

Xpert.press

Die Reihe **Xpert.press** vermittelt Professionals  
in den Bereichen Softwareentwicklung,  
Internettechnologie und IT-Management aktuell  
und kompetent relevantes Fachwissen über  
Technologien und Produkte zur Entwicklung  
und Anwendung moderner Informationstechnologien.

Thomas Winkler

# ABAP/4 Programmiertechniken

Trainingsbuch

Mit 350 Abbildungen und 51 Tabellen

 Springer

Thomas Winkler  
Gerhard-Hauptmann-Str. 26  
15537 Erkner  
th.winkler1001@t-online.de

Bibliografische Information der Deutschen Bibliothek  
Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen  
Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über  
<http://dnb.ddb.de> abrufbar.

ISSN 1439-5428  
ISBN 3-540-40486-4 Springer Berlin Heidelberg New York

Dieses Werk ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere die der Übersetzung, des Nachdrucks, des Vortrags, der Entnahme von Abbildungen und Tabellen, der Funksendung, der Mikroverfilmung oder der Vervielfältigung auf anderen Wegen und der Speicherung in Datenverarbeitungsanlagen bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Eine Vervielfältigung dieses Werkes oder von Teilen dieses Werkes ist auch im Einzelfall nur in den Grenzen der gesetzlichen Bestimmungen des Urheberrechtsgesetzes der Bundesrepublik Deutschland vom 9. September 1965 in der jeweils geltenden Fassung zulässig. Sie ist grundsätzlich vergütungspflichtig. Zuwiderhandlungen unterliegen den Strafbestimmungen des Urheberrechtsgesetzes.

Springer ist nicht Urheber der Daten und Programme. Weder Springer noch die Autoren übernehmen die Haftung für die CD-ROM und das Buch, einschließlich Ihrer Qualität, Handels- und Anwendungseignung. In keinem Fall übernehmen Springer oder die Autoren Haftung für direkte, indirekte, zufällige oder Folgeschäden, die sich aus der Nutzung der CD-ROM oder des Buches ergeben.

Springer ist ein Unternehmen von Springer Science+Business Media  
[springer.de](http://springer.de)

© Springer-Verlag Berlin Heidelberg 2005  
Printed in Germany

Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Werk berechtigt auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne der Warenzeichen- und Markenschutzgesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürften. Text und Abbildungen wurden mit größter Sorgfalt erarbeitet. Verlag und Autor können jedoch für eventuell verbliebene fehlerhafte Angaben und deren Folgen weder eine juristische Verantwortung noch irgendeine Haftung übernehmen.

Satz und Herstellung: LE-TeX Jelonek, Schmidt & Vöckler GbR, Leipzig  
Umschlaggestaltung: KünkelLopka Werbeagentur, Heidelberg  
Gedruckt auf säurefreiem Papier 33/3142/YL - 5 4 3 2 1 0

# Einführung

Wieder ein neues Buch zu ABAP/4 – wer soll es lesen und wie unterscheidet es sich von den anderen Werken, die sich mit der Programmiersprache des SAP R/3-Systems befassen?

So wie das Boxen nicht durch die Analyse von Kämpfen guter Boxer zu erlernen ist, kann auch das Programmieren nicht allein durch Analysieren von Syntaxdiagrammen, Lernen von Definitionen und Klauseln gemeistert werden. Programmieren lernt man am besten durch - Programmieren.

*Inhaltübersicht*

Deshalb vermittelt Ihnen das vorliegende Buch grundlegende und weiterführende ABAP/4-Programmiertechniken, wie z.B. das Erstellen von interaktiven Listen und die Nutzung aller klassischen Komponenten der Dialogprogrammierung, am Beispiel der Entwicklung eines „Literaturrecherche- und -verwaltungsprogrammes“ für die East-Side-Library.

Ein Ausblick in die objektorientierte Programmierung mit ABAP-Objects erklärt wichtige Begriffe und Prinzipien dieser integrierten, objektorientierten Komponente der Programmiersprache ABAP – ebenfalls am Beispiel des „Literaturrecherche- und verwaltungsprogrammes“, das mit einer objektorientierten Komponente zur komfortablen Tabellenausgabe von Daten, dem ALV Grid Control, vervollkommen wird.

Das vorliegende Buch versteht sich als handlungsorientierte Ergänzung zu den ABAP/4-Standardwerken. In den einzelnen Kapiteln programmieren Sie, in leicht überschaubaren Programmierschritten, das „Literaturrecherche- und -verwaltungsprogramm“ der East-Side-Library. Zu jedem Programmierschritt wird eine kurze Einführung gegeben, den Abschluss bildet jeweils eine Programmieraufgabe deren Lösung sowohl im Buch als auch auf der mitgelieferten CD zu finden ist.

*Methode*

Durch den Transport der für die einzelnen Programmierschritte benötigten Entwicklungsobjekte von der Buch-CD in Ihr R/3-

*Vorkenntnisse* System, können Sie selbst bestimmen, mit welchem Schwierigkeitsgrad Sie Ihr ABAP/4-Training beginnen. Vorkenntnisse sind demnach nicht notwendig, wenn Sie Ihr Training mit dem ersten Kapitel beginnen. Der Anhang enthält eine detaillierte Anleitung zum Transport der Entwicklungsobjekte in Ihr R/3-System.

*Entwicklungssystem* Alle Übungen können Sie auf dem „Mini SAP-System“ durchführen, dessen neueste Version Sie für 29,00 € bei der SAP erwerben können (<http://www.sap.com/company/shop>, Link: SAP Knowledge Shop, Suchbegriff „Mini SAP“). Dieses System läuft unter Windows 2000 oder Windows XP. Sie benötigen 199 Mbyte RAM und 3,5 Gbyte freie Plattenkapazität.

*Zeichenerklärung* Im Buch werden zur Verbesserung der Übersichtlichkeit die folgenden Icons benutzt:

	Der Student macht auf wichtige Definitionen und Begriffe aufmerksam.
	Das Stoppschild warnt vor Aktionen, deren Ausführung weitreichende Konsequenzen für das R/3-System hat.
	Auf Vorgehensweisen, wie z.B. das Anlegen einer Tabelle oder eines Dynpros, weist Sie das Computersymbol hin.
	Und dieses Symbol zeigt Ihnen an, dass durch Sie eine Aufgabe zur Programmierung des „Literaturrecherche- und –verwaltungsprogramm“ zu erledigen ist.



Ich wünsche Ihnen viel Freude beim Programmieren, denn die Freude über ein funktionierendes Programm garantiert den Erfolg beim Bewältigen der nächsten Aufgabe (Hinweis: Dieses rekursive Prinzip wirkt nicht nur beim Programmieren).

*Thomas Winkler*

# Inhaltsverzeichnis

<b>1</b>	<b>Projektmanagement.....</b>	<b>1</b>
1.1	Komponenten eines SAP-R/3-Systems.....	1
1.2	Datenstruktur eines R/3-Systems .....	3
1.3	Änderungen an R/3-Datenobjekten.....	6
1.3.1	Änderungsebenen .....	6
1.3.2	Änderungsstrategien.....	8
1.4	Die Drei-System-Landschaft.....	10
1.5	Transporte durchführen .....	12
1.5.1	Transporte innerhalb eines R/3-Systems .....	12
1.5.2	Transporte in andere R/3-Systeme.....	39
<b>2</b>	<b>Wegweiser .....</b>	<b>69</b>
2.1	Projektbeschreibung.....	69
<b>3</b>	<b>Das ABAP-Dictionary.....</b>	<b>79</b>
3.1	Einführung .....	79
3.2	Domäne, Datenelement, Datenbankfeld .....	85
3.2.1	Domänen anlegen .....	86
3.2.2	Datenelemente anlegen.....	88
3.3	Eigenschaften von Tabellen.....	92
3.3.1	Tabellenarten .....	92
3.3.2	Schlüsselfelder und Primärindex.....	98
3.3.3	Sekundärindizes.....	100
3.3.4	Fremdschlüssel .....	101
3.3.5	Pufferungsarten.....	102
3.3.6	Synchronisation von Puffern.....	106
3.3.7	Änderungen an Tabellen .....	108
3.3.8	Anlegen der Tabellen für das Bibliothekprojekt .....	111
3.3.9	Anlegen und Einbinden von Suchhilfen .....	118
3.3.10	Tabellen mit Werten laden .....	124



	3.3.11 Übungsaufgaben .....	125
	3.3.12 Lösungen .....	129
<b>4</b>	<b>Grundlegende Techniken der Listenprogrammierung</b>	<b>135</b>
4.1	Zielstellung des Kapitels .....	135
4.2	Grundaufbau eines ABAP-Programmes .....	136
4.3	Ausgabe von Texten .....	140
4.4	Datentypen und Datenobjekte .....	155
	4.4.1 Eingebaute Datentypen .....	156
	4.4.2 Deklaration von Datenobjekten .....	158
	4.4.3 Arithmetische Operationen .....	168
	4.4.4 Operationen mit Zeichenketten .....	173
	4.4.5 Strukturen .....	179
	4.4.6 Interne Tabellen .....	186
	4.4.7 Globale Datentypen .....	220
4.5	Kontrollstrukturen .....	225
	4.5.1 Bedingte Verzweigungen .....	225
	4.5.2 Programmschleifen .....	228
	4.5.3 Logische Ausdrücke .....	232
4.6	Lesen von Daten aus Datenbanktabellen .....	234
	4.6.1 Die „SELECT-Anweisung“ als Schleife .....	235
	4.6.2 Einzelsatzzugriff mit der „Select single- Anweisung“ .....	242
	4.6.3 Array-Fetch – Laden einer internen Tabelle mit Daten aus einer Datenbanktabelle .....	243
	4.6.4 Der Selektionsbildschirm .....	247
<b>5</b>	<b>Spezielle Techniken der Listenerstellung</b>	<b>259</b>
5.1	Zielstellung des Kapitels .....	259
5.2	Modularisierung durch Unterprogramme .....	260
	5.2.1 Anlegen eines Includes .....	270
	5.2.2 Anlegen und Einbinden eines Unterprogrammes .....	275
5.3	Ikonen in Listen .....	281
5.4	Verzweigungslisten .....	285
	5.4.1 Anlegen von Verzweigungslisten .....	286
5.5	Die Programmoberfläche .....	297
5.6	Dynamische Auswahl von Datensätzen der Ausgabeliste .....	308
5.7	Dynamisches Sortieren der Ausgabeliste .....	314
5.8	Ein Freund des Programmierers – Der Debugger .	323
	5.8.1 Start des Debuggers .....	323
	5.8.2 Programm debuggen .....	325

5.9	Ausgabe von Meldungen (Messages) .....	331
5.10	Modularisierung mit Funktionsbausteinen.....	339
<b>6</b>	<b>Grundlagen der Dynproprogrammierung.....</b>	<b>351</b>
6.1	Zielstellung des Kapitels.....	351
6.2	Dynpros und ihre Komponenten .....	355
6.3	Statischer und dynamischer Dynproaufruf.....	357
6.4	Dateneingabe und –ausgabe mit Dynpros.....	359
6.4.1	Dynproelemente.....	361
6.4.2	Dynproelemente zur Ausgabe .....	362
6.4.3	Dynproelemente zur Ein-/Ausgabe.....	368
6.5	Programmierung der Ablauflogik .....	382
6.5.1	Module und Modulaufruf .....	382
6.5.2	Benutzeraktionen auswerten .....	388
6.6	GUI-Status und GUI-Titel des Dynpros .....	396
6.7	Eigenschaften der Dynproelemente dynamisch ändern .....	399
6.8	Eingabepfahrungen mit der FIELD-Anweisung.....	406
6.9	Bedingtes bzw. vorrangiges Ausführen von Modulen .....	414
<b>7</b>	<b>Subscreens, Listen und Tabellen in Dynpros .....</b>	<b>427</b>
7.1	Zielstellung des Kapitels.....	427
7.2	Subscreenbereiche und SubscreenDynpros .....	430
7.3	Ausgabe von Listen auf einem Dynpro.....	437
7.4	Datenausgabe mit Table Controls .....	445
7.4.1	Anlegen eines Table Controls .....	446
7.4.2	Datentransport zum Table Control und zurück.....	450
<b>8</b>	<b>Tabstrips .....</b>	<b>471</b>
8.1	Zielstellung des Kapitels.....	471
8.2	Allgemeine Eigenschaften Einsatzbedingungen....	472
8.3	Tabstrip-Elemente.....	473
8.4	Blättern im Tabstrip .....	474
8.4.1	Tabstrip mit statischer Blätterfunktion.....	474
8.4.2	Tabstrip mit dynamischer Blätterfunktion...	475
8.5	Tabstrip anlegen.....	476
<b>9</b>	<b>Datenbankänderungen programmieren .....</b>	<b>491</b>
9.1	Zielstellung des Kapitels.....	491
9.2	Datenbankändernde Anweisungen.....	493

9.2.1	Die INSERT-Anweisung.....	494
9.2.2	Die UPDATE-Anweisung.....	496
9.2.3	Die MODIFY-Anweisung.....	501
9.2.4	Die DELETE-Anweisung.....	502
9.3	Datenbankänderungen organisieren .....	508
9.3.1	Das LUW-Konzept.....	508
9.3.2	Bündelung durch Unterprogramme.....	512
9.3.3	Bündelung durch Verbucherbausteine .....	515
9.4	Das SAP-Sperrkonzept .....	520
9.4.1	Prinzip des SAP-Sperrkonzepts.....	521
9.4.2	Grundsätzliche Arbeitsweise beim Sperren und Freigeben.....	522
9.4.3	Technische Realisierung .....	522
9.4.4	Die Sperrtabelle .....	530
9.5	Nummernkreise .....	532
<b>10</b>	<b>Ausblick: ABAP Objects .....</b>	<b>563</b>
10.1	Zielstellung des Kapitels.....	563
10.2	Ein Wort zu ABAP-Objects.....	564
10.3	Objekte, Attribute, Methoden und Klassen .....	565
10.4	Klassen in ABAP Objects.....	567
10.5	Instanz- und statische Methoden, Instanz- und statische Attribute.....	570
10.6	Methoden in ABAP Objects .....	571
10.7	Anlegen von Objekten .....	572
10.8	Methodenaufrufe .....	573
10.8.1	Aufruf einer Instanzmethode .....	573
10.8.2	Aufruf einer Klassenmethode.....	574
10.9	Externer Zugriff auf öffentliche Attribute .....	575
10.10	Funktionale Methoden .....	576
10.11	Der Konstruktor, eine besondere Methode.....	579
10.12	Objekte löschen .....	581
10.13	Referenzen in internen Tabellen speichern .....	581
10.14	Globale Klassen .....	584
10.15	Vererbung und Polymorphie.....	588
10.16	Kurzer Überblick über GUI-Controls am Beispiel des ALV-Grid-Controls .....	594
	<b>Anlage.....</b>	<b>605</b>
	Installation des Übungsszenarios .....	605
	<b>Index .....</b>	<b>611</b>

# 1 Projektmanagement

## 1.1 Komponenten eines SAP-R/3-Systems

Ein SAP-R/3-System basiert, wie die meisten Leser sicher wissen, auf einer Client-Server-Architektur. Dieser Begriff lässt sich sowohl aus software- als auch aus hardwareorientierter Sicht betrachten.

*Client-Server-Architektur*

### **Softwareorientierte Sicht**

Unter „Service“ ist ein Dienst zu verstehen, der von einer Softwarekomponente angeboten wird. Solche Softwarekomponenten können aus einem einzelnen Prozess oder aus mehreren Prozessen (Prozessgruppe) bestehen und werden dann „Server“ (Diener) genannt. Softwarekomponenten, die einen „Service“ nutzen, werden als „Clients“ (Kunden) bezeichnet.



*Abb. 1.1 Softwareorientierte Sicht*

Aus softwareorientierter Sichtweise besteht ein R/3-System immer aus drei Komponenten, die oft auch als „Schichten“ bezeichnet werden:

### ■ Datenbankschicht

Die Datenbankschicht übernimmt die Datenhaltung. Sie speichert alle Anwendungstabellen und Programme im darunterliegenden Datenbanksystem und stellt sie der Anwendungsschicht zur Verfügung.

*Datenbankschicht*

*Anwendungs-  
schicht*

■ Anwendungsschicht

Diese Schicht ist verantwortlich für alle Abläufe und Funktionen der R/3-Anwendung. Hier laufen alle System- und ABAP/4-Programme.

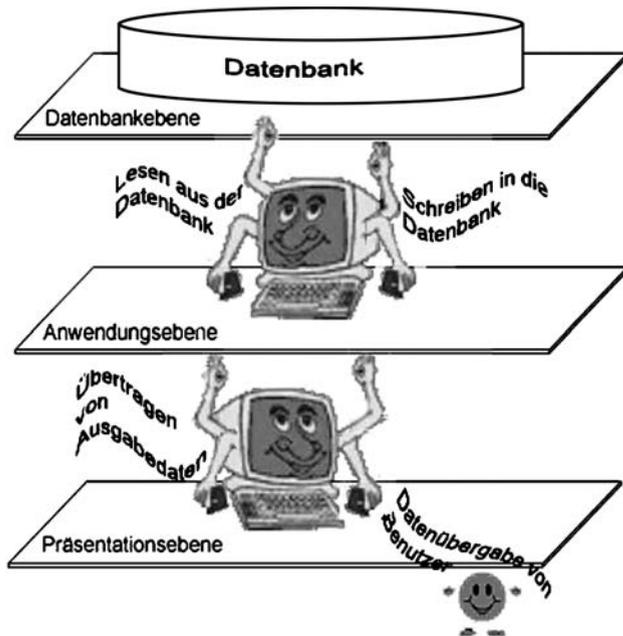
*Präsentations-  
schicht*

■ Präsentationsschicht

Die Präsentationsschicht hat folgende Hauptaufgaben:

- Entgegennehmen und Weiterleiten von Benutzeraktionen (Maus, Tastatureingaben),
- Entgegennehmen, Aufbereiten und Darstellen der Anwendungsdaten.

*Abb. 1.2  
Schichtenmodell  
eines SAP-R/3-  
Systems*



**Hardwareorientierte Sicht**

In der hardwareorientierten Sicht wird unter „Server“ ein Rechner verstanden, der bestimmte „Services“ (Dienste) anbietet. Der „Client“ ist bei dieser Betrachtungsweise ein Rechner, der Dienste des Servers in Anspruch nimmt.

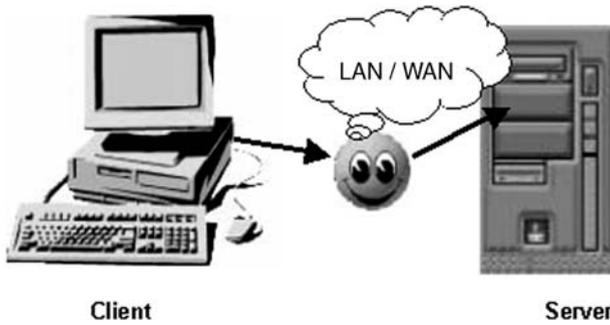


Abb. 1.3  
Hardwareorientierte Sicht

Der Vorteil der Client-Server-Architektur besteht darin, dass verschiedene Dienste (softwareorientierte Sicht) auf verschiedene Hardwarekomponenten aufgeteilt werden können. Zwei mögliche Verteilungen (Konfigurationen) eines R/3-Systems zeigt die folgende Abbildung:

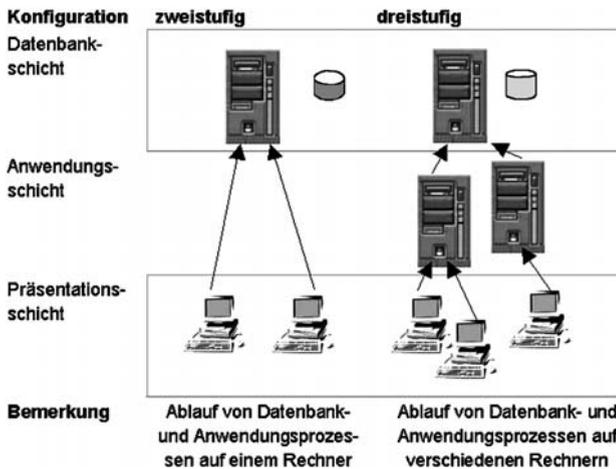


Abb. 1.4  
R/3-Konfigurationsmöglichkeiten

Theoretisch können alle Softwarekomponenten eines R/3-Systems auch in einem einzigen Rechner installiert werden. Das ist dann eine einstufige Konfiguration, die in der Praxis jedoch eher selten vorkommt.

## 1.2 Datenstruktur eines R/3-Systems

Jedes SAP-R/3-System besitzt genau eine Datenbank. In dieser können verschiedene organisatorische Bereiche, die als Mandanten bezeichnet werden, angelegt werden. Jedem dieser Mandanten ist ein

„mandantenabhängiger“ Speicherbereich zugeordnet, auf dessen Datenobjekte

- Anwendungsdaten,
- Customizingdaten,
- Userdaten,

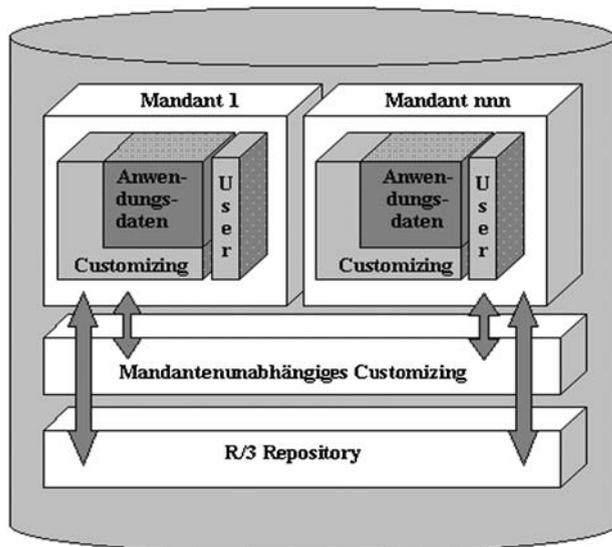
nur der jeweilige Mandant zugreifen kann.

Zusätzlich gibt es noch zwei „mandantenunabhängige“ Speicherbereiche für

- Mandantenunabhängige Customizingdaten,
- Repositorydaten,

auf deren Datenobjekte alle Mandanten zugreifen.

Abb. 1.5  
Datenstruktur  
des R/3-Systems



■ Mandant

Organisatorischer Bereich in der Datenbank, der Customizingdaten, Anwendungsdaten und Benutzerdaten kapselt.

Standardmandanten:

- Mandant 000 (SAP-R/3-Auslieferungsmandant)
- Mandant 001 (SAP-R/3-Entwicklungs- oder Customizingmandant)
- Mandant 066 (SAP-R/3-Mandant zur Durchführung der Fernwartung im Rahmen des SAP-Services EarlyWatch)

## ■ Customizingdaten

Customizing ist das Anpassen des R/3-Systems an die konkreten Bedingungen im Kundenunternehmen ohne Programmierung. Dabei werden Unternehmensdaten, wie z.B. Buchungskreise, Kostenstellen, Werke usw., in Customizingtabellen eingetragen. Jede Customizingtabelle existiert einmal pro Mandant.

*Mandanten-  
abhängige  
Customizing-  
daten*

## ■ Anwendungsdaten

Anwendungsdaten sind z.B. Dokumente, Materialstammsätze und Lieferantenstammsätze. Solche Anwendungsdaten sind nur in ihrer jeweiligen Customizing-Umgebung betriebswirtschaftlich sinnvoll. Beispiel: Material (Anwendungsdaten) wird einem Lager (Customizingdaten) zugeordnet, ein Lieferant (Anwendungsdaten) wird einem Buchungskreis zugeordnet (Customizingdaten).

*Anwendungs-  
daten*

## ■ Benutzerdaten

Für jeden R/3-Benutzer wird vom Systemadministrator ein Benutzerstammsatz angelegt. Darin stehen z.B.

- der Benutzername,
- das Kennwort,
- die Benutzerrechte.

*Benutzerdaten*

Diese Daten sind mandantenabhängig, deshalb kann sich jeder Benutzer mit seinen Daten (Benutzername und Kennwort) nur an den Mandanten anmelden, in denen ein entsprechender Benutzerstammsatz angelegt wurde.

## ■ Mandantenunabhängiges Customizing

Neben dem mandantenabhängigen Customizing gibt es auch das mandantenunabhängige. Die hier bearbeiteten Customizingdaten stehen dann für alle Mandanten zur Verfügung. Beispiele dafür sind:

- die Druckereinstellungen und
- der Werktagskalender

*Mandanten-  
unabhängiges  
Customizing*

## ■ R/3 Repository

Das Repository ist ein Teil der zentralen Datenbank, in dem alle Objekte, die in der ABAP Workbench angelegt wurden, gespeichert sind. Zu diesen Daten gehören:

- Programme
- Funktionsbausteine
- Klassen

*R/3 Repository*

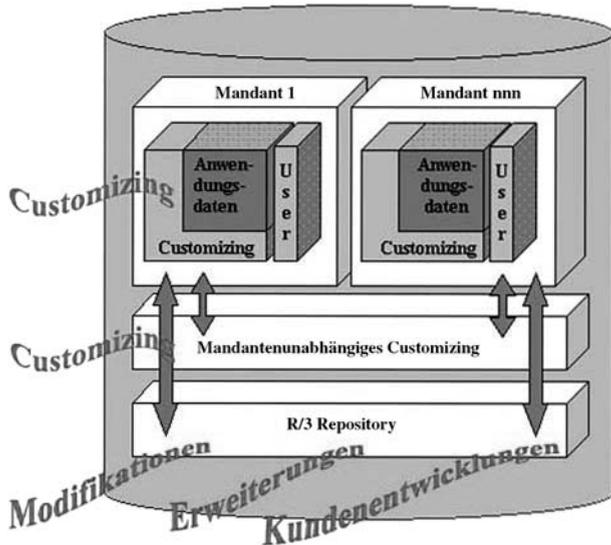
- Tabellen
- Views
- Datenelemente
- Domänen
- Suchhilfen
- Sperrobjekte

## 1.3 Änderungen an R/3-Datenobjekten

### 1.3.1 Änderungsebenen

Änderungen können sowohl an mandantenabhängigen als auch an mandantenunabhängigen Datenobjekten durchgeführt werden. Im folgenden Kapitel sollen unterschiedliche Methoden zur Änderung von R/3-Datenobjekten gezeigt werden.

Abb. 1.6  
Ebenen von  
Datenbankänderungen und  
Anpassungen



*Kundenentwicklungen*

- Kundenentwicklungen

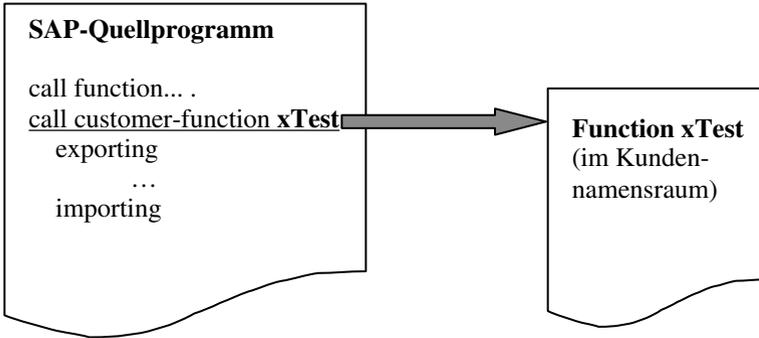
Der Kunde entwickelt selbst neue Repository-Objekte (Programme, Tabellen, Views, ...).

*Erweiterungen*

- Erweiterungen

Der Kunde nutzt von der SAP vorgedachte Programmschnittstellen, um die Funktionalität der Standardprogramme zu erweitern. Diese Schnittstellen werden auch als „Exits“ bezeichnet.

Prinzipdarstellung:



*Abb. 1.7 Ebenen von Datenbankänderungen und Anpassungen*

Wie an der Prinzipdarstellung zu erkennen ist, muss der SAP-Programmierer einen Exit in das SAP-Programm eingearbeitet haben. In diesem Fall sorgt ein „Funktionsbaustein-Exit“ für den Aufruf eines Funktionsbausteins im Kundennamensbereich. Es gibt folgende Exits:

- Funktionsbausteinexit  
Vom SAP-Programm wird ein Funktionsbaustein im Kundennamensbereich aufgerufen.
- Dynproexit  
Im Dynpro ist ein Subscreenbereich, der ein Subscreen im Kundennamensbereich aufruft, implementiert.
- Menüexits  
Im Menü (GUI-Oberfläche) der SAP-Anwendung befindet sich ein Menüpunkt, der einen Funktionsbausteinexit aufruft.
- Modifikationen

*Funktionsbausteinexit*

*Dynproexit*

*Menüexit*

*Modifikationen*

Modifikationen sind Änderungen, die der Kunde direkt am SAP-Programm durchführt. Modifikationen sollten nach Möglichkeit nicht angewendet werden, weil sie zu Komplikationen bzw. Mehraufwand beim Update führen.

### 1.3.2 Änderungsstrategien

Das Durchführen von Änderungen erfordert bei so umfangreichen Systemen, wie es das SAP-R/3 darstellt, einen hohen administrativen Aufwand. Folgende Hauptanforderungen sollten durch technische und administratorische Maßnahmen gewährleistet sein:

- Die Tests der Änderungen und Anpassungen sollten immer auf einem genau definierten Stand basieren.
- Test- und Entwicklungsarbeiten sollten parallel erfolgen können.
- Nicht getestete Änderungen dürfen nicht praxiswirksam werden.

Um diese Anforderungen zu erfüllen, bedarf es unterschiedlicher Voraussetzungen bei Änderungen an mandantenunabhängigen und mandantenabhängigen Daten.

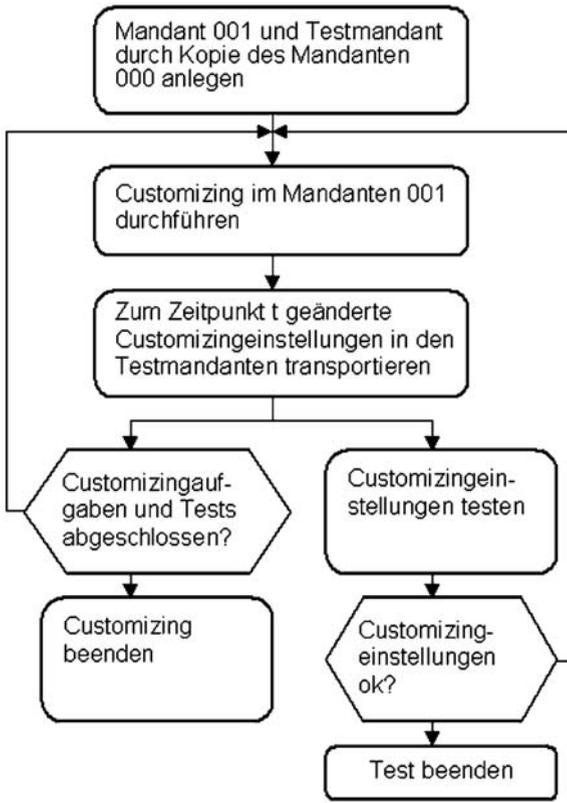
#### ***Änderungsstrategie für mandantenabhängige Daten***

Die Änderungen an mandantenabhängigen Daten (Customizing) werden im Entwicklungsmandanten (Mandant 001), der zunächst eine 1:1 Kopie des Auslieferungsmandanten (Mandant 000) ist, durchgeführt.

Zu einem bestimmten Zeitpunkt werden alle mandantenabhängigen Änderungen in einen dritten Mandanten (Testmandant) transportiert und dort getestet. Während der Testphase können die Entwicklungsarbeiten im Entwicklungsmandanten weitergeführt werden, ohne dass sie sich auf den Test auswirken.

Durch den Test entdeckte Fehler werden im Entwicklungsmandanten korrigiert und erneut in den Testmandanten transportiert. Am Ende enthält der Entwicklungsmandant alle (ausreichend getesteten) mandantenabhängigen Einstellungen. In Abb. 1.8 ist die Änderungsstrategie für mandantenabhängige Daten grafisch dargestellt.

Abb. 1.8  
 Änderungen an  
 mandantenab-  
 hängigen Daten



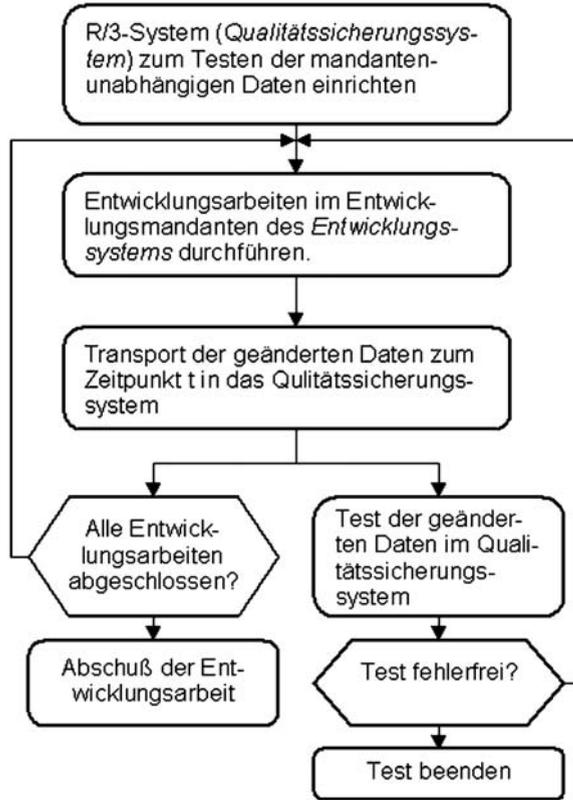
### **Änderungsstrategie für mandantenunabhängige Daten**

Änderungen an mandantenunabhängigen Datenobjekten wirken sich sofort auf alle Mandanten eines SAP-R/3-Systems aus. Daraus ergeben sich folgende Konsequenzen:

- Es ist nicht möglich, innerhalb eines SAP-R/3-Systems Test- und Entwicklungsarbeiten an mandantenunabhängigen Datenobjekten parallel durchzuführen.
- Der Mandant, auf dem der Produktivbetrieb läuft, und der Entwicklungsmandant sind in verschiedenen R/3-Systemen zu installieren. Änderungen an mandantenunabhängigen Datenobjekten würden sich sonst sofort, d.h. ohne getestet worden zu sein, auf die Produktivumgebung auswirken.

Sollen also Änderungen an Repository-Objekten oder an Objekten des mandantenunabhängigen Customizings vorgenommen werden, müssen im Unternehmen mindestens zwei SAP-R/3-System betrieben werden – ein Entwicklungssystem und ein Produktivsystem.

Abb. 1.9  
 Änderungen an  
 mandanten-  
 unabhängigen  
 Daten



## 1.4 Die Drei-System-Landschaft



Systemland-  
 schaft

Unter einer Systemlandschaft sind alle miteinander durch Transportwege verbundenen SAP-R/3-Systeme mit den darin angelegten Mandanten zu verstehen.

Im Abschnitt „Änderungsstrategie für mandantenunabhängige Daten“ (Seite 9) wurde die Notwendigkeit begründet, mindestens zwei R/3-Systeme zu betreiben. Die dort beschriebene Systemlandschaft (Entwicklungssystem und Produktivsystem) hat jedoch den Nachteil, dass Entwicklung und Test nicht parallel erfolgen können. Deshalb empfiehlt SAP, diese Zwei-System-Landschaft um ein R/3-System, das Qualitätssicherungssystem (QAS), zur Drei-System-Landschaft zu erweitern.

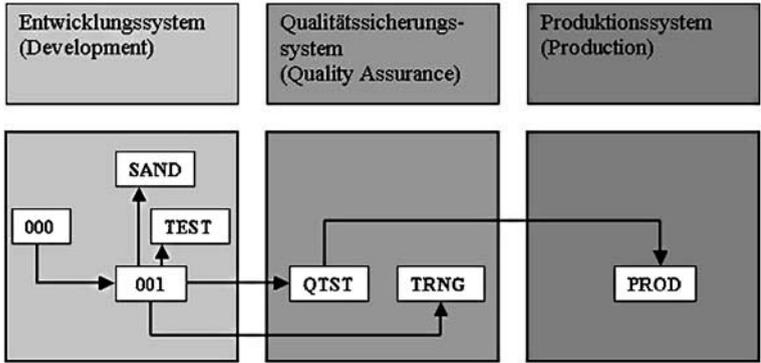


Abb. 1.10  
Drei-System-  
Landschaft

Grundsätzlich erfolgen alle Entwicklungsarbeiten im Entwicklungssystem (Mandant 001). Die geänderten Datenobjekte werden nach Abschluss der Entwicklungsarbeiten in das Qualitätssicherungssystem transportiert und unabhängig von weiteren Entwicklungsarbeiten getestet. Alle in das Qualitätssicherungssystem transportierten Objekte lassen sich automatisch in das Produktivsystem übernehmen.

R/3-System	Mandant	Aktivitäten
<b>Entwicklungssystem</b>	001	Entwicklung
	TEST	Testen der mandantenabhängigen Einstellungen
	SAND	Experimentieren mit den Customizingwerkzeugen
<b>Qualitätssicherungssystem</b>	QTST	Customizingeinstellungen und Programmentwicklungen werden in einer realitätsnahen Umgebung getestet
	TRNG	Schulung der Endanwender
<b>Produktionssystem</b>	PROD	produktive Arbeit

Tabelle 1.1  
Mandantenrollen

# 1.5 Transporte durchführen

## 1.5.1 Transporte innerhalb eines R/3-Systems

### 1.5.1.1 **Voraussetzungen und Durchführung**

Um geänderte mandantenabhängige Daten unabhängig von weiteren Änderungen an den Customizingtabellen testen zu können, werden sie in den Testmandanten des Entwicklungssystems transportiert. Dazu müssen vorher folgende Voraussetzungen geschaffen worden sein:

*Voraussetzungen für das Customizing*

- der Testmandant ist angelegt (siehe Seite 12),
- der Projektleiter hat für alle Entwickler einen Customizingauftrag angelegt (siehe Seite 16),
- im Entwicklungsmandanten (001) ist die Eigenschaft „automatische Aufzeichnung von Änderungen“ aktiviert (siehe Seite 18).

Die Datenänderungen und der Transport der geänderten Daten erfolgt in folgenden Schritten:

*Schritte zum Transport von Änderungen innerhalb eines R/3-Systems*

- Die Daten werden mit den Customizingwerkzeugen gepflegt. Beim Sichern der Änderungen werden diese in einen speziellen Abschnitt des Customizingauftrages, der dem jeweiligen Entwickler zugeordnet ist, eingetragen. Dieser spezielle Abschnitt wird als „Aufgabe“ bezeichnet (siehe Seite 17).
- Hat der Entwickler seine Customizingaufgaben abgeschlossen, gibt er seine Aufgabe frei (siehe Seite 27).
- Hat jeder Entwickler seine Aufgabe freigegeben, gibt der Projektleiter den gesamten Auftrag frei (siehe Seite 27).
- Zum Schluss werden vom Projektleiter alle im Customizingauftrag gekapselten Änderungen in den Zielmandanten transportiert (siehe Seite 30).

### 1.5.1.2 **Anlegen des Testmandanten**

Der Testmandant wird durch Kopieren des Mandanten 000 (Auslieferungsmadant) erzeugt. Die Kopie wird in zwei Schritten erstellt:

### Erster Schritt: Deklaration des Testmandanten

Alle Mandanten eines Systems werden in die Tabelle T000 eingetragen.

#### Vorgehensweise: Mandantendeklaration

Die Mandantendeklaration wird mit SCC4 vorgenommen.  
(Werkzeuge → Administration → Verwaltung → Mandantenverwaltung → SCC4 Mandantenpflege)

Das Einstiegsbild *Sicht „Mandanten“ ändern: Übersicht* zeigt alle Mandanten des R/3-Systems an dem Sie angemeldet sind.



Mandant	Bezeichnung	Ort	Währ.	geändert am
000	SAP AG Konzern	Walldorf	DEM	11.08.2000
001	Entwicklungsmandant	Berlin	EUR	24.06.2003

Abb. 1.11  
*Sicht „Mandanten“ ändern: Übersicht*  
(Anzeige der Datensätze der Tabelle T000)

Im Bild *Neue Einträge: Detail Hinzugefügte* werden die Eigenschaften des neuen Mandanten festgelegt (siehe Abb. 1.11).

Mandant: 002 | Testmandant

Ort: Berlin | Letzter Änderer: | Datum: |

Logisches System: |

Std. Währung: EUR

Rolle des Mandanten: Test

Anderungen und Transporte:

- Änderungen ohne auto
- automatische Aufzeichn
- keine Änderungen erla
- Änderungen ohne autom. Aufzeichnung, keine Transporte erlaubt

Anderungen an mandantenübergreifenden Objekten:

Anderungen an Repository und mand.unabh. Customizing erlaubt

Abb. 1.12  
*Deklaration eines neuen Mandanten in der Tabelle T000*

### Zweiter Schritt: Mandantenkopie durchführen

Nachdem der Testmandant deklariert ist, müssen die Customizingtabellen vom Quellmandanten (000) in den Testmandanten (002) kopiert werden.



**Achtung:** Bei der Mandantenkopie werden *mehrere Tausend* Tabellen kopiert. Das erfordert erhebliche Systemressourcen. Außerdem wird für die Zeit der Ausführung der Mandantenkopie der Quellmandant für weitere Anmeldungen gesperrt. Sollten Sie beabsichtigen, die Mandantenkopie wirklich auszuführen, sprechen Sie mit Ihrem Systemadministrator – am besten vorher, damit kein Frust aufkommt.



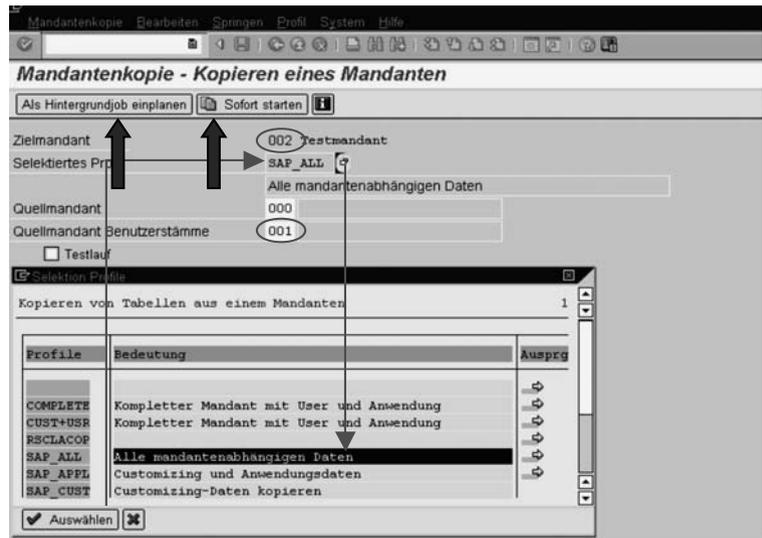
*Vorgehensweise: Mandantenkopie*

**Wichtig:** Sie müssen sich am Testmandanten (Zielmandant) anmelden. Weil im Testmandanten jedoch noch keine Benutzerstammdaten angelegt sind, benutzen Sie folgende Login-Daten:

Mandant: 002 (Nr. des Zielmandanten)  
Benutzer: SAP\*  
Kennwort: Pass

Im Testmandanten starten Sie die Mandantenkopie über die Transaktion SCCL (Werkzeuge → Administration → Verwaltung → Mandantenverwaltung → Mandantenkopie → SCCL lokale Kopie)

Abb. 1.13  
Mandantenkopie  
– Kopieren eines  
Mandanten



Gehen Sie wie folgt vor:

- Kontrollieren Sie, ob der richtige Zielmandant (002) eingetragen ist.
- Wählen Sie im Feld „Selektiertes Profil“ ein Profil aus. Über das Profil steuern Sie, welche Daten vom Quellmandan-

ten in den Zielmandanten transportiert werden. Ihnen stehen folgende Selektions- Standardprofile zur Verfügung:

<b>Profil</b>	<b>Daten</b>
SAP_ALL	Alle mandantenabhängigen Daten
SAP_APPL	Customizing- und Anwendungsdaten
SAP_CUST	Customizingdaten
SAP_CUSV	Customizingdaten und Reportvarianten
SAP_UAPP	Benutzerstämme, Reportvarianten, Anwendungsdaten
SAP_UCSV	Customizingdaten, Reportvarianten und Benutzerstämme
SAP_UCUS	Customizingdaten und Benutzerstämme
SAP_USER	Benutzerstämme und Berechtigungsprofile

*Tabelle 1.2  
Selektionsprofile*

- **Hinweis:** Die Customizingtabellen sind einmal pro Mandant vorhanden. Sie werden beim Anlegen des Mandanten vom Quell- in den Zielmandanten kopiert. Tabellen für Anwendungsdaten, z.B. die Debitorenstammtabelle existiert nur einmal pro R/3-System. Die Mandantenabhängigkeit wird dadurch erreicht, dass die Mandantennummer Bestandteil des Datensatzes ist. Die Tabellen für die Anwendungsdaten werden also nicht kopiert, sondern die Datensätze dieser Tabellen werden mit der Nummer des Zielmandanten neu angelegt.
- Geben Sie im Feld „Quellmandant“ den Quellmandanten (000) an.
- Geben Sie im Feld „Quellmandant Benutzerstämme“ einen Mandanten an, der die Benutzerstammsätze enthält, die im Zielmandanten benötigt werden. Das könnte zum Beispiel der Mandant 001 (Entwicklungsmandant) sein, weil in dieser Phase der Entwickler seine Änderungen meist selbst testet.
- Starten Sie die Mandantenkopie über eine der folgenden Schaltflächen:
  - „Als Hintergrundjob einplanen“ oder
  - „Sofort starten“.

Während der Mandantenkopie dürfen weder am Quellmandanten noch am Zielmandanten Daten geändert werden.

### 1.5.1.3

## Anlegen eines Customizingauftrages

Das Customizing ist im Allgemeinen projektorientiert, d.h. unterschiedliche Aufgabenkomplexe werden von verschiedenen Projektteams bearbeitet. Der Transport-Organizer unterstützt diese Arbeitsweise. Auf der Grundlage eines Customizingauftrages werden folgende administratorischen Aufgaben der Projektarbeit gelöst:

- Zuordnung von Änderungen zum jeweiligen Entwickler
- Kapselung aller Änderungen die ein Projektteam durchführt.
- Sicherstellen, dass *alle* zu einem Projekt gehörenden Änderungen zum richtigen Zeitpunkt transportiert werden.
- Prüfen, ob alle Entwickler ihre Aufgaben beendet haben.



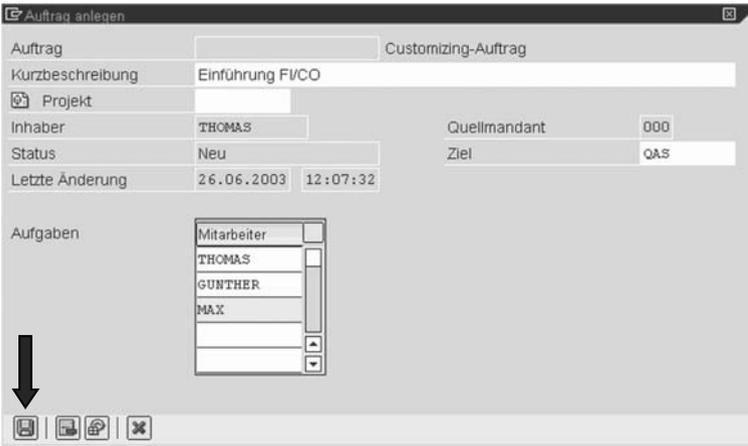
*Vorgehensweise: Customizingauftrag anlegen*

Customizingaufträge werden im Transport-Organizer (SE10) angelegt (Werkzeuge → AcceleratedSAP → Customizing → SE10 Transport Organizer). Füllen Sie das Einstiegsbild des Transport-Organizers wie in Abb. 1.14 aus. Klicken Sie dann die Schaltfläche „Anlegen“.

Abb. 1.14  
Einstiegsbild  
Transport-  
Organizer



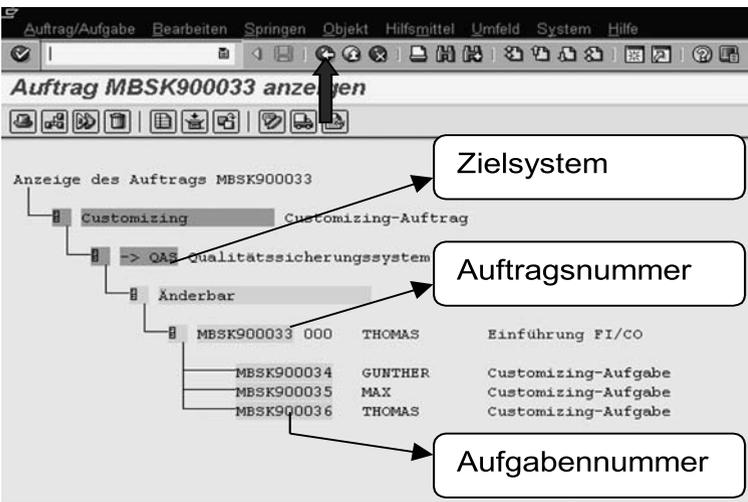
Abb. 1.15  
Auftrag anlegen



Geben Sie in das Eingabefeld „Kurzbeschreibung“ einen Titel für Ihren Customizingauftrag ein. Wenn Sie in der komfortablen Lage sind, die anfallenden Customizingarbeiten auf andere Teammitarbeiter delegieren zu können, tragen Sie in die Datengruppe *Aufgaben* deren Benutzernamen ein. Ihr eigener Benutzername wird vom System in diese Datengruppe übernommen.

Ist die Systemlandschaft bereits angelegt ist, wird im Feld „Ziel“ das R/3-System angegeben, in welches die Änderungen nach Abschluss der Entwicklungsarbeiten transportiert werden sollen. Bei einer Drei-System-Landschaft ist das das Qualitätssicherungssystem QAS.

Abb. 1.16  
Auftrag anzeigen



In diesem Fenster ist zu erkennen:

- Für jeden Auftrag wird vom System automatisch eine Auftragsnummer ermittelt. Diese setzt sich zusammen aus dem Systemnamen (hier: MBS), einer Konstanten „K9“ und einer 5-stelligen laufenden Nummer (hier „00033“).
- Für jeden Entwickler, der beim Anlegen des Auftrages in die Datengruppe „Aufgaben“ eingetragen wurde (siehe Abb. 1.14), ist automatisch eine „Aufgabe“ angelegt worden. Beim Sichern der Änderungen wählt der Entwickler den Customizingauftrag aus. Das System ermittelt den Benutzernamen des Entwicklers und trägt die Änderungen in die entsprechende Aufgabe des Customizingauftrages ein.
- Ist die Systemlandschaft angelegt, wird das Zielsystem (hier das Qualitätssicherungssystem QAS) in die Auftragshierarchie eingetragen. Dadurch kann man die Änderungen später auch in das Zielsystem transportieren.

#### 1.5.1.4

#### **Automatische Aufzeichnung von Änderungen**

Die automatische Aufzeichnung von Änderungen bewirkt, dass der Entwickler beim Sichern seiner Änderungen vom System nach der Auftragsnummer gefragt wird. Die Zuordnung der Änderungen zu einem Auftrag erfolgt nur, wenn diese Eigenschaft des Mandanten aktiviert ist. Durch die „automatische Aufzeichnung von Änderungen“ werden die geänderten Objekte an das Korrektur- und Transportwesen (CTW) des R/3-Systems angeschlossen. Damit ist sichergestellt, dass keine Änderungen beim Transport in den Zielmandanten bzw. in das Zielsystem „vergessen“ werden – ein Highlight des R/3-Systems.



*Vorgehensweise: Automatische Aufzeichnung von Änderungen aktivieren*

Die Eigenschaften der Mandanten eines R/3-Systems werden mit der Transaktion SCC4 (Werkzeuge → Administration → Verwaltung → Mandantenverwaltung → SCC4 Mandantenpflege) geändert. Stellen Sie im *Bild Sicht* „Mandanten“ ändern: Übersicht (Abb. 1.17) durch Anklicken der Schaltfläche  (Ändern → Anzeigen) den Änderungsmodus ein und wählen Sie danach durch Doppelklick den Entwicklungsmandanten aus.

Mandant	Bezeichnung	Ort	Währ.	geändert am
000	SAP AG Konzern	Walldorf	DEM	11.08.2000
001	Entwicklungsmandant	Berlin	EUR	24.06.2003
002	Testmandant	Berlin	EUR	25.06.2003

Abb. 1.17  
Sicht Mandanten ändern: Übersicht

**Sicht "Mandanten" ändern: Detail**

Mandant: 001 Entwicklungsmandant

Ort: Berlin      Letzter Änderer: DDIC

Logisches System:      Datum: 24.06.2003

Std. Währung: EUR

Rolle des Mandanten: Customizing

**Änderungen und Transporte für mandantenabhängige Objekte**

- Änderungen ohne automat. Aufzeichnung
- automatische Aufzeichnung von Änderungen
- keine Änderungen erlaubt
- Änderungen ohne autom. Aufzeichnung, keine Transporte erlaubt

**Änderungen an mandantenübergreifenden Objekten**

Änderungen an Repository und mand.unabh. Customizing erlaubt

Abb. 1.18  
Sicht Mandanten ändern: Detail

### 1.5.1.5 Ausflug ins Customizing

Das Customizing ist die Anpassung der Standardsoftware R/3 an die konkreten Bedingungen des Anwenderunternehmens. Dieser Teil der Kundenanpassung wird ohne ABAP-Programmierung ausgeführt. Beispiele dafür sind:

- das Festlegen allgemeiner Einstellungen (z.B. im Unternehmen gebräuchliche Maßeinheiten)
- die Abbildung der Unternehmensstruktur im R/3-System.

Die allgemeinen Einstellungen bzw. die Struktureinheiten, wie Buchungskreise, Kostenstellen, Werke, Lager, werden mit speziellen R/3-Werkzeugen in die entsprechenden Customizingtabellen eingetragen.

Ausgangspunkt des Customizings ist der Referenz-Einführungsleitfaden, auch Referenz-IMG genannt

Referenz-IMG

(IMG → Implementation Guide). Der Referenz-IMG enthält

- alle Customizingaufgaben und
- die notwendigen Customizingwerkzeuge.

*Projekt-IMG*

Daraus wird der Projekt-Einführungsleitfaden, auch als Projekt-IMG bezeichnet, abgeleitet. Der Projekt-IMG enthält

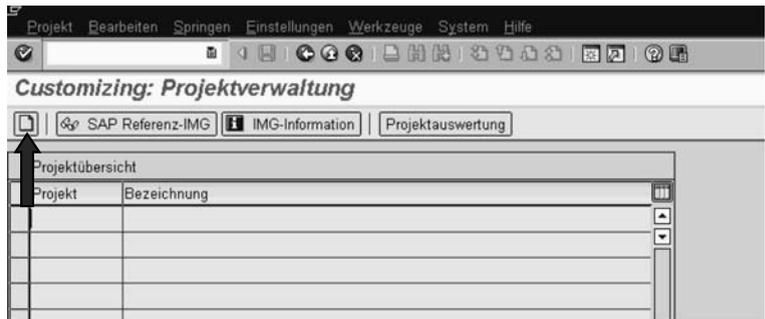
- alle Customizingaufgaben für ein Projektteam,
- alle notwendigen Customizingwerkzeuge,
- Tools zur Verwaltung des Projektes.



*Vorgehensweise: Projekt-IMG anlegen*

Der Projekt-IMG wird mit der Transaktion SPRO\_ADMIN (Werkzeuge → AcceleratedSAP → Customizing → SPRO\_ADMIN-Projektverwaltung) angelegt.

**Abb. 1.19**  
*Customizing:  
Projekt-  
verwaltung*



Um den Referenz-IMG anzuzeigen, klicken Sie auf die Schaltfläche *SAP Referenz-IMG*. Einen neuen Projekt-IMG erzeugen Sie über die Schaltfläche „Anlegen“.

**Abb. 1.20**  
*Projekt anlegen  
mit Vorlage*



Legen Sie einen Projektnamen fest und drücken Sie dann die ENTER-Taste. Sie gelangen in das Fenster *Projekt anlegen*. Alle Registerkarten dieses Werkzeuges hier zu erläutern würde den „Ausflug ins Customizing“ in eine eher knochige Bergtour verwandeln. Deshalb beschränkt sich dieses Buch auf die Registerkarten „Allge-