Das SAP-Paketkonzept – Erfahrungen bei der Modularisierung bestehender Anwendungssysteme

Jörg Ackermann

SAP AG, Neurottstr. 16, 69190 Walldorf/Baden, Tel. +49 (6227) 747474, E-mail: joerg.ackermann@sap.com , *URL:* www.sap.com

Zusammenfassung. Neben der Kombination und Integration von Komponenten im Rahmen von mySAP.com™ gibt es bei SAP Bestrebungen zur weiteren Modularisierung innerhalb einzelner Anwendungssysteme. In diesem Zusammenhang steht das SAP-Paketkonzept, welches eine stärkere Entkopplung einzelner Anwendungsteile erlaubt. Erste Erfahrungen dazu stellen wir in diesem Beitrag vor. Dabei konzentrieren wir uns vor allem auf die Modellierung betriebswirtschaftlich geeigneter Pakete (Modularisierungseinheiten).

Schlüsselworte: Anwendungssysteme, Modularisierung, Modellierung, Vorgehensmodell

1 Einleitung

Mit mySAP.com™ stellt SAP seinen Kunden eine Komplettlösung zur Verfügung, die die Umsetzung einer Vielzahl betrieblicher Anwendungsfälle unterstützt. Das Spektrum reicht dabei von innerbetrieblichen Planungsabläufen bis zu kollaborativen e-Business-Szenarien zwischen verschiedenen Unternehmen. Komponenten gewinnen im verteilten Umfeld von Anwendungen, die über das Internet integriert werden, immer mehr an Bedeutung und sind auch für SAP von Interesse. (Zum Komponentenparadigma allgemein siehe z.B. Szyperski 1997). mySAP.com™ besteht aus verschiedenen Komponenten wie Financials, Human Resources, Customer Relationship Management oder Advanced Planner and Organizer. Die semantische Integration erfolgt über Business-Objekte und zugehörige BAPI-Schnittstellen. Das Internet Business Framework bietet eine Architektur, die die technische Integration auf der Grundlage von XML-Schnittstellen über das Internet ermöglicht.

Neben diesen Komponenten im Großen gibt es bei SAP weitere Bestrebungen zur verstärkten Modularisierung im Kleinen. Ziel ist eine stärkere technische Entkopplung von Anwendungsteilen unter Beibehaltung der hohen betriebswirtschaftlichen Integration. Wir versprechen uns davon eine höhere technische Verständlichkeit der Systeme, geringere Wartung und einfachere Weiterentwicklung. Dafür wurde das sogenannte Paketkonzept entwickelt, welches sich derzeit in der Pilotphase befindet. Dieser Beitrag diskutiert unser Vorgehen bei der Paketbildung und die gesammelten Erfahrungen.

Grundidee des Paketkonzepts ist, eine Anwendung (z.B. SAP R/3 oder SAP CRM) vollständig in Pakete zu zerlegen. Pakete können Schnittstellen definieren und den Zugriff durch andere Paketen auf ausgewählte Entwicklungsobjekte einschränken. Dadurch werden bestehende

Abhängigkeiten noch transparenter und nicht gewünschte Abhängigkeiten können (toolunterstützt) abgebaut werden. Der Grundgedanke von SAP-Paketen ist vergleichbar mit UML-Paketen plus einer UML-Abhängigkeitsbeziehung (zu UML siehe OMG 2000). Über die Modellierung hinaus werden die Pakete bei SAP auch in der Entwicklungsumgebung verankert, wodurch eine Toolunterstützung ermöglicht wird. Die technischen Aspekte des Paketkonzepts werden im Kapitel 2 näher erläutert.

Die richtige Definition der konkreten Pakete (Zerlegung des Systems) ist von entscheidender Bedeutung. Die Pakete sollen die betriebswirtschaftliche Struktur des Systems widerspiegeln, möglichst wenige Abhängigkeiten untereinander aufweisen und langfristig stabil sein. In den Kapiteln 3 und 4 werden unser Vorgehensmodell, die Modellierung der Pakete und die Dokumentation der Ergebnisse ausführlich vorgestellt.

Das Ziel des Paketkonzepts liegt in der SAP-internen Softwarestrukturierung. Es gibt derzeit keine Planungen, aus den Paketen (technisch echte) Komponenten zu machen. Allerdings sind aus unserer Sicht die betriebswirtschaftlich gewählten Pakete gerade (semantisch) geeignete Kandidaten für Fachkomponenten. Der Ansatz gibt damit Erfahrungen wieder, die für eine Komponentisierung bestehender Anwendungen sowie allgemein für Fachkomponenten interessant sind.

2 Das SAP-Paketkonzept

Die weiteren Ausführungen beziehen sich auf SAP R/3, gelten aber analog für SAP CRM, SAP APO etc.

2.1 Funktionsweise

SAP R/3 ist eine 3-Tier-Client-Server-Lösung (Datenbank, Applikationsserver, Frontend). Die betriebswirtschaftlichen Anwendungen werden mit Hilfe einer SAP-eigenen Entwicklungsumgebung und Programmiersprache (ABAP Objects) erstellt.

Das System R/3 setzt sich aus einer Vielzahl von eigenständigen *Entwicklungsobjekten* zusammen. Dabei kann es sich um Programme, Funktionsbausteine, Tabellen, Screens, Datentypen u.a. handeln. Die Entwicklungsobjekte sind heute in sogenannten *Entwicklungsklassen* gruppiert. Diese Container enthalten außer der Gruppierung keine weitere Semantik. (Der bei SAP verwendete Begriff „Entwicklungsklasse“ für eine Gruppierung von Entwicklungsobjekten ist sicher nicht so ganz glücklich. Er wird im folgenden immer dann verwendet, wenn zwischen dem bisherigen und dem geplanten Zustand unterschieden werden soll.) Das System R/3 besteht aus über 800.000 Entwicklungsobjekten in etwa 3.000 Entwicklungsklassen.

Pakete sind eine Weiterentwicklung der heutigen Entwicklungsklassen mit neuer zusätzlicher Semantik. Sie dienen zur besseren Modularisierung und Kapselung.

Ein *Paket* ist ein Container, der einzelne Entwicklungsobjekte (*Paket-Elemente*) und/oder andere Pakete enthält. Jedes Entwicklungsobjekt liegt in genau einem Paket. Ein Paket kann in (höchstens) einem anderen Paket enthalten sein.

Ein *Hauptpaket* ist ein Paket, das in keinem anderen Paket enthalten ist (d.h. ein Paket ohne Vater).

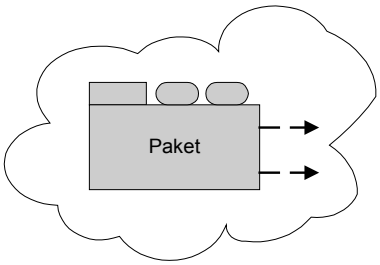
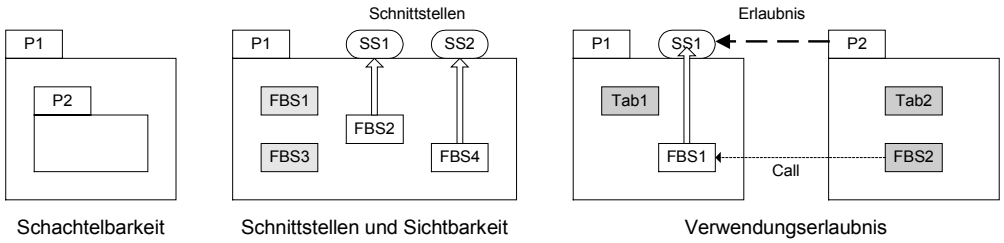


Bild 1: Grundidee eines Pakets

Pakete zeichnen sich gegenüber den bisherigen Entwicklungsklassen durch zusätzliche semantische Eigenschaften aus: Schachtelbarkeit, Schnittstellen und Sichtbarkeit, Verwendungserklärungen.

- *Schachtelbarkeit*: ist die Fähigkeit von Paketen, andere Pakete in sich einzubetten.
- *Sichtbarkeit*: ist eine Eigenschaft von Paket-Elementen. Ein Element kann außerhalb seines Pakets sichtbar sein. Es ist immer innerhalb seines Pakets sichtbar. Es ist nie in eingebetteten Paketen sichtbar. Ein Element kann nur nach außen sichtbar sein, wenn es in mindestens einer Paket-Schnittstelle liegt. Eine Schnittstelle ist eine Menge von Entwicklungsobjekten (des Pakets), die außerhalb des Pakets sichtbar (verwendbar) sein sollen.
- *Verwendungserklärung*: dokumentiert das einseitige Recht eines Paketes, die Elemente einer Schnittstelle eines anderen Pakets zu verwenden.



(Legende: P1, P2 = Pakete; SS1, SS2 = Schnittstellen; FBS1 – FBS4 = Funktionsbausteine; Tab1, Tab2 = Tabellen)

Bild 2: Die 3 Grundkonzepte von Paketen

Mittels Schnittstellen und Sichtbarkeit kann ein Paket sein Angebot an Dienstleistungen deutlich machen ("Was dürfen andere nutzen"). Was sichtbar ist, können andere Pakete potentiell nutzen. Was nicht sichtbar ist, können andere Pakete auch nicht nutzen. Damit kann ein Paket seine Elemente vor beliebiger Fremdverwendung schützen ("Was sollen andere nicht nutzen") und sein Inneres kapseln.

Verwendungserklärungen schränken die Nutzung von sichtbaren Elementen fremder Schnittstellen ein ("Nicht jeder darf alles nutzen"). Nur bei Vorliegen einer Verwendungserklärung auf eine Schnittstelle eines fremden Paketes kann ein Paket die sichtbaren Elemente in dieser Schnittstelle nutzen. Verwendungserklärungen werden im Einvernehmen zwischen beiden beteiligten Paketen vergeben.

Schachtelbarkeit erlaubt das technische, hierarchische Zergliedern von größeren Einheiten. Damit ist das System einfacher zu verstehen. Im Zusammenspiel mit Schnittstellen und Verwendungserklärungen lassen sich zusätzlich sehr viele Paketelemente auf einfache Weise "verstecken" und vor Nutzung schützen.

Jedes Paket kann sowohl in der Rolle des Anbieters (=Server) als auch der des Nutzers (=Client) von Services auftreten.

Mit diesen drei Fähigkeiten von Paketen besteht die Möglichkeit, R/3 in technische Einheiten (in Pakete) zu zerlegen und zu kapseln, die hohen Abhängigkeiten zu verringern und die Verständlichkeit des Systems zu erhöhen.

Sind Sichtbarkeiten und Verwendungserklärungen festgelegt, so kann deren Einhaltung sowohl zur Bauzeit als auch zur Laufzeit überprüft werden. Beispiele dafür sind: Darf beim …

- ... Anlegen eines komplexen Datentyps ein bestimmter einfacher Datentyp verwendet werden?
- ... Kompilieren eines Programms eine bestimmte Tabelle genutzt werden?
- ... dynamischen Aufruf eines Funktionsbausteins dieser wirklich aufgerufen werden?

Die Prüfungen und die Reaktionen auf Verstöße (Ignorieren, Warnung, Abbruch) sind in Stufen einstellbar. Je nach Bedarf (z.B. aus Performance-Gründen oder in produktiven Kundensystemen) können auch alle Prüfungen abgeschaltet werden.

2.2 Ziele und Nutzen

Zweck der Einführung von Paketen ist, die Voraussetzungen für eine höhere technische Modularisierung bei Erhalt der logischen Integration zu schaffen. Im einzelnen werden die folgenden Ziele für die (Weiter-) Entwicklung, Wartung, Administration und Nutzung von R/3 angestrebt:

- Festlegen des Angebots und Schutz vor Verwendung

Ein Paket kann seine Dienstleistungen über die Paket-Schnittstelle deutlich machen ("was soll verwendet werden") und gleichzeitig verhindern, dass jedes seiner Elemente zugänglich ist ("was darf nicht verwendet werden"). Pakete sind damit gekapselt.

- Reduzieren und Verdeutlichen von Abhängigkeiten

Durch geeignete Zerlegung der Pakete lässt sich die Lokalität innerhalb der Pakete erhöhen. Pakete werden weniger fremde Elemente verwenden. Anhand der Verwendungserklärungen sind die Abhängigkeiten zwischen Paketen jederzeit ersichtlich. Mittels Schachtelung oder gerichteten Verwendungserklärungen lassen sich Pakete leicht in Schichten anordnen.

- Erkennen inkompatibler Änderungen; leichteres Reorganisieren

Es besteht die Möglichkeit, die Schnittstellen von Paketen einzufrieren bzw. nur noch kompatibel zu erweitern. Syntaktische Änderungen an Schnittstellen können so verhindert oder zumindest maschinell erkannt werden. Außerdem können bei konstanten Schnittstellen die Pakete ohne Schaden für ihre Verwender beliebig reorganisiert werden.

- Verringern der technischen Komplexität; höhere technische Verständlichkeit

Aufgrund der reduzierten und ersichtlichen Abhängigkeiten und aufgrund der Schnittstellen

verringert sich aus technischer Sicht die Komplexität von R/3. Das System wird technisch einfacher zu verstehen sein.

- Schnellere und kostengünstigere Änderbarkeit

Direkte Folge der verringerten Komplexität sind einfachere, schnellere und weniger aufwendige (Weiter-) Entwicklung, Wartung und Implementierung. Auslieferung, Releasewechsel und interne Transporte werden ebenfalls einfacher und schneller.

- Höhere Unabhängigkeit von Entwicklungsbereichen innerhalb und außerhalb der SAP

Die organisatorische Entkopplung der Entwicklungsbereiche (bis hin zu Partner- oder Kundenentwicklungen) kann durch die technische Entkopplung und das Erkennen von Abhängigkeiten unterstützt werden. Die einzelnen Bereiche sind weniger aufeinander angewiesen und können mit geringerem Aufwand ihre (Teil-) Produkte erstellen.

2.3 Aktueller Status und geplantes Vorgehen

Um die genannten Ziele zu erreichen, sind ein geeignetes Vorgehen und die richtige Definition der Pakete (Zerlegung des Systems) von entscheidender Bedeutung. Das geplante Vorgehen setzt sich aus den folgenden Schritten zusammen:

1. Analyse der aufzubauenden Paketlandschaft

In der Analysephase werden die Hauptpakete und deren Aufgaben bestimmt. Für jedes Hauptpaket werden als Soll-Beschreibung seine inhaltlichen Aufgaben, anzubietende Services, Beteiligung an Prozessen und seine Abgrenzung zu anderen Hauptpaketen festgelegt. Bei der Ist-Beschreibung wird ermittelt, welche heutigen Entwicklungsklassen dem Hauptpaket zuzuordnen sind und welche Defizite noch zum Soll-Zustand bestehen. Außerdem sollte schon eine Vorstellung entstehen, in welcher Form das Hauptpaket seine Services anbieten wird und welche Teilpaketkandidaten existieren.

Das Vorgehen und die Erkenntnisse aus der Analysephase werden ausführlich im Kapitel 3 diskutiert.

Status des Projektes: Für SAP R/3 wurde ein erster, flächendeckender Paketvorschlag erstellt. Darüber hinaus wurde in ausgewählten Bereichen auch die Analyse durchgeführt. Die folgenden Punkte wurden noch nicht durchgeführt und spiegeln nur unsere derzeitigen Planungen wider.

2. Definition der Pakete im System

Die Pakete werden zu einem Stichtag flächendeckend eingeführt. Dazu werden die in der Analysephase bestimmten Hauptpakete definiert. Jede heutige Entwicklungsklasse wird genau einem Hauptpaket zugeordnet und wird zu einem Teilpaket.

Es werden vom System alle bestehenden Verwendungen zwischen Paketen errechnet. Alle verwendeten Entwicklungsobjekte eines Pakets werden in eine Altschnittstelle aufgenommen. Dadurch wird ein vorläufiger Bestandsschutz gewährleistet. Die Altschnittstelle darf jedoch nicht mehr erweitert werden und muss schrittweise abgebaut werden.

Es ist nicht immer möglich, eine heutige Entwicklungsklasse vollständig einem Hauptpaket zuzuordnen. Deshalb müssen die Entwicklungsklassen vorher bereinigt werden. Dazu werden einzelne Entwicklungsobjekte anderen Entwicklungsklassen zugewiesen oder ganze Ent-

wicklungsklassen zerlegt. Schwieriger ist der Fall, wenn einzelne Entwicklungsobjekte (z.B. Programme) zerlegt werden müssen. Ein solches Redesign der Anwendung wird nicht immer im Vorfeld möglich sein.

3. Anbieten von Services über Schnittstellen

Jedes Hauptpaket legt fest, welche Services es in welcher Form anbietet. Diese werden dann in neu zu definierende Schnittstellen aufgenommen und ersetzen mittelfristig die Altverwendungen. Ein Paket-Architekturteam legt fest, welche Arten von Verwendung möglich sind (in Abstufung erwünscht, erlaubt, vorläufig geduldet). So könnte beispielsweise der direkte Zugriff auf Tabellen und Daten eines anderen Paketes ausgeschlossen werden.

4. Umstellen der Verwender auf die neuen Schnittstellen

Die Nutzer von Services müssen sich dann (mit Übergangsfristen) auf die neuen Schnittstellen umstellen und die Verwendungserklärungen mit dem „Server"-Paket absprechen.

Geplant ist, dass ein Paket-Architekturteam pro Release Vorgaben zur Umsetzung des Paketskonzept gibt. Solche Vorgaben können erlaubte Typen von Verwendungen (z.B. Funktionsbausteinsverwendung vs. Tabellenzugriff), Auslauffrist der Altschnittstelle oder eine Schichtung von bestimmten Paketen sein. Gesteuert werden solche zentralen Vorgaben durch Tools und die geeignete Wahl der Fehlerschwere (Warnung vs. Abbruch) bei Verstößen.

3 Modellierung geeigneter Pakete

In diesem Kapitel beschäftigen wir uns mit der Modellierung von Hauptpaketen. Durch das Paketkonzept soll eine bessere Kapselung der einzelnen Anwendungsteile erreicht und die bestehenden Abhängigkeiten expliziter erkennbar werden. Dafür ist vor allem die richtige Definition der Hauptpakete von entscheidender Bedeutung, da diese die oberste Paketschicht bilden und im wesentlichen die Zerlegung des Systems darstellen. Innerhalb von Hauptpaketen können Anwendungsbereiche durch Teilpakete weiter strukturiert werden. Dies ist zumeist nur von lokaler Bedeutung und hat wenig Einfluss auf die übergreifende Paketlandschaft.

Deswegen liegt der primäre Fokus zunächst auf dem Bestimmen geeigneter Hauptpakete. Alle Ausführungen in diesem Kapitel beziehen sich auf Hauptpakete, auch wenn der Kürze halber oft nur von Paketen gesprochen wird.

3.1 Kriterien bei der Modellierung

In diesem Abschnitt erläutern wir unsere Kriterien bei der Modellierung von Hauptpaketen. Die Reihenfolge spiegelt in etwa ihre Wichtigkeit wider. Es ist klar, dass im Einzelfall nicht alle Kriterien gleichzeitig erfüllt werden können, da diese sich teilweise widersprechen können. Im jeweils konkreten Fall muss ein möglichst guter Kompromiss gefunden werden.

- Stabilität

Für die Reduktion der technischen Komplexität und den Abbau von Abhängigkeiten ist es notwendig, dass die Pakete langfristig stabil bleiben. Änderungen an der Paketlandschaft müssen zwar möglich sein, sollen aber minimiert werden.

- Abbilden der betriebswirtschaftlichen Struktur

Als primäres Kriterium für das Bilden der Pakete dient die betriebswirtschaftliche Struktur

des Anwendungssystems. Die heutige technische Struktur oder organisatorische Struktur spielen nur eine untergeordnete Rolle. Nur so kann die langfristige Stabilität gewährleistet werden.

- Richtige Zuordnung von Services

Die Services, welche ein Paket für andere erbringt, konstituiert einen Hauptteil der Definition eines Pakets. Deshalb ist die richtige Zuordnung der Services von entscheidender Bedeutung. Auch hier ist das primäre Kriterium die betriebswirtschaftlich sinnvolle Zuordnung und weniger die heutigen technischen und organisatorischen Gegebenheiten.

- Eindeutige personelle Verantwortlichkeit

Die erfolgreiche Umsetzung des Konzepts verlangt organisatorische Maßnahmen im Entwicklungsprozess. Deshalb muss es für jedes Paket einen eindeutigen Verantwortlichen geben.

Im Einzelfall sind Konflikte zwischen der betriebswirtschaftlichen Struktur und der personellen Verantwortung zu erwarten. Dann muss ein möglichst guter Ausgleich zwischen diesen Kriterien erreicht werden.

- Minimierung der Abhängigkeiten / Schnittstellen

Haben zwei Pakete viele, detaillierte Abhängigkeiten untereinander, so ist dies oft ein Zeichen dafür, dass die Pakete falsch geschnitten wurden. Es kann aber auch sein, dass die heutige technische Struktur nicht der betriebswirtschaftlichen Struktur entspricht und ein Redesign sinnvoll wäre. Auch hier muss im Einzelfall ein geeigneter Ausgleich zwischen den Kriterien gefunden werden.

- Ausgewogene Granularität

Die Pakete müssen eine ausgewogene Granularität („nicht zu grob und nicht zu fein“) besitzen. Bei „zu großen“ Paketen wird die angestrebte Entkopplung kaum unterstützt und es können keine klaren Verantwortlichen gefunden werden. Bei „zu kleinen“ Paketen wird das Gesamtsystem zu unübersichtlich und es entstehen zu viele und potentiell unnötige Abhängigkeiten.

- Gleichmäßige Granularität

Die Größe der Pakete sollte über die verschiedenen Anwendungen hinweg vergleichbar sein. Dies muss durch zentrale Vorgaben und organisatorische Begleitung des Prozesses sichergestellt werden. Es wird aber natürlich in allen Anwendungsbereichen eher kleinere (z.B. Tools) und eher größere Pakete geben.

Der Vollständigkeit halber ist hier noch ein technisches Kriterium angegeben, welches aufgrund des Paketkonzepts an sich in jedem Fall erfüllt sein muss:

- Vollständigkeit und Disjunktheit

Alle Entwicklungsobjekte werden genau einem Paket zugeordnet. Damit wird eine Anwendung vollständig in Pakete zerlegt und es gibt keine Überschneidungen zwischen den Paketen. Entwicklungsobjekte können nicht mehreren Paketen gehören, aber natürlich von anderen Paketen verwendet werden.

Darüber hinaus wurde vor Beginn des Projektes die Grundannahme getroffen, dass sich das R/3-System aus 150-250 Hauptpakten zusammensetzen sollte. Damit war eine grobe Entscheidung über die Granularität der Pakete gegeben. Diese Vorgabe schien uns am Besten den sinnvollen Kompromiss in Bezug auf die Granularität darzustellen. Dies bedeutet insbesondere, dass Entwicklungsgruppen von bis zu 20 Entwicklern ein Hauptpaket betreuen werden.

3.2 Vorgehensweise

Die Analyse der möglichen Hauptpakete erfolgt in den einzelnen Entwicklungsbereichen. Unterstützt werden diese von einer zentralen Paketgruppe, die aus modellierungserfahrenen Kollegen besteht. Die Aufgaben der Paketgruppe sind:

- Erstellung eines ersten Vorschlags der Hauptpaketlandschaft
- Festlegen einer geeigneten Vorgehensweise und der Modellierungskriterien
- intensive Betreuung von Pilotkandidaten (und Rückführung der Erfahrungen in den Prozess)
- Unterstützung der Bereiche bei der Analyse der Pakete (insbesondere bei kritischen Aspekten)

Im folgenden stellen wir das Vorgehen in der Analysephase vor. Den Projektgruppen wurden die zu durchlaufenden Schritte, die jeweils zu erfassenden Informationen und die Modellierungskriterien zur Verfügung gestellt. Auf dieser Grundlage konnten dann die notwendigen Entscheidungen über die geeigneten Pakete gefällt werden.

Erläutert werden die diskutierten Fragestellungen am Beispiel des Bereichs Bestandsführung. Dieser Bereich gehörte zu den Pilotbereichen, in denen die gesamte Analysephase schon abgeschlossen wurde. Er ist auch deshalb von Interesse, da sich in diesem gleich mehrere kritische Fragestellungen ergaben. Die angegebenen Beispiele dienen primär zur Erläuterung der einzelnen Punkte des Vorgehens. Sie spiegeln deshalb nur wenige ausgewählte Aspekte der Bestandsführung wider. Eine vollständige Diskussion dieses Beispiels würde über den Rahmen dieses Beitrags hinausgehen.

3.2.1 Grobvorschlag einer Hauptpaketlandschaft

Die zentrale Paketgruppe erstellte zunächst einen ersten, flächendeckenden Vorschlag für die möglichen Hauptpakete. Grundlage dafür waren frühere Modellierungsergebnisse und Kurzinterviews in den einzelnen Entwicklungsbereichen. Dabei wurde bestimmt (jeweils undetailliert):

- Hauptpaket, Funktionsumfang und wichtigste Aufgaben,
- Abhängigkeiten zu anderen Paketen
- auftretende Probleme und offene Fragen,
- Liste der beteiligten (heutigen) Entwicklungsklassen.

Gehört eine der Entwicklungsklassen nicht eindeutig zu einem Paket, wurde sie (mit einem Vermerk) dem geeignetsten Paketkandidaten zugeordnet.

Der so entstandene Vorschlag ist vollständig und disjunkt. Änderungen an diesem Vorschlag

sind natürlich möglich. Er dient aber als Rahmen für die weitergehende Analyse in den einzelnen Entwicklungsbereichen. Aus unserer Sicht ist es wichtig, dass dieser flächendeckende Grobvorschlag vorliegt, bevor mit der eigentlichen Analyse begonnen wird. Dieser garantiert eine gleichmäßige Granularität über die verschiedenen Bereiche hinweg und sichert, dass einzelne Anwendungsteile nicht vergessen oder stillschweigend von mehreren Gruppen für sich reklamiert werden.

3.2.2 Anforderungsanalyse und Zielkonsens

Für alle weiteren Schritte wird pro Hauptpaketkandidat eine Arbeitsgruppe gebildet. Diese besteht aus Kollegen aus den entsprechenden Entwicklungsbereichen und kann im Einzelfall durch einen Vertreter der zentralen Paketgruppe ergänzt werden. Die weiteren Schritte wurden in ausgewählten Pilotbereichen durchgeführt, aber noch nicht flächendeckend umgesetzt.

Diese Phase dient zur Ist-Beschreibung und zur groben Abgrenzung des Hauptpakets. Außerdem soll ein Konsens über die grundsätzlichen Ziele der beteiligten Entwicklungsabteilungen beim Paketkonzept gefunden werden. Im einzelnen sind folgende Fragen zu beantworten:

1. *Ausgangslage / Ist-Situation*

- Beschreibung des zu bildenden Pakets erstellen: vorhandene Dokumente zusammenstellen

  Beispiel: Paket Bestandsführung; vorhandene Kurzbeschreibung beschreibt das Gebiet hinreichend

- Liste der beteiligten Entwicklungsklassen (inklusive teilweiser beteiligter); Abgleich mit Grobvorschlag
- bekannte Defizite der existierenden Entwicklungsklassen
- Kandidaten für Paketverantwortlichen

2. *Zentrale betriebswirtschaftliche Begriffe (Inhalt des Pakets)*

- Liste der Begriffe zur betriebswirtschaftlichen Beschreibung des Pakets

  Beispiel:

Begriff	*Definition*
Wareneingang (WE) zur Bestellung	*WE eines bestellten Materials in des Lager oder in den Verbrauch*
Warenausgang	*Warenausgang aus dem Lager*
Umbuchung	*Zustandsänderung eines Materials*
...	

3. *Einflüsse / Abhängigkeiten / Umfeld*

- Liste der Pakete/Gebiete, die verwendet werden (möglichst vollständig)
- Liste der Pakete/Gebiete, die mich verwenden (einige typische)

  Beispiel: Bestellung, Fertigungsauftrag (beide Richtungen), Materialstamm, Material-

Ledger (wird verwendet), Vertrieb (verwendet), ...

4. *Anforderung und Ziele an das Paketkonzept (aus Sicht der Entwicklungsabteilung)*

- Liste der Anforderungen und deren Gewichtung

Beispiel:

Erhoffter Nutzen	*Erläuterung*	*Priorität (A/B/C)*
Arbeitserleichterung	*Weniger Seiteneffekte bei Änderungen, leichtere Wartbarkeit*	*B*
Bessere Kapselung	*z.B. direkte Zugriffe durch Funktionsbausteine ersetzen*	*B*
...		

Für einige der Punkte können die Ergebnisse bei der Erstellung des Grobvorschlags wiederverwendet werden. Diese Phase sollte nach einer Gruppensitzung abgeschlossen sein.

3.2.3 Feinanalyse des Hauptpakets

In dieser Phase soll das Paket in seinen wesentlichen Zügen festgelegt werden. Dazu sollten ein bis drei Gruppensitzungen verwendet werden. Es müssen die folgenden Fragen beantwortet werden:

1. *Abgrenzung des Pakets*

- zukünftige Kapselung (Soll-Zustand), Vergleich mit Ist-Zustand
- Validierung der Festlegungen des Grobvorschlags, gegebenenfalls verändern
- Überprüfung der personellen Verantwortlichkeit für das Paket

Beispiel: Aufgabe der Bestandsführung ist die Fortschreibung von Warenzugängen oder –abgängen inklusive ihrer Bewertung. Damit gehört insbesondere die wertmäßige Bestandsführung mit zum Paket. Die Bilanzbewertungsverfahren gehören nicht zur Bestandsführung. Fortschreibung von Preisänderung ist die Aufgabe eines Pakets Material-Ledger.

Wem sollen die Materialbestände (Verwaltung und Fortschreibung) zugeordnet werden: der Bestandsführung oder dem Materialstamm? Da die Materialbestände eine sehr enge Kopplung mit ihren Fortschreibungsregeln und eine eher lose Kopplung mit Materialbearbeitungsservices (z.B. Anlegen eines Materials) haben, gehören sie semantisch zur Bestandsführung. Allerdings liegen die Materialbestände technisch und organisatorisch in Verantwortung des Materialstamms. Ein Redesign bedeutet einen größeren Aufwand.

Lösung: Die Bestandsservices („Bestände fortschreiben“ bzw. „Bestände holen“) gehören technisch dem Materialstamm. Die Fortschreibung erfolgt jedoch durch die Bestandsführung und wird von ihr in Form von „Services zu Warenbewegungen“ anderen Paketen angeboten. Dazu werden die Services in der Bestandsführung (entsprechend „verpackt“) in Form von „Warenbewegungen anlegen/stornieren“ bzw. „Bestände holen“ angeboten. Die Services bei der Bestandsführung sind allgemein verfügbar, die am Materialstamm nur für das Paket Bestandsführung.

2. *Geschäftsvorfälle*

- Liste aller Geschäftsvorfälle, an denen das Paket beteiligt ist

  Beispiel: Wareneingang zur Bestellung, Wareneingang zum Fertigungsauftrag, Warenausgang, Umbuchungen, Umlagerungen, Materialreservierung, Mengenmäßige Inventur

3. *Pakete im Umfeld*

- Ermittlung aller vorausgesetzten und aller abhängigen Pakete (dies soll aus den Geschäftsprozessen abgeleitet werden)

  Beispiel:

Fremdes Paket/Gebiet	*Geschäftsvorfall*	*Rolle des fremden Pakets*
Bestellung	*WE zu Bestellung*	*verwendet / wird verwendet*
Fertigungsauftrag	*WE zu Fertigungsauftrag*	*verwendet / wird verwendet*
Vertrieb	*Wareausgang zu Lieferung*	*verwendet*
Materialstamm	*Lesen der Bestände*	*wird verwendet*
...		

4. *Beteiligte Business-Objekte*

- Liste aller anzubietenden Dienste (unabhängig von heutiger Realisierung)
- Liste aller benötigten Business-Objekte (vorhandene plus zusätzliche) und deren Beziehungen

  Beispiel: Business-Objekt GoodsMovement

anzubietender Dienst	*heute realisiert in Form von*	*Bemerkung*
Create (Anlegen)	*BAPI & lokaler Funktionsbaustein*	*beides weiterhin sinnvoll*
Cancel (Stornieren)	*BAPI & lokaler Funktionsbaustein*	*beides weiterhin sinnvoll*
...		

5. *Teilpakete identifizieren (falls zutreffend)*

   Beispiel: mögliche Teilpakete sind Warenbewegung, Materialreservierung, Inventur und Services Bestandsführung; vollständige Kapselung der Teilpakete hat vorläufig eine geringere Priorität

3.2.4 Technische Detailanalyse

Diese Phase dient der objektweisen Festlegung des Paketinhalts. Es wird im Detail entschieden, welchem Paket jedes einzelne Entwicklungsobjekt zuzuordnen ist. Diese Phase muss in mehreren Gruppensitzungen behandelt werden und bedarf einer umfangreicheren Vor- und Nachbereitung. Dazu wird untersucht:

1. *Objektweise Analyse der (bisherigen) Entwicklungsklassen*

- Überprüfen der Zuordnung jeder Entwicklungsklasse zu einem Paket; Überprüfung aller Entwicklungsobjekte auf die richtige Zuordnung; Bereinigung der Entwicklungsklassen

2. *Technische Validierung des Pakets*

- testweise Definition des Pakets; Überprüfung aller Abhängigkeiten; Validierung der in der Analyse getroffenen Annahmen zu Abhängigkeiten; ggf. zur Feinanalyse zurückgehen
- Überprüfung, ob der Schnitt des Pakets rundum abgesichert ist

3. *Festlegen geeigneter Schnittstellen pro Geschäftsvorfall*

Bei der technischen Detailanalyse handelt es nicht mehr um die Modellierung der Pakete an sich. Deshalb soll darauf an dieser Stelle nicht näher eingegangen werden.

3.3 Aufgetretene Modellierungsprobleme

Mit den vorgegebenen Modellierungskriterien und vorhandenem Domänenwissen war es im allgemeinen relativ einfach, geeignete Pakete zu definieren. In einigen Fällen jedoch führten die unterschiedlichen Kriterien zu stärkeren Widersprüchen. Dann wurden verschiedene Alternativen formuliert und im Einzelfall eine der Alternativen ausgewählt. Bei diesen Problemfällen wiederholten sich bestimmte Fragestellungen. Einige dieser typischen Konflikte bei der Modellierung und mögliche Lösungen wollen wir in diesem Abschnitt diskutieren.

- Betriebswirtschaftliche versus organisatorische Strukturierung

Bei der Bildung der Hauptpakete soll nicht die heutige organisatorische Struktur abgebildet werden. Andererseits sollte jedes Paket einen eindeutigen Verantwortlichen haben. Findet sich zu einem semantischen Paketkandidaten kein Verantwortlicher, dann sind kompakte Teilbereiche mit jeweils klarer Verantwortung zu identifizieren. Dies ist in den meisten Fällen möglich. Sollte eine Veränderung der Organisation nicht sinnvoll sein, sollte geprüft werden, ob die Teilbereiche stattdessen die Hauptpakete werden. Dies ist ein sinnvoller Kompromiss zwischen der betriebswirtschaftlichen und einer rein organisatorischen Strukturierung. Es ist denkbar, die Pakete zu einem späteren Zeitpunkt wieder zusammenzufassen. Dies wäre weder für die Paketbesitzer noch für die Verwender mit einem größeren Aufwand verbunden.

- Abgeschlossenheit versus Disjunktheit

Beim Bilden von Softwarekomponenten ergibt sich oft ein Widerspruch bei der Forderung

nach in sich abgeschlossenen, aber disjunkten Komponenten (siehe z.B. Fellner/Rautenstrauch/Turowski 1999). Bei unserem Paketkonzept tritt das Problem in dieser Form nicht auf: Jedes Entwicklungsobjekt muss genau einem Paket zugeordnet werden. Damit ist die (technische) Disjunktheit automatisch gegeben. Eine vollständige Abgeschlossenheit der Pakete kann damit nicht immer garantiert werden. Dies ist aber auch gar nicht das Ziel. Unsere Pakete müssen nicht einzeln lauffähig sein und dürfen von anderen Paketen abhängen. Muss z.B. ein Paket A einen Service X von Paket B nutzen, um seine Aufgaben erfüllen zu können, dann ist die Abhängigkeit des Pakets A von Paket B gewollt und sinnvoll. Handelt es sich jedoch um eine Abhängigkeit rein technischer Natur (Paket A nutzt aus historischen Gründen ein bestimmtes Entwicklungsobjekt von Paket B), dann sollte die Abhängigkeit im Idealfall abgebaut werden.

- Richtige Zuordnung von Services

Das Problem der semantischen Abgeschlossenheit kann allerdings bei der betriebswirtschaftlich richtigen Zuordnung der Services auftreten. Hierbei kann es zu einem Konflikt zwischen der betriebswirtschaftlichen und einer eher technischen Zuordnung kommen. So könnte zum Beispiel ein Service X semantisch zu einem Paket A gehören, ist aber heute technisch und organisatorisch eng mit dem Paket B verwoben. Eine mögliche Lösung ist: Der Service X wird vom Paket A seinen Verwendern zur Verfügung gestellt. Ist ein Redesign (vorläufig) nicht möglich, verbleibt die eigentliche Realisierung des Services beim Paket B. Paket A nutzt seinerseits den Service X beim Paket B, den letzteres in einer „privaten“ Schnittstelle Paket A anbietet. Ein eventuelles späteres Redesign betrifft dann nur die Pakete A und B und nicht die Verwender des Services X.

- Mehrfach genutzte Tabellen

In einigen Fällen wird eine Datenbanktabelle von mehreren Paketen genutzt. Eine Aufteilung in mehrere Tabellen ist oft nicht sinnvoll, da dies für die Kunden im allgemeinen einen erheblichen Umstellungsaufwand bedeutet. Welchem Paket soll die Tabelle zugeordnet werden? Mögliche Lösung: Die Tabelle wird einem der Pakete zugeordnet. Dieses Paket bietet eher technische Services zum Zugriff auf diese Tabelle an, die nur von den anderen „Tabellenbesitzern“ verwendet werden dürfen. Jeder der Tabellenbesitzer bietet seinerseits allgemeine Services an, über die die betriebswirtschaftliche Funktionalität durch alle andere Pakete genutzt werden kann.

4 Spezifikation und Dokumentation der Pakete

Damit die Pakete sinnvoll verwendet werden können, müssen sie ausreichend dokumentiert und ihre Eigenschaften spezifiziert sein. Dazu werden für jedes Paket einige Attribute, seine Bestandteile und Dokumentation erfasst.

Attribute und Bestandteile sind die eher formal spezifizierten Eigenschaften eines Pakets. Diese werden im System selbst abgelegt und sind maschinell auswertbar.

Darüber hinaus wird Dokumentation in Prosaform erfasst, die Semantik und Inhalt eines Paketes beschreiben. Um eine einheitliche Darstellung zu erreichen, wurden Richtlinien und Templates erstellt. Die Dokumentation ist integraler Bestandteil des Pakets und wird im System selbst erfasst und verwaltet. Bei der Dokumentation wird unterschieden zwischen

- Dokumentation des Pakets an sich,

- Dokumentation aller seiner Schnittstellen,
- Dokumentation der Entwicklungsobjekte in den Schnittstellen.

Die einzelnen Entwicklungsobjekte (Funktionsbausteine, Datentypen etc.) werden unabhängig vom Paketkonzept nach entsprechenden Richtlinien dokumentiert. Wir gehen in den Abschnitten 4.2 und 4.3 näher auf die Dokumentation von Paketen und ihren Schnittstellen ein.

4.1 Attribute und Bestandteile von Paketen

Für jedes Paket werden die folgenden Attribute im System definiert:

- Allgemeine Daten zum Paket (Name, Kurzbeschreibung, gegebenenfalls Vaterpaket),
- Verwaltungsdaten (angelegt von, Anlegerelease, Transportschicht,...),
- Zuordnung eines Pakets zur betriebswirtschaftlichen Komponentenhierarchie (FI, CO etc.).

Die Bestandteile eines Pakets ergeben sich aus

- zugeordneten Entwicklungselementen und Teilpaketen,
- Schnittstellen eines Pakets, Entwicklungselemente in Schnittstellen,
- Verwendungserklärungen (von anderen Paketen, zu anderen Paketen),
- Freigabestatus einzelner Schnittstellenelemente (es kann unterschieden werden zwischen (nur) verwendbar und (langfristig) garantiert).

Alle diese Informationen werden in der Entwicklungsumgebung verwaltet und stehen damit sowohl zur Entwicklungs- als auch zur Laufzeit maschinell auswertbar zur Verfügung.

4.2 Dokumentation der Pakete

Die Dokumentation von Paketen ist für Entwickler (SAP, Partner, Kunden) bestimmt. Sie soll Aufschluss geben, wie sich die betriebswirtschaftlichen Inhalte technisch im R/3-System verteilen. Aus der Dokumentation geht hervor, welches Paket welche Services anbietet bzw. anbieten sollte. In der Dokumentation inhaltlich zusammengehöriger Pakete sollte gegenseitig aufeinander verwiesen werden. Die Dokumentation eines Hauptpakets sollte insgesamt 2 Seiten nicht überschreiten. Sie sollte nicht auf technische Namen oder Details eingehen und insbesondere nicht die Funktionsweise einzelner Paket-Elemente erläutern.

Im folgenden findet sich eine Aufstellung, aus welchen Themenblöcken eine Paketdokumentation besteht und welchen Inhalt diese haben. Auf die Veranschaulichung anhand unseres Beispiels Bestandsführung wird an dieser Stelle verzichtet, da dies über den Rahmen dieses Beitrags hinausgehen würde.

1. *Verantwortlichkeiten des Paketes*

- Welchen abgegrenzten Teilbereich umfasst das Paket ?
- Welches sind die betriebswirtschaftlichen Inhalte des Paketes ?
- Welche Aufgaben und Verantwortlichkeiten nimmt das Paket konkret wahr ?
- (Grob:) An welchen Geschäftsprozessen ist das Paket beteiligt (z.B. Kundenauftrags-

abwicklung, Beschaffung, Planungsprozesse in Produktion und Kostenrechnung, ...) ?

- Welcher Teilprozess wird gegebenenfalls realisiert (z.B. Lieferung, Bezugsquellenfindung,...) ?

2. *Zentrale Services des Paketes*

- Welche prozessorientierten Services bietet das Paket an ?
- Welche weiteren Services werden angeboten? (Stammdatenpflege, Reporting, Customizing,...)

  Bemerkung: Standardservices wie Anlegen, Anzeigen, Ändern müssen nicht dokumentiert werden.

- Welche Business-Objekte realisieren diese Services?

3. *Zentrale Abhängigkeiten des Paketes*

- Welche zentralen Services anderer Pakete werden verwendet ?
- Welche Services werden vorausgesetzt ?

4. *Verweise auf inhaltlich zugehörige Pakete*

- Welche Pakete enthalten verwandte oder weiterführende Funktionalität ?

4.3 Dokumentation der Paketschnittstellen

Ein Paket kann mehrere Schnittstellen zu seiner Funktionalität anbieten. Dies eröffnet die Möglichkeit, Verwendungserklärungen differenzierter zu verteilen. Bietet z.B. ein Paket X die Services A und B an, dann kann einem Paket Y Zugriff auf A und einem Paket Z Zugriff auf A und B gestattet werden. Entsprechend können Entwicklungsobjekte auch mehreren Schnittstellen ihres Pakets zugeordnet sein.

Das Design der einzelnen Schnittstellen sollte sich an den Services ausrichten, die das Paket anbietet. Im allgemeinen wird es pro Geschäftsvorfall eine Schnittstelle geben, die alle benötigten Elemente enthält, um den Service in Anspruch nehmen zu können. Genau daran orientieren sich auch die Richtlinien, wie Paketschnittstellen zu dokumentieren sind:

1. *Kurzbeschreibung der Paketschnittstelle*

- Welchen Hauptverwendungszweck hat die Schnittstelle ?
- Welchen Funktionalitätsbereich umfasst sie ?

2. *Zentrale Services der Schnittstelle*

- Welche Services bietet diese Schnittstelle an ?
- Welches Business-Objekt steht dahinter ?

5 Zusammenfassung

Das SAP-Paketkonzept erlaubt eine bessere technische Strukturierung unserer bestehenden Anwendungen. Mit den Konstrukten Kapselung, Schnittstellen und Verwendungserklärung

können einzelne Anwendungsteile noch stärker entkoppelt werden, und bestehende Abhängigkeiten werden transparenter.

Für die langfristige Stabilität spielt die richtige Wahl der Pakete eine wichtige Rolle. Die Kriterien bei der Modellierung und das Vorgehensmodell wurden in diesem Beitrag vorgestellt. Mit diesen Hilfsmitteln und dem vorhandenen Domänenwissen war es im allgemeinen relativ einfach, die geeigneten Pakete zu finden bzw. konkurrierende Alternativen zu formulieren.

Das Projekt befindet sich in der Pilotphase. Daher können zum jetzigen Zeitpunkt keine Aussagen zum tatsächlichen Nutzen getroffen werden. In der Zukunft sollen aus den weiteren Erfahrungen geeignete Verbesserungen am Vorgehensmodell und den Modellierungskriterien abgeleitet werden.

Danksagung

Die hier vorgestellten Konzepte und Erfahrungen sind unter Beteiligung einer Vielzahl von Kollegen bei SAP entstanden. Insbesondere zu nennen sind die Kollegen der zentralen Paketgruppe (Kurt Reiner) und der beteiligten Pilotbereiche. Bei der Erstellung dieses Beitrages konnte auf bestehende SAP-interne Dokumente zurückgegriffen werden: Pakete – White Paper zum Paketkonzept (Achim Magel, Manfred Schneider); Vorgehensweise bei Paketbildung (Kurt Reiner, Michael Wagener, Jürgen Wagner, Maic Wintel); Kriterien für die Bestimmung von Paketen - Erfahrungen aus den Pilotprojekten (Claus von Riegen); Ergebnisprotokoll Bestandsführung (Gerhard Maier, Marcus Pfeifer, Kurt Reiner, Claus von Riegen, Jürgen Wagner, Karl Wagner). Ich bedanke mich bei allen Kollegen, die mich bei der Erstellung dieses Beitrags unterstützt haben.

Literatur

Fellner, K.; Rautenstrauch, C.; Turowski, K.: Fachkomponenten zur Gestaltung betrieblicher Anwendungssysteme. In: IM Information Management & Consulting 14 (1999) 2, S. 25-34.

OMG (Hrsg.): Unified Modelling Language Specification, Version 1.3. OMG. http:///www.omg.org , Abruf am 2000-07-12.

SAP (Hrsg.): SAP Homepage, http://www.sap.com , Abruf am 2000-07-12.

Szyperski, C.: Component Software – Beyond Object-Oriented Programming. Addison Wesley Ltd., 1997.