Effiziente Konstruktion von ergonomischen Benutzungsoberflächen aus konfigurierbaren Bausteinen

Udo Arend
SAP AG
Neurottstr. 16
69190 Walldorf/Baden
udo.arend@sap.com
http://www.sapdesignguild.org/

Abstract
Generische Bausteine, sogenannte User Interface Patterns, können zu generischen Grundrissen zusammengestellt werden. Grundrisse bilden Klassen von Applikationsarchetypen ab. Durch geeignete Konfigurationswerkzeuge können aus den Grundrissen User Interfaces für bestimmte Geschäftsprozesse erstellt werden. Dazu werden Grundrisse an Web-Services, welche die Business Logik beinhalten, gebunden. In dem Beitrag wird ein Ansatz vorgestellt, wie Grundrisse und User Interface Patterns aus generischen Aufgaben hergeleitet werden können. Beispiele illustrieren die Abbildung bestimmter Ge-schäftsprozesse auf reale Benutzungsoberflächen. Die Vorteile dieses Ansatzes werden diskutiert: neben einer effizienten Erstellung von Benutzungsoberflächen erlaubt der Ansatz eine einfache, aber weitgehende Konfiguration der Benutzungsoberflächen (Customizing, Personalisierung, Adaptierung); dem Benutzer können hochkonsistente und bzgl. der Be-dienbarkeit optimierte User Interfaces zur Verfügung gestellt werden.

Keywords
Usability, User Interface Pattern, Floor Plans, Komponenten

1.0 Einleitung

Haben sie sich jemals als User Interface Designer beim Entwurf der Benutzungsoberfläche einer Anwendung gefragt: »Soll ich in meiner Anwendung Iko-nen verwenden? Wo soll ich die Kopfdaten und wo die Detaildaten platzieren? Soll ich klappbare Bildbe-reiche verwenden? Wie suchen Benutzer nach Objekten oder Daten? Wie sollen neue Zeilen in eine Tabelle eingefügt werden? Aus welchen Arbeitsberei-chen soll ein Bild aufgebaut werden? Wie soll...? Wenn sie solche Fragen bereits gestellt haben, wissen sie auch, wie schwierig es ist, die richtige Antwort zu finden. SAPs Antwort auf diese Frage ist die Einführung von wiederverwendbaren generischen »User Interface Pattern« und von universellen »Grundrissen« (Floor Plans) 1, 2, 3, 4. Im Rahmen eines User-Interface-First Prozesses können diese Konzepte – unterstützt durch geeignete Tools – von User Interface Designern eingesetzt werden um Geschäftsanwendungen effizient und konsistent zu entwerfen. User Interface Entwickler können die Benutzungsoberfläche von Anwendungen anschließend mit geeigneten Entwicklungswerkzeugen konfigurieren.

2.0 Das User Interface Pattern Konzept

Die Verwendung von wiederverwendbaren Bausteinen, auch von User Interface Komponenten, ist nicht neu. Van Duyne et al. 4 definieren »Patterns« wie folgt »Patterns communicate insights into design problems, capturing the essence of the problems and their solution in a compact form«. Was also ist das spezifisch Neue an unserem Ansatz?

1 Trennen der Beschreibung der (Teil-)Aufgabe von ihrer Abbildung auf das User Interface
2 Finden von generischen Lösungen für generische Aufgaben

Zur Strukturierung des Lösungsansatzes wurde davon ausgegangen, dass sowohl Aufgaben als auch User Interface Elemente in eine Dekompositionshierarchie gepackt werden können 5. Abbildung 1 zeigt, auf der rechten Seite die Aufgabenhierarchie und auf der linken Seite die User Interface Hierarchie.

Hassenzahl M., Peissner M.,(Hrsg.): Usability Professionals 2004

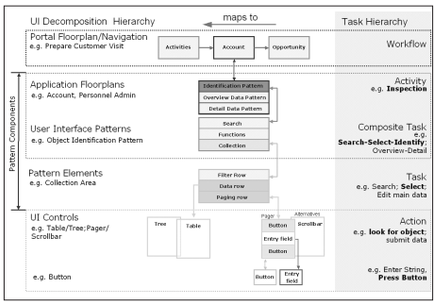

Abbildung 1: Das UI Pattern Konzept

Die grundsätzliche Herangehensweise zeichnet sich dadurch aus, eine generische User Interface Lösung für jede identifizierte generische Aufgabe zu erzeugen. Die Aufgabe »Suche nach einem Arbeitsobjekt« zum Beispiel ist generisch. Ob es sich dabei um eine Bestellung, oder eine Lieferung, oder einen Rechnungsbeleg, oder ein Material handelt, spielt keine Rolle. Die Annahme ist: wenn generische Aufgaben für eine bestimmte Anzahl von Geschäftsprozessen gleichartige Eigenschaften aufweisen, lässt sich eine vereinheitlichte Lösung auf der Benutzungsoberfläche finden. So stellt z.B. die »Drucktaste« auf einem User Inter-face eine Lösung für das generische Pro-blem »Funk-tion auslösen« dar. Das Neuartige besteht darin, die Ebene der elementaren Aktionen und der elementaren Lösungen zu verlassen, und komplexere Aufgaben zu identifizieren.
Es werden elementare Bausteine, die »Controls«, zu höherwertigen Bausteinen«, den »Pattern-Elementen« zusammengesetzt, die dann wiederum zu noch komplexeren Bausteinen kombiniert werden können.

Diese neuen Bausteine werden »User Interface Pattern« genannt. Jedes User Interface Pattern bedient eine oder mehrere generische Aufgaben, wie z.B. Suche nach Businessobjekten und Auswahl eines bestimmten Objektes.
Bei der Applikationsentwicklung werden User Interface-Patterns zu Applikations-Grundrissen kombiniert. Dabei werden sie auf dem Bildschirm räumlich angeordnet. Für gleiche Anwendungstypen (»Business-Aktivitäten«) können solche Applikations-Grundrisse standardisiert werden. Eine erste Version des »People-centric« CRM von SAP besteht aus einem einzigen Grundriss mit zwei Varianten und benutzt nur zwei unterschiedliche User Interface Pattern. Dadurch wird eine hochgradige Konsistenz der Anwendungen erzielt. Zugunsten einer optimierten Aufgabenangemessenheit kann der »puristische« Ansatz natürlich dahingehend erweitert werden, dass weitere Grundrisse und weitere User Interface Pattern zugelassen werden.

3.0 Design Methode

3.1 Fokus-Areas

Im Rahmen der Contextual-Design Methode von Holtzblatt 6, werden »Focus Areas« hergeleitet, aus denen dann ein »User Environment Modell« erstellt wird. Eine »Fokus-Area« beschreibt eine kohärente Benutzer-Aktivität. Sie wird definiert durch ihren Zweck für den Benutzer, durch Funktionen, die auf sie ausgeübt werden können, durch Objekte auf denen operiert wird und durch Links zu denen sich der Benutzer bewegen kann (S. 306f). Die Benutzungsoberfläche soll nach Holtzblatt eine Fokus-Area als einen zusammenhängenden Bereich auf dem Bildschirm abbilden. Dadurch kann der Benutzer sich auf diesen Bildbereich fokussie-ren und an diesem Platz seine Aktivität vollständig ausführen.

3.2 Generische Fokus-Areas und generische Aufgaben

Normalerweise werden Fokus-Areas für einen bestimmten Geschäftsprozess hergeleitet und sind nur für diesen gültig. Für unseren Ansatz ist es aber entscheidend, generische Fokus-Areas für generische Aufgaben herzuleiten. Nur so kann sichergestellt werden, dass gleich strukturierte, aber unterschiedliche Applikationen, eine konsistent aufgebaute Benutzungsoberfläche erhalten.
Außerdem ist die Generalisierung die Voraussetzung dafür, generische, wie-derverwendbare Software Komponenten zu bauen, die dann nur noch für den jeweiligen Anwendungsfall konfiguriert werden müssen.

3.3 Herleitung einiger generischer Komponentenaufgaben

Der ganze Ansatz kann nur funktionieren, wenn es gelingt, eine überschaubare Zahl von »generischen« Aufgaben zu identifizieren, mit denen jedoch viele, auch sehr unterschiedliche Business-Aktivitäten be-schrieben werden können. Nur in diesem Fall kann die Anzahl der User-Interface Komponenten klein gehalten werden.

3.3.1 Business-Aktivität auswählen

Bevor eine Business-Aktivität bearbeitet werden kann, muss eine zugehörige Applikation ausgewählt werden.
Die folgende Abbildung zeigt wie beispielsweise eine Applikation in einem mehrstufigen Konstrukt ausgewählt werden kann, welches in Form eines Portal-Bar permanent verfügbar ist.

Abbildung 2: Eine Applikation auswählen

3.3.2 Arbeitsobjekt suchen

Ein Arbeitsobjekt zu suchen oder zu bestimmen, kann z.B. durch folgende Lösung erreicht werden:

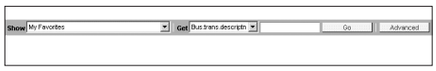

Abbildung 3: Arbeitsobjekt suchen und auswählen

Die Dropdown-Box »Show« erlaubt die Auswahl einer vordefinierten Suchabfrage, die Dropdown-Box »Get« erlaubt alternativ die Auswahl eines Schlüsselfeldes und die Eingabe eines Wertes im zugehörigen Eingabefeld. Hinter der Drucktaste »Advanced« verbirgt sich ein Fenster, in dem eine erweiterte Suchanfrage gestellt werden kann.

3.3.3 Auswählen aus gesammelten Arbeitsob-jek-ten

Als Ergebnis der Suche nach Arbeitsobjekten, können mögliche Arbeitsobjekte in einem Bereich gesammelt werden.
Der Benutzer kann entscheiden, wie er dann weiter verfährt, d.h. ein oder mehrere Arbeitsobjekte auswählen.

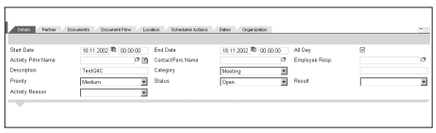

Abbildung 4: Mögliche Arbeitsobjekte in einer Ergebnisliste

3.3.4 Orientieren

Der Benutzer benötigt Informationen über das momentane Arbeitsobjekt, seinen Status und wohin er navigieren kann.

3.3.5 Identifizieren

Dem Benutzer ermöglichen, dass zu bearbeitende Arbeitsobjekt eindeutig zu identifizieren. Dazu benötigt er die beschreibenden Daten, die auch editierbar sein müssen.

3.3.6 Daten bearbeiten

Dem Benutzer die zu bearbeitenden Daten in einer geeigneten Repräsentation zur Verfügung stellen. Das kann in einem Formular, in einer Tabelle, in ei-nem Baum oder grafisch erfolgen.

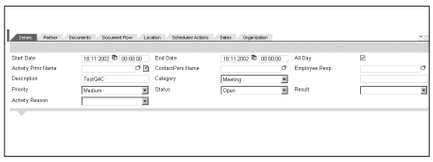

Abbildung 5: Operativer Bereich zur Bearbeitung von Daten

3.3.7 Detaildaten bearbeiten

Dem Benutzer die zu bearbeitenden Detaildaten in einer geeigneten Repräsentation zur Verfügung stellen. Das kann in einem Formular, in einer Tabelle, in einem Baum oder grafisch erfolgen. Gleichzeitig den Bezug zum übergeordneten Objekt deutlich machen.

3.3.8 Analysieren

Der Benutzer möchte größere, strukturierte Datenmengen analysieren. Dazu müssen die Daten in geeignete Sichten gebracht werden können. Um Beziehungen zwischen/innerhalb der Daten zu erkennen, müssen entsprechende Funktionen bereitgestellt werden (z.B. Sortierungen, Drilldown, Dice and Slice).

3.4 Struktur von Aktivitäten

Wir wählten eine »Business-Aktivität« als Korngröße um eine Applikation zu definieren, also z.B. Erfassung einer Rechnung, Bearbeiten eines Kundenauftrages usw. Die Annahme ist, dass jede Business-Aktivität aus einer leicht beschreibbaren, zugrundeliegenden Struktur besteht, die vom Benutzer auch so oder ähnlich mental repräsentiert wird. Diese deklarative Struktur setzt sich aus Begriffen (Objekten) unterschiedlicher Abstraktion zusammen. Jedes Objekt kann vom Typ »kategorial« oder »aufzählend« sein. Daraus ergibt sich folgende Objektstruktur:

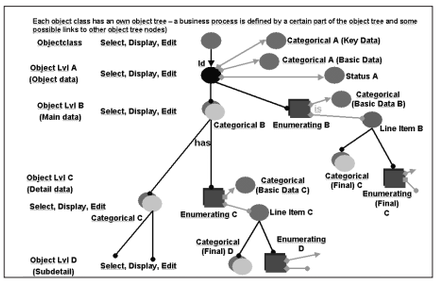

Abbildung 6: Struktur von Business-Aktivitäten

Beim Reengineering zahlreicher Business-Aktivitäten konnten wir feststellen, dass sich aus der allgemeinen Objektstruktur immer die spezifi-sche

Objektstruktur herleiten lässt. Für Konstruktions-, Planungs- und Analyseaufgaben können sich natürlich andere Strukturen ergeben. Das folgende Beispiel zeigt die Objektstruktur für eine Rechnung:

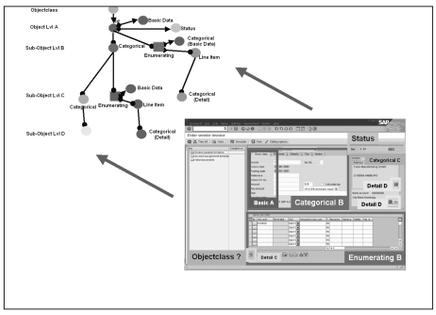

Abbildung 7: Beispiel -Objektstruktur für eine Rechnung

3.5 Was ist ein Grundriss (Floor Plan)

Die Fokus-Areas eines Geschäftsprozesses können zu einem Grundriss (Floor Plan) zusammengestellt werden. Jeder »Raum« des Grundrisses bestimmt einen Platz auf dem User Interface, also z.B. oben horizontal, links senkrecht, rechts unten usw., jedes Stockwerk des Grundrisses impliziert einen Bildwechsel. Nach dem zuvor gesagten gilt: Für jede Fokus-Area, die eine spezifische Aufgabe beschreibt, lässt sich eine dahinterliegende generi-sche Aufgabe identifizieren. Zu jeder generischen Aufgabe gibt es aber eine Abbildung auf das User Interface in Form einer konfigurierbaren Komponente. Für das CRM-System haben wir einen Applikations Grundriss definiert, der für die Aufgabe des Bearbeitens von Objekten geeignet ist. Andere Aufgaben erfordern natürlich andere Grundrisse.

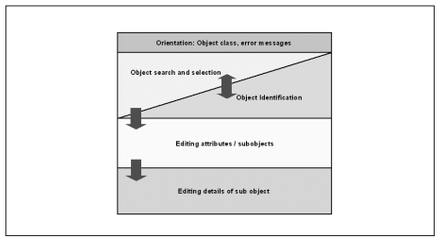

Abbildung 8: Applikations-Grundriss für generische Aufgaben vom Typ »Bearbeiten von Objekten«.

Der Grundriss der generischen Aufgaben kann dann auf den Grundriss ausprogrammierter User Interface Patterns abgebildet werden:

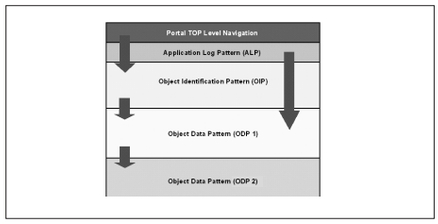

Abbildung 9: Applikations-Grundriss mit User Interface Patterns.

4.0 Mappen von generischen Aufgaben auf das User Interface

Ziel ist, generische Aufgaben in einer geeigneten Weise auf UserInterface Komponenten abzubilden. Dabei gilt:

1 Eine oder mehrere generische Aufgaben werden auf eine User-Interface Komponente, genannt «User Interface Pattern« abgebildet.

2 Diese Komponente wird räumlich auf dem Bildschirm entsprechend eines vordefinierten Grundrisses zugeordnet

3 Die Komponente enthält alle Funktionen, Daten, Links um die entsprechende Aufgabe durchführen zu können

4 Die Komponente verfügt über eigene Personalisierungs- und Hilfefunktionen

Die folgende Abbildung zeigt wie generische Aufgaben auf bestimmte User Interface Patterns in einem vordefinierten Grundriss abgebildet werden können:

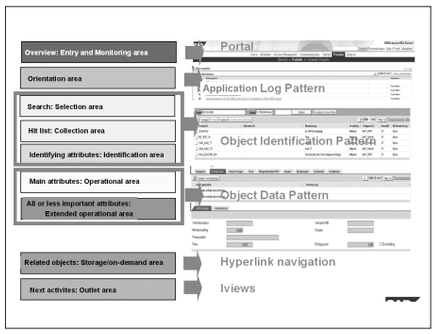

Abbildung 10: Beispiel 1 - Abbildung generischer Aufgaben auf User-Interface Komponenten.

5.0 Beispiele

Das folgende Beispiel des SAP CRM enthält ein Object Identification Pattern und ein Object Data Pattern, beide in Formulardarstellung.

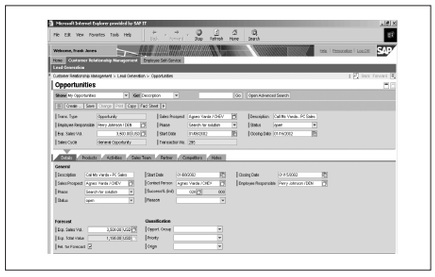

Abbildung 11: Beispiel – Opportunity Management

Man beachte, dass die Oberflächen vom Entwickler nur konfiguriert werden. Das eigentliche Erscheinungsbild wird

komplett generiert. So wird beispielsweise eine Formularsicht durch Layoutalgorithmen erzeugt.

6.0 Bewertung und Ausblick

Was haben wir aus dem ersten Projekt mit User Interface Pattern gelernt?

- Die gewählte Abstraktionsebene hat sich als geeignet erwiesen, funktional mächtige und allgemeine Bausteine abzuleiten
- Die Zahl der identifizierbaren, gener ischen Funktionen bleibt über schaubar
- Generische Funktionen lassen sich auf generische User Interface Pattern mappen. Hierbei ist eine prä zise Beschreibung der Funktionen notwendig.
- Die Komposition zu Grundrissen ist relativ einfach. Zu bestimmen ist, wie Daten zwischen den User Interface Pattern ausgetauscht werden.
- Die Konfiguration der Benutzungsoberflächen statt ihrer Programmierung ist technisch umset zbar und bietet dem Entwickler große Hilfen.

Der große Vorteil vordefinierter User Interface Komponenten ergibt sich in a) der einfachen Herstellung hochkonsistenter Benutzungsoberflächen b) der ergonomischen Optimierbarkeit einzelner Komponenten auch im nachhinein c) des hohen Produktivitätsgewinns bei der Entwicklung, da die Oberflächen nur konfiguriert werden müssen.
Herausforderungen ergeben sich durch die vorhandene Unterschiedlichkeit der Business-Aktivitäten, d.h. der Tradeoff Beziehung zwischen Aufgabenangemessenheit und Konsistenz, aber auch dadurch, dass Entwicklungsanfragen im Laufe der Zeit die klare Struktur eines User Interface Patterns verwässern können. Welche Grundrisse für welche Applikations-Archetypen am Geeigneten sind, lässt sich auch erst aus der Erfahrung erkennen. Weitere Forschungsfragen finden sich in Kapitel 4 bei Wesson und Cowley 7.

7.0 Referenzen

1 Alexander C., The Timeless Way of Building. Oxford University Press (1979).

2 Tidwell J., Common Ground: A Pattern Language for Human-Computer Interface Design (1999). http://www.mit.edu/~jtidwell/ui_pat terns_essay.html

3 Van Welie M., Web Design Patterns (2003). http://www.welie.com/patterns/ index.html

4 Van Duyne D.K., Landay J.A. and Hong J.I., The Design of Sites – Patterns, Principles, and Processes for Crafting a Customer-Centered Web Experience. Boston: Addison-Wesley (2003).

5 Card S.K., Moran T.P. and Newell A., The Psychology of Human-Computer Interaction. Hillsdale, NJ: Lawrence Erlbaum Ass (1983).

6 Beyer H. and Holtzblatt K., Contextual Design: Customer-Centered Systems. San Francisco: Morgan Kaufman (1998).

7 Wesson J. and Cowley L., Designing with Pat-terns: Possibilities and Pitfalls. In Rauterberg, M., Menozzi, M., Wesson, J., »Human-Computer Interaction, Interact 2003, Workshop Software and Usability Crosspollina tion - The Role of Usability Patterns«, Zürich (2003). http://wwwswt.infor matik.uni-rostock.de/deutsch/Interact/ index.html