Der software-ergonomische Stellenwert von Rahmenwerken Erfahrungen mit einem bankfachlichen Anwendungssystem

Peter W. Fach

RWG, Stuttgart

Zusammenfassung

Rahmenwerke sind Strukturierungsmittel in der modernen, objektorientierten Software-Entwicklung und verfolgen den Sinn, die Wiederverwendung ganzer Entwurfsstrukturen zu ermöglichen. Dabei ist der software-technische Aspekt der Wiederverwendung nur eine Seite der Medaille, die andere, ob und ggf. wie sich ein benutzungsfreundliches Design wiederverwenden läßt, ist nach wie vor eine interessante software-ergonomische Fragestellung. In diesem praxisorientierten Bericht werden Methoden und Erfahrungen präsentiert, wie mit Hilfe des Reifungsprozesses in Rahmenwerken und der sog. „Entwurfsmuster" benutzungsfreundliche und wiederverwendbare Anwendungssysteme für eine Universalbank erstellt werden können.

Abstract

Frameworks are structural aids in the development of modern, object-oriented software with the objective to facilitate the reuse of already existing design structures. However, the technical aspect of software reuse is only one side of the coin; the other question which presents an interesting challenge is if and how a user-friendly design can be reused. This report describes methods and experiences concerning how to develop reusable and user-friendly application systems for a universal bank, the central issues being the process of framework maturation and the so-called "design patterns".

1 Einleitung

An den Arbeitsplätzen großer Dienstleister wie Versicherungen oder Banken verändert sich die Rolle der Anwendungssoftware kontinuierlich vom ehemaligen Datenerfassungsterminal hin zu multimedialen Informations- und Beratungsinstrumenten. So sind beispielsweise Serviceinseln im Bankenbereich immer stärker gefragt, an denen einerseits eine umfassende Anlageberatung andererseits aber auch Reiseschecks ausgestellt oder Versicherungen organisationsnaher Institute abgeschlossen werden können. Dieses breite Spektrum an „Diensten", das Bankberater und -beraterinnen für ihre alltägliche Arbeit benötigen (analysieren und bewerten von Wertpapierverläufen, Auszahlen von Fremdwährungen, elektronisches Abschließen von Versicherungsverträgen), stellt allerhöchste Anforderungen an deren Benutzungsfreundlichkeit. Besonders hervorzuheben sind dabei:

- Aufgabenangemessenheit
- Performanz
- Durchgängig konsistente Bedienung

Weil sich dieser Trend, Kunden einen „Rundumservice" anzubieten, branchenweit verfestigt, und die Anforderungen an eine typische Bank-Anwendungssoftware zunehmend ähnlicher werden, entsteht auch das Potential für standardisierte, branchenweite Lösungen. So sehr sich aber einerseits die Geschäftsbereiche gleichen, so sehr müssen andererseits die einzelnen Ar-

beitsplätze an die jeweiligen Arbeitsplatzanforderungen adaptierbar sein. Rahmenwerke und Entwurfsmuster, das Thema dieses Beitrags, stellen eine software-technische Antwort auf solche und ähnliche, im Grunde sich gegenseitig ausschließende, Standardisierungs- und Flexibilisierungsanforderungen dar.

Die RWG als zentrales Dienstleistungsunternehmen für Daten- und Informationsverarbeitung der württembergischen Genossenschaftsorganisation erstellt für über 350 württembergische Raiffeisen- und Volksbanken Rahmenwerke für banktypische Anwendungsbereiche. Das auf diesen Rahmenwerken aufbauende Anwendungssystem GEBOS wird seit über acht Jahren erfolgreich an zwischenzeitlich über 12.000 z.T. stark divergierenden Arbeitsplätzen eingesetzt (vgl. z.B. [1]). In diesem Beitrag sollen nicht nur die zahlreichen Vorteile von Rahmenwerken aus software-ergonomischer Sicht dargelegt werden, sondern auch Probleme und Hindernisse, die auftreten können. Denn Rahmenwerke werden evolutionär entwickelt, sie „reifen" oder besser festigen sich erst im Laufe von Jahren. Deshalb ist es extrem wichtig, Probleme insbesondere software-ergonomischen Ursprungs in den frühen Stadien zu erkennen und zu bereinigen, denn die Korrektur wird mit zunehmendem Reifegrad aufwendiger und kostspieliger. Darüber hinaus soll auch deutlich werden, welche zusätzlichen Anforderungen auf Software-Ergonominnen und -Ergonomen zukommen, wenn sie die Entwicklung derartiger Rahmenwerke betreuen.

2 Rahmenwerke - eine kurze Einführung

Das Schlüsselkonzept beim Einsatz von Rahmenwerken lautet „Wiederverwendbarkeit des Designs". Im Gegensatz zu bisherigen software-technischen Lösungsansätzen, wo sich Wiederverwendbarkeit auf einzelne Software-Funktionen bezog, z.B. mathematische Funktionen (Sinus, Exponentialfunktionen), setzen Rahmenwerke auf die Wiederverwendbarkeit kompletter fachlicher Einheiten, wie Kundenakten, Konten oder Wertpapierdepots. Mit anderen Worten versucht man, die Entwicklungsarbeiten wie Aufgabenanalyse, fachlicher Klassenentwurf und Werkzeugdesign zu konservieren, indem man ein Anwendungsskelett bzw. -rahmen erstellt, der beispielsweise die software-technische Implementierung eines typischen Kontos darstellt. Ein solcher Rahmen muß von Anwendungsentwicklern dann nur noch auf die Besonderheiten des jeweiligen Anwendungsbereichs zugeschnitten werden. Um den Unterschied auf den Punkt zu bringen: In herkömmlichen Entwicklungsansätzen erstellt man ein Anwendungsprogramm *neu* unter Verwendung *bestehender* Klassenbibliotheken für Datenbankzugriffe, mathematische Funktionen oder die Benutzungsoberfläche. Im Sinne der Wiederverwendung des Designs kommt ein *fertiges*, ausgereiftes Rumpf-*Anwendungssystem* zum Einsatz, das von Programmierern nur an die Besonderheiten des jeweiligen Anwendungsbereichs angepaßt werden muß. Rahmenwerke geben also bereits die „Programmlogik" vor.

2.1 Schichtenbildung durch Sedimentation

Zunächst wird die Anwendungsentwicklung mit einem relativ kleinen und überschaubaren Tätigkeitsbereich begonnen, z.B. mit der Abwicklung von Daueraufträgen. Durch Aufgabenanalysen werden die typischen Tätigkeitsfelder erhoben, also Arbeitsgegenstände und Handlungsabläufe, sowie die an eventuellen Kooperationen beteiligten Rollen beschrieben. Daraus lassen sich beispielsweise mit Hilfe von CRC-Karten die fachlichen Klassen (Dauerauftrag, Konten, Auftraggeber und Empfänger, etc.) modellieren. Auf der Basis dieser fachlichen Klassen wird schließlich damit begonnen, die Benutzungsoberfläche zu entwickeln. Prototy-

pen und Autor-Kritiker-Zyklen unterstützen bei dieser Vorgehensweise die Benutzerpartizipation. Dieses oder ein vergleichbares Vorgehen hat sich im Laufe der letzten Jahre bewährt und wurde auch in verschiedenen Arbeiten dokumentiert (siehe z.B. [2], [3]). Nach der gleichen Vorgehensweise werden nun Zug um Zug Softwarelösungen für weitere Teile des Anwendungsbereichs erstellt; diese werden teilweise Überlappungen mit den bereits existierenden Lösungen aufweisen, teils aber auch neue Aspekte hinzufügen.

Vereinfacht dargestellt besteht nun die „Kunst" beim Erstellen von Rahmenwerken darin, während der Entwicklung des Anwendungssystems diejenigen Implementierungsteile (im OO-Jargon: Klassen) zu finden, die beispielsweise für alle Konto-Implementierungen immer wieder neu erstellt werden müßten. Diese werden dann als Rahmenwerk in einer eigenständigen Schicht zusammengefaßt, dem *Gegenstandsbereich* (siehe Abb. 1). Am Beispiel des Kontos führt diese Vorgehensweise dazu, daß alle Klassen, in denen das Ein- und Auszahlen von Beträgen implementiert ist, in eine tiefere Schicht sedimentieren. Das Hinzufügen und Prüfen eines „Kreditlimits" ist dagegen eine Besonderheit von Girokonten, oder das Anwenden gestaffelter Zinssätze eine von bestimmten Sparkonten. Solche konkreten Ausprägungen von Konten gehören immer auch zu einem bestimmten Typ von Tätigkeit, wie etwa die Kreditsachbearbeitung oder die Anlageberatung. Daher werden alle Implementierungsteile, die zur Unterstützung einer solchen Tätigkeit dienen, in einem Rahmenwerk zusammengefaßt, das in der Schicht *Tätigkeitsbereich* angesiedelt ist (siehe Abb. 1). Alle Rahmenwerke dieser Schicht können dann gemeinsam die Rahmenwerke aus dem Gegenstandsbereich nutzen. Im Verlaufe mehrerer Entwicklungszyklen, die sich u.U. über mehrere Jahre erstrecken können, sedimentieren gewisse Implementierungsteile in die tiefere Schicht ab und bilden dort stabile Rahmenwerke heraus [4], die in ihrer Gesamtheit alle diejenigen Arbeitsgegenstände (Konten, Kunden, Formulare, usw.) repräsentieren, die typisch für den Geschäftsbereich einer Universalbank sind [5].

2.2 Hohes Wiederverwendungspotential durch Sedimentation

Beachtenswert ist, daß diese Gegenstände nicht unmittelbar durch eine Aufgabenanalyse erhoben werden können, weil es beispielsweise das Arbeitsmittel „Konto" auf einer Bank zunächst einmal überhaupt nicht gibt, sondern nur Sparkonten, Girokonten, Darlehenskonten usw. Die Summe der Gemeinsamkeiten, die alle Kontoarten zusammen nutzen können, kristallisiert sich erst durch den Sedimentationsprozeß heraus.

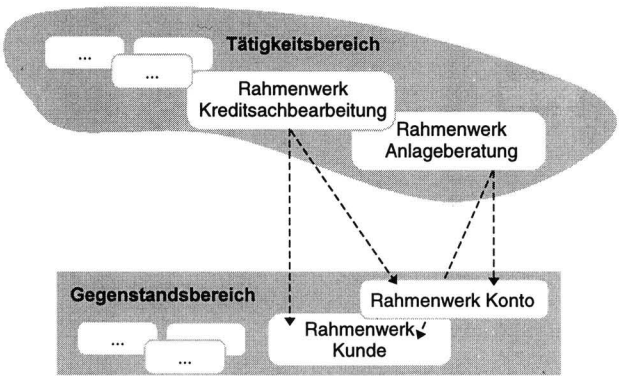

Abbildung 1: Stabile und wiederverwendbare Rahmenwerke durch den Sedimentationsprozeß. Rahmenwerke des Tätigkeitsbereichs verwenden diejenigen des Gegenstandsbereichs wieder. Eigenschaften, die in mehreren Rahmenwerken auftreten, sedimentieren so in den Gegenstandsbereich.

Genauso, wie Rahmenwerke für die „Gegenstände" eines Anwendungsbereichs entstehen, bilden sich auch solche für den Interaktionsbereich heraus. Das Ausfüllen eines Kontoeröffnungs-Antrags am Bildschirm unterscheidet sich im Prinzip nur unwesentlich vom Erfassen der Kontodaten, wie Kontonummer, Dispolimit, usw. Beide Gegenstände aus dem Anwendungsbereich lassen sich mit einem Editor bearbeiten. Auch hier kommt das gleiche Prinzip zum Einsatz, das oben bereits beschrieben wurde: Im Laufe mehrerer Entwicklungszyklen „setzen sich" die Funktionalitäten, die allen Editoren gemein sind, wie z.B. die Möglichkeit Felder anzuordnen, fehlerhafte Werte hervorzuheben, Art und Weise der Darstellung von Rückmeldungen. Auf diese Weise entsteht ein Rahmenwerk, mit dem Editoren erstellt werden können, die das Bearbeiten aller gängigen Gegenstände des Anwendungsbereichs ermöglichen. Auf die gleiche Weise bilden sich weitere Rahmenwerke heraus, die immer wiederkehrende Formen der Kooperation unterstützen, wie Versende-Mechanismen, elektronisches Unterschreiben im Rahmen einer Kompetenzverteilung, usw. Diese Rahmenwerke werden dann ebenfalls dem Gegenstandsbereich zugeordnet (siehe Abb.1).

3 Entwurfsmuster - Transparenz zwischen Benutzungsoberfläche und Implementierung

Software-technisch ist ein Rahmenwerk eine Sammlung kooperierender Klassen (z.B. in Java oder C++), die „von außen" als in sich geschlossene Einheiten genutzt werden. Die Anzahl solcher Klassen und deren Kooperation untereinander kann allerdings sehr umfassend werden, so daß es häufig schwierig ist, die Funktionsweise eines Rahmenwerkes zu dokumentieren oder auch neuen Mitarbeitern zu vermitteln. Aus dieser Problematik heraus entstanden erstmals die sog. *Entwurfsmuster* (z.B. [6]). Einerseits sind Entwurfsmuster im wesentlichen Metaphern, die Verhalten und Struktur eines einzelnen oder das Zusammenspiel mehrerer Rahmenwerke beschreiben und somit deren Funktionsweise transparent machen. Andererseits bieten sie aber auch einzigartige Möglichkeiten, um Anwenderinnen und Anwender in den evolutionären Prozeß der Entwicklung von Rahmenwerken miteinzubeziehen und somit die

Voraussetzungen für ein benutzungsfreundliches System entscheidend zu verbessern. Dieser Sachverhalt wird im folgenden anhand zweier Entwurfsmuster erörtert, die im GEBOS-Anwendungssystem erfolgreich zum Einsatz kommen.

3.1 Das Werkzeug-Material-Muster

Ein bekanntes Entwurfsmuster, welches das Zusammenspiel zwischen Objekten aus dem Anwendungsbereich (z.B. Konto) und Objekten aus dem Interaktionsbereich (z.B. Editor) beschreibt, leitet sich aus der Werkzeug-Material-Metapher ab [7]. Diese Metapher, daß nämlich mit Werkzeugen Materialien bearbeitet werden, ist ein Bild dafür, wie zwei (software-technische) Objekte zueinander in Beziehung stehen: Ein Werkzeug (Editor) wird für die Bearbeitung spezieller Materialien (Konten) hergestellt, um beispielsweise Ein- und Auszahlungen zu bewerkstelligen. Materialien haben somit keinen direkten Bezug zur Benutzungsoberfläche, dieser wird ausschließlich durch Werkzeuge hergestellt. Die Metapher wird nun für die Implementierung handlungsleitend, weil sie die Abhängigkeiten zwischen Objekten definiert: Für die Implementierung eines Werkzeugs ist Kenntnis über die Beschaffenheit der relevanten Materialien von Nöten; dagegen bleibt die Implementierung eines Materials völlig unabhängig von der eines bestimmten Werkzeugs[1]. Damit ist gewährleistet, daß Änderungen an der Benutzungsoberfläche keinerlei Auswirkungen z.B. auf die Rahmenwerke des Gegenstandsbereichs haben.

Handlungsleitend ist diese Metapher aber auch für die Strukturierung der Benutzungsschnittstelle. So „hantieren" qualifizierte GEBOS-Benutzer bewußt mit Materialien und wenden auf diese auch gezielt Werkzeuge an, z.B. einen Kundensucher, um anhand gegebener Kriterien eine gewünschte Menge von Kunden aufzulisten. Dies versetzt Benutzer einerseits in die Lage, auftretende Problembereiche samt Arbeitskontext präzise zu benennen (z.B. „der Kundensucher kann in dieser oder jener Situation nicht eingesetzt werden, weil ...") und ermöglicht Entwicklern anderseits derartige Anforderungen fachlich einzuordnen, zu lokalisieren und zu beurteilen (vgl. [4]). Anwender und Entwickler sprechen also dieselbe Sprache und beziehen sich dabei auch auf dieselben technischen Entitäten! Damit kommt diesem Entwurfsmuster eine zentrale Rolle hinsichtlich der Benutzerpartizipation in der evolutionären Entwicklung von Rahmenwerken zu, die ihrerseits wiederum eine notwendige Voraussetzung für rasch konvergierende Sedimentationsprozesse ist. So ist das Entwurfsmuster „Werkzeug-Material" komplett in den GEBOS-Entwicklungsprozeß integriert ([2], [3], [8]) und ein herausragendes Beispiel dafür, wie konzeptionelle Modelle und Entwurfsmodelle (Normans „conceptual model" und „design model" [9], [10]) durch den Einsatz einer geeigneten Metapher zur Konvergenz gebracht werden können.

Auch in diesem Beitrag wird im folgenden von *Materialien* gesprochen, wenn es sich um fachlich motivierte Implementierungen (Konten oder Daueraufträge) handelt und von *Werkzeugen*, wenn es um Implementierungen der Benutzungsoberfläche geht, mit denen Materialien angezeigt oder bearbeitet werden (Auflister, Editor). Daß dieses Entwurfsmuster, das seinen eigentlichen Ursprung in den Entwicklungs-Labors hat, auch in diesem Beitrag als Metapher handlungsleitend werden kann, ist ein weiteres und aussagekräftiges Indiz für das Potential so verstandener Entwurfsmuster. Ein zweites Muster, das in GEBOS ebenfalls erfolgreich eingesetzt wird, das sog. Rollenmuster, wird im nächsten Abschnitt beschrieben.

---

1 Für OO-Methodiker: Das Entwurfsmuster „Werkzeug-Material" entspricht dem sog. MVC-Paradigma (vgl. z.B. [11]) und wurde hier nur in Teilen dargestellt.

3.2 Das Rollenmuster

In einer Universalbank treten Kunden häufig in verschiedenen Rollen auf: als Anleger, Kreditnehmer, Bürge oder auch als Kunde organisationsnaher Institute wie Versicherungen oder Fondsgesellschaften. Entsprechend vielfältig sind auch die Anforderungen an die jeweiligen Arbeitsplätze der Bankangestellten. Bei der Kreditsachbearbeitung muß beispielsweise ein Gesamtüberblick über die wirtschaftlichen Verhältnisse eines Kunden gewährleistet sein, in den o.a. Serviceinseln, wo eine schnelle und effiziente Abwicklung eines Kundenwunsches im Vordergrund steht, genügen die wichtigsten Kundendaten und ggf. ein Kontenüberblick. Dagegen dürfen spezielle Kundendaten generell nur autorisierten Personen zugänglich sein, usw. Derartige Anforderungen an die Adaptierbarkeit der Arbeitsplätze ziehen u.a. softwaretechnische Herausforderungen wie die folgenden nach sich:

- *Performanz.* In Serviceinseln müssen alle relevanten Kundendaten unmittelbar verfügbar sein, dennoch sollten einzelne Rollen so von einander entkoppelt werden, daß auch nur diejenigen „geladen" werden, die in der jeweiligen Situation erforderlich sind; kurze Antwortzeiten wäre sonst nicht leistbar.
- *Sedimentation.* Häufige Änderungen der Kundendaten, z.B. wegen gesetzlicher Anforderungen, oder Erweiterungen des Anwendungssystems, verlangsamen oder verhindern im Extremfall sogar die Reifung des betroffenen Rahmenwerks.

Anders als im Fall des Entwurfsmusters „Werkzeug-Material", das einen software-technischen Ursprung hat, dient in diesem Fall der bankfachliche Alltag als Metapher für die Strukturierung software-technischer Lösungen, woraus das sog. „Rollenmuster" entstand [5].

In Abb. 1 wurde dargestellt, auf welche Weise Rahmenwerke in unterschiedliche Schichten eingeteilt werden können. Die Idee beim Rollenmuster besteht nun darin, daß sich die Kernimplementierung eines Kunden, die von allen Kundenrollen (z.B. Bankkunde oder Bürge) genutzt werden, im Gegenstandsbereich setzt. Jene Kundenimplementierungen, die rollenspezifisch sind, verbleiben dagegen in Rahmenwerken des Tätigkeitsbereichs. Fallen nun Änderungen an einer bestimmten Rolle an, wirken sich diese nur in dem betroffenen Rahmenwerk aus. Der Kundenkern im Gegenstandsbereich sowie die Implementierungen der anderen Rollen bleiben davon unberührt. Denn die Idee besteht ja gerade darin, daß alle Gemeinsamkeiten im Gegenstandsbereich sedimentieren, und die Implementierungen im Tätigkeitsbereich voneinander disjunkt sind.

3.3 Adaptierbarkeit des Anwendungssystems

Wegen dieser Aufteilung in unterschiedliche Rahmenwerke, die alle denselben Kundenkern nutzen, untereinander jedoch völlig unabhängig sind, können auf sehr elegante Weise und ohne weiteren Implementierungsaufwand arbeitsplatz-spezifische Anwendungen erstellt werden[2]. Beispielsweise stellt Abb. 2 ein GEBOS-Werkzeug dar, das einen Gesamtüberblick über einen Bankkunden vermittelt. Im oberen Teil des Werkzeugs werden die invarianten Kundendaten präsentiert, die zum Kundenkern des Gegenstandsbereichs gehören. Dagegen listet der untere, linke Bereich des Werkzeugs nur diejenigen Rollen auf, die an dem jeweiligen Arbeitsplatztyp zur Verfügung stehen müssen. Die Auswahl einer bestimmten Rolle, z.B. Bank-

2 Für OO-Methodiker: Die Technik hierfür sind unterschiedliche DLLs; technische Voraussetzungen sind u.a. die Trennung der Rahmenwerke in einen konzeptionellen und einen Implementierungsteil, siehe [5].

kunde, listet dann im rechten Bereich alle Informationen auf, die typischerweise mit dieser Rolle assoziiert werden.

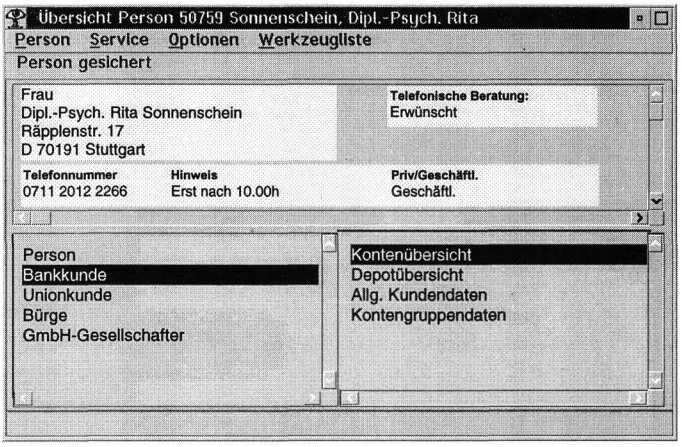

Abbildung 2: Struktur der Implementierung und der Benutzungsoberfläche fallen zusammen. Im unteren, linken Teil werden die arbeitsplatz-spezifischen Rollen und im rechten die dazugehörenden Informationen aufgelistet; letztere sind als Rahmenwerke des Tätigkeitsbereichs implementiert.

Wie auch schon das Werkzeug-Material-Muster, strukturiert das Rollenmuster die Benutzungsoberfläche und die software-technische Implementierung vor dem selben bankfachlichen Hintergrund und fördert damit die Konvergenz zwischen konzeptionellem Modell der Benutzer und dem Entwicklungsmodell. Darüber hinaus bietet das Rollenmuster fachlich sinnvolle und software-technisch elegante Möglichkeiten, wie ein Anwendungssystem auf bestimmte Arbeitsplatztypen zugeschnitten werden kann. Gleichwertige Möglichkeiten bietet übrigens auch das Werkzeug-Material-Muster, indem auf bestimmten Arbeitsplatztypen nur ein eingeschränkter oder aber ein erweiterter Satz an Werkzeugen und Materialien verfügbar gemacht wird.

Dieses Kapitel sollte verdeutlichen, welches software-ergonomische Potential in der Anwendung von Rahmenwerken steckt: beste Voraussetzungen für eine kompetente Benutzerpartizipation, fachlich adaptierbare Arbeitsplatztypen und vor allem anderen: die Wiederverwendbarkeit eines „guten" Designs (vgl. Abb. 1), was in der Deckungsgleichheit von technischer Implementierung (Rahmenwerk) und fachlicher Entität gründet. Ein weiterer entscheidender Faktor ist jedoch, daß nicht nur Materialien, sondern ganze Werkzeuge wiederverwendet werden können, und nicht wie bislang üblich nur einzelne Controls bzw. Widgets (z.B. Pushbuttons oder Listboxen). Dadurch ändert sich jedoch die Rolle des Styleguides bei der Entwicklung von Rahmenwerken drastisch.

4 Styleguides - Verwendungskataloge für bewährte Lösungen

Bei traditionellen Entwicklungsmethoden wurde die Rolle des Styleguides mit wachsendem Umfang der Anwendung immer wichtiger. Denn immer mehr Personen sind an der Entwicklung beteiligt, immer neue Komponenten kommen hinzu und die Gefahr, daß ein einheitliches

Look&Feel verloren geht, steigt kontinuierlich an. So umfaßt z.B. der Styleguide der Sparkassen-Organisation [12] über 800 Seiten. Auch die ursprüngliche, von der RWG als firmeninterner Styleguide ausgearbeitete Richtlinie umfaßte knapp 300 Seiten. Werke, die aller Erfahrung nach bei Anwendungsentwicklern auf Akzeptanzprobleme stoßen.

Beim Einsatz von Rahmenwerken verändert sich dagegen die Rolle des Styleguides gravierend, was seinen Grund in der bereits erwähnten Wiederverwendbarkeit der Werkzeuge hat. So werden im GEBOS-Anwendungssystem nur selten Werkzeuge neu konzipiert; in den meisten Fällen können bestehende wiederverwendet werden (die einfachsten Beispiele hierfür sind Auflister und Editoren). Dadurch konzentriert sich jedoch die Standardisierungsarbeit auf eine sehr frühe Phase in der Evolution der Anwendungsentwicklung und betrifft nur einen kleinen und überschaubaren Kreis von Entwicklern, so daß sich hier in Zusammenarbeit mit Software-Ergonomen das notwendige Know-how aufbaut und in den Werkzeugen selbst konserviert. Dadurch stellt sich auch die Frage nach der korrekten Verwendung einzelner Controls bzw. Widgets immer seltener. Dagegen muß ein Styleguide für die Entwicklung von Rahmenwerken die folgenden Schwerpunkte berücksichtigen:

- *Wiederverwendungskriterien.* Die Situationen, in denen ein spezielles Werkzeug wiederverwendet werden darf, müssen präzise definiert werden. Dies läßt sich am besten durch einen Katalog realisieren, der alle zur Wiederverwendung bereitstehenden Werkzeuge darstellt und deren Einsatzbedingungen definiert. Ansonsten besteht die Gefahr, daß Werkzeuge - weil sie bereits implementiert und deswegen mit verhältnismäßig geringem Aufwand nutzbar sind - außerhalb des vorgesehenen Kontexts eingesetzt werden. Aller Erfahrung nach sind die Werkzeuge dann nicht mehr handhabbar, müssen mittel- oder langfristig angepaßt werden, und machen letztlich die Vorteile der Wiederverwendbarkeit wieder zunichte.
- *Handhabungsmodelle.* Die Menge aller Metaphern (in diesem Zusammenhang als Handhabungsmodell bezeichnet), die für die Bedienung eines Anwendungssystems handlungsleitend sind, wie z.B. die Werkzeug-Material-Metapher oder die Rollenmetapher, muß frühzeitig standardisiert werden. Dies verhindert, daß bei der parallelen Entwicklung von Rahmenwerken alternative Handhabungsmodelle angewendet werden, deren spätere Vereinheitlichung zu Überarbeitungen der Rahmenwerke aus dem Gegenstandsbereich (vgl. Abb. 1) führen kann.

Nach wie vor muß ein Styleguide für Rahmenwerke z.B. Menüeinträge und damit assoziierte Aktionen standardisieren. Aller Erfahrung nach sind Rahmenwerke jedoch gegenüber derartigen Veränderungen sehr tolerant, weil nur durch wenige, lokal begrenzte Änderungen (z.B. im Gegenstandsbereich, vgl. Abb. 1), auf das systemweite Look&Feel Einfluß genommen werden kann (vgl. auch [4]).

5 Diskussion

Wie ein roter Faden zog sich der Begriff der Sedimentation durch diesen Beitrag und letztlich zielte alles darauf ab, diesen evolutionären Reifungsprozeß von der software-ergonomischen Seite von Beginn an zu unterstützen, sei es durch geeignete Metaphern für Entwurfsmuster, die ein hohes Maß an Benutzerpartizipation sicherstellen, sei es durch spezielle Kataloge, die die Wiederverwendung reglementieren. Natürlich ist dieses „vorauseilende" Wesen der Software-Ergonomie nicht erst durch das Aufkommen der Rahmenwerke aktuell geworden, sondern wird schon seit fast zehn Jahren immer wieder hervorgehoben [13]. Gereifte Rahmen-

werke tragen jedoch ein extrem hohes Trägheitspotential in sich, und Änderungen am Design von Rahmenwerken haben oft weitreichende Folgen und destabilisieren im ungünstigsten Fall das gesamte System. Rein software-ergonomisch bedingte Änderungen finden deshalb in den späten Phasen der Rahmenwerke tendenziell nur noch wenige Realisierungschancen. Dies bewirkt natürlich eine Veränderung im Tätigkeitsfeld der Software-Ergonominnen und -Ergonomen. Zu den typischen Aufgabenfeldern kommt noch die Notwendigkeit hinzu, den technischen Entwurfsprozeß zu verstehen, im Rahmen des Möglichen auf die Entwicklung passender Metaphern für Entwurfsmuster hinzuwirken und vor allen Dingen darauf zu achten, daß deren Realisierung im Entwicklungsprozeß konsequent eingehalten wird.

Die in diesem Beitrag immer wieder hervorgehobenen Entwurfsmuster sind zunächst einmal Metaphern, um technische Lösungen transparent zu machen und den Grad der Wiederverwendbarkeit zu erhöhen, mit dem Ziel, mittel- und langfristig die Entwicklungskosten zu minimieren. So gibt es zwischenzeitlich ganze Kataloge [6] mit technischen Lösungen, die anhand von Metaphern als Entwurfsmuster dokumentiert sind. Viele dieser Entwurfsmuster werden die Entwicklungslabors sicherlich niemals verlassen; andere aber, und hierzu gehören das Werkzeug-Material-Muster und das Rollenmuster, haben das Potential, Entwicklungsmodelle und konzeptionelle Modelle zur Konvergenz zu führen. So entstehen transparente und auch für Anwender leicht nachvollziehbare technische Lösungen, die in vergleichbaren Kontexten wiederverwendet werden und so einen wertvollen Beitrag zur Erwartungskonformität von Anwendungssoftware leisten.

6 Literatur

[1] RWG & Volksbank Herrenberg: Nutzungspotentiale von GEBOS-Passiv. Diebold-Studie. Stuttgart,1996: RWG.

[2] U. Bürkle, G. Gryczan & H. Züllighoven: Object-Oriented System Development in a Banking Project: Methodology, Experience, and Conclusions. Human Computer Interaction, 10, 1995, 4, pp. 293-336.

[3] H. Züllighoven: Das objektorientierte Konstruktionshandbuch. Nach dem Werkzeug- und Materialansatz. Heidelberg, 1998; dpunkt.verlag.

[4] W.A. Kellogg et al.: NetVista: Growing an Internet solution for schools. *http://www.almaden.ibm.com/journal/sj/371/kellogg.txt*, 1998.

[5] D. Bäumer et al.: Framework Development for Large Systems. Commun. ACM, 40, 1997, 10, pp. 52 - 95.

[6] E. Gamma et al.: Design Patterns: Elements of Reusable Software. Reading, Massachusetts, 1994: Addison Wesley.

[7] R. Budde & H. Züllighoven: Software-Werkzeuge in einer Programmierwerkstatt. München, 1990: Oldenbourg.

[8] RWG: Die GEBOS-Methode, Interner Entwicklungsleitfaden. Stuttgart, 1996: RWG.

[9] D.A. Norman: Cognitive engineering. In: D.A. Norman & S.W.Draper (eds.): User centered system design. Hillsdale, NJ, 1986: Lawrence Erlbaum.

[10] M.B. Rosson & S. Alpert: The cognitive consequences of object-oriented design. Human-Computer Interaction, 5, 1990, 4, pp. 345-379.

[11] T. Reenskaug: Working with Objects. New York, 1996: Prentice-Hall.

[12] Sparkassen-Organisation: SIZ Gestaltungsleitlinien für Grafische Oberflächen. Stuttgart, 1997: Deutscher Sparkassenverlag.

[13] A. Grünupp & K.-P. Muthig: Software-Ergonomie als prospektive Gestaltung der Funktionalität von Werkzeugen. In: A. Reuter (Hrsg.): GI - 20. Jahrestagung: Informatik auf dem Weg zum Anwender, Band 1. Berlin, 1990: Springer Verlag, S. 532-542.

Adresse des Autors

Dr. Peter Fach
RWG GmbH
Produktplanung / Software-Ergonomie
Räpplenstr. 17
70191 Stuttgart
Tel: +49 711 2012 2266
Fax: +49 711 2012 6266
Email: peter.fach@rwg.de
http://www.rwg.de