
- Erstellung einer Spezifikation
  - Wie mächtig ist die Notation?
  - Wie vollständig und wie groß ist die entstandene Spezifikation?
  - Wie gut wird Teamarbeit durch Modellierungstechnik und Werkzeug unterstützt?
  - Wie hoch ist der Aufwand und was sind Probleme bei der Erstellung einer Spezifikation?
  - Wieviele und welche Art von Mängeln in informellen Anforderungen werden durch die Erstellung einer Spezifikation aufgedeckt?
  - Wie ist die Qualität der entstandenen Spezifikation?
- Überprüfung einer Spezifikation
  - Wie umfassend ist die automatische Überprüfung einer Spezifikation?
  - Wie ausführlich ist die Simulation bzw. wie umfangreich ist die Kodegenerierung?
  - Wie umfassend ist die Generierung von Dokumentation für Reviews?

Der Aspekt der Benutzerfreundlichkeit (Useability) der Werkzeuge wurde bewußt ausgelassen, da dies schwer ist auch nur annähernd objektiv zu bewerten.
Die Fallstudie lief in drei Schritten ab: Einarbeitung, Spezifikation, Review und Überarbeitung.

1. Einarbeitung in Modellierungstechnik und Werkzeug (4 Wochen)
   Die Studenten erhielten eine kurze Anforderungsbeschreibung einer Notabschaltung eines Reaktors als Trainingsbeispiel und Literatur zur Technik und Handbücher zum Werkzeug. Die Aufgabe bestand darin, die Literatur durchzuarbeiten und eine Spezifikation des Trainingsbeispiels mit Hilfe des Werkzeugs zu erstellen.
2. Spezifikation und Simulation des Tamagotchis (5 Wochen)
   Die Studenten erhielten das Anforderungsdokument für das Tamagotchi. Zunächst wurden die Anforderungen unter den Teammitgliedern aufgeteilt und ein Arbeitsplan für die Erstellung der Spezifikation aufgestellt. Beides wurde dann mit dem Betreuer abgestimmt. Anschließend wurde die Spezifikation geschrieben. Schritthaltend mit dem Spezifizieren wurde auch simuliert, sofern das Werkzeug dies zulies, um frühzeitig Fehler zu eliminieren.
3. Review und Rework (1 Woche)
   Die Teams tauschten nach der vorläufigen Fertigstellung der Spezifikation diese paarweise untereinander aus. Die Spezifikation wurden zunächst inspiziert und anschließend in einem Meeting diskutiert. Außerdem wurden im Meeting die Spezifikationen auf Basis von Fragen stichprobenartig simuliert, die sich in der Inspektion ergeben haben.

3 Resultate

Im folgenden sind die Ergebnisse der Fallstudie beispielhaft dargestellt; sie wurden aus Fragebögen, den schriftlichen Ausarbeitungen und Vorträgen der Teams sowie abschließenden Interviews entnommen.

Einarbeitung. Der Aufwand für die Einarbeitungsphase betrug zwischen 20 und 70 Stunden. Tabelle 1 zeigt die Aufwandsverteilung über die verschiedenen Modellierungstechniken. Im Aufwand enthalten ist das Lesen der Literatur und Manuals, sowie die Erstellung der Trainings-

spezifikation. Der Aufwand ist über die Teammitglieder summiert.

Focus AutoFocus	Petrinetze PEP	Octopus StP/omt	ROOM ObjecTime	SCR SCR*	SDL ObjectGEODE	Statecharts Statemate	UML Rhapsody
20	70	60	50	40	70	30	40

Tabelle 1: Aufwand für Einarbeitungsphase (Stunden)

Die Aufwandsdifferenzen entstanden im wesentlichen durch den Umfang der Literatur bzw. Handbücher, die bei den kommerziellen Werkzeugen natürlich umfangreicher ausfallen als bei Forschungsprototypen. Der Einarbeitungsaufwand des Statemate-Teams ist deswegen so gering, weil zur Modellierung der Reaktornotabschaltung ein State-chart ausreichend war. Die komplette Statemate-Vorgehensweise mit Activity- und Module-charts wurde erst in der Spezifikationsphase gelesen und verstanden. Der niedrige Aufwand für Rhapsody erklärt sich durch Vorkenntnisse eines Teammitglieds (hatte an Rhapsody-Schulung bei Berner&Mattner teilgenommen). Die Qualität der Dokumentation ist erwartungsgemäß bei den kommerziellen Werkzeugen sehr gut. Die Dokumentation zum Statemate-Ansatz beispielsweise ist zwar umfangreich (ca. 300 Seiten), ließ aber keine Fragen offen.

Größe der Spezifikation. Die Seitenanzahl der ausgedruckten Spezifikation gibt nur eine grobe Annäherung, da durch das automatisches Layout der Informationsgehalt von Seite zu Seite stark schwankt. Die Tabelle 2 zeigt die Größe der einzelnen Spezifikationen. Einige Werkzeuge erfordern die Beschreibung bestimmter Aspekte direkt in C++ Kode, in diesen Fällen ist zusätzlich noch die Zeilenanzahl des gesamten (einschließlich des generierten) Kodes angegeben.

Focus AutoFocus	Petrinetze PEP	Octopus StP/omt	ROOM ObjecTime	SCR SCR*	SDL ObjectGEODE	Statecharts Statemate	UML Rhapsody
25	20	20	84 (33)	36	136	40	(23)
-	-	-	5.5	-	-	-	16

Tabelle 2: Größe der Spezifikationen (Seitenanzahl, #KLOC)

Die Größe der Spezifikationen ist von der Beherrschung der Modellierungstechnik und ihrer Darstellungseffizienz abhängig. Die Größe (d.h. Darstellungseffizienz) der Spezifikationen in Focus, Petrinetzen, Octopus, SCR, und Statecharts ist relativ ähnlich. Die ROOM und UML-Spezifikationen sind etwas größer, da hier ein Teil graphisch (Seitenanzahl in Klammern gesetzt), und der andere Teil textuell als C++ Kode beschrieben wird. Die Größe der SDL Spezifikation ist relativ hoch, was zum Teil darauf zurückzuführen ist, daß Fallunterscheidungen, in SDL ausgedrückt, viel Platz benötigen und das automatische Seitenlayout nicht optimal ist.

Teamarbeit. Vier der acht Teams konnten aus technischen Gründen (Lizenzen, o.ä.) nicht arbeitsteilig arbeiten (ObjecTime, ObjectGEODE, Rhapsody und StP/OMT). Das SCR* Werkzeug sieht Teamarbeit nicht vor, es erlaubt weder die Aufteilung der Spezifikation in unabhängig bearbeitbare Teile noch die Zusammenführung getrennt entwickelter Spezifikationen. Der SCR Notation fehlt ein Konzept zur Kapselung von Spezifikationsteilen, was bei der Teamarbeit hilfreich wäre. Statemate unterstützt Teamarbeit vollständig, soweit dies im Rahmen der Fallstudie bewertbar ist; die Spezifikation konnte arbeitsteilig entwickelt und simuliert werden. AutoFocus unterstützt durch die Verwendung eines Versionsverwaltungssystems mit Mehrbenutzerprinzip auf der einen Seite sowie der modularen Modellierungstechniken auf der anderen Seite die Entwicklung im Team im wesentlichen sehr gut.

Aufwand. Der Aufwand für die Spezikationsphase betrug zwischen 100 und 230 Stunden. Tabelle 1 zeigt die Aufwandsverteilung über die verschiedenen Modellierungstechniken. Im Aufwand enthalten ist nur die Erstellung der Tamagotchi-Spezifikation mit dem Werkzeug. Der Aufwand ist über die Teammitglieder summiert.

Focus AutoFocus	Petrinetze PEP	Octopus StP/omt	ROOM ObjecTime	SCR SCR*	SDL ObjectGEODE	Statecharts Statemate	UML Rhapsody
140	110	100*	110*	100	210 *	100	230 *

Tabelle 3: Aufwand für Spezifikationphase (Stunden, *=keine Arbeitsteilung)

Die Aufwände für Focus, Petrinetzen, SCR, und Statecharts können nur indirekt mit den Aufwänden von OCTOPUS, ROOM, SDL und UML verglichen werden. Bei den Teams, die die Spezifikation aufgeteilt haben, ist zusätzlicher Koordinierungsaufwand entstanden. Bei den Teams, die die Spezifikation gemeinsam erstellt haben, hätte im Grunde ein Student die Spezifikation auch alleine erstellen können, d.h. der Aufwand würde dann nur ein Drittel bzw. die Hälfte (je nach Teamgröße) betragen. Teilt man beispielsweise den Aufwand für UML durch die Teamgröße ergibt sich ein Aufwand von 80 Stunden; zieht man bei Statemate einen gewissen Koordinierungsaufwand ab, kommt man auf einen sehr ähnlichen Wert. Fazit ist, daß keine signifikanten Unterschiede im Aufwand zwischen den verschiedenen Modellierungstechniken und Werkzeugen zu verzeichnen sind, mit Ausnahme von ROOM. Die ROOM-Spezifikation hätte arbeitsteilig einen Aufwand von ca. 35 Stunden + Koordinierungsaufwand benötigt, was daran liegt, daß große Teile der Funktionalität des Tamagotchi durch C++ - Kodefragemente in den Transitionen realisiert sind; im Umgang mit CASE-Werkzeugen unerfahrenen Studenten fällt dies vermutlich leichter als alle Möglichkeiten der eigentlichen Modellierungssprachen auszuschöpfen. Im Vergleich zur ersten Modellierungs-Fallstudie, die mit dem Tamagotchi jedoch ohne CASE-Werkzeuge durchgeführt wurde [KKR+98], zeigt sich, daß zum einen größere Differenzen im Aufwand zwischen Techniken durch den Einsatz von CASE-Werkzeugen nivelliert wurden. Zum anderen stieg der Aufwand durch CASE-Werkzeuge nur um ca. 10-50%, wobei sich aber auch die Qualität der Spezifikationen stark verbesserte.

4 Literatur

[BDD+92] M. Broy, C. Dendorfer, F. Dederichs, M. Fuchs, T. Gritzner, and R. Weber. The design of distributed systems - An introduction to Focus. TUM-I 0192, Technical University of Munich, 1992.

[Gra97] Bernd Grahlmann. The PEP Tool. In *Tool Presentations of ATPN'97 (Application and Theory of Petri Nets)*, June 1997.

[HBGL95] C. Heitmeyer, A. Bull, C. Gasarch, and B. Labaw. SCR*: A toolset for specifying and analyzing requirements. In *Proceedings of the 10th Annual Conf. on Computer Assurance*, pages 109–122, Gaithersburg, MD, USA, June 1995.

[HMR+98] Franz Huber, Sascha Molterer, Andreas Rausch, Bernhard Schätz, Marc Sihling, and Oscar Slotosch. Tool supported specification and simulation of distributed systems. In Bernd Krämer, Naoshi Uchihira, Peter Croll, and Stefano Russo, editors, *Proceedings International Symposium on Software Engineering for Parallel and Distributed Systems*, pages 155–164, 1998.

[KKR+98] Antje von Knethen, Erik Kamsties, Ralf Reussner, Christian Bunse, and Bin Shen. A comparative case study with industrial requirements engineering methods. In *Proceedings of the 11th International Conference on Software Engineering an its Applications*, Paris, France, December 8–10 1998.