Spezifikation des Parametrisierungsspielraums von Fachkomponenten – Erste Überlegungen

Jörg Ackermann

Von-der-Tann-Str. 42, 69126 Heidelberg, Deutschland, E-Mail: joerg.ackermann.hd@t-online.de

Zusammenfassung. Die geeignete Spezifikation einer Softwarekomponente ist eine wesentliche Voraussetzung für ihre erfolgreiche Wiederverwendung. Von der GI-Arbeitsgruppe 5.10.3. wurde ein Vorschlag erstellt, wie die Spezifikation von Fachkomponenten standardisiert werden kann. Nicht berücksichtigt wurde bisher, wie möglicher Parametrisierungsspielraum von Fachkomponenten spezifiziert werden soll. Dieser Beitrag enthält dazu erste Überlegungen. Nach einem Überblick über den Stand der Forschung wird ein Teilbereich des Customizings von SAP R/3 analysiert. Aus diesen Erkenntnissen werden einige Thesen abgeleitet, welche Aspekte zu berücksichtigen sind, um die Parametrisierbarkeit von Fachkomponenten zu spezifizieren. Außerdem wird ein Ausblick gegeben, wie diese Aspekte in einer Spezifikation abgebildet werden können. Es handelt sich um eine laufende Forschungsarbeit, deren erste Zwischenergebnisse vorgestellt werden.

Schlüsselworte: Fachkomponente; Softwarekomponente; Spezifikation; Standardisierung; Customizing; Parametrisierung; Anpassbarkeit, SAP R/3

Mit der Komponententechnologie wird die Hoffnung verbunden, dass - wie in anderen Ingenieursdisziplinen üblich - komplexe Systeme aus kleineren, vorgefertigten Bausteinen zusammengesetzt werden können. Neben einer stärkeren Wiederverwendung und der damit verbundenen Erhöhung von Effektivität und Qualität verspricht man sich insbesondere für betriebliche Anwendungen, dass so gebaute Anwendungssysteme spezifisch auf die Bedürfnisse der Verwender zugeschnitten werden können. Dies ist mit der heutigen Standardsoftware nur eingeschränkt möglich.

Eine wichtige Voraussetzung dafür ist, dass Softwarekomponenten zur Verfügung stehen, die möglichst genau auf die spezifischen Kundenwünsche ausgerichtet sind. Dazu sind prinzipiell zwei Vorgehensweisen denkbar: a) es gibt sehr viele Komponenten, welche verschiedenen Anforderungen jeweils punktgenau genügen, und ein Verwender wählt die passende Komponente; b) es gibt weniger, dafür variablere Komponenten und ein Verwender passt eine Komponente auf seine konkreten Bedürfnisse an.

Man kann zwar davon ausgehen, dass für Anforderungen, die sich deutlich voneinander unterscheiden, verschiedene Komponenten entstehen werden. Aus praktischen Gesichtspunkten ist es jedoch unwahrscheinlich, dass ein Hersteller für jede Kombination von Detailanforderungen eine eigene Komponente produzieren wird. Deshalb ist es notwendig, dass Softwarekomponenten in gewissem Rahmen anpassbar sind. Es müssen geeignete Mechanismen entwickelt werden, die es erlauben, Komponenten effektiv auf Kundenbedürfnisse anzupassen. Ein solcher Mechanismus ist zum Beispiel die Anpassung über Parametrisierung.

Neben den technischen Hilfsmitteln zur Anpassung sind dem Verwender einer Komponente auch alle

inhaltlichen Informationen zur Verfügung zu stellen, die dieser benötigt, um die möglichen und notwendigen Einstellungen sachgerecht und effektiv vorzunehmen. Dies bedeutet, der Spielraum für Anpassungen und deren Konsequenzen für die Arbeitsweise der Komponente müssen geeignet spezifiziert werden.

Dieser Beitrag beschäftigt sich mit diesem Thema und stellt erste Überlegungen dazu vor. Im Kapitel 1 diskutieren wir den derzeitigen Stand der Forschung. Im Kapitel 2 wird definiert, was wir in dieser Arbeit unter Parametrisierung verstehen. Im Kapitel 3 untersuchen wir das Customizing eines Teilbereichs von SAP R/3 und erhalten dadurch einen besseren Einblick in den Parametrisierungsspielraum von betrieblichen Anwendungen. Der Schwerpunkt der Untersuchungen liegt darauf, Parameter und ihre Belegungen inklusive Bedingungen und Abhängigkeiten zu analysieren. Im Kapitel 4 werden einige Thesen aufgestellt, welche Aspekte zu berücksichtigen sind, um mögliche Parameterbelegungen zu spezifizieren. Im Kapitel 5 wird schließlich ein Ausblick darauf gegeben, wie diese Aspekte in eine Spezifikation einfließen können. Dieser Beitrag stellt erste Zwischenergebnisse einer laufenden Forschungsarbeit vor.

1 Stand der Forschung

Zum Thema „Spezifikation der Parametrisierbarkeit von Fachkomponenten“ sind uns keine Arbeiten bekannt. Es gibt jedoch zu Teilaspekten verschiedene Arbeiten. Abschnitt 1.1 beschäftigt sich mit der Spezifikation von Softwarekomponenten für den betrieblichen Einsatz, Abschnitt 1.2 im Allgemeinen mit der Anpassbarkeit von Softwarekomponenten und Abschnitt 1.3 mit der Parametrisierung von betrieblicher Standardsoftware. Dabei werden diese Teilaspekte immer vor dem Hintergrund des oben genannten Themas betrachtet.

1.1 Spezifikation von Fachkomponenten

Die präzise und geeignete Spezifikation von Komponenten ist eine wesentliche Voraussetzung, damit sich der Bau von Anwendungssystemen aus am Markt gehandelten Softwarekomponenten etablieren kann. Es gibt verschiedene Arbeiten, die sich mit dieser Thematik beschäftigen. Besonders hervorzuheben ist das Memorandum „Vorschlag zur Vereinheitlichung der Spezifikation von Fachkomponenten“ (Turowski et al. 2002). Dieser Vorschlag wurde im Rahmen der Arbeitsgruppe 5.10.3. der Gesellschaft für Informatik von Vertretern aus Forschung und Praxis erstellt.

Dem Vorschlag liegt als „Leitbild, im Sinne eines idealen zukünftigen Zustands, die Idee einer kompositorischen, plug-and-play-artigen Wiederverwendung von Black-Box-Komponenten zu Grunde, deren Realisierung einem Verwender verborgen bleibt und die auf einem Softwaremarkt gehandelt werden.“ Im Memorandum werden die Begriffe (Software-) Komponente und Fachkomponente wie folgt definiert:

- „Eine *Komponente* besteht aus verschiedenartigen (Software-) Artefakten. Sie ist wiederverwendbar, abgeschlossen und vermarktbar, stellt Dienste über wohldefinierte Schnittstellen zur Verfügung, verbirgt ihre Realisierung und kann in Kombination mit anderen Komponenten eingesetzt werden, die zur Zeit der Entwicklung nicht unbedingt vorhersehbar ist.“
- „Eine *Fachkomponente* ist eine Komponente, die eine bestimmte Menge von Diensten einer betrieblichen Anwendungsdomäne anbietet.“

Im weiteren Verlauf dieses Beitrages folgen wir sowohl diesen Definitionen als auch dem vorgestellten Leitbild.

Um Fachkomponenten verschiedener Hersteller effektiv verwenden zu können, müssen diese eindeutig und vollständig spezifiziert werden. Turowski et al. formulieren dazu methodische Standards. Es wird „postuliert, dass die Spezifikation von Fachkomponenten auf verschiedenen Abstraktionsebenen erfolgen muss“: Schnittstellenebene, Verhaltensebene, Abstimmungsebene, Qualitätsebene, Terminologieebene, Aufgabenebene, und Vermarktungsebene. Zu allen Abstraktionsebenen werden die zu spezifizierenden Objekte, eine primäre Notationstechnik sowie gegebenenfalls noch ergänzende sekundäre Notationstechniken festgelegt.

Der Aspekt der Variabilität und Anpassbarkeit von Fachkomponenten konnte aufgrund fehlender Vorarbeiten bisher nicht berücksichtigt werden.

1.2 Anpassbarkeit von Softwarekomponenten

Wie oben gezeigt ist davon auszugehen, dass beim Erstellen von betrieblichen Anwendungen aus Fachkomponenten der Bedarf nach der Anpassbarkeit einzelner Komponenten sowie der gesamten Anwendung besteht. Verschiedene Autoren haben sich mit der Anpassbarkeit von Softwarekomponenten beschäftigt.

Bergner et al. beschreiben ein Computer Aided Engineering System, welches aus Komponenten aufgebaut ist (Bergner/Rausch/Sihling/Vilbig 2000). Die Autoren identifizieren die folgenden Anpassungsstrategien: Wrapperkomponenten, Komposition mit einer Adaptorkomponente, Adaptorinterface, Vererbung und Reimplementierung. Dabei wird allerdings davon ausgegangen, dass die Komponenten selbst keine Variabilität erlauben. Die gewünschte Anpassung erfolgt entweder außerhalb der Komponente (Wrapper, Adaptoren) oder durch Modifikation auf der Codingebene.

Reussner schlägt eine bessere Anpassbarkeit an verschiedene Konfigurationen durch flexiblere Kontrakte vor (Reussner 2001). Statt fester Vor- und Nachbedingungen sollen mehrere Varianten solcher Bedingungen angegeben werden. Durch eine zusätzliche Beschreibung der Abhängigkeiten zwischen angebotenen und benötigten Interfaces kann ermöglicht werden, dass zu einer vorhandenen Systemkonfiguration (gegebene Vorbedingung) ermittelt wird, welche Nachbedingungen die Komponente in dieser Konfiguration erfüllen kann. Dieses Vorgehen erscheint vor allem für die Anpassung auf verschiedene Systemumgebungen von Interesse zu sein.

Weis diskutiert die Möglichkeit einer Trennung von Anpassung und Deployment (Weis 2001). Komponentenhersteller spezifizieren eine Komponente und mögliche Variationen. Der Komponentenkäufer bestellt die Komponente in der für ihn gewünschten Ausprägung, welche dann durch den Hersteller kundenspezifisch hergestellt wird. Zur Erstellung der Komponente verwendet der Hersteller Techniken wie Aspect-weaving, so dass der Komponenten- und Wiederverwendungsgedanke trotz allem gewahrt bleibt. Als Anwendungsfälle für dieses Vorgehen werden Plattform, Bildschirmgröße, Netzwerk und andere technische Attribute angegeben.

Floch und Gulla stellen ein industrielles Wiederverwendungsprojekt im Bereich der Telekommunikationssysteme vor (Floch/Gulla 1996). Im Laufe der Zeit sind eine Vielzahl von variablen Komponenten entstanden, die zu kundenspezifischen Systemen assembliert werden. Mit der PROTEUS Konfigurationssprache kann die genaue Konfiguration solcher Systeme abgebildet werden. Die Beschrei-

bung der Variabilität erfolgt hier vor allem auf der Architekturebene.

Ebenfalls auf der Ebene der Komponentenarchitektur beschäftigen sich Stiermeling und Cremers mit dem Thema Anpassbarkeit (Stiermeling/Cremers 1998). Im Kontext von Groupware-Anwendungen werden die nachfolgenden Anpassungsaktivitäten identifiziert, die eine Komponentenarchitektur unterstützen muss: Parametereinstellung bei einzelnen Komponenten, Änderung der Komposition von Komponenten und Änderung der Implementierung einzelner Komponenten.

Neben der Betrachtung von Softwarekomponenten ist es auch interessant, allgemein einen Blick auf das Thema Variabilität und Wiederverwendung zu werfen. Jacobson/Griss/Jonsson beschäftigen sich in ihrem Standardwerk „Software Reuse“ ausführlich mit dieser Problematik (Jacobson et al. 1997). Sie definieren sogenannte Variationspunkte (variation points), an denen die Anpassung von Komponenten stattfindet. Dabei unterscheiden sie zwischen folgenden Anpassungsmechanismen (Seite 102): Vererbung, Erweiterungen (Extension classes), Use Case Vererbung (Uses), Konfiguration, Parameter, Template-Instanziierung und Generierung. Bei dem Stichwort Parameter handelt es sich um das Setzen von Parameter mit vorgedachtem Wertebereich; dies kann entweder das einfache Setzen von Parameterwerten als auch die Instanziierung eines abstrakten Datentyps (Parameter-Polymorphismus) sein. Die von Jacobson et al. vorgeschlagenen Mechanismen zur Anpassung bewegen sich eher auf der Programmierebene. Dadurch sind vielfältige und sehr flexible Anpassungen von Programmierobjekten möglich. Dieses Vorgehen ist bei der allgemeinen Wiederverwendung (im Gegensatz zur Anpassung von Softwarekomponenten) sinnvoll, da der Black-Box-Gedanke nicht unbedingt im Vordergrund steht.

Zusammenfassend lässt sich feststellen, dass verschiedenste Mechanismen und Techniken zur Anpassung und Variabilität von komponentenbasierten Anwendungen bekannt sind. Diese können grob in zwei Gruppen unterteilt werden:

- Anpassung der Komponentenarchitektur sowie der Auswahl und des Zusammenspiels der einzelnen Komponenten,
- Anpassung einzelner Komponenten selbst.

Bei der zweiten Gruppe bewegen sich die meisten Techniken auf der Programmierebene und gehen nicht davon aus, dass eine Variabilität schon vom Komponentenhersteller vorgesehen wurde. Relativ wenig untersucht wird die Situation, dass eine Komponente schon eine gewisse Variabilität erlaubt. Mögliche Techniken dafür sind vor allem das Setzen von Parameterwerten, das Programmieren eigener Teilaspekte in vorgedachten Erweiterungspunkten und das Einbinden alternativer, vorgedachter Varianten. Eine detaillierte Beschreibung dieser Techniken erfolgte nicht. Vorgedachte Erweiterungen stoßen zwar von Natur aus an Grenzen, bieten aber gerade beim Upgrade auf neuere Releaseversionen große Vorteile.

1.3 Parametrisierung betrieblicher Standardsoftware

Zur Parametrisierung und Anpassbarkeit betrieblicher Anwendungssysteme liegen umfangreiche Erfahrungen für monolithische Standardsoftware vor. Solche Anwendungssysteme sind hochkomplex parametrisierbar, um eine möglichst genaue Anpassung an die konkreten Kundenwünsche zu ermöglichen. Eine solche Parametrisierung wird bei betrieblicher Standardsoftware oft auch als Customizing bezeichnet.

Die SAP AG bietet mit dem SAP Implementation Guide (IMG) ein Werkzeug an, welches die Anpassung ihrer Produkte unterstützt. Alle elementaren Einstellungen, die zum selben betriebswirtschaftlichen Kontext gehören und zusammen durchzuführen sind, werden zu sogenannten Customizing-Aktivitäten (CAs) zusammengefasst. Beispiele für CAs aus dem Bereich Einkauf sind Anlegen von Einkäufergruppen, Definieren von Einkaufsbelegarten oder Definieren möglicher Bestellgründe. Alle CAs werden in einen Baum einsortiert, der ähnlich zur Komponentenhierarchie im SAP-Referenzmodell aufgebaut ist. Man spricht dabei auch vom komponentenorientierten SAP Referenz-IMG. (Der Begriff Komponente ist hier im Sinne eines funktionalen Teilbereichs von z.B. R/3 und nicht im Sinne der in Kapitel 1 definierten Fachkomponente zu sehen.) Beispiele für solche Komponenten auf der obersten Ebene sind Finanzbuchhaltung, Controlling, Vertrieb und Materialwirtschaft. Zusätzlich gibt es einen prozessorientierten SAP Referenz-IMG. CAs werden dabei Prozessen zugeordnet, wenn die Ausführung der CA notwendig für die Verwendung des Prozesses ist. Ein Kunde kann die von ihm benötigten Komponenten und Prozesse auswählen und erhält einen (auf seine Bedürfnisse) reduzierten Kunden-IMG. Dieser führt ihn Schritt für Schritt durch die notwendigen Arbeitsschritte. Mit diesem Vorgehen werden mehrere Ziele verfolgt: modellgestütztes Customizing, Aufzeigen von Abhängigkeiten zwischen CAs und Reduktion der Komplexität auf die kundenspezifischen Anforderungen. Die einzelnen CAs werden überblicksartig dokumentiert. Es erfolgt jedoch keine formale Beschreibung und oft sind Bedingungen und Abhängigkeiten auf der Ebene einzelner Parameter nur anhand von Systemmeldungen nachvollziehbar. Für weitere Details siehe das R/3 Referenz(prozeß)modell (SAP 1997).

Umfangreiche Untersuchungen über die Kopplung von Geschäftsprozessmodellen und Anwendungssystemen wurden im WEGA-Projekt vorgenommen. Bei WEGA (Wiederverwendbare und erweiterbare Geschäftsprozess- und Anwendungssystemarchitekturen) handelt es sich um ein Verbundprojekt zwischen der Universität Bamberg, der SAP AG und der KPMG Unternehmensberatung. „Ziel des Projektes WEGA [war] die Entwicklung und Untersuchung von Architekturen für Geschäftsprozessmodelle und Anwendungssysteme in industriellen Größenordnungen ...“. Besonders berücksichtigt werden sollte dabei, dass die Beherrschung von Komplexität, Variantenreichtum und Interoperabilität unterstützt wird. Die Ergebnisse des Projekts sind u.a. in das oben dargestellte Vorgehen beim SAP-Customizing eingeflossen. Für weitere Details und umfangreiche Literaturverweise siehe den WEGA-Abschlussbericht (Ferstl et al. 1998).

Teil des WEGA-Projektes war die Diplomarbeit von Guntermann, welche Anpassungsmechanismen für Business-Objekte in objektorientierten Anwendungssystemen untersucht (Guntermann 1998). In Anlehnung an (Jacobson et al. 1997) werden fünf Mechanismen identifiziert und verglichen, wie gesamte Anwendungssysteme angepasst werden können: Vererbung, Erweiterungen mit Hilfe von Extensions, Parametrisierung von Objekten, Componentware und Propagation Pattern. Die Verwendung von Componentware (eventuell unterstützt durch Propagation Patterns) entspricht weitgehend dem in Kapitel 1 formulierten Leitbild. Es wird aber angemerkt, dass auch die Komponenten selbst noch anpassbar sein müssen. Bei den Mechanismen Vererbung und Erweiterungen handelt es sich um Methoden, wie Variabilität vom Hersteller vorgesehen werden kann. Für den Anwender des Anwendungssystem ist Parametrisierung der Standardmechanismus zur Auswahl zwischen vorgedachten Varianten. Zum Beispiel könnte der Hersteller verschiedene Varianten eines Business-Objektes in verschiedenen Subklassen des Objektes implementieren, und der Anwender entscheidet sich durch das Setzen eines Parameter für die gewünschte Ausprägung.

Zusammenfassend lässt sich folgendes feststellen: Trotz ihrer Grenzen ist die Parametrisierung eine einfache und ziemlich mächtige Technik, die gerade für die Abdeckung betriebswirtschaftlicher Variabilität gut geeignet ist (siehe dazu auch Kapitel 3). Es kann also davon ausgegangen werden, dass Parametrisierung auch bei der Anpassung von Fachkomponenten eine wichtige Rolle spielen wird. Eine genauere Untersuchung und Beschreibung der Parametrisierung von Fachkomponenten erfolgte bisher nicht, erscheint aber wünschenswert.

2 Anpassung durch fachliche Parametrisierung

Wie in Kapitel 1 diskutiert, ist Parametrisierung eine Möglichkeit, wie man Fachkomponenten (oder ganze Anwendungssysteme) auf spezifische Kundenwünsche anpassen kann. In diesem Kapitel wollen wir näher definieren, was wir im Folgenden unter Parametrisierung verstehen.

Parametrisierung heißt eine Anpassungsstrategie, die sich durch folgende Merkmale auszeichnet:

- Die Anpassungsmöglichkeiten werden vom Hersteller der Software vorgedacht.
- Der Hersteller definiert Parameter inklusive deren Bedeutung und deren Auswirkungen auf die Funktionsweise der Software.
- Der Verwender prägt die Parameter aus, indem er diese mit den Werten belegt, die den von ihm gewünschten Verhalten entsprechen.
- Die Parameterbelegungen sind persistent und werden zur Laufzeit ausgewertet.

Die vom Hersteller definierten Parameter befinden sich also auf der Typebene, die vom Verwender festgelegten Parameterwerte (Belegungen der Parameter) auf der Instanzebene. Während des Ablaufs von Transaktionen und Prozessen verhalten sich die Parameter rein technisch wie Konstanten und unterscheiden sich damit von Programmiervariablen. Allerdings sind Parameter nicht auf alle Zeiten konstant, sondern können zwischen verschiedenen Transaktionen (in gewissem Rahmen) geändert werden.

Unter *Parametrisierung* verstehen wir einerseits den Prozess, die vorgegebenen Parameter mit geeigneten Parameterwerten zu füllen. Andererseits wird der Begriff *Parametrisierung* auch allgemein als Bezeichnung für diese spezielle Anpassungsstrategie verwendet. Der oft verwendete Begriff Customizing ist weitgehend synonym zum Begriff Parametrisierung.

Der Begriff *Parametrisierungsspielraum* beschreibt die Gesamtheit der Variabilität (in Datenstrukturen und Verhalten) einer Fachkomponente, welche durch Parametrisierung erreicht werden kann. Der Begriff *Parametrisierbarkeit* steht für die Eigenschaft der Fachkomponente, über Parametrisierung anpassbar zu sein.

Unter *fachlicher Parametrisierung* verstehen wir alle solchen Einstellungen, die betriebswirtschaftlich und aufgabenbezogen sind. Mögliche Beispiele dafür sind:

- Definition von organisatorischen Einheiten und Stammdaten (z.B. Werke, Materialen),
- Auswahl unter vorgegebenen Prozessvarianten,
- Definition von Steuerdaten (z.B. erlaubte Anlieferungszeiten an einem Lager),
- Definition von Daten zur Dialog- und Benutzersteuerung (z.B. Vorschlagswerte).

Unter *technischer Parametrisierung* verstehen wir alle die Einstellungen, die sich auf einer rein technischen Ebene befinden (z.B. verwendete Datenbank, verwendetes Betriebssystem).

In unseren weiteren Untersuchungen interessiert uns nur die fachliche Parametrisierung. Der Einfachheit halber werden wir oft von Parametrisierung sprechen, meinen damit aber immer die fachliche Parametrisierung.

Parametrisierung kann zu verschiedenen Zeiten stattfinden: initial oder während des laufenden Betriebs. Bei Änderungen von Parameterwerten im laufenden Betrieb sind allerdings einige Einschränkungen zu beachten. Entweder sind die Parameterwerte zeitbehaftet oder aber es muss sicher gestellt sein, dass die Änderungen keine Inkonsistenzen bei den bisherigen Daten und Prozessen erzeugen.

Um Parametrisierungsspielraum zu spezifizieren, müssen zwei Aspekte berücksichtigt werden: die Spezifikation der einzustellenden Parameter selbst inklusive eventueller Bedingungen und Abhängigkeiten, sowie die Spezifikation der Auswirkungen dieser Einstellungen auf den laufenden Betrieb. Um die Parametrisierung von Fachkomponenten spezifizieren zu können, müssen damit die folgenden vier Problemfelder von der Forschung untersucht werden:

1. Analyse von Parametern und möglichen Parameterwerten inklusive Bedingungen und Abhängigkeiten
2. Entwickeln von Techniken zur Spezifikation dieser Parameter(werte)
3. Analyse der Auswirkungen einzelner Parameter(werte) auf Daten und Verhalten einer Fachkomponente
4. Entwickeln von Techniken zur Spezifikation dieser Auswirkungen

Bei den Fragestellungen 1und 3 muss also untersucht werden, was beschrieben werden soll, bei den Fragestellungen 2 und 4, wie die Beschreibung erfolgen soll.

In dieser Arbeit liefern wir erste Beiträge für die Fragestellung 1. Dazu untersuchen wir exemplarisch das Customizing von SAP R/3 (siehe Kapitel 3) und leiten daraus einige Thesen für die Parametrisierung von Fachkomponenten ab (siehe Kapitel 4). Im Kapitel 5 geben wir einen Ausblick auf mögliche Lösungen für die Fragestellung 2. Mit den Fragestellungen 3 und 4 beschäftigen wir uns in dieser Arbeit nicht.

3 Analyse von Customizing-Aktivitäten in SAP R/3

Um die Parametrisierbarkeit von Fachkomponenten spezifizieren zu können, sollte zunächst im Detail untersucht werden, welche Einstellungen typischerweise vorgenommen werden. Wie oben beschrieben, gibt es dazu im Bereich von Fachkomponenten kaum Erfahrungen. Es kann jedoch davon ausgegangen werden, dass die Erfahrungen beim Customizing von Standardsoftware zu größeren Teilen auf Fachkomponenten übertragbar sind.

Wir stellen hier die Ergebnisse einer Fallstudie vor, in welcher das Customizing von SAP R/3 untersucht wurde. Der Fokus liegt auf der Analyse, wie Customizing-Einstellungen erfolgen und welche Bedingungen und Abhängigkeiten existieren. Es wird nicht untersucht, welche Auswirkungen die Einstellungen auf die Funktionsweise der Software haben. Für einen allgemeinen Überblick über das Customizing in SAP R/3 siehe auch Abschnitt 1.3.

Für eine beispielhafte Analyse des Customizings von SAP R/3 wurde der Bereich Bestellung innerhalb der Anwendung Einkauf ausgewählt. Dieser Bereich ist für die Steuerung und Abwicklung von Bestellungen zuständig. Betrachtet werden alle Customizing-Aktivitäten (CAs), die im komponentenbasierten SAP Referenz-IMG dem Teilbaum Bestellung zugeordnet sind. Zusätzlich betrachtet wurde noch die CA *Einkäufergruppen definieren*, die der übergeordneten Ebene Einkauf zugeordnet ist. Insgesamt handelt es sich um 37 CAs, die näher untersucht wurden (Stand: SAP R/3 Enterprise, Release 4.70). Für eine detailliertere Beschreibung der CAs sei auf den Anhang verwiesen.

3.1 Beschreibung von Customizing-Aktivitäten

In diesem Abschnitt werden die wichtigsten Eigenschaften von Customizing-Aktivitäten erfasst und analysiert. Das Kriterium „wichtig“ richtet sich dabei nach dem Zweck, Parameter und ihre möglichen Werte spezifizieren zu können.

Ziel dieses Abschnitts ist es, die CAs und bestimmte wiederkehrende Bedingungen so zu beschreiben, dass darauf aufbauend geeignete Spezifikationstechniken entwickelt werden können. Vereinzelt auftretende Arten von Bedingungen sollen vorläufig nicht näher untersucht werden. Sie müssen jedoch erfasst werden, damit auch sie in der Spezifikation berücksichtigt werden können.

Im SAP R/3 ist Customizing sehr datenzentriert und beruht zumeist auf dem Setzen von Parameterwerten. Diese werden in Datenbanktabellen abgelegt. Zur Laufzeit fragen ablaufende Programme die entsprechenden Werte ab und bestimmen dadurch die gewünschte Funktionsweise von Transaktionen und Prozessen. Das Setzen der Parameterwerte erfolgt über spezielle Daten-Views, welche das gleichzeitige Erfassen aller zusammen zu pflegenden Parameter erlauben.

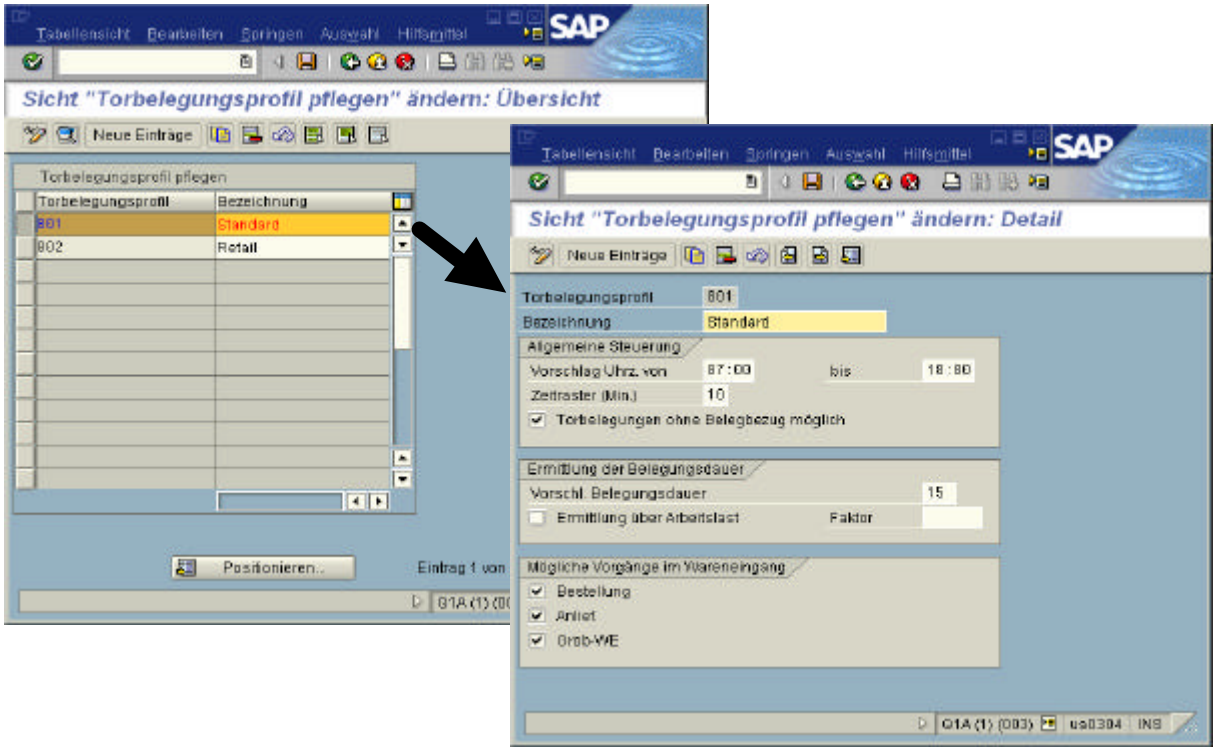

Abb. 1: Customizing-Aktivität „Torbelegungsprofil pflegen“ (Quelle: SAP)

Beispiel: Betrachten wir zum Beispiel die (durchschnittlich komplexe) CA zur Pflege von Torbelegungsprofilen für die Anlieferung. Beim Ausführen der CA erscheint eine Tabelle mit den Feldern Schlüssel des Torbelegungsprofils und Bezeichnung des Torbelegungsprofils. In dieser Tabelle können beliebig viele Torbelegungsprofile definiert werden (siehe Abb. 1 links). Pro Eintrag ist es möglich, auf ein Detailbild zu verzweigen, auf welchem die Steuerungsdaten zu diesem Profil definiert werden. Es handelt sich z.B. um die tägliche Anlieferungszeit (von-bis) oder um ein Kennzeichen, ob Anlieferungen auch ohne Einkaufsbeleg möglich sind (siehe Abb. 1 rechts). In einer folgenden CA wird jedem konkreten Lager eines der abstrakten Torbelegungsprofile zugewiesen, welches dann die Torbelegung für dieses Lager steuert.

Die 37 untersuchten Customizing-Aktivitäten lassen sich in folgende Gruppen unterteilen:

- einfache CAs (31 von 37),
- komplexe CAs (4 von 37),
- Programmierexit (1 von 37),
- Prüfprogramm (1 von 37).

Bei dem Programmierexit und dem Prüfprogramm handelt es sich um keine Anpassungsmöglichkeit über Parametrisierung. Diese sollen deshalb nicht weiter betrachtet werden. Die Begriffe *einfache CA* und *komplexe CA* wurden von uns gewählt. Ihre Bedeutung wird im Folgenden erklärt. (Bei SAP wird diese Unterscheidung durch die sehr technischen Begriffe *View* und *View-Cluster* getroffen.)

Die 31 einfachen Customizing-Aktivitäten folgen nachstehendem Schema: Die CA definiert auf der Typebene einen strukturierten Datentyp mit einer bestimmten Anzahl von Parametern. Die CA erlaubt dem Benutzer, verschiedene Datensätze / Instanzen zu diesem Datentyp anzulegen, zu ändern oder auch wieder zu löschen.

Interessant ist der Fakt, dass bei allen untersuchten CAs mehrere Instanzen zu einer CA ausgeprägt werden können. Beispiele dafür sind: Anlegen verschiedener Belegarten; Anlegen von Steuerdaten, die nicht global gültig, sondern werksspezifisch sind.

Diese Datensätze / Instanzen zu CAs werden wir im Folgenden auch die *Ausprägungen einer CA* nennen.

Jede Ausprägung einer CA besteht also aus einer Reihe von Parametern. Die Anzahl und der Datentyp der Parameter werden von der CA festgelegt. Es gibt einige ausgezeichnete Parameter, welche die Existenz eines Datensatzes bestimmen. Sie bilden damit einen semantischen Schlüssel. Es kann keine zwei Datensätze geben, die eine identische Wertebelegung für die Schlüsselparameter haben.

Beispiel: Wir betrachten die Customizing-Aktivität *Umlagerungsbestellung einstellen: Belegart, Einschrittverfahren und Unterlieferungstoleranz zuordnen*. Eine Umlagerungsbestellung ist eine Bestellung, die nicht an einen externen Lieferanten geht. Stattdessen werden die benötigten Materialien bei einem anderen Werk der selben Firma bestellt. Im Bereich *Umlagerungsbestellung einstellen* werden verschiedene Einstellungen für solche Bestellungen durchgeführt. So wird in der o.g. CA pro Kombination von lieferndem und empfangenden Werk folgendes festgelegt: welche Belegart wird verwendet, soll das Einschrittverfahren angewendet werden, soll die Unterlieferungstoleranz berücksichtigt werden.

Die CA definiert einen Datentyp mit 5 Parametern: Lieferndes Werk, Empfangendes Werk, Belegart, Kennzeichen Einschrittverfahren, Kennzeichen Unterlieferungstoleranz.

Lieferndes und Empfangendes Werk sind die semantischen Schlüssel, d.h. für jede Kombination der Werke kann es maximal einen Datensatz geben. Die Belegart und die beiden Kennzeichen sind Eigenschaften, die die Datensätze näher beschreiben.

Werk und Belegart sind Entitäten, die in anderen CAs definiert werden. Entsprechend können bei lieferndem und empfangendem Werk bzw. bei der Belegart als Werte nur zuvor definierte Werke bzw. Belegarten verwendet werden. Bei den Kennzeichen für Einschrittverfahren und Unterlieferungstoleranz handelt es sich um Parameter mit dem Wertebereich Boolean.

Die vier komplexen Customizing-Aktivitäten bestehen aus mehreren Teilschritten. Dabei sind bis zu 5 Teilschritte aufgetreten und in einem Fall bestand ein Teilschritt aus weiteren Teilschritten. Jeder dieser elementaren Teilschritte entspricht einer einfachen CA. Die Teilschritte lassen sich damit durch das obige Schema für einfache CAs beschreiben. Es gibt die zusätzliche Semantik, dass eine komplexe CA nur dann abgeschlossen ist, wenn die einzelnen Teilschritte nacheinander ausgeführt wurden.

Unabhängig von ihren konkreten Parametern haben einfache Customizing-Aktivitäten und ihre Ausprägungen allgemeine Eigenschaften. Zur Erfassung dieser Eigenschaften schlagen wir das Klassifikationsschema in Tabelle 1 vor.

MERKMAL	MERKMALSAUSPRÄGUNG			
Bwl. Zweck	Org. Einheiten / Stammdaten (1)	Strategische Steuerdaten (14)	Administrative Steuerdaten (19)	Benutzersteuerung / Darstellung (12)
Max. Anzahl Ausprägungen	Eine (0)	Feste Anzahl (0)	Abhängig von anderen CA (21)	Beliebig (25)
Abhängigkeiten	Keine (15)	Innerhalb des Bereichs (14)	Außerhalb des Bereichs (5)	Innerhalb und außerhalb des Bereichs (12)

Tabelle 1: Klassifikationsschema für einfache Customizing-Aktivitäten

Wir haben insgesamt 46 einfache CAs untersucht: 31 an sich einfache CAs und 15 Teilschritte der 4 komplexen CAs. Die in Klammern angegebenen Zahlen sind die Häufigkeiten, mit welcher die Merkmalsausprägungen bei diesen 46 CAs aufgetreten sind.

Das Merkmal *Betriebswirtschaftlicher Zweck* beschreibt die Funktion einer CA auf betriebswirtschaftlicher Ebene und ihren Einfluss auf die Systemsteuerung. Dieses Merkmal wird eher nicht in eine formale Spezifikation eingehen, da insbesondere die Abgrenzung zwischen den Merkmalsausprägungen nicht eindeutig ist. Als Zusatzinformation für den menschlichen Leser ist dieses Merkmal jedoch eine sinnvolle Ergänzung.

Im SAP- Objektmodell (SAP 1996) werden Anwendungsobjekte und -daten wie nachfolgend unterschieden: strategische Objekte (organisatorische Einheiten, Stammobjekte, Regeln, Strukturen), administrative Objekte (verdichtete Werte / Mengen, Vorschriften) und operative Objekte (Geschäftsdokumente, Geschäftsvorfälle). Unter Berücksichtigung der Besonderheiten von Customizing schlagen wir folgende Merkmalsausprägungen vor:

- Im Customizing werden *organisatorische Einheiten* (z.B. Werke) und *Stammdaten* (z.B. Einkäufergruppen) ausgeprägt. Diese haben mittelbaren Einfluss auf die Ablaufsteuerung, indem sie die Träger verschiedener Ablaufvarianten sind

- Steuerdaten steuern den Ablauf von Prozessen und Geschäftsvorfällen. Dabei definieren *strategische Steuerdaten* abstrakte Regeln und Strukturen für die Prozessabläufe. *Administrative Steuerdaten* steuern konkrete Prozessabläufe und sind oft Ausprägungen der strategischen Steuerdaten.
- Unter *Benutzersteuerung / Darstellung* werden CAs zusammengefasst, welche Darstellung und Bildschirmdetails sowie Benutzerbesonderheiten steuern. Sie haben meistens keinen Einfluss auf den Ablauf von Prozessen.

Die Merkmalsausprägungen sind nicht vollkommen disjunkt. So werden in manchen CAs mehrere Aspekte gleichzeitig eingestellt. Bei Unschärfen in der Zuordnung wird immer der wesentlichere Aspekt einer CA als Entscheidungskriterium zu Grunde gelegt.

Das Merkmal *Maximale Anzahl Ausprägungen* beschreibt, wie viele Datensätze in dieser CA maximal definiert werden können. *Feste Anzahl* heißt, die maximale Anzahl ist vom System selbst vorgegeben, z.B. durch Festwerte. *Abhängig von anderer/n CA* heißt, die Anzahl wird durch die Anzahl von Ausprägungen anderer CAs festgelegt (siehe auch Beispiel unten). Im letzten Fall heißt das gleichzeitig, dass die Datensätze in ihrer Existenz von anderen CAs abhängen.

Das Merkmal *Abhängigkeit* beschreibt, ob die CA von anderen CAs abhängig ist. (D.h. müssen bestimmte Ausprägungen anderer CAs vorhanden sein, damit ich diese CA ausführen kann?) Bei bestehenden Abhängigkeiten wird nochmals unterschieden, ob die benötigte CA innerhalb oder außerhalb des betrachteten Bereichs liegt. (In unserem Fall haben wir den Bereich Bestellung des R/3 analysiert.) Diese Unterscheidung wurde deshalb aufgenommen, weil dadurch bei Fachkomponenten beschrieben wird, ob die Abhängigkeiten komponentenintern oder komponentenübergreifend bestehen. Dies ist im Gegensatz zu monolithischen Anwendungen ein qualitativer Unterschied. Da für eine CA mehrere Abhängigkeiten bestehen können, kann auch die Merkmalsausprägung *Innerhalb und außerhalb des Bereichs* auftreten.

Beispiel: Betrachten wir wieder die CA *Umlagerungsbestellung einstellen: Belegart ... zuordnen*. Diese legt für verschiedene Werke fest, wie die Umlagerungsbestellung durchgeführt werden soll.

Bei der CA handelt es sich dem *Betriebswirtschaftlichen Zweck* nach um *Administrative Steuerdaten*.

Wie schon oben gesehen, ist die mögliche Anzahl von Ausprägungen von der Anzahl gepflegter Werke abhängig. Damit trägt das Merkmal *Maximale Anzahl Ausprägungen* den Wert *Abhängig von anderer/n CA*.

Die CA ist einerseits von Werken abhängig, welche in einer CA außerhalb des Bereichs Bestellung definiert werden. Andererseits besteht aber auch eine Abhängigkeit zu den Belegarten, welche innerhalb des Bereichs Bestellung definiert werden. Damit ergibt sich für das Merkmal *Abhängigkeit* die Ausprägung *Innerhalb und außerhalb des Bereichs*.

3.2 Eigenschaften von Parametern

Neben der Analyse ganzer Ausprägungen von Customizing-Aktivitäten sind natürlich auch die Eigenschaften der einzelnen Parameter einer CA interessant. Die Parameterbelegungen bestimmen gerade die Eigenschaften der einzelnen Ausprägungen von CAs. Es muss beschrieben werden, welche Möglichkeiten zum Setzen der Parameterwerte bestehen und ob Abhängigkeiten und Bedingungen zu beachten sind. Um die Eigenschaften von Parametern zu erfassen, schlagen wir das Klassifikationsschema in Tabelle 2 vor.

MERKMAL	MERKMALSAUSPRÄGUNG			
Wertebereich	Boolean (44)	Festwerte (28 + 120)	Customizing-abhängig (95)	Beliebig (73)
Notwendigkeit	Optional (97)	Optional mit Default (58 + 120)	Obligatorisch (85)	

Tabelle 2: Klassifikationsschema für Parameter

Die 46 untersuchten einfachen CAs haben insgesamt 360 Parameter. Die in Klammern angegebenen Zahlen sind die Häufigkeiten, mit welcher die Merkmalsausprägungen aufgetreten sind. (Die CA *Bildaufbau auf Belegebene festlegen* hat 120 Parameter vom gleichen Typ und fällt von der Anzahl der Parameter aus dem üblichen Rahmen. Wir haben deshalb die Eigenschaften der 120 Parameter getrennt aufgeführt, um die Aussage der Häufigkeiten nicht zu verfälschen.)

Das Merkmal *Wertebereich* gibt an, welche Freiheiten es bezüglich der Werte gibt, die der Parameter annehmen kann. Einerseits gibt es Parameter, deren Wertebereich (im Rahmen ihres Datentyps) nicht eingeschränkt ist (*Beliebig*). Weiterhin gibt es Parameter, deren Wertebereich von SAP fest vorgegeben ist. Da viele boolsche Parameter aufgetreten sind, unterscheiden wir diese Gruppe noch mal in die Merkmalsausprägungen *Boolean* und *Festwerte* (Nicht-Boolean). Außerdem gibt es Parameter, deren Wertebereich *customizingabhängig* ist, d.h. er ist durch die zuvor definierten Ausprägungen einer anderen CA vorgegeben. Da im letzteren Fall starke statische Abhängigkeiten zu anderen CAs entstehen, wird dieser im nächsten Abschnitt noch genauer untersucht.

Das Merkmal *Notwendigkeit* gibt an, ob bei dem Erfassen eines Datensatzes der Parameter gefüllt werden muss oder nicht. Neben den beiden Standardausprägungen *Optional* und *Obligatorisch* wird auch noch *Optional mit Default* betrachtet. Letzteres bedeutet, dass auf die Pflege des Parameters verzichtet werden kann, dieser dann aber einen vorgegebenen Vorschlagswert annimmt. (Ein initialer Wert ohne semantische Bedeutung wird dabei nicht als Vorschlagswert betrachtet.) Parameter mit boolschem Wertebereich betrachten wir immer als *optional mit Default*, wobei *false* der Defaultwert ist.

3.3 Parameter mit customizingabhängigem Wertebereich

Es gibt Parameter, deren Wertebereich durch die Ausprägungen einer anderen CA vorgegeben ist. Dadurch entsteht eine statische Abhängigkeit zu der referenzierten CA. Der Parameter kann nur dann gepflegt werden, wenn zuvor geeignete Ausprägungen der anderen CA definiert wurden. Dadurch kommt es zu einer Anforderung an die Reihenfolge zwischen beiden CAs.

Beispiel: Wir betrachten wieder die CA *Umlagerungsbestellung einstellen: Belegart... zuordnen*. Diese hat z.B. den Parameter ‚Lieferndes Werk'. Der Wertebereich dieses Parameters ist durch die Menge der Werke vorgegeben, die in einer CA *Werke einstellen* definiert wurden. Umlagerungsbestellungen können damit nur eingestellt werden, wenn zuvor die notwendigen Werke definiert wurden.

Auf diese Weise entsteht eine besondere Beziehung zwischen zwei CAs. Im betrachteten Beispiel traten viele solcher Beziehungen auf, weswegen eine nähere Untersuchung erfolgt. Interessant ist vor allem zu untersuchen, welche Eigenschaften die so entstandene Beziehung zwischen den zwei CAs

aufweist. Zur Erfassung dieser Eigenschaften schlagen wir das Klassifikationsschema in Tabelle 3 vor.

MERKMAL	MERKMALSAUSPRÄGUNG				
Semantischer Charakter	Gliederung / Strukturierung (42)	Aufbau von Zuordnungen (18)	Vorschrift (22)	Klassifizierung (4)	Prozessintegration (9)
Art	Hierarchie (12)	Aggregation (37)	Obligatorische Referenz (6)	Optionale Referenz (40)	
Abhängigkeiten	Innerhalb des Bereichs (67)	Außerhalb des Bereichs (28)			

Tabelle 3: Klassifikationsschema für die Beziehungen zwischen CAs, die durch einen customizingabhängigen Wertebereich entstehen

Von den insgesamt 360 Parametern hatten 95 einen customizingabhängigen Wertebereich. Die in Klammern angegebenen Zahlen sind die Häufigkeiten, mit welcher die Merkmalsausprägungen bei diesen Parametern aufgetreten sind.

Zur besseren Erläuterung der Merkmale führen wir folgende Bezeichnungen ein: Die CA mit dem customizingabhängigen Parameter nennen wir die betrachtete CA und bezeichnen sie immer als CA X. Die CA, welche den Wertebereich für den Parameter konstituiert, nennen wir die verwendete CA und bezeichnen sie immer als CA Y. Die betrachtete CA X ist von der verwendeten CA Y abhängig, da sie nur dann ausgeführt werden kann, wenn zuvor die CA Y ausgeführt wurde.

Beispiel: Im oben genannten Beispiel wäre die CA *Umlagerungsbestellung einstellen: Belegart ... zuordnen* die CA X. Die Ausprägungen dieser CA haben einen Parameter Lieferart. Dieser Parameter definiert eine Beziehung zur CA *Belegarten definieren*, welche hier als CA Y bezeichnet wird.

Das Merkmal *Semantischer Charakter* beschreibt den Charakter der Beziehung, d.h. aus welchem Grund werden die Ausprägungen der zwei CAs über eine Beziehung verknüpft. Dies hilft vor allem zu verstehen, warum überhaupt Beziehungen gebildet werden. Das Merkmal wird eher nicht in eine formale Spezifikation eingehen, da insbesondere die Abgrenzung zwischen den Merkmalsausprägungen nicht eindeutig ist. Als Zusatzinformation für den menschlichen Leser ist dieses Merkmal jedoch eine sinnvolle Ergänzung.

Es werden folgende Merkmalsausprägungen unterschieden:

- *Gliederung / Strukturierung*: In der betrachteten CA X werden Daten beschrieben, die z.B. einen Ablauf steuern. Diese Daten sollen allerdings nicht global gelten, sondern können sich in Abhängigkeit von einer organisatorischen Einheit oder einem Geschäftsvorfall unterscheiden. In solch einem Fall werden die Ausprägungen der betrachteten CA X in Abhängigkeit von Ausprägungen einer anderen CA Y (z.B. einer organisatorischen Einheit) definiert. Zu beachten ist dabei, dass die Ausprägungen der CA Y zumeist keinen sachlich-logischen Bezug zu den Daten haben, sondern nur als Gliederungselement dienen.

Beispiel: Es gibt eine CA *Berechtigungsprüfung für Sachkonten einstellen*. Hier wird mittels eines Kennzeichens eingestellt, ob bei Zugriff auf ein Sachkonto eine Berechtigungsprüfung durchgeführt werden soll. Es wäre denkbar, dieses Kennzeichen global zu setzen. Im SAP R/3 System wurde allerdings eine flexiblere Variante gewählt. Dieses Kennzeichen kann pro Buchungskreis gesetzt werden. Damit dient der Buchungskreis als Gliederungselement für die betrachtete CA.

- *Aufbau von Zuordnungen*: In manchen Fällen werden einfache Zuordnungen zwischen Entitäten definiert. Oft ist die Semantik dabei, dass nur die explizit definierten Paare gültige Kombinationen bilden.

  Beispiel (außerhalb des Bereichs Bestellung): Es gibt die zwei Entitäten *Werk* und *Material*. Es soll definiert werden, welche Materialien in einem Werk relevant sind. Dazu dient eine CA *Werksmaterial definieren*. Diese CA hat Referenzen zum Werk und zum Material und beide Referenzen dienen dazu, erlaubte Werk-Material-Zuordnungen aufzubauen.

- *Vorschrift*: Die verwendete CA Y beschreibt eine abstrakte Regel oder Struktur. Jeder Ausprägung der betrachteten CA X wird eine solche Regel oder Struktur (d.h. eine konkrete Ausprägung der CA Y) zugeordnet, welche dann die Eigenschaften der CA X näher beschreiben. (Ziel dieses Vorgehens ist es, bestimmte Regeln oder Strukturen nur einmal zu definieren und an verschiedenen Stellen wiederverwenden zu können.) Zumeist beschreiben solche Regeln und Strukturen Daten, die prozesssteuernden Charakter haben.

  Beispiel: In der oben schon beschriebenen CA *Umlagerungsbestellung einstellen: Belegart ... zuordnen* wird für eine Kombination von Werken u.a. festgelegt, welche Belegart bei der Bestellabwicklung verwendet wird. Belegarten beschreiben auf einer abstrakten Ebene, wie Bestellungen abgearbeitet werden können. Durch die Zuordnung einer Belegart wird hier festgelegt, wie eine konkrete Umlagerungsbestellung ablaufen soll. Die in der verwendeten CA definierten Belegarten dienen also als Vorschrift, wie die Umlagerungsbestellung ausgeführt werden soll.

- *Klassifizierung*: In einer CA Y kann eine Liste von Bezeichnern für eine Entität gepflegt werden. Diese tragen keine weiteren Eigenschaften und haben damit, im Gegensatz zu den oben beschriebenen Regeln und Strukturen, keinen steuernden Charakter. Wird in der betrachteten CA X solch ein Bezeichner benötigt, ist keine Freitexteingabe möglich, sondern es muss einer der in der CA Y definierten Bezeichner gewählt werden. Dieses Vorgehen hat folgende Gründe: es erfolgt eine Vereinheitlichung bestimmter Bezeichner, eine Klassifizierung / Gruppierung nach Ausprägungen dieses Bezeichners ist einfacher und einem Verwender kann bei vorgegebenen Bezeichnern der Aufwand erspart werden, diese immer wieder einzugeben.

  Beispiel: In der CA *Termintyp zur Rechnungsplanart pflegen* wird für jeden Termintyp ein Terminbezeichner festgelegt. Statt einer Freitexteingabe sind jedoch nur solche Bezeichner erlaubt, die in einer anderen CA *Terminbezeichner pflegen* zuvor definiert wurden. Damit klassifizieren / gruppieren die Terminbezeichner die verschiedenen Ausprägungen von Termintypen bezüglich des Aspekts Terminbezeicher.

- *Prozessintegration*: Nach Abschluss von Prozessen werden häufig Folgeprozesse angestoßen. Für die Überleitung zu den anderen Prozessen werden oft Inputparameter benötigt. In manchen Fällen ist es sinnvoll, diese im Rahmen einer CA X zu definieren. Handelt es sich bei den Inputparametern um Ausprägungen einer anderen CA Y, dann handelt es sich um eine Beziehung, welche der Prozessintegration dient.

  Beispiel: Bei einer Umlagerungsbestellung erfolgt nach der Abarbeitung der einkaufsspezifischen Aufgaben eine Überleitung ins Lieferwesen, damit die bestellten Materialien von einem Werk zum anderen geliefert werden. Um eine Lieferung automatisch anzustoßen (d.h. einen Dienst *Lieferung anlegen* auszuführen), muss z.B. die Lieferart bekannt sein. In der CA *Umlagerungsbestellung einstellen: Lieferart und Prüfregel zuordnen* wird für eine Kombination von Lieferwerk und Belegart festgelegt, welche Lieferart verwendet werden

soll.

Das Merkmal *Art* beschreibt die Form der Beziehung. Dabei werden in Anlehnung an (SAP 1996) folgende Merkmalsausprägungen unterschieden:

- *Hierarchie*: Eine Hierarchie ist ein Beziehungstyp, bei dem die CA X eine semantische Verfeinerung der CA Y darstellt. Die Ausprägungen der CA Y gehen erzeugend in die Ausprägungen der CA X ein.
- *Aggregation*: Eine Aggregation ist ein Beziehungstyp, bei dem die Ausprägungen der CA X durch eine Ausprägung der CA Y und mindestens einer anderen Entität (nicht notwendigerweise eine Ausprägung einer CA) erzeugt werden. Dies bedeutet im Allgemeinen, dass der semantische Schlüssel der Ausprägungen von CA Y in den semantischen Schlüssel der Ausprägungen von CA X eingeht.
- *Referenz*: Eine Referenz ist ein Beziehungstyp, in der Ausprägungen der CA Y die Ausprägungen der CA X näher beschreiben, diese aber nicht erzeugen.
- Bei Referenzen wird nochmals zwischen *optional* und *obligatorisch* unterschieden. Aggregation und Hierarchie sind immer obligatorisch.

Das Merkmal *Abhängigkeiten* beschreibt, ob die Beziehung innerhalb des betrachteten Bereichs (hier Bestellung) besteht oder ob die verwendete CA Y außerhalb des Bereichs definiert wird. Dieses Merkmal wurde deshalb aufgenommen, weil dadurch bei Fachkomponenten beschrieben wird, ob die Abhängigkeiten komponentenintern oder komponentenübergreifend bestehen. Dies ist eine wichtige Information. Es ist jedoch davon auszugehen, dass bei Fachkomponenten viel weniger mit übergreifenden Beziehungen gearbeitet wird, als dies bei einer monolithischen Anwendung geschieht.

3.4 Abhängigkeiten zwischen Customizing-Aktivitäten

Bisher haben wir uns hauptsächlich mit der Beschreibung einzelner Customizing-Aktivitäten (CA) beschäftigt. Daneben gibt es noch Abhängigkeiten zwischen CAs. Diese sollen in diesem Abschnitt näher diskutiert werden.

In der Analyse des Bereichs Bestellung wurden zwei häufig wiederkehrende Arten von Abhängigkeiten gefunden:

- Abhängigkeiten zwischen CAs aufgrund eines customizingabhängigen Wertebereichs
- Zusammenfassung mehrerer einfacher CAs zu einer komplexen CA

Die Abhängigkeit über den Wertebereich hatten wir zuvor ausführlich diskutiert. Eine Ausprägung einer CA X verwendet als einen ihrer Parameterwerte eine Ausprägung einer CA Y. Dies bedeutet, in der CA Y muss diese Ausprägung definiert worden sein, bevor die Ausprägung der CA X definiert werden kann.

Bei der Beschreibung komplexer CAs ist es wichtig, dass alle Teilschritte vollständig und in der richtigen Reihenfolge abgearbeitet werden.

Darüber hinaus sind allgemeiner auch noch die nachfolgenden Abhängigkeiten zwischen CAs denkbar. Für diese wurden allerdings im untersuchten Bereich von SAP R/3 keine Beispiele gefunden. (Wobei nicht auszuschließen ist, das solche Abhängigkeiten existieren, nur aufgrund fehlender Do-

kumentation nicht ersichtlich sind.)

- Eine CA Y muss immer vor einer CA X ausgeführt werden, obwohl keiner der beiden oben genannten Fälle vorliegt.
- Das Ausführen einer CA Y führt dazu, dass eine CA X nicht mehr ausgeführt werden kann / muss.
- Das Ausführen von zwei CAs schließt einander aus.

Eine Beschreibung des Parametrisierungsspielraums von Fachkomponenten sollte diese Abhängigkeiten zwischen CAs ausdrücken können. Es handelt sich dabei um Reihenfolgebeziehungen zwischen einzelnen Customizing-Aktivitäten.

Außerdem gibt es beim SAP Customizing, wie oben beschrieben, noch folgende Auswahlmöglichkeit für CAs: Durch die Abwahl benötigter Komponenten oder Prozesse werden ganze Bereiche von CAs nicht benötigt und werden damit abgeschalten. Diese Auswahlmöglichkeit erfolgt allerdings auf ziemlich hoher Ebene. So könnte ein Kunde entscheiden, ob er die Funktionalität des Einkaufs einsetzen möchte oder nicht. Dieser Mechanismus dient zur Komplexitätsreduktion im Großen. Die Auswahl ganzer Bereiche des R/3 entspricht von der Ebene her also eher der Auswahl und der Integration von Fachkomponenten in ein Anwendungssystem. Daher ist diese Auswahl und ihr Einfluss auf CAs für die Beschreibung der Parametrisierung von Fachkomponenten nicht von Interesse.

4 Parametrisierung von Fachkomponenten

Für Fachkomponenten liegen noch keine Erfahrungen bezüglich ihrer Parametrisierung vor. Wir wollen deshalb in diesem Kapitel einige Thesen aufstellen, welche Parameter und mögliche Parameterwerte von Fachkomponenten betreffen. Wir stützen uns dabei hauptsächlich auf die Erkenntnisse aus den Untersuchungen zum Customizing von SAP R/3 und bewerten diese bezüglich ihrer Übertragbarkeit auf Fachkomponenten. Die Aussagen bilden den Kern davon, was in einer Spezifikation zu berücksichtigen und damit von einer Spezifikationstechnik zu unterstützen ist. Diese Thesen müssen anhand weiterer Untersuchungen von betriebswirtschaftlicher Standardsoftware bzw. von Fachkomponenten validiert bzw. gegebenenfalls angepasst werden. Es ist zu beachten, dass sich die Thesen nur auf die Fragestellung 1 aus Kapitel 2 (Parameter und mögliche Parameterwerte) und nicht auf die Fragestellung 3 (Auswirkungen einzelner Parameter(werte)) beziehen.

1. Parametrisierung von Fachkomponenten bedeutet, dass der Entwickler verschiedene Parameter vordefiniert und der Anwender diese mit geeigneten Werten belegt.

   Dies ergibt sich aus unserem Begriff Parametrisierung und wurde hier nur der Vollständigkeit halber nochmals aufgenommen.

2. Es lassen sich mehrere Parameter zu logischen Einheiten zusammenfassen, die immer zusammen gepflegt werden. Solche Zusammenfassungen können durch einen strukturierten Datentyp beschrieben werden.

   Die Gründe für die Zusammenfassung von Parametern in SAP R/3 erfolgt aus rein fachlichen Gründen und ist damit auf Fachkomponenten übertragbar. Ein Spezialfall der These ist, dass man einzelne Parameter als Zusammenfassung eines Elements betrachtet.

3. Zu den Zusammenfassungen von Parametern kann es mehrere Ausprägungen (Instanzen) geben. Jede dieser Ausprägungen entspricht vom Typ dem oben genannten strukturierten Datentyp.

Auch dieser Fakt ist fachlich motiviert, weil dadurch verschiedene Varianten von möglichen Parameterbelegungen (z.B. für verschiedene organisatorische Einheiten) definiert werden können. Der denkbare Fall, dass es zu einer Zeit nur eine Ausprägung geben kann, ist ein Spezialfall der These.

4. Es stehen Möglichkeiten zur Verfügung, diese Ausprägungen anzulegen, zu ändern und wieder zu löschen.

5. Es können Bedingungen auftreten, die die Reihenfolge beeinflussen, in welcher die Parameter durch Werte zu belegen sind.

   Auch diese Erkenntnis aus der Untersuchung von SAP R/3 ist fachlich motiviert und lässt sich deshalb auf Fachkomponenten übertragen. In jedem betrieblichen System bestehen Abhängigkeiten, die sich auf die Reihenfolge von Arbeitsschritten auswirken.

6. Der Wertebereich einzelner Parameter kann entweder fest oder durch Werte anderer Parameter vorgegeben sein. Die Pflege von Parametern kann optional oder obligatorisch sein.

   In einem betrieblichen Datenmodell ist es üblich, dass Referenzen zwischen Entitäten bestehen. Es gibt keinen Grund, warum dies bei parametrisierungsrelevanten Entitäten nicht so sein sollte. Deshalb muss man davon ausgehen, dass der Wertebereich nicht aller Parameter fest ist, sondern teilweise von anderen Parameterwerten vorgegeben wird.

7. Bei der Belegung der Parameter können weitere Bedingungen auftreten, die sich aus anderen Parameterwerten ergeben. Wir treffen die Annahme, dass sich solche Bedingungen mithilfe der üblichen Beschreibungssprachen für Bedingungen (z.B. der UML OCL) ausdrücken lassen.

   Bedingungen zwischen einzelnen Parametern sind ebenfalls üblich und nicht SAP-spezifisch. Die in der SAP-Fallstudie gefundenen Bedingungen genügen der oben formulierten Annahme. Um eine Grundlage für die Spezifikation zu haben, sollten wir diese Annahme solange beibehalten, bis komplexere Bedingungen auftreten, die der Annahme nicht genügen.

8. Auftretende Bedingungen werden meist innerhalb einer Fachkomponente bestehen, können aber auch komponentenübergreifend sein.

   Im Gegensatz zu monolithischen Anwendungen werden Bedingungen und Abhängigkeiten zumeist innerhalb einer Fachkomponente bestehen. Allerdings ist davon auszugehen, dass eine Fachkomponente nicht nur Dienste bereitstellt, sondern auch welche von anderen Fachkomponenten benötigt (siehe z.B. Turowski et al. 2002). Daher kann nicht ausgeschlossen werden, dass auch bei der Parametrisierung Abhängigkeiten zu den Parameterwerten anderer Fachkomponenten bestehen.

Wir gehen davon aus, dass die hier formulierten Thesen für Fachkomponenten gültig sind. Wir können jedoch nicht voraussetzen, dass diese vollständig sind. D.h. es werden in weiteren Untersuchungen vermutlich neue Aspekte hinzukommen, die in der Spezifikation einer Fachkomponente zu berücksichtigen sind. Zum Beispiel erfolgt in SAP R/3 die Einstellung von Prozessabläufen zumeist datenorientiert. Bei eher prozessorientierten Parametrisierungen (z.B. über eine Workflow Engine) ergeben sich eventuell zusätzliche Aspekte.

5 Ausblick: Spezifikation des Parametrisierungsspielraums

In diesem Kapitel formulieren wir erste Ideen, wie der Parametrisierungsspielraum von Fachkomponenten spezifiziert werden kann. Wir beschäftigen uns also mit der zuvor formulierten Fragestellung 2 (siehe Kapitel 2). Es handelt sich um einen Ausblick und entsprechend erheben wir keinen Anspruch darauf, alle Aspekte vollständig zu berücksichtigen.

Für die Spezifikation von Parametern und ihren möglichen Belegungen können verschiedene Techni-

ken in Betracht gezogen werden. Wir wählen an dieser Stelle die OMG Unified Modeling Language (UML) aus folgenden Gründen:

- Die in Kapitel 4 formulierten Thesen beschreiben eine Mischung aus Daten- und Prozesssicht, wofür die objektorientierte Sichtweise der UML gut geeignet ist.
- Die UML hat die Mächtigkeit, alle relevanten Aspekte abbilden zu können.
- Die UML ist eine formale Sprache, d.h. sie hat eine eindeutige Syntax und Semantik.
- Bei der UML handelt es sich um einen weit verbreiteten und bekannten Standard.
- Im Memorandum „Vorschlag zur Vereinheitlichung der Spezifikation von Fachkomponenten“ (Turowski et al. 2002) werden auf der Verhaltens- und der Abstimmungsebene ebenfalls Elemente der UML verwendet. Dadurch entsteht bei einer späteren Integration der Parametrisierungsaspekte in das Memorandum kein Methodenbruch.

Im Folgenden unterbreiten wir für die Thesen aus Kapitel 4 Vorschläge, wie diese mit Hilfe der UML abgebildet werden könnten.

Thesen 2 und 3: Alle zusammen zu pflegenden Parameter werden zu einem Entitätstyp zusammengefasst, welcher in UML durch eine Klasse abgebildet wird. Die einzelnen Parameter sind Attribute oder Assoziationen dieser Klasse. Verschiedene Ausprägungen sind auf natürliche Weise möglich, da eine Klasse verschiedene Instanzen haben kann. Um die parametrisierungsrelevanten Entitätstypen von anderen Entitätstypen zu unterscheiden, könnten die entsprechenden Klassen durch einen Stereotyp „Parametrisierung“ ergänzt werden.

These 4: Alle Klassen mit dem Stereotyp „Parametrisierung“ erhalten drei Standardmethoden: Anlegen, Ändern und Löschen. Die Methoden Anlegen und Ändern haben alle Attribute als Inputparameter. Alle Methoden haben als Export geeignete Statusmeldungen, die über den Erfolg der Methode aufklären.

These 5: Reihenfolgebedingungen können am besten mit den temporalen Operatoren ausgedrückt werden, welche als Ergänzung für die OCL vorgeschlagen wurden (Conrad/Turowski 2000).

These 6: Jedes Attribut einer UML-Klasse kann mit einem Datentyp versehen werden, welches den Wertebereich des Parameters beschreibt. Festwerte können durch einen Datentyp beschrieben werden, der die einzelnen Werte als Aufzählung enthält. Wird der Wertebereich eines Parameters durch Werte eines anderen Parameter vorgegeben, beschreiben wir dies durch eine Assoziation zur entsprechenden Klasse.

These 7: Bedingungen zwischen einzelnen Parametern können durch OCL-Ausdrücke beschrieben werden, welche die beteiligten Klassen und Attribute enthalten. Sind Bedingungen immer gültig, können sie als Invarianten abgebildet werden. Treten Bedingungen nur in Zusammenhang mit dem Anlegen, Ändern oder Löschen auf, dann handelt es sich um Vor- und Nachbedingungen zu den entsprechenden Methoden.

These 8: Bestehen Bedingungen an Entitätstypen außerhalb der Fachkomponente, müssen diese Entitätstypen ebenfalls als Klassen modelliert werden. Analog zum Vorgehen bei der Spezifikation der Dienste einer Fachkomponente (Ackermann 2001, Turowski et al. 2002) sollten diese Entitätstypen als *Extern* gekennzeichnet werden.

Beispiel: Wir hatten in Kapitel 3 schon die Torbelegung für die Anlieferung betrachtet. In einer ersten CA *Torbelegungsprofile pflegen* können verschiedene Torbelegungsprofile definiert werden. Neben dem Torbelegungsprofil und seiner Bezeichnung werden verschiedene Steuerungsdaten zu diesem Profil erfasst. Es handelt sich z.B. um die tägliche Anlieferungszeit (von-bis) oder um ein Kennzeichen, ob Anlieferungen auch ohne Einkaufsbeleg möglich sind. In einer zweiten CA *Zuordnung Profil zu Lagernummer* wird jedem konkreten Lager eines der abstrakten Torbelegungsprofile zugewiesen, welches dann die Torbelegung für dieses Lager steuert.

Für die CA *Torbelegungsprofile pflegen* definieren wir einen Entitätstyp *Torbelegungsprofil*, welcher durch eine gleichnamige Klasse repräsentiert wird. Diese Klasse hat neben dem Namen und der Bezeichnung des Profils noch 10 weitere Attribute für die Steuerdaten. Jedes der Attribute hat einen Datentyp. Die Klasse *Torbelegungsprofil* hat die drei Standardmethoden Anlegen, Ändern und Löschen. Es bestehen keine weiteren Bedingungen bei der Pflege der Parameter zu diesem Entitätstyp. (Siehe dazu auch das UML-Diagramm in Abb. 2.)

Für die CA *Zuordnung Profil zu Lagernummer* definieren wir einen Entitätstyp *ZuO_Torbelegungsprofil_Lager*, welcher ebenfalls durch eine gleichnamige Klasse repräsentiert wird. Diese Entität hat die zwei Parameter Lager und Profil. Beides sind Parameter, deren Wertebereich durch einen anderen Entitätstyp (Lager bzw. Torbelegungsprofil) vorgegeben ist. Deswegen werden diese Parameter als Assoziationen modelliert. Die Entität *Lager* wird nicht innerhalb des Bereichs Einkaufs eingestellt. Daher definieren wir eine gleichnamige Klasse im Bereich *Extern*. Durch die Definition von Assoziationen beschreiben wir nicht nur den Wertebereich der zwei Parameter *Lager* und *Torbelegungsprofil*, sondern wir drücken gleichzeitig die Existenzabhängigkeit aus. Wir müssen für diese Abhängigkeit also keine zusätzliche OCL-Bedingung formulieren.

Es können nur dann Profile zu Lagern zugeordnet werden, wenn zuvor die entsprechenden Lager und Profile definiert wurden. Es handelt sich also um eine Reihenfolgebeziehung. Diese lässt sich mit Hilfe der temporalen Operatoren ausdrücken. (Für die genaue Syntax und ihre Bedeutung siehe Turowski et al. 2002)).

Bestellung

```
inv:  ( after(Torbelegungsprofil::Anlegen(pnr: Nummer))
          and after(Extern::Lager::Anlegen(lnr: Nummer)))
             before before(ZuO-Torbelegungsprofil-Lager::Anlegen(pnr, lnr))
```

Streng genommen müsste noch berücksichtigt werden, dass das Torbelegungsprofil und das Lager nach ihrem Anlegen nicht wieder gelöscht wurden. (Auf diese temporale Bedingung könnte auch ganz verzichtet werden, da diese durch die Assoziation schon statisch formuliert wurde.)

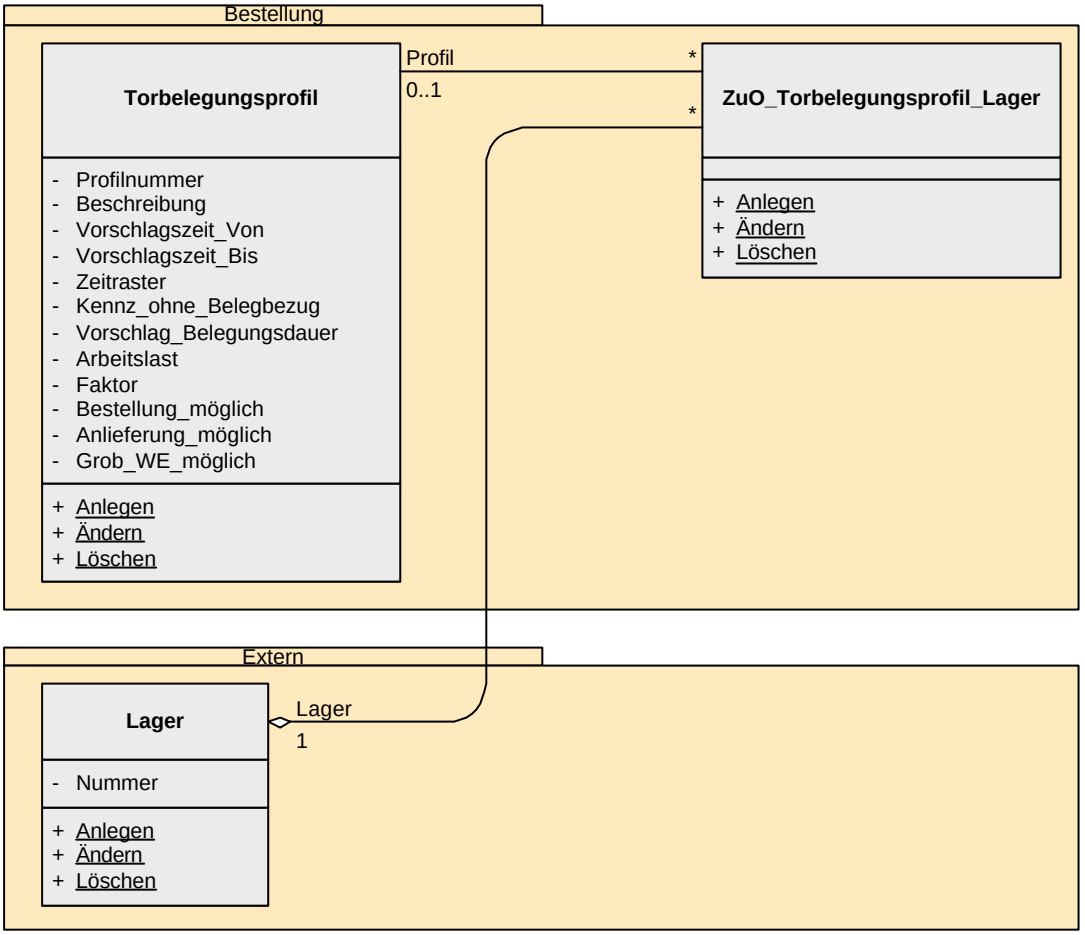

Abbildung 2: Modellierung customizingrelevanter Entitätstypen mit einem UML-Diagramm

6 Zusammenfassung und Ausblick

Fachkomponenten sollten so erstellt werden, dass ein Verwender sie in gewissem Rahmen an seine Bedürfnisse anpassen kann. Eine einfache und recht mächtige Technik steht mit der Parametrisierung zur Verfügung. Sieht der Hersteller vor, dass eine Fachkomponente parametrisierbar ist, so muss der Parametrisierungsspielraum auch spezifiziert werden.

In diesem Beitrag wurden erste Überlegungen dazu präsentiert. Nach einigen Begriffsklärungen formulierten wir im Kapitel 2 vier Fragestellungen, die in diesem Zusammenhang zu beantworten sind. Dabei wurden zwei Aspekte der Parametrisierung unterschieden: die Spezifikation der Parameter und möglicher Parameterwerte, sowie die Spezifikation der Auswirkungen von Parameterwerten auf die Funktionsweise der Fachkomponente. Da wenige Vorarbeiten existieren, muss bei beiden Aspekten einerseits untersucht werden, was genau beschrieben werden soll, und andererseits, wie das geschehen kann.

Im Kapitel 3 wurde eine Fallstudie vorgestellt, die einen Teilbereich des Customizings von SAP R/3 untersucht. Dabei wurde exemplarisch ermittelt, welche Einstellungen bei komplexen betrieblichen Anwendungen vorzunehmen sind. Daraus wurden im Kapitel 4 Thesen abgeleitet, welche Aspekte zu berücksichtigen sind, um Parameter und Parameterwerte von Fachkomponenten zu spezifizieren. Im

Kapitel 5 wurden schließlich erste Vorschläge unterbreitet, wie die möglichen Parameter(werte) bei Fachkomponenten mit Hilfe der OMG UML spezifiziert werden können.

In der Zukunft sollten die Thesen aus Kapitel 4 anhand weiterer Untersuchungen von betrieblicher Standardsoftware oder von Fachkomponenten validiert werden. Danach sollte die Fragestellung, wie Parameter(werte) zu spezifizieren sind, vollständig beantwortet werden. Außerdem wurde noch gar nicht untersucht, wie man beschreiben kann, welche Auswirkungen die Parameter(werte) auf den Betrieb von Fachkomponenten haben. Letztendlich sollten dann noch die Ergebnisse geeignet in das Memorandum „Vorschlag zur Vereinheitlichung der Spezifikation von Fachkomponenten“ integriert werden.

Literatur

Ackermann, J.: Fallstudie zur Spezifikation von Fachkomponenten. In: K. Turowski (Hrsg.): Modellierung und Spezifikation von Fachkomponenten: 2. Workshop, Bamberg, 2001.

Bergner, K.; Rausch, A.; Sihling, M.; Vilbig, A.: Adaptation Strategies in Componentware. In: Proceedings 2000 Australian Software Engineering Conference. IEEE Computer Society 2000, S. 87 – 95.

Conrad, S.; Turowski, K.: Vereinheitlichung der Spezifikation von Fachkomponenten auf der Basis eines Notationsstandards. In: Tagungsband Modellierung 2000. St. Goar, 2000.

Davies, R.: Techniques for developing reusable business components. In: Journal of Object-Oriented Programming, 9 (1996) 7, S. 40-43.

Ferstl, O.K.; Sinz, E.J.; Hammel, C.; Schlitt, M.; Wolf, S.; Popp, K.; Kehlenbeck, R.; Pfister, A.; Kniep, H.; Nielsen, N.; Seitz, A.: WEGA – Wiederverwendbare und erweiterbare Geschäftsprozeß- und Anwendungssystemarchitekturen. Abschlussbericht des Verbundprojektes. Walldorf, 1998.

Firesmith, D.: Using parameterized classes to achieve reusability while maintaining the coupling of application-specific objects. In: Journal of Object-Oriented Programming, o.Jg. (1994) 6, S. 41-44.

Floch, J.; Gulla, B.: Enabling Reuse with a Configuration Language. In: Proceedings of Fourth International Conference on Software Reuse, Los Alamitos, USA. IEEE Computer Society 1996, S. 176 – 185.

Guntermann, T.: Business-Objekte bei der Einführung von SAP R/3 – Customizing durch Anpaßbarkeit von Business-Objekten. Diplomarbeit. Universität Hohenheim, 1998.

Jacobson, I.; Griss, M.; Jonsson, P.: Software Reuse. ACM Press/Addison Wesley Longman. New York, 1997.

OMG (Hrsg.): Unified Modeling Language Specification: Version 1.4, September 2001. URL: http://www.omg.org/technology/documents/formal/uml.htm. Abruf am 2001-12-18.

Reussner R.H.: The Use of Parameterised Contracts for Architecting Systems with Software Components. In: Proceedings of the 6th International Workshop on Component-Oriented Programming. At ECOOP 2001, Budapest, Hungary, 2001.

SAP (Hrsg.): SAP Data Model – Reference Model for Business Objects. Walldorf, 1996.

SAP (Hrsg.): R/3-Referenz(prozeß)modell 4.0 im R/3 Business Engineer – Zielsetzung, Inhalte, Vorgehensweise. Walldorf, 1997.

Schütte, R.: Grundsätze ordnungsmäßiger Referenzmodellierung. Gabler Verlag, Wiesbaden, 1998.

Software Development Online: Beyond Objects. Discussion column by different authors. URL: http://www.sdmagazine.com./features/uml/beyondobjects/?topic=uml. Abruf am 2001-08-03.

Stiemerling, O.; Cremers, A. B.: Tailorable Component Architectures for CSCW-Systems. In: Proceedings of the 6th Euromicro Workshop on Parallel and Distributed Programming, Jan 21-24, 1998, Madrid, Spain. IEEE Press, 1998, S. 302-308.

Szyperski, C.: Component Software: Beyond Object-Oriented Programming. 2. Aufl., Addison-Wesley, Harlow 1998.

Turowski, K., et al.: Vorschlag zur Vereinheitlichung der Spezifikation von Fachkomponenten. Memorandum des Arbeitskreises 5.10.3 der Gesellschaft für Informatik, 2002. URL: http://www.fachkomponenten.de/. Abruf am 2002-03-01.

Turowski, K.: Standardisierung von Fachkomponenten: Spezifikation und Objekte der Standardisierung. In: 3. Meistersingertreffen, Schloss Thurnau, 1999.

Weis, T.: Component Customization. In: Proceedings of the 6th International Workshop on Component-Oriented Programming. At ECOOP 2001, Budapest, Hungary, 2001.