

Jürgen Schwaninger

ABAP®-Programmierung für die SAP®-Materialwirtschaft – Kundeneigene Erweiterungen



Galileo Press

Bonn • Boston

Inhalt

Vorwort	9
Einleitung	11

1 Allgemeines zu User-Exits und BAdIs 15

1.1	Verwendung von User-Exits	15
1.1.1	Erweiterungen finden und anschauen	16
1.1.2	Projekt anlegen und Erweiterungen zuordnen	16
1.1.3	Komponenten des Projektes verwenden	17
1.1.4	Projekte aktivieren und deaktivieren	19
1.2	Verwendung von klassischen BAdIs	19
1.2.1	BAdIs finden und anschauen	20
1.2.2	BAdI-Implementierung anlegen	22
1.2.3	Arbeiten mit Methoden	23
1.2.4	BAdIs aktivieren und deaktivieren	25
1.2.5	Erweiterte Bearbeitungsmöglichkeiten	25
1.3	Verwendung von neuen BAdIs (Erweiterungsspots)	26
1.3.1	SAP Enhancement Framework	26
1.3.2	Erweiterungsspots finden und anschauen	27
1.3.3	Erweiterungsimplementierung anlegen	28
1.3.4	Arbeiten mit Methoden	30
1.3.5	BAdIs aktivieren und deaktivieren	31

2 User-Exits und BAdIs im Einkauf 33

2.1	Kundeneigene Felder in Bestellungen	33
2.1.1	Überblick über die Implementierung	34
2.1.2	Implementierung eigener Bestelldaten und der Funktionsgruppe	38
2.1.3	Integration der eigenen Felder in die BAdIs	49
2.1.4	Anbindung der Kundenfelder an die Geschäftslogik	56
2.1.5	Initialisierung, Lesen und Verbuchung der Daten	63
2.1.6	Ausgabe von Fehlermeldungen	65
2.2	Archivierung der Daten	67
2.2.1	BAdI ARC_MM_EKKO_CHECK	67
2.2.2	BAdI ARC_MM_EKKO_WRITE – Eigene Daten archivieren	68

2.2.3	BAdI ARC_MM_EKKO_WRITE – Eigene Daten löschen	72
2.3	Anpassung der Belegübersicht in Bestellanforderungen oder Bestellungen	75
2.3.1	Entfernung einer Standardselektionsvariante	76
2.3.2	Eigene Selektionsvarianten einfügen	78
3	User-Exits und BAdIs in der Dienstleistungsabwicklung	87
3.1	Kontierung für Leistungszeilen vorbelegen	87
3.2	Eingabeüberprüfung der Leistungszeilen	90
3.2.1	Felder in EXIT_SAPLMLSP_030 vorbelegen	91
3.2.2	Eingabeüberprüfung in EXIT_SAPLMLSP_031	92
3.3	Vorbelegung der Kopfdaten im Erfassungsblatt	93
4	User-Exits und BAdIs in der Bestandsführung	97
4.1	Eigene Felder in Transaktion MIGO	97
4.1.1	Kundeneigene Felder – Ein Überblick	98
4.1.2	Vorbereitungen im ABAP Dictionary	101
4.1.3	Vorbereitung der Funktionsgruppe	102
4.1.4	Vorbereitung und Statusverwaltung in BAdI MB_MIGO_BADI	109
4.1.5	Aktivierung der eigenen Kopfdaten	114
4.1.6	Aktivierung der eigenen Positionsdaten	117
4.1.7	Verbuchung der Daten	121
4.2	Weitere Funktionen des BAdIs MB_MIGO_BADI	123
4.2.1	Merken der eigenen Daten	123
4.2.2	Eingabeüberprüfungen in Transaktion MIGO	127
4.3	Standardfelder prüfen und vorbelegen	130
4.3.1	Vorbelegung von Lagerort und Positionstext	130
4.3.2	Prüfung der Standardfelder	130
4.4	Prüfung des frühesten Lieferdatums	132
4.5	Toleranzgrenzen zu Lieferplänen	134
4.5.1	Überlieferungsmenge überschreiben	134
4.5.2	Vorschlagsmenge überschreiben	136
4.6	Erweiterung von Reservierungen	138
4.6.1	Felder vorbelegen	139
4.6.2	Überprüfung der Eingabe	142

5	User-Exits und BAdIs im Bereich Bewertung und Kontierung	145
5.1	WE/RE-Verrechnungskonto	145
5.2	Übersteuerung der Kontenfindung im User-Exit	147
6	User-Exits und BAdIs in der Logistik-Rechnungsprüfung	151
6.1	Kundeneigene Felder in Transaktion MIRO	151
6.1.1	Überblick über die Lösung per BAdI	152
6.1.2	BAdI im Detail – Anpassungen im ABAP Dictionary	154
6.1.3	Eigenes Dynpro mit Table Control erstellen	157
6.1.4	Vorbereitung der Daten im BAdI	160
6.1.5	Zurück im Dynpro	164
6.2	Toleranzprüfungen übersteuern	170
6.2.1	Toleranzgrenzen im Customizing	171
6.2.2	Nutzung der Erweiterung	172
7	User-Exits und BAdIs im Materialstamm	177
7.1	Eingabeüberprüfungen und einfache Vorbelegungen	177
7.1.1	Überblick zum User-Exit EXIT_SAPLMGMU_001	177
7.1.2	Prüfung der erfassten Materialstammdaten	178
7.1.3	Vorbelegen einzelner Materialstammdaten	181
7.2	Komplexe Vorbelegung von Materialstammdaten	183
7.2.1	BADI_MATERIAL_REF – Was Sie vorab wissen müssen	184
7.2.2	Vorbelegen von werkspezifischen Daten	185
8	Validierung und Substitution von Buchhaltungsbelegen ...	193
8.1	Validierung von Buchhaltungsbelegen	194
8.1.1	Zeitpunkte	194
8.1.2	Schritte	195
8.1.3	Beispiel ohne Exit-Routine	196
8.1.4	Beispiel mit Exit-Routine	199
8.2	Substitution von Buchhaltungsbelegen	204
8.2.1	Substitution ohne Exit-Routine	205

8.2.2	Substitution mit Exit-Routine	207
8.2.3	Lesender Zugriff auf Daten des Ursprungsbelegs	210

Anhang 215

A	User-Exits und BADs in der SAP-Materialwirtschaft	215
A.1	Einkauf	216
A.1.1	Bestellanforderungen allgemein	216
A.1.2	Bestellungen allgemein	221
A.1.3	Rahmenverträge (Lieferpläne/Kontrakte)	227
A.1.4	Preisfindung	230
A.1.5	Obligofunktionen	232
A.1.6	Belegübergreifend	234
A.1.7	Lieferantenbeurteilung	238
A.1.8	IDoc-Verarbeitung	238
A.1.9	Logistik-Informationssystem	240
A.1.10	Archivierung	240
A.2	Dienstleistungsabwicklung	243
A.3	Bestandsführung	250
A.3.1	Materialbelege allgemein	250
A.3.2	Wareneingang	256
A.3.3	Reservierungen	258
A.3.4	Archivierung	259
A.4	Bewertung und Kontierung	261
A.5	Logistik-Rechnungsprüfung	264
A.5.1	Allgemein	264
A.5.2	Archivierung	278
A.5.3	Herkömmliche Rechnungsprüfung	278
A.6	Materialstamm	280
A.6.1	Allgemein	280
A.6.2	Archivierung	283
B	Der Autor	285
	Index.....	287

Vorwort

Manche Dinge geschehen einfach überraschend. So auch die plötzliche Anfrage vor mittlerweile drei Jahren aus dem Lektorat von SAP PRESS, ob ich nicht ein Buch über User-Exits und BAdIs in der SAP-Materialwirtschaft verfassen möchte. Ich hatte zwar zu diesem Zeitpunkt bereits begonnen, ein kleines Online-Buch zum ABAP Debugger zu erstellen, aber ein richtiges Buch zu schreiben – das war schon etwas ganz anderes. Doch in den vielen Jahren, in denen ich bereits als Logistikberater und -entwickler tätig war, hatte ich wahrscheinlich Hunderte von Erweiterungen für Kunden ausprogrammiert, und so dachte ich mir, warum nicht?

Schwieriger als diese Entscheidung zu treffen, war die Frage, wie das Buch aufgebaut sein soll. Es war mir von vornherein klar, dass ich unmöglich auf alle Erweiterungsmöglichkeiten im Detail eingehen kann, dafür ist deren Anzahl in der Materialwirtschaft einfach zu groß. Auf der anderen Seite wollte ich auf keinen Fall eine unvollständige Übersicht erstellen, es reichen schon die Lücken, die selbst die offizielle Dokumentation in Teilen aufweist. Entstanden ist daher eine, so hoffe ich, gelungene Mischung: In den Hauptkapiteln dieses Buches werden einzelne ausgewählte Erweiterungen ausführlich erläutert – insbesondere die komplexeren und nicht ganz einfach zu verwendenden Erweiterungen stehen dabei im Vordergrund –, während Sie im Anhang eine strukturierte und vollständige Übersicht über alle User-Exits und BAdIs finden.

Sie halten nun bereits die zweite Auflage dieses Buches in der Hand. Neben vielen kleinen Verbesserungen habe ich vor allem ein neues Kapitel zu Erweiterungen im Materialstamm hinzugefügt.

Zu verdanken ist dieses Buch auch allen, die mich bei der Umsetzung dieses Projektes unterstützt haben, insbesondere seien hier Kristina Noe, Philipp Grimm und Timo Ellenberger genannt. Auch Stefan Proksch und Janina Schweitzer aus dem Lektorat SAP PRESS danke ich ganz herzlich für die hervorragende Unterstützung bei der Umsetzung dieses Projektes.

Vorwort

Ganz besonderer Dank geht an meine Familie, die mir den notwendigen Freiraum gegeben hat, um neben der normalen Arbeitszeit dieses Buch zu schreiben.

Jürgen Schwaninger

Senior Consultant Logistics & ABAP

conlutio UG

Einleitung

Die Komponente – früher auch: das Modul – für Materialwirtschaft gehört mit Sicherheit zu den größten ihrer Art in SAP ERP, gerade daher sind die Einstellungsmöglichkeiten im Customizing sehr umfassend. Doch auch die Prozesse bei Kunden sind in diesem Bereich hoch komplex und variantenreich, sodass man früher oder später an die Grenzen des Customizings stößt.

Zielsetzung

Um den geforderten Prozessen dennoch gerecht zu werden, bietet SAP eine große Anzahl an User-Exits und BAdIs in der Materialwirtschaft an, die es ermöglichen, höchst individuell Anforderungen und Prozesse zu realisieren. Dieses Buch zeigt Ihnen die vorhandenen Möglichkeiten und erklärt Ihnen zu einer Auswahl von Erweiterungen das genaue Vorgehen, sodass auch Sie diese Technik zur Optimierung Ihrer Prozesse einsetzen können.

Sie lernen zunächst allgemein den Umgang mit Erweiterungen, BAdIs und Erweiterungsslots kennen, sodass Sie die Beispiele aus diesem Buch problemlos nachvollziehen können. Zu ausgewählten Erweiterungen wird ihre Nutzung und Ausprogrammierung in ABAP anhand von Schritt-für-Schritt-Anleitungen erklärt. Alle verwendeten ABAP-Listings sind vollständig dargestellt und mit ausführlichen Kommentaren versehen, sodass Sie sie leicht verstehen und in Ihren eigenen Anwendungen einsetzen können.

Aufbau und Inhalt

Wenn Sie nur selten Erweiterungen programmieren, finden Sie gleich in **Kapitel 1** eine ausführliche Einführung in das Konzept von User-Exits, BAdIs und Erweiterungsslots. Mit kurzen Beispielen erläutere ich Ihnen, wie Sie diese Erweiterungsmöglichkeiten verwenden und aktivieren.

Es folgen Kapitel für jeden der großen Bereiche in der Materialwirtschaft, das heißt Einkauf (**Kapitel 2**), Dienstleistungsabwicklung (**Kapitel 3**), Bestandsführung (**Kapitel 4**), Bewertung und Kontierung (**Kapitel 5**), Logistik-Rechnungsprüfung (**Kapitel 6**) sowie Materialstamm (**Kapitel 7**). Die

wichtigsten und umfangreichsten Erweiterungsmöglichkeiten werden wiederum mit Beispielen erläutert, wobei die enthaltenen Beispiele dabei bewusst einfach gehalten sind, um unnötige Verwirrung zu vermeiden. Selten gibt es bei mehreren Unternehmen eine identische Problemstellung, daher vermitteln die Beispiele die grundsätzlichen Funktionen und Möglichkeiten. Mit diesem Wissen sind Sie jedoch in der Lage, Ihre individuelle Anforderung in die Erweiterung zu übertragen.

Ein Ausflug in die Validierung und Substitution von Buchhaltungsbelegen in **Kapitel 8** rundet das Buch ab. Bei der Buchung von Warenbewegungen und Eingangsrechnungen in MM werden als Folgebelege auch automatisch Buchhaltungsbelege in SAP ERP Financials erzeugt. Häufig besteht der Wunsch, diesen FI-Beleg mit zusätzlichen Daten aus der Materialwirtschaft anzureichern oder zusätzliche Prüfungen aus Buchhaltungssicht zu verwenden. Interessant ist diese Technik auch als Ersatz für eventuell vorhandene eigene Prüfungen in User-Exits oder BADIs, da Sie so ein zentrales Regelwerk an einer Stelle aufbauen können, ungeachtet dessen, ob ein Beleg aus der Bestandsführung, der Logistik-Rechnungsprüfung oder dem FI-System selbst kommt.

In **Anhang A** finden Sie schließlich eine Übersicht über die User-Exits und BADIs in der SAP-Materialwirtschaft. Auch der Anhang ist dabei in die genannten Bereiche unterteilt. Gibt es in einem Bereich eine große Anzahl von Erweiterungen, habe ich diesen Bereich weiter strukturiert, sodass Sie schnell alle vorhandenen Erweiterungen zu einer bestimmten Transaktion oder zu einem bestimmten Vorgang finden können.

Zielgruppe

Dieses Buch richtet sich vor allem an MM-Berater, die nur über grundlegende ABAP-Kenntnisse verfügen, aber auch an erfahrene ABAP-Programmierer, die keine oder nur wenige Kenntnisse in der Materialwirtschaft besitzen. Doch auch wenn Sie ein erfahrener MM-Berater und Programmierer sind, können Sie dieses Buch als Nachschlagewerk verwenden und vielleicht sogar noch den einen oder anderen Trick kennenlernen.

Zu welcher Gruppe Sie auch gehören, ich habe versucht, sowohl auf ABAP-Seite als auch aufseiten der Logistik nicht zu oberflächlich zu sein, gleichzeitig aber auch nicht zu sehr in die Grundlagen abzugleiten.

Voraussetzungen

Auch wenn sich dieses Buch unter anderem an Berater mit wenig Übung im Programmieren richtet, sollten Sie grundlegende ABAP-Kenntnisse auf dem Niveau der SAP-Schulung BC400 besitzen. Auf den Einsatz von ABAP Objects wird, wenn es nicht unbedingt erforderlich ist, verzichtet, dennoch gibt es einzelne BADs, die einen objektorientierten Ansatz haben, der in der Ausprogrammierung beibehalten werden muss.


Sie benötigen dennoch kein tief gehendes Wissen in objektorientierter Programmierung, die Kenntnis der wesentlichen Grundbegriffe der Objektorientierung kann in diesen Fällen jedoch nicht schaden. Auf einige Spezialitäten der Objektorientierung, wie zum Beispiel die Interfaces oder die Durchführung eines Upcasts, gehe ich in diesem Buch aber kurz ein.


Die Beispiele lassen sich prinzipiell auf jedem R/3-System ab Release 4.6C oder einem ECC-System ab Release 5.0 nachvollziehen. Manche Erweiterungen wurden erst zu einem späteren Release eingeführt und stehen somit nur auf neueren SAP-Systemen zur Verfügung. Welche Voraussetzung eine bestimmte Erweiterung hat, können Sie Anhang A entnehmen.


Hinweise zur Lektüre

In diesem Buch finden Sie mehrere Orientierungshilfen, die Ihnen die Arbeit erleichtern sollen.

In hervorgehobenen Informationskästen sind Inhalte zu finden, die wissenschaftlich wertvoll und hilfreich sind, aber etwas außerhalb der eigentlichen Erläuterung stehen. Damit Sie die Informationen in den Kästen sofort einordnen können, haben wir die Kästen mit Symbolen gekennzeichnet:

Die mit diesem Symbol gekennzeichneten *Tipps* geben Ihnen spezielle Empfehlungen, die Ihnen die Arbeit erleichtern können. 

In Kästen, die mit diesem Symbol gekennzeichnet sind, finden Sie Informationen zu *weiterführenden Themen* oder wichtigen Inhalten, die Sie sich merken sollten. 

Dieses Symbol weist Sie auf *Besonderheiten*, die Sie beachten sollten, hin. Es warnt Sie außerdem vor häufig begangenen Fehlern oder Problemen, die auftreten können. 

6 User-Exits und BAdIs in der Logistik-Rechnungsprüfung

Auch in der Logistik-Rechnungsprüfung, also der Erfassung von Eingangsberechnungen in der SAP-Materialwirtschaft, existiert ein großes Repertoire an User-Exits und BAdIs. Dennoch war es lange Zeit nicht möglich, in Transaktion MIRO eigene Felder an einen Rechnungsbeleg anzubinden, ohne das System zu modifizieren. Mit der technischen Basis SAP ECC 6.0 hat sich dies geändert, endlich existiert eine saubere Lösung zur Einbindung eines eigenen Screens in Form eines BAdIs. Doch auch dieses hat seine Tücken, daher wird die Nutzung dieses BAdIs einen großen Teil dieses Kapitels beanspruchen. In Abschnitt 6.1 wird sowohl die Möglichkeit der Modifikation als auch die Nutzung des BAdIs beschrieben.

In Abschnitt 6.2 finden Sie zudem eine Erweiterung, mit der Sie die Einstellungen der Toleranzprüfung im Customizing dynamisch überschreiben können.

6.1 Kundeneigene Felder in Transaktion MIRO

Mit dem BAdI `MRM_ITEM_CUSTFIELDS` existiert ab Release SAP ECC 6.0 eine Möglichkeit, einen eigenen Screen auf Positionsebene in den Rechnungsbeleg zu integrieren. Wie Sie dieses BAdI einsetzen und wie die Kommunikation mit Ihrem Screen funktioniert, werden Sie im Folgenden Schritt für Schritt erfahren.

Eigene Felder per Modifikation

Wenn Sie ein älteres Release als SAP ECC 6.0 verwenden, können Sie eigene Felder auch per Modifikation realisieren. Das Vorgehen hierzu ist in den SAP-Hinweisen 352701 (Belegposition) und 174413 (Sachkontenreiter) beschrieben. In diesen Hinweisen wird erklärt, wie man ein Feld durch die Modifikation der Standard-Dynpros und Dictionary-Strukturen hinzufügen kann.

Für Felder auf dem Sachkontenreiter müssen Sie zusätzlich beachten, dass das Feld mit identischem Namen als Append-Struktur an die Tabelle `BSEG` angehängt werden muss, da es anderenfalls nicht auf der Datenbank gesichert werden kann.

[+]

Auch muss das Feld in einige Steuerungstabellen aufgenommen werden. Dies erreichen Sie über Transaktion OXK3. Wählen Sie hier zunächst das Menü KONTIERUNGSFELDER • EXPERTEN-MODUS. Anschließend klappen Sie den Bereich KUNDENDEF. KONTIERUNGEN • KUNDEN INCLUDE STRUKTUR • CI_COBL auf. Setzen Sie nun den Cursor auf das betreffende Feld, und wählen Sie den Menüpunkt KONTIERUNGSFELD • ST.EINTRÄGE AUFNEHM. • TESTLAUF BZW. ECHTLAUF.

6.1.1 Überblick über die Lösung per BAdI

Um eigene Felder im Rechnungsbeleg (siehe Abbildung 6.1) abzubilden, benötigen Sie wieder mehrere Elemente. Für einen besseren Überblick finden Sie hier zunächst eine grobe Betrachtung der Abläufe. Im folgenden Abschnitt werden aber alle Schritte noch einmal im Detail besprochen.

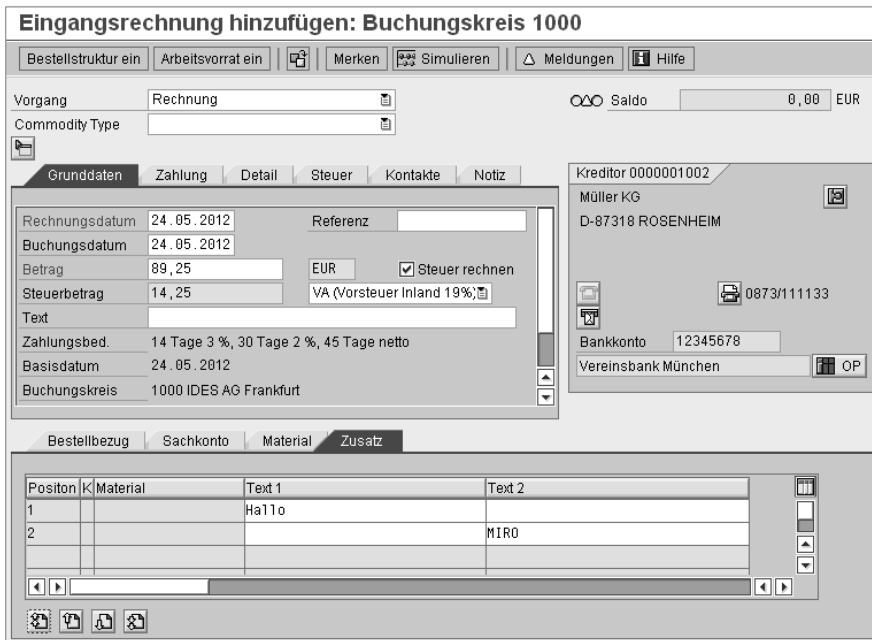


Abbildung 6.1 Eigene Felder in der Rechnungserfassung

1. Zunächst einmal müssen Sie das eigentliche Dynpro definieren, das als Subscreen im Rechnungsbeleg erscheinen soll. Hierbei ist zu beachten, dass dieser Subscreen alle vorhandenen Belegpositionen abbilden muss, was bedeutet, dass Sie hier in einem eigenen Dynpro in einem Z-Programm ein Table Control mitsamt der erforderlichen Ablauflogik erstellen müssen.
2. Als Nächstes muss der Subscreen dem Rechnungsbeleg bekannt gemacht werden. In der BAdI-Implementierung müssen Sie daher den Namen des

zuvor erstellten Programms und die Dynpro-Nummer eintragen. Außerdem müssen Sie dem Karteireiter einen Namen geben, dies erfolgt in der BAdI-Methode `TABPAGE_LABEL_SET`.

Wenn Sie die BAdI-Implementierung nun aktivieren würden, wären diese beiden Schritte bereits ausreichend, um den Subscreen auf dem Bildschirm erscheinen zu lassen, wenn zum Beispiel Transaktion MIRO gestartet wird. Darüber hinaus hätten die neuen Eingabefelder selbstverständlich noch keine weitere Funktion.

- Die Felder, die aus Ihrem Subscreen in den Rechnungsbeleg übernommen werden sollen, müssen in der Dictionary-Struktur `CI_DRSEG_CUST` angelegt werden. Diese Struktur ist bereits Bestandteil der Struktur `DRSEG_CI`, die ihrerseits der Struktur `DRSEG` zugeordnet ist. `DRSEG` ist die zentrale Struktur, die alle Positionsfelder eines Rechnungsbelegs abbildet.

Um die Felder auch in der Datenbank zu speichern, müssen die Felder mit gleichem Namen als `APPEND` an die entsprechende Datenbanktabelle angehängt werden.

- Nun wird es Zeit, die BAdI-Methoden mit Leben zu füllen. Die Hauptaufgabe der Methoden ist es, die Kommunikation zwischen der Positionsstruktur `DRSEG` und Ihrem Dynpro herzustellen. Hierzu werden die in Tabelle 6.1 genannten Methoden in der aufgeführten Reihenfolge automatisch aufgerufen.

Schritt	Methode/Zeitpunkt	Beschreibung
1	<code>CUSTOMDATA_MODIFY</code>	Dies ist die erste aufgerufene Methode des BAdIs. Diese wird durchlaufen, sobald der eigene Karteireiter ausgewählt wurde. Sie können diese Methode verwenden, um die eigenen Felder mit Daten vorzubelegen.
2	<code>INVOICE_DATA_TRANSFER</code>	Als Nächstes wird diese Methode durchlaufen. Sie stellen hier die Daten, die auf dem Dynpro angezeigt werden sollen, in den Attributen der Klasse bereit.
3	Anzeige des Dynpros	Das Dynpro wird jetzt angezeigt und kann vom Anwender gepflegt werden.
4	<code>CUSTOM_DATA_GET</code>	Sobald der Anwender eine Aktion auslöst, wird diese Methode aufgerufen. Sie müssen hier die vom Dynpro bereitgestellten, eventuell geänderten Daten zurückholen und in das Datenmodell zurückschreiben.

Tabelle 6.1 BAdI-Aufruf in einem Dialogschritt

5. Zuletzt müssen Sie noch die Ablauflogik Ihres Dynpros anpassen, um die zuvor bereitgestellten Daten in das Table Control zu übertragen oder zurückzugeben. Hierzu rufen Sie aus der Ablauflogik heraus die zwei verbleibenden Methoden des BAdIs auf. Diese Methoden werden nicht automatisch an einer vorgegebenen Stelle gestartet, sondern dienen nur zum Datenaustausch mit Ihrem Dynpro. Das genaue Vorgehen hierzu finden Sie in Tabelle 6.2.

Zeitpunkt	Beschreibung
PROCESS BEFORE OUTPUT (PBO)	Sobald dieser Teil der Ablauflogik durchlaufen wird, müssen Sie die darzustellenden Daten abholen. Hierzu rufen Sie aus dem PBO die Methode <code>INVOICE_DATA_GET</code> auf. Diese Methode liefert Ihnen die Daten, die Sie zuvor per <code>INVOICE_DATA_TRANSFER</code> bereitgestellt haben.
PROCESS AFTER INPUT (PAI)	Nachdem der Anwender einen Dialogschritt ausgelöst hat, prüfen Sie, ob sich eine Änderung ergeben hat. Die neuen Daten schreiben Sie durch einen Aufruf der Methode <code>CUSTOM_DATA_TRANSFER</code> in das BAdI zurück (um die Daten später in <code>CUSTOM_DATA_GET</code> weiterzuverarbeiten).

Tabelle 6.2 Übersicht über Ablauflogik Ihres Dynpros

Sobald Sie diese Schritte ausgeführt haben, können Sie Ihre eigenen Felder verwenden und zusammen mit dem Beleg speichern. In den folgenden Abschnitten schauen wir uns dieses Vorgehen noch einmal im Detail an.

6.1.2 BAdI im Detail – Anpassungen im ABAP Dictionary

Im folgenden Beispiel sollen zwei Felder, `ZZTEXT_1` und `ZZTEXT_2`, je Position erfasst und gesichert werden können. Felder im Kundennamensraum sollten immer mit einem Doppel-Z beginnen. Beide Felder sollen das Datenelement `CHAR32` verwenden, das in jedem System vorhanden ist.

Wie bereits erläutert, müssen diese Felder zunächst in die Arbeitsstruktur `DRSEG` aufgenommen werden, die für alle Positionen im Rechnungsbeleg verwendet wird. Die Struktur `DRSEG` enthält bereits ein Include `DRSEG_CI`, das drei feststehende Felder sowie die Include-Struktur `CI_DRSEG_CUST` beinhaltet. In diese Struktur können Sie Ihre eigenen Felder aufnehmen.

1. Gehen Sie in das ABAP Dictionary (Transaktion SE11), und bearbeiten Sie die Struktur `CI_DRSEG_CUST` (Auswahl `DATENTYP`). Normalerweise ist diese

Struktur noch nicht vorhanden, und Sie müssen sie anlegen. Wechseln Sie in den Bearbeitungsmodus.

2. Tragen Sie die beiden Felder ZZTEXT_1 und ZZTEXT_2 ein, und verwenden Sie jeweils das Datenelement CHAR32.
3. Über das Menü ZUSÄTZE • ERWEITERUNGSKATEGORIE können Sie die Kategorie NICHT ERWEITERBAR setzen. Tun Sie dies nicht, ist dies nicht weiter schlimm, Sie werden jedoch bei der Aktivierung eine Warnmeldung erhalten.
4. Sichern und aktivieren Sie Ihre Struktur. Die Aktivierung kann etwas Zeit in Anspruch nehmen, da alle Objekte, die die Struktur DRSEG verwenden, ebenfalls neu generiert werden.
5. Betrachten Sie nun die Struktur DRSEG_CI. Diese sollte die Felder wie in Tabelle 6.3 enthalten. Wie Sie sehen, enthält die Struktur neben Ihren eigenen Feldern auch die Positionsnummer, zu der Ihre Felder gehören. Damit beschreibt diese Struktur genau eine Zeile des Belegs aus Sicht Ihrer eigenen Felder. Sie werden diese Struktur im weiteren Verlauf daher häufiger verwenden.

Feldname	Beschreibung
C_RBLGP	Positionsnummer des Belegs
C_KOART	Kontoart: <ul style="list-style-type: none"> ▶ A – Anlagen ▶ D – Debitoren ▶ K – Kreditoren ▶ M – Material ▶ S – Sachkonten
C_MATNR	Materialnummer bei Kontoart M
ZZTEXT_1	Ihr erstes eigenes Feld
ZZTEXT_2	Ihr zweites eigenes Feld

Tabelle 6.3 Struktur DRSEG_CI

Damit sind alle Voraussetzungen erfüllt, um später mit Ihren Feldern innerhalb des BAdIs und in Ihrem Dynpro zu arbeiten. Damit die Daten jedoch auch in der Datenbank gesichert werden, müssen Sie die Felder zusätzlich per APPEND an die entsprechende Tabelle anhängen. In Tabelle 6.4 sehen Sie alle Möglichkeiten, die Ihnen hierfür zur Verfügung stehen.

Tabelle/Struktur	Beschreibung
RSEG	Datenbanktabelle zur Speicherung der Belegpositionen
RBDRSEG	Datenbanktabelle zur temporären Speicherung der Belegpositionen in der Hintergrundprüfung
RBMA	Datenbanktabelle zur Speicherung von materialbezogenen Feldern
COBL_MRM	Struktur der Kontierungsfelder (Sachkontenreiter). Felder, die Sie dieser Struktur hinzufügen, werden in der Datenbanktabelle RBCO gesichert.

Tabelle 6.4 Tabellen zur Speicherung der eigenen Felder

Da die Felder ZZTEXT_1 und ZZTEXT_2 keine Kontierungsinformationen darstellen und nicht materialbezogen sind, sollen sie in Tabelle RSEG gesichert werden.

1. Lassen Sie sich im Dictionary (Transaktion SE11) die Datenbanktabelle RSEG anzeigen. Klicken Sie dann oben rechts auf die Schaltfläche APPEND-STRUKTUR.
2. Im erscheinenden Auswahlfenster klicken Sie auf das Icon APPEND ANLEGEN. Als Namen für die Append-Struktur können Sie zum Beispiel ZZRSEG verwenden.
3. Geben Sie nun wieder die beiden Felder ZZTEXT_1 und ZZTEXT_2 an. Achten Sie unbedingt auf die absolut identische Schreibweise mit den Feldnamen in der Struktur CI_DRSEG_CUST. Verwenden Sie außerdem unbedingt dasselbe Datenelement (CHAR32) für diese Felder (siehe Abbildung 6.2).

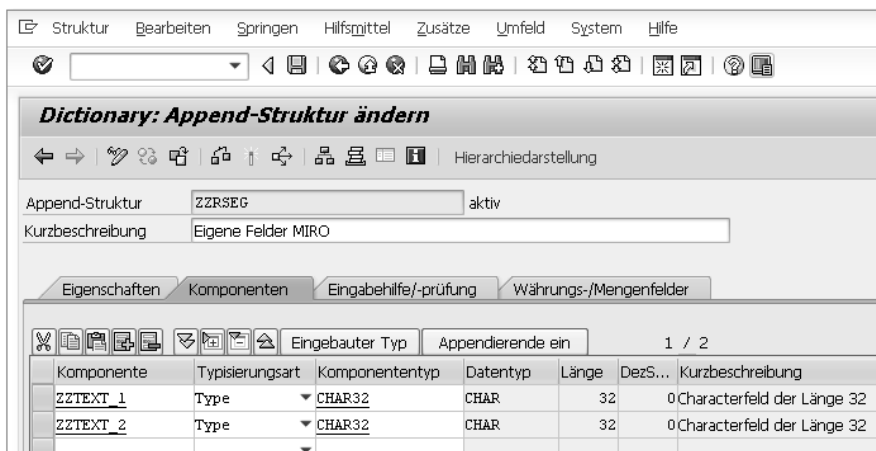


Abbildung 6.2 Append-Struktur ZZRSEG

4. Pflegen Sie auch hier wieder die Erweiterungskategorie (Menü ZUSÄTZE • ERWEITERUNGSKATEGORIE) als NICHT ERWEITERBAR.
5. Sichern und aktivieren Sie die Append-Struktur.

Wenn Sie die Hintergrundprüfung verwenden und über Transaktion MIRA Belege erfassen, die Sie anschließend über Transaktion MIR6 oder MIR4 weiterbearbeiten und dort zur Hintergrundprüfung einplanen, werden die Positionsdaten temporär in der Tabelle RBDRSEG gespeichert. Erst wenn die Hintergrundprüfung (Transaktion MRBP) ohne Fehler durchlaufen wurde, werden die Sätze wieder aus Tabelle RBDRSEG gelöscht und in Tabelle RSEG gesichert.

Damit auch in diesem Verfahren zu jeder Zeit die eigenen Felder erfasst werden können, sollten Sie auch die Tabelle RBDRSEG identisch mit Tabelle RSEG um die eigenen Felder erweitern.

6.1.3 Eigenes Dynpro mit Table Control erstellen

Als Nächstes benötigen Sie natürlich auch das Dynpro, das als Subscreen in den Karteireiter eingebunden werden soll. Diesen packen Sie am besten in ein eigenes Programm in Form einer Funktionsgruppe oder in einen Report vom Typ MODULPOOL.

1. Starten Sie Transaktion SE80, und legen Sie ein neues Programm an. Für das Beispiel wurde der Name SAPMZMRM und als Programmtyp ein MODULPOOL verwendet (siehe Abbildung 6.3).

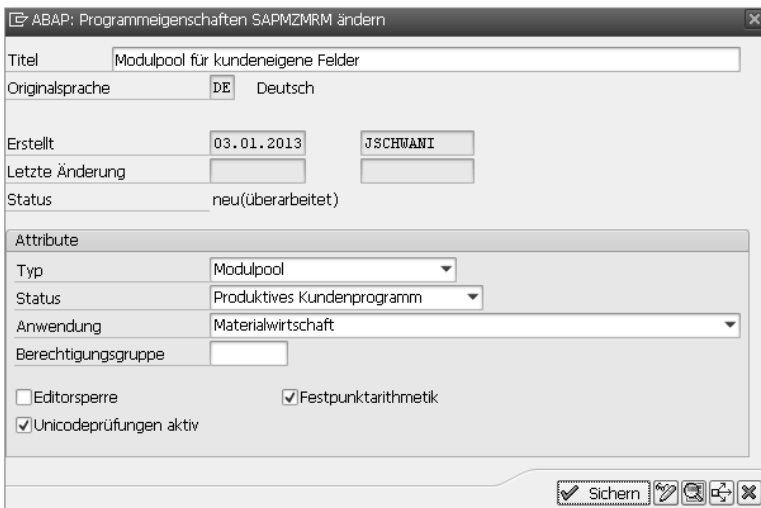


Abbildung 6.3 Modulpool zur Aufnahme des Dynpros


2. Bevor Sie das Dynpro mit dem Table Control erzeugen, sollten Sie überlegen, welche Felder im Table Control dargestellt werden sollen. Sie benötigen eine interne Tabelle und eine Arbeitsstruktur für diese Felder, um das Table Control daraus mit Daten zu versorgen.

Sie können einfach die Struktur `DRSEG_CI` verwenden, diese enthält neben Ihren eigenen Feldern auch die Positionsnummer und eignet sich daher gut für eine Anzeige auf dem Bildschirm. Erstellen Sie deshalb im Programm `SAPMZMRM` zwei globale Datenobjekte wie in Listing 6.1.

```
PROGRAM   sapmzmrn.

* Globale Daten für Table Control
DATA: gs_drseg_ci TYPE drseg_ci,
      gt_drseg_ci TYPE TABLE OF drseg_ci.
```

Listing 6.1 Globale Daten für Table Control

3. Aktivieren Sie zunächst Ihr Programm. Klicken Sie dann in Transaktion `SE80` links im Repository Browser mit der rechten Maustaste auf den Programmnamen. Im erscheinenden Kontextmenü wählen Sie `ANLEGEN • DYNPRO`. Als Dynpro-Nummer geben Sie die `0100` an.
4. Vergeben Sie eine kurze Beschreibung für dieses Dynpro, und wählen Sie auf dem Karteireiter `EIGENSCHAFTEN` im Abschnitt `DYNPROTYP` unbedingt die Option `SUBSCREEN`, da andere Dynpro-Typen nicht eingebunden werden können und Sie beim Starten von Transaktion `MIRO` einen Kurzdump erhalten würden.
5. Wechseln Sie dann über das Icon `LAYOUT` in den Screen Painter. Hier erzeugen Sie als Erstes ein Table Control. Hierfür gibt es einen nützlichen Wizard, der Ihnen viel Arbeit abnimmt. Klicken Sie hierzu auf der linken Seite auf das Icon , `TABLE CONTROL (MITTELS WIZARD)`, und ziehen Sie anschließend einen Kasten. Beginnen Sie am besten ganz links oben, und zeichnen Sie Ihr Table Control ungefähr 100 Zeichen breit und 18 Zeilen hoch. Sie sehen rechts oben während der Erstellung einen Zähler mit Breite und Höhe des Table Controls (siehe Abbildung 6.4).
6. Sobald Sie das Table Control gezeichnet haben, erscheint nun der Wizard, wie in Abbildung 6.5 zu sehen ist. Überspringen Sie den Einführungsbildschirm durch einen Klick auf die Schaltfläche `WEITER`.

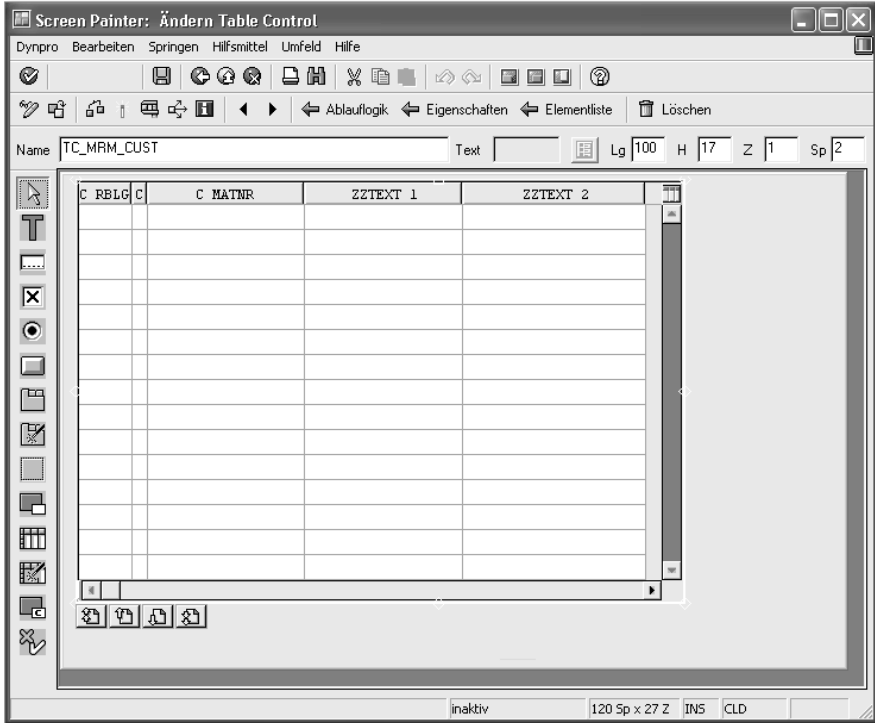


Abbildung 6.4 Fertiges Table Control im Screen Painter

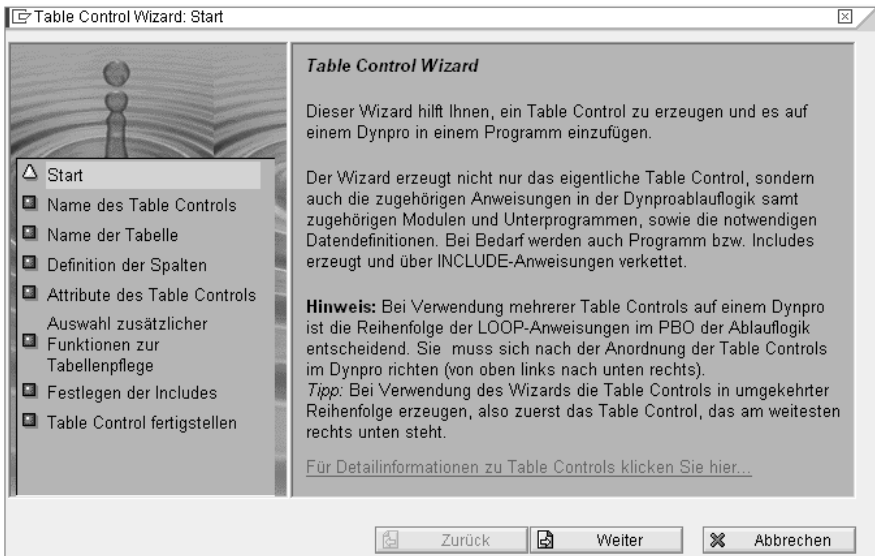


Abbildung 6.5 Table Control Wizard

7. Im Abschnitt NAME DES TABLE CONTROLS geben Sie einen eindeutigen Namen an, im Beispiel wurde TC_MRM_CUST verwendet. Klicken Sie auf WEITER.
8. In NAME DER TABELLE wählen Sie als Typ die INTERNE PROGRAMMTABELLE und tragen dort die Tabelle GT_DRSEG_CI ein. Da diese interne Tabelle keine Kopfzeile besitzt, müssen Sie auch die Option TABELLEN-ARBEITSBEREICH aktivieren und die Struktur GS_DRSEG_CI eintragen. Klicken Sie auf WEITER.
9. Im Schritt DEFINITION DER SPALTEN markieren Sie einfach alle angebotenen Spalten (die aus der Struktur DRSEG_CI übernommen wurden). Klicken Sie auf WEITER.
10. Im Schritt ATTRIBUTE DES TABLE CONTROLS markieren Sie im Abschnitt EIN-/AUSGABE-ATTRIBUTE die Option EINGABE, damit sind die Felder im Standardzustand eingabebereit. Alle weiteren Optionen lassen Sie unverändert. Klicken Sie auf WEITER.
11. Im Schritt AUSWAHL ZUSÄTZLICHER FUNKTIONEN ZUR TABELLENPFLEGE können Sie noch die Option BLÄTTERN aktivieren. Hierdurch werden unter das Table Control vier Schaltflächen gesetzt, mit denen der Anwender schnell durch die Zeilen des Controls blättern kann. Klicken Sie auf WEITER.
12. Im Schritt FESTLEGEN DER INCLUDES werden Ihnen Bezeichnungen für verschiedene Includes vorgeschlagen, in denen die notwendigen Daten-deklarationen und Ablaufmodule erstellt werden sollen. Übernehmen Sie den Vorschlag unverändert, und klicken Sie auf WEITER.
13. Im letzten Schritt müssen Sie nur noch auf FERTIGSTELLEN klicken, und alle für das Table Control notwendigen Objekte und Module werden erzeugt.
14. Sichern Sie Ihr Dynpro, und verlassen Sie den Screen Painter. Anschließend sollten Sie das Dynpro aktivieren, markieren Sie dabei auch alle anderen erzeugten Objekte, die Ihnen vorgeschlagen werden, und aktivieren Sie diese ebenfalls.

Damit steht das Grundgerüst Ihres Dynpros, noch fehlt aber die Kommunikation mit dem BAdI, um das Table Control mit Daten zu füllen und geänderte Werte zurückzuschreiben. Doch bevor Sie diesen Teil angehen, sollten Sie sich zunächst einmal dem BAdI widmen.

6.1.4 Vorbereitung der Daten im BAdI

Nachdem Sie diese Vorbereitungen getroffen haben, können Sie sich jetzt den ersten Methoden des BAdIs zuwenden. Betrachten Sie zunächst noch einmal Tabelle 6.1 und Tabelle 6.2. Bei der Bearbeitung von Eingangsrech-

nungen werden vom System die BAdI-Methoden der Tabelle 6.1 aufgerufen, hier müssen Sie die Daten gegebenenfalls aufbereiten und im BAdI zwischenspeichern. Dies geschieht, indem Sie die Daten in Attributen der zugehörigen Klasse ablegen. Wenn Sie die Daten dann später (siehe Tabelle 6.2) in der Ablauflogik Ihres Dynpros aus dem BAdI abholen, werden letztendlich die Daten aus diesen Attributen verwendet.

1. Legen Sie zunächst in Transaktion SE19 eine Implementierung zum BAdI `MRM_ITEM_CUSTFIELDS` an. Als Name für die Implementierung können Sie zum Beispiel `ZMRM_ITEM_CUSTFIELDS` verwenden. Sichern Sie die Implementierung.
2. Als Erstes sollten Sie Ihr zuvor erstelltes Dynpro in der Implementierung hinterlegen. Wechseln Sie hierzu auf den Karteireiter `SUBSCREENS`, und tragen Sie den Namen und die Dynpro-Nummer als gerufenes Programm ein (siehe Abbildung 6.6).

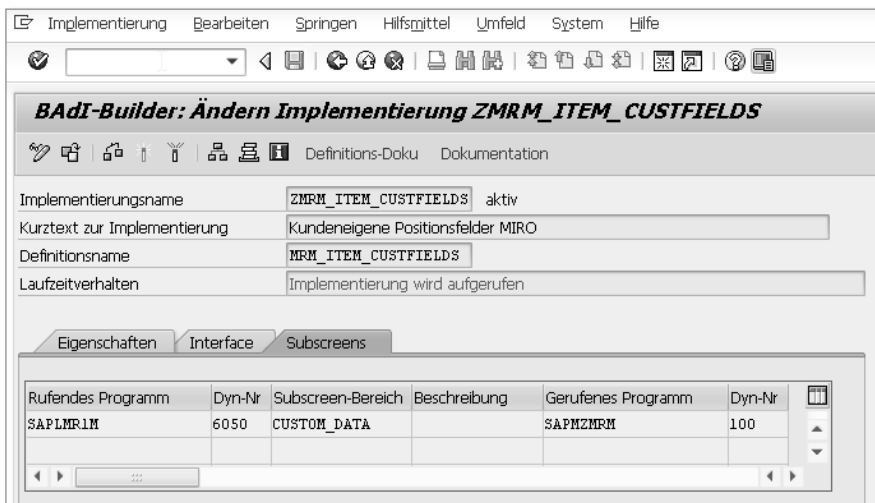


Abbildung 6.6 Karteireiter »Subscreens« der BAdI-Implementierung

3. Wechseln Sie nun auf den Karteireiter `INTERFACE`, und navigieren Sie durch einen Doppelklick in die implementierende Klasse (`ZCL_IM_MRM_ITEM_CUSTFIELDS`).
4. In der Klasse sollten Sie nun zunächst einmal einen Blick auf den Karteireiter `ATTRIBUTE` werfen. Sie werden hier sechs Attribute finden (siehe Tabelle 6.5), die automatisch angelegt wurden. Diese Attribute wurden bereits im Interface `IF_EX_MRM_ITEM_CUSTFIELDS` definiert, von dem das BAdI abgeleitet wurde.

Attribut	Beschreibung
IF_EX_MRM_ITEM_CUSTFIELDS~ TRANSACTION_TYPE	Dieses Feld enthält den Transaktionstyp. Die wichtigsten möglichen Werte sind folgende: A – Anzeigen H – Hinzufügen (neuer Beleg) V – Verändern
IF_EX_MRM_ITEM_CUSTFIELDS~ S_RBKPV	Diese Struktur enthält alle Daten des Belegkopfes.
IF_EX_MRM_ITEM_CUSTFIELDS~ TAB_DRSEG	In dieser internen Tabelle befinden sich alle Daten der Belegpositionen, einschließlich Ihrer eigenen Felder.
IF_EX_MRM_ITEM_CUSTFIELDS~ TAB_DRSEG_CUSTOM	Diese interne Tabelle enthält nur Ihre eigenen Felder (Typ DRSEG_CI, siehe Tabelle 6.3).
IF_EX_MRM_ITEM_CUSTFIELDS~ H_SORT	Wenn die Reihenfolge der Datensätze in Ihrem Dynpro verändert (sortiert) wurde, können Sie dieses Kennzeichen setzen, um die Sortierung beizubehalten.
IF_EX_MRM_ITEM_CUSTFIELDS~ H_CHANGE	Kennzeichen: Die eigenen Daten wurden durch den Anwender verändert.

Tabelle 6.5 Attribute aus Interface IF_EX_MRM_ITEM_CUSTFIELDS

[+] Automatische Erzeugung der Attribute

Eine wesentliche Grundeigenschaft von Interfaces ist folgende: Wenn eine Klasse (und nichts anderes versteckt sich hinter der Implementierung eines BADs) ein Interface implementiert, muss die Klasse alle im Interface definierten Methoden und Attribute übernehmen. Aus diesem Grund wurden die Attribute an dieser Stelle automatisch erzeugt. Für Sie bedeutet dies, dass Sie an dieser Stelle nichts weiter tun müssen. Alle Attribute, die Sie im weiteren Verlauf benötigen, sind schon da!

5. Wechseln Sie nun wieder auf den Karteireiter **METHODEN**. Zunächst einmal sollen die Methoden implementiert werden, die automatisch aufgerufen werden (siehe Tabelle 6.1). Die Methode `CUSTOMDATA_MODIFY` können Sie verwenden, um die eigenen Felder vorzubelegen, für die grundlegende Funktion wird diese jedoch nicht benötigt.

Stattdessen widmen Sie sich der Methode `INVOICE_DATA_TRANSFER`. Diese Methode besitzt vier Eingangsparameter (Transaktionstyp, Rechnungsbekopf, Rechnungsbelegpositionen und Ihre eigenen Felder). Sie müssen in dieser Methode nichts weiter tun, als die Parameter, die hier übergeben

werden, in die Attribute der Klasse zu kopieren. Das Coding hierzu finden Sie in Listing 6.2.

```
METHOD if_ex_mrm_item_custfields~invoice_data_transfer.
* Übernahme der Eingangspartner in Attribute
me->if_ex_mrm_item_custfields~transaction_type =
    i_transaction_type.
me->if_ex_mrm_item_custfields~s_rbkpv =
    is_rbkpv.
me->if_ex_mrm_item_custfields~tab_drseg =
    it_drseg.
me->if_ex_mrm_item_custfields~tab_drseg_custom =
    it_drseg_custom.
ENDMETHOD.
```

Listing 6.2 Methode INVOICE_DATA_TRANSFER

6. Nach der Änderung der Daten durch den Anwender werden die Daten im Attribut TAB_DRSEG_CUSTOM den neuen Stand beinhalten (dieser Teil wird später realisiert). Auch wird das Attribut H_CHANGE ein Kennzeichen enthalten, ob die Daten geändert wurden. In der Methode CUSTOM_DATA_GET müssen Sie nun nichts weiter tun, als diese Informationen wieder an das Standardprogramm zurückzugeben. Das entsprechende Coding sehen Sie in Listing 6.3.

```
METHOD if_ex_mrm_item_custfields~custom_data_get.
* Rückgabe der geänderten Attribute an Aufrufer
et_drseg_cust =
    me->if_ex_mrm_item_custfields~tab_drseg_custom.
e_change =
    me->if_ex_mrm_item_custfields~h_change.
ENDMETHOD.
```

Listing 6.3 Methode CUSTOM_DATA_GET

7. Als Nächstes folgen die Methoden aus Tabelle 6.2. Die Methode INVOICE_DATA_GET soll aus dem Zeitpunkt PROCESS BEFORE OUTPUT der Ablauflogik des Dynpros aufgerufen werden. Die Aufgabe der Methode ist es demnach, die aktuellen Inhalte der Attribute zu den Positionsdaten an das Dynpro zurückzuliefern. Hierzu versorgen Sie einfach die Exportparameter der Methode mit den passenden Attributen Ihrer Klasse. Das entsprechende Coding finden Sie in Listing 6.4.

```
METHOD if_ex_mrm_item_custfields~invoice_data_get.
* Daten aus Attributen bereitstellen für Dynpro
e_transaction_type =
    me->if_ex_mrm_item_custfields~transaction_type.
```

```
es_rbkpv          =
  me->if_ex_mrm_item_custfields~s_rbkpv.
et_drseg         =
  me->if_ex_mrm_item_custfields~tab_drseg.
et_drseg_cust    =
  me->if_ex_mrm_item_custfields~tab_drseg_custom.
ENDMETHOD.
```

Listing 6.4 Methode INVOICE_DATA_GET

8. Die Methode `CUSTOM_DATA_TRANSFER` wird aus dem Zeitpunkt `PROCESS AFTER INPUT` der Ablauflogik des Dynpros aufgerufen. Das Dynpro liefert an dieser Stelle die eventuell geänderten eigenen Felder sowie ein Kennzeichen zurück, ob eine Datenänderung erfolgt ist. Diese Daten legen Sie in den Attributen ab. Damit kann die zuvor implementierte Methode `CUSTOM_DATA_GET` genau diese Daten weiterverarbeiten (siehe Listing 6.5).

```
METHOD if_ex_mrm_item_custfields~custom_data_transfer.
  me->if_ex_mrm_item_custfields~tab_drseg_custom =
    it_drseg_cust.
  me->if_ex_mrm_item_custfields~h_change =
    i_change.
ENDMETHOD.
```

Listing 6.5 Methode CUSTOM_DATA_TRANSFER

9. Zu guter Letzt benötigen Sie noch die Methode `TABPAGE_LABEL_SET`, um dem Karteireiter eine Bezeichnung zu geben. Verwenden Sie ein Textsymbol, um die Bezeichnung übersetzbar zu halten. Auf dem Karteireiter werden maximal elf Zeichen dargestellt (siehe Listing 6.6).

```
METHOD if_ex_mrm_item_custfields~tabpage_label_set.
* Bezeichnung des Karteireiters, max. elf Zeichen
* text-001 enthält den Text 'Zusatz'
  e_screen_name = text-001.
ENDMETHOD.
```

Listing 6.6 Methode TABPAGE_LABEL_SET

6.1.5 Zurück im Dynpro

Nachdem Sie nun auch im BAdI alle Vorbereitungen getroffen haben, müssen Sie noch die Datenkommunikation zwischen BAdI und Dynpro einrichten. Wie zuvor erwähnt, läuft die Kommunikation zwischen Dynpro und BAdI über die Methoden `INVOICE_DATA_GET` und `CUSTOM_DATA_TRANSFER`, die wiederum auf die Attribute der Klasse zugreifen. In der objektorientierten

Programmierung ist es jedoch so, dass der Inhalt von Attributen jeweils genau an eine Instanz der Klasse gebunden ist. Wenn Sie nun daher einfach eine neue Instanz zur implementierenden Klasse `ZCL_IM_MRM_ITEM_CUST-FIELDS` erzeugen würden, um dann die genannten Methoden aufzurufen, wäre dies erfolglos, denn die neue Instanz hätte auch eigene Attribute, die schlicht und ergreifend leer wären.

Daher müssen Sie für einen Zugriff auf genau dieselbe Instanz sorgen, die auch vom BAdI verwendet wurde. Dazu existiert die statische Klasse `CL_EXITHANDLER` (siehe Abbildung 6.7) mit der Methode `GET_INSTANCE_FOR_SUBSCREENS`. Dieser Methode müssen Sie nur eine passende Referenzvariable übergeben, die sich auf das Interface des BAdIs bezieht. Die Methode erkennt dann anhand des Interface die passende Instanz und gibt diese zurück.

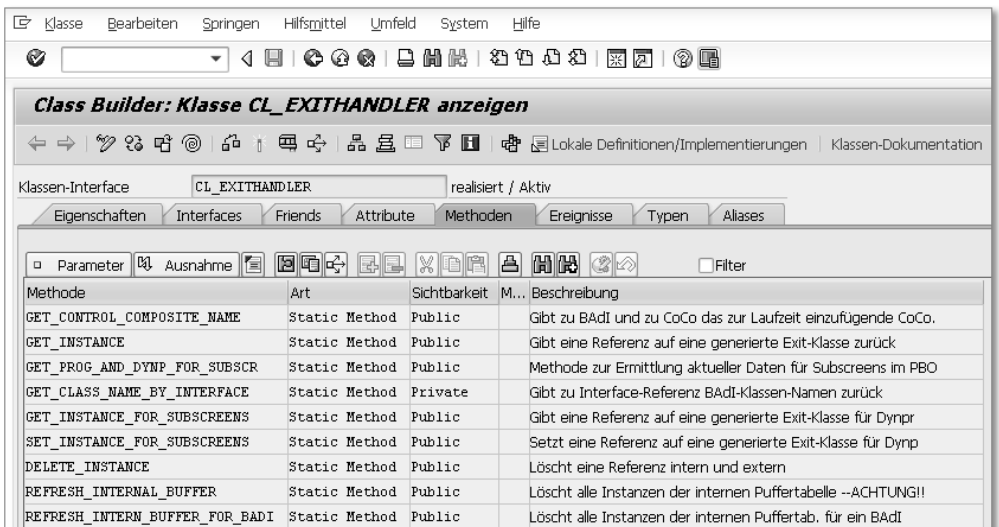


Abbildung 6.7 Methoden der Klasse `CL_EXITHANDLER`

1. Gehen Sie in Transaktion SE80 in Ihr Programm `SAPMZMRM`, und wechseln Sie in das Dynpro 0100. Fügen Sie in der Ablauflogik an erster Stelle im Zeitpunkt `PROCESS BEFORE OUTPUT` ein neues `MODULE` mit dem Namen `GET_DATA` ein. Doppelklicken Sie auf `GET_DATA`, um das Modul zu erzeugen. Übernehmen Sie das vorgeschlagene Include (`MZMRM_GET_DATA001`).
2. Erstellen Sie im Modul zunächst eine neue Variable als Referenz auf das Interface des BAdIs. Sie finden den Namen des Interface in Transaktion SE18 oder SE19 auf dem Karteireiter `INTERFACE`. In diesem Fall ist es das

Interface IF_EX_MRM_ITEM_CUSTFIELDS. Im Beispiel wurde diese Variable MRM_CUSTFIELDS genannt.

3. Rufen Sie nun die Methode GET_INSTANCE_FOR_SUBSCREENS der Klasse CL_EXITHANDLER auf, und übergeben Sie Ihre zuvor definierte Variable. Nach dem Aufruf zeigt Ihre Variable auf die Instanz des BAdIs.
4. Holen Sie nun über die Methode INVOICE_DATA_GET alle zur Verfügung stehenden Daten aus dem BAdI ab. Um die Daten aufnehmen zu können, müssen Sie noch passende Variablen in Ihrem Programm anlegen. Legen Sie diese Variablen im globalen Bereich ab, sodass Sie auch später noch Zugriff darauf haben.
5. Zuletzt müssen Sie die Daten in Ihr Table Control übertragen. Da das Table Control alle Felder der Struktur DRSEG_CI verwendet, können Sie den Inhalt der internen Tabelle ET_DRSEG_CUST, die Sie im vorangegangenen Schritt erhalten haben, einfach in die Tabelle GT_DRSEG_CI kopieren, die Ihr Table Control versorgt. Das entsprechende Coding für dieses Modul finden Sie in Listing 6.7 und Listing 6.8.

```
PROGRAM  sapmzmr.
...
* Datendeklarationen für BAdI-Kommunikation
TYPE-POOLS: mrm, mmcr.

DATA: gv_trtyp LIKE t169-trtyp,
      gs_rbkpv TYPE mrm_rbkpv,
      gt_drseg TYPE mmcr_tdrseg,
      gt_drseg_custom TYPE tdrseg_cust.
```

Listing 6.7 Datendeklarationen in Programm SAPMZMRM

```
*-----*
***INCLUDE MZMRM_GET_DATA001 .
*-----*
*&      Module GET_DATA  OUTPUT
*&-----*
module GET_DATA output.

* Referenz auf das Interface
DATA: mrm_custfields TYPE REF TO if_ex_mrm_item_custfields.
* Instanz holen
CALL METHOD cl_exithandler=>get_instance_for_subscreens
  CHANGING
    instance = mrm_custfields
  EXCEPTIONS
    OTHERS   = 6.
```

```

* Daten aus BAdI holen
CALL METHOD mrm_custfields->invoice_data_get
  IMPORTING
    e_transaction_type = gv_trtyp
    es_rbkpv           = gs_rbkpv
    et_drseg          = gt_drseg
    et_drseg_cust     = gt_drseg_custom.

* Übertrag in Table Control
gt_drseg_ci[] = gt_drseg_custom[].

endmodule.                " GET_DATA  OUTPUT

```

Listing 6.8 Modul GET_DATA

6. Wechseln Sie nun wieder zurück in die Ablauflogik Ihres Dynpros. Legen Sie ein neues Modul mit Namen WRITE_DATA als letztes Modul im Zeitpunkt PROCESS AFTER INPUT an. Klicken Sie doppelt auf den Modulnamen, um das Modul erzeugen zu lassen, und verwenden Sie wieder das vorgeschlagene Include (MZMRM_WRITE_DATAI01).
7. Sie müssen an dieser Stelle die Daten wieder zurück in das BAdI übertragen. Grundsätzlich kann keine Änderung erfolgen, wenn sich der Anwender nur im Anzeigemodus befindet. Daher müssen Sie die folgenden Prüfungen nur durchführen, wenn der Transaktionstyp, den Sie in GV_TRTYP zwischengespeichert haben, ungleich 'A' ist.
8. Anschließend machen Sie einen LOOP über die aktuellen Daten Ihres Table Controls (GT_DRSEG_CI) und vergleichen jede Zeile mit dem ursprünglichen Zustand, der im PBO in der internen Tabelle GT_DRSEG_CUSTOM gespeichert wurde. Hat sich eine Zeile geändert, merken Sie sich dies, und schreiben Sie die Änderung in die Tabelle GT_DRSEG_CUSTOM zurück.
9. Danach können Sie die Methode CUSTOM_DATA_TRANSFER anlegen. Die im PBO definierte Referenzvariable MRM_CUSTFIELDS steht noch zur Verfügung und kann hierfür verwendet werden. Geben Sie die Tabelle GT_DRSEG_CUSTOM zurück. Falls Daten verändert wurden, setzen Sie das Kennzeichen I_CHANGE in der Parameterschnittstelle der aufgerufenen Methode. Den Parameter I_SORT verwenden wir im Beispiel nicht, Sie können hier einfach ABAP_FALSE zurückgeben.

Das vollständige Coding zum Modul WRITE_DATA finden Sie in Listing 6.9.

```

*-----*
***INCLUDE MZMRM_WRITE_DATAI01 .
*-----*

```

```

*&      Module  WRITE_DATA  INPUT
*&-----
MODULE write_data INPUT.

* Lokale Deklarationen
  DATA lv_change TYPE c.
  DATA ls_drseg_cust TYPE drseg_ci.

* Nur wenn nicht im Anzeigemodus
  CHECK gv_trtyp NE 'A'.

* Wurden Daten geändert?
  LOOP AT gt_drseg_ci INTO gs_drseg_ci.
    READ TABLE gt_drseg_custom
      INTO ls_drseg_cust INDEX sy-tabix.
    IF ls_drseg_cust NE gs_drseg_ci.
*      Daten wurden geändert
      lv_change = 'X'.
      MODIFY gt_drseg_custom FROM gs_drseg_ci
        INDEX sy-tabix.
    ENDIF.
  ENDLOOP.

* Daten zurückschreiben
  CALL METHOD mrm_custfields->custom_data_transfer
    EXPORTING
      i_sort      = abap_false
      it_drseg_cust = gt_drseg_custom
      i_change     = lv_change.

ENDMODULE.                " WRITE_DATA  INPUT

```

Listing 6.9 Modul WRITE_DATA

Aktivieren Sie jetzt Ihre letzten Änderungen, und achten Sie auch darauf, dass die BAdI-Implementierung aktiv ist. Anschließend können Sie die neue Funktion in Transaktion MIRO testen, der zusätzliche Karteireiter erscheint nun, und auch die eingegebenen Daten werden dargestellt.

Es ist jedoch nicht ideal, dass alle Felder immer eingabebereit sind. Besser ist es, wenn die drei Standardfelder, zum Beispiel die Positionsnummer, grundsätzlich nicht eingabebereit sind. Außerdem sollte darauf geachtet werden, dass auch Ihre eigenen Felder deaktiviert werden, wenn eine reine Anzeigetransaktion darauf zugreift. Darüber hinaus ist Ihnen vielleicht aufgefallen, dass der Anwender auch in neue Positionen Werte eingeben kann. Da Ihre eigenen Felder jedoch positionsbezogen sind, darf die Eingabe nur möglich sein, wenn auch tatsächlich eine Position im Beleg vorhanden ist. Dies lässt

sich einfach über das Feld C_RBLGP prüfen – ist hier keine Positionsnummer vorhanden, darf auch keine Eingabe möglich sein.

1. Wechseln Sie nochmals in die Ablauflogik Ihres Dynpros. Der Table Control Wizard hat Ihnen bereits ein Modul vorgeschlagen, das Sie verwenden können, um die Attribute der Felder zu verändern (TC_MRM_CUST_CHANGE_FIELD_ATTR).
2. Entfernen Sie den Kommentar vor dieser Zeile, und klicken Sie doppelt auf den Namen, um das Modul anzulegen. Da es sich um ein weiteres PBO-Modul zum Table Control handelt, können Sie es im vorhandenen Include MZMRM001 erstellen lassen.
3. Anschließend können Sie dort die Felder ändern, wie zuvor erläutert. Da dies keine spezielle Technik des BADIs ist, sondern zur allgemeinen Programmierung von Table Controls gehört, wurde nicht detailliert darauf eingegangen. Sie finden das vollständige Coding jedoch in Listing 6.10. Die vollständige Ablauflogik zum Dynpro können Sie in Listing 6.11 überprüfen.

```
*&-----*
*&      Module  TC_MRM_CUST_CHANGE_FIELD_ATTR  OUTPUT
*&-----*
MODULE tc_mrm_cust_change_field_attr OUTPUT.
  DATA ls_col TYPE cxtab_column.

* Für alle Spalten der aktuellen Zeile
  LOOP AT tc_mrm_cust-cols INTO ls_col.

* Standardfelder: Eingabe verhindern
  IF ls_col-screen-name EQ 'GS_DRSEG_CI-C_RBLGP' OR
     ls_col-screen-name EQ 'GS_DRSEG_CI-C_KOART' OR
     ls_col-screen-name EQ 'GS_DRSEG_CI-C_MATNR'.
     ls_col-screen-input = 0.
  ENDIF.

* Im Anzeigemodus oder falls keine Positionsnummer
* zugeordnet ist - für eigene Felder Eingabe verhindern
  IF ls_col-screen-name EQ 'GS_DRSEG_CI-ZZTEXT_1' OR
     ls_col-screen-name EQ 'GS_DRSEG_CI-ZZTEXT_2'.
     IF gv_trtyp = 'A' OR
        gs_drseg_ci-c_rblgp IS INITIAL.
        ls_col-screen-input = 0.
     ELSE.
        ls_col-screen-input = 1.
     ENDIF.
  ENDIF.
```

```
        MODIFY tc_mrm_cust-cols FROM ls_col.  
    ENDLLOOP.  
  
ENDMODULE.          " TC_MRM_CUST_CHANGE_FIELD_ATTR  OUTPUT
```

Listing 6.10 Modul TC_MRM_CUST_GET_LINES OUTPUT

```
PROCESS BEFORE OUTPUT.  
    MODULE get_data.  
  
*&SPWIZARD: PBO FLOW LOGIC FOR TABLECONTROL 'TC_MRM_CUST'  
    MODULE tc_mrm_cust_change_tc_attr.  
*&SPWIZARD: MODULE TC_MRM_CUST_CHANGE_COL_ATTR.  
    LOOP AT  gt_drseg_ci  
        INTO gs_drseg_ci  
        WITH CONTROL tc_mrm_cust  
        CURSOR tc_mrm_cust-current_line.  
        MODULE tc_mrm_cust_get_lines.  
        MODULE tc_mrm_cust_change_field_attr.  
    ENDLLOOP.  
  
PROCESS AFTER INPUT.  
*&SPWIZARD: PAI FLOW LOGIC FOR TABLECONTROL 'TC_MRM_CUST'  
    LOOP AT gt_drseg_ci.  
        CHAIN.  
            FIELD gs_drseg_ci-c_rblgp.  
            FIELD gs_drseg_ci-c_koart.  
            FIELD gs_drseg_ci-c_matnr.  
            FIELD gs_drseg_ci-zztext_1.  
            FIELD gs_drseg_ci-zztext_2.  
            MODULE tc_mrm_cust_modify ON CHAIN-REQUEST.  
        ENDCHAIN.  
    ENDLLOOP.  
  
    MODULE tc_mrm_cust_user_command.  
  
    MODULE write_data.
```

Listing 6.11 Ablauflogik zu Dynpro 0100

6.2 Toleranzprüfungen übersteuern

Mit den Toleranzgrenzen im Customizing der Logistik-Rechnungsprüfung stehen Ihnen vielfältige Möglichkeiten zur Verfügung, um das Sperrverhalten von Eingangsrechnungen bei Preis- oder Mengenabweichungen zu beeinflussen. Dabei werden bei der Erfassung der Eingangsrechnung absolute oder

prozentuale Ober- und Untergrenzen geprüft. Weicht ein Wert (Menge/Preis) über diese Grenzen hinaus von der zugehörigen Bestellung ab, wird der Beleg gesperrt. Durch die Sperrung kann die Rechnung in der Finanzbuchhaltung nicht zur Zahlung angewiesen werden, solange die Sperre nicht beseitigt wurde. Mit der Erweiterung MM08R002 können Sie zusätzlich diese Ober- und Untergrenzen individuell übersteuern.

6.2.1 Toleranzgrenzen im Customizing

Im Customizing (Transaktion SPRO) finden Sie die Einstellungen zu Toleranzgrenzen unter MATERIALWIRTSCHAFT • LOGISTIK-RECHNUNGSPRÜFUNG • RECHNUNGSSPERRE • TOLERANZGRENZEN FESTLEGEN. Abhängig vom Buchungskreis und dem Toleranzschlüssel, können Sie hier absolute und prozentuale Ober- und Untergrenzen für Differenzen pflegen (siehe Abbildung 6.8).

The screenshot shows the SAP Customizing (SPRO) interface for 'Toleranzgrenzen ändern: Detail'. The top menu bar includes 'Tabellensicht', 'Bearbeiten', 'Springen', 'Auswahl', 'Hilfsmittel', 'System', and 'Hilfe'. Below the menu is a toolbar with various icons. The main content area is titled 'Sicht "Toleranzgrenzen" ändern: Detail' and contains the following settings:

- Toleranzschlüssel: PP (Preisabweichung)
- Buchungskreis: 1000 (IDES AG)
- Beträge in: EUR (Euro)

The 'Untergrenze' (Lower Limit) section is divided into two sub-sections:

- Absolut:** Nicht prüfen, Grenze prüfen. Wert: 10,00
- Prozentual:** Nicht prüfen, Grenze prüfen. Toleranzgrenze %: 20,00

The 'Obergrenze' (Upper Limit) section is also divided into two sub-sections:

- Absolut:** Nicht prüfen, Grenze prüfen. Wert: 5,00
- Prozentual:** Nicht prüfen, Grenze prüfen. Toleranzgrenze %: 5,00

Abbildung 6.8 Toleranzgrenzen im Customizing

Setzen Sie bei einer Grenze die Option NICHT PRÜFEN, kann die Rechnungsposition in dieser Richtung unbegrenzt abweichen. Werden diese Grenzen überschritten, kann die Rechnung zwar gesichert werden (in der Standard-

auslieferung ist die zugehörige Meldung als Warnung eingestellt), wird jedoch mit einem Sperrgrund versehen.

Der Toleranzschlüssel steht für eine SAP-seitig festgelegte Art der Abweichung zwischen der Eingangsrechnung und Bestellung oder dem Wareneingang. In Tabelle 6.6 sehen Sie eine Kurzübersicht über alle vom System geprüften Toleranzschlüssel. In der Dokumentation zum zuvor genannten Customizing-Punkt finden Sie zusätzlich ausführliche Erläuterungen zu den einzelnen Schlüssel.

Toleranzschlüssel	Bedeutung
AN	Betragshöhe Position ohne Bestellbezug
AP	Betragshöhe Position mit Bestellbezug
BD	Kleindifferenzen automatisch ausbuchen
BR	prozentuale Bestellpreismengenabweichung (RE vor WE)
BW	prozentuale Bestellpreismengenabweichung (RE nach WE)
DQ	Überschreiten Betrag Mengenabweichung
DW	Mengenabweichung bei Wareneingangsmenge = null
KW	Preisabweichung Bezugsnebenkosten
LA	Betragshöhe Limitbestellung
LD	Zeitüberschreitung Limitbestellung
PP	Preisabweichung
PS	Preisabweichung Schätzpreis
ST	Terminabweichung
VP	V-Preisabweichung

Tabelle 6.6 Übersicht über Toleranzschlüssel

6.2.2 Nutzung der Erweiterung

Auch die Erweiterung MM08R002 soll wieder anhand eines kleinen Beispiels erläutert werden. Nehmen wir an, Sie haben die Toleranzgrenzen für Preisabweichungen zur Bestellung (Toleranzschlüssel PP) so eingestellt, dass nur zwei Prozent Abweichung nach oben und unten erlaubt sind. Sie haben nun jedoch eine Warengruppe, bei der die Preise für die Waren starken Marktschwankungen unterliegen, daher soll in diesem Fall eine Abweichung von fünf Prozent erlaubt sein. Sie benötigen zur Umsetzung daher zunächst einmal die Information, welcher Warengruppe das Material einer Rechnungsposition angehört, und müssen dann abhängig davon die Grenzen neu defini-

nieren. Leider erhalten Sie diese Information jedoch in einem anderen User-Exit als dem User-Exit, der verwendet wird, um die Grenzwerte zu überschreiben.

Werfen Sie einen Blick auf Tabelle 6.7: Wenn Sie eine Position in der Eingangsrechnung erfassen und eine Abweichung auftritt, wird abhängig von der Art der Abweichung zunächst der EXIT_SAPLMR1M_001 oder EXIT_SAPLMRMP_001 aufgerufen. Sie müssen die Kopf- und Positionsdaten in die globalen Daten der Funktionsgruppe kopieren. Anschließend startet der EXIT_SAPLMRMC_001, in dem Sie die Toleranzgrenzen neu definieren können. Über die globalen Daten der Funktionsgruppe können Sie hier wieder auf die Kopf- und Positionsdaten zugreifen.

User-Exit	Beschreibung
EXIT_SAPLMR1M_001	Wird für jede Position bei Preis- und Mengenabweichungen durchlaufen und enthält Kopf- und Positionsdaten der Position.
EXIT_SAPLMRMP_001	Wird für jede Position bei Termin- und Betragsabweichungen durchlaufen und enthält Kopf- und Positionsdaten der Position.
EXIT_SAPLMRMC_001	Wird für jede Position und je relevanten Toleranzschlüssel einmal durchlaufen und bietet die Möglichkeit, die Toleranzgrenzen neu zu definieren.

Tabelle 6.7 User-Exits der Erweiterung MM08R002

Mit der folgenden Anleitung setzen Sie die Theorie in die Praxis um:

1. Legen Sie zunächst ein neues Projekt in Transaktion CMOD an, und nehmen Sie die Erweiterung MM08R002 auf. Sichern Sie anschließend das Projekt, und wechseln Sie dann in die KOMONENTEN.
2. Gehen Sie jetzt in den EXIT_SAPLMR1M_001, und wählen Sie dort den Menüpunkt SPRINGEN • GLOBALE DATEN. Sie können die globalen Daten eines User-Exits nicht direkt ändern, dies wäre eine Modifikation, aber Sie finden in den globalen Daten eine Zeile mit dem Inhalt INCLUDE ZXMO8TOP. Dieses Include liegt im Kundennamensraum und kann so modifikationsfrei von Ihnen angepasst werden.

Klicken Sie doppelt auf den Namen des Includes, um diesen anzulegen. Definieren Sie anschließend zwei Strukturen zur Aufnahme der Kopf- und Positionsdaten aus der Schnittstelle des Exits. Die Datentypen können Sie aus der Schnittstelle des User-Exits übernehmen (siehe Listing 6.12).

```
*&-----*
*& Include          ZXM08TOP
*&-----*
* Globale Daten: Kopf- und Positionsdaten
DATA: zz_rbkpv TYPE MRM_RBKPV,
      zz_drseg TYPE MMCR_DRSEG.
```

Listing 6.12 Globale Daten der Erweiterung

3. Gehen Sie nun zurück in den User-Exit, und legen Sie dort das Include ZXM08U04 durch einen Doppelklick an. Übertragen Sie nun einfach die Daten aus der Schnittstelle in die soeben definierten globalen Strukturen (siehe Listing 6.13).

```
*&-----*
*& Include          ZXM08U04 (EXIT_SAPLMR1M_001)
*&-----*
*"*"Lokale Schnittstelle:
*" IMPORTING
*" VALUE(I_RBKPV) TYPE MRM_RBKPV
*" VALUE(I_YRSEG) TYPE MMCR_DRSEG
*" EXCEPTIONS
*" CALL_FAILURE
*"-----*
zz_rbkpv = i_rbkpv.
zz_drseg = i_ydrseg.
```

Listing 6.13 User-Exit EXIT_SAPLMR1M_001

4. Wechseln Sie nun in den User-Exit EXIT_SAPLMRMP_001, und legen Sie das Include ZXM08U17 an. Übertragen Sie auch hier die Daten der Schnittstelle in die globalen Daten. Das Coding ist identisch mit Listing 6.13.
5. Zuletzt gehen Sie in den EXIT_SAPLMRMC_001 und legen dort das Include ZXM08U19 an. In diesem User-Exit werden zwei Importparameter geliefert, der Buchungskreis und der Toleranzschlüssel. Sie sollten daher Ihre Neudefinition der Grenzen abhängig von diesen Parametern gestalten.
Die neuen Toleranzgrenzen können Sie anhand von Tabelle 6.8 festlegen. Die Felder E_XW1JA, E_XW2JA, E_XP1JA und E_XP2JA bestimmen, ob die zugehörige Grenze geprüft werden soll, identisch mit den Optionen NICHT PRÜFEN und GRENZE PRÜFEN im Customizing (siehe Abbildung 6.8). Setzen Sie das Flag nicht, ist auch hier eine unbegrenzte Abweichung erlaubt.
6. Sie müssen das Flag E_CHECK setzen, wenn Ihre Änderungen angewendet werden sollen, anderenfalls werden weiterhin die Einstellungen aus dem Customizing verwendet.

Parameter	Bedeutung
E_WERT1	Untergrenze absolut
E_XW1JA	Untergrenze absolut prüfen
E_WERT2	Obergrenze absolut
E_XW2JA	Obergrenze absolut prüfen
E_PROZ1	Untergrenze prozentual
E_XP1JA	Untergrenze prozentual prüfen
E_PROZ2	Obergrenze prozentual
E_XP2JA	Obergrenze prozentual prüfen
E_CHECK	Kennzeichen: neue Grenzen aus User-Exit verwenden

Tabelle 6.8 Parameter des EXIT_SAPLMRMC_001

Standardwerte aus Customizing einbinden

Wenn Sie E_CHECK setzen, sonst jedoch kein Feld füllen, bedeutet dies, dass grundsätzlich unbegrenzte Abweichungen erlaubt sind. Da in unserem Beispiel nur die prozentuale Grenze angepasst werden soll, können Sie optional alle anderen Werte aus dem Customizing (Tabelle T169G) nachlesen. Das Beispiel hierzu finden Sie in Listing 6.14.

```
*&-----*
*& Include          ZXM08U19 (EXIT_SAPLMRMC_001)
*&-----*
DATA ls_t169g TYPE t169g.

* Abweichende Logik für alle Buchungskreise
* und Preisabweichung (Toleranzschlüssel PP)
* für Warengruppe 007
IF i_tolsl = 'PP' AND
   zz_drseg-matk1 = '007'.
* Standardwerte aus Customizing holen
SELECT SINGLE * FROM t169g INTO ls_t169g
  WHERE bukrs = i_bukrs AND
        to1sl = i_tolsl.
* Absolute Grenzen aus Customizing zuordnen
e_wert1 = ls_t169g-wert1.
e_wert2 = ls_t169g-wert2.
e_xw1ja = ls_t169g-xw1ja.
e_xw2ja = ls_t169g-xw2ja.
* Prozentuale Abweichung erhöhen
e_proz1 = 5.
e_xplja = 'X'.
e_proz2 = 5.
e_xp2ja = 'X'.
```

```
* Änderung verwenden  
  e_check = 'X'.  
ENDIF.
```

Listing 6.14 User-Exit EXIT_SAPLMRMC_001

Index

A

ABAP Dictionary 101, 154
ABAP Editor 112
ABAP-Liste 81
Abgrenzung, freie 78
Abladestelle 140, 142
Ablaufart 107
Ablauflogik 41, 104, 154, 163, 169
Ablaufmodul 160
Abnahme 94
Abrufbestellung 229
Abweichung, unbegrenzte 173, 175
AC03 244
Aktion 99
Aktivierungsgrad 199, 207
Änderungsbeleg 15, 226
Anfrage 33
Anlieferung 237, 238
Anzahlung 271
Anzeigemodus 104, 105, 167
Anzeigetransaktion 49
AOBJ 67
Append-Struktur 156
Arbeitsbereich 79, 80
Arbeitsgebiet 200, 207
ARC_MM_EBAN_CHECK 241
ARC_MM_EBAN_PRECHECK 240
ARC_MM_EBAN_WRITE 241
ARC_MM_EINA_CHECK 241
ARC_MM_EINA_WRITE 242
ARC_MM_EKKO_CHECK 67
ARC_MM_EKKO_WRITE 67, 243
ARC_MM_INVBEL_CHECK 260
ARC_MM_INVBEL_WRITE 261
ARC_MM_MATBEL_CHECK 259
ARC_MM_MATBEL_WRITE 260
ARC_MM_REBEL_CHECK 278
ARC_MM_REBEL_WRITE 278
Archivierung 67, 240, 259, 278, 283
Archivierungsobjekt
 MM_INVBEL 260
 MM_MATBEL 259
Attribut 26, 51, 58, 109
Auftragsbestätigung 239
Auswahlliste 85
AUTHORITY-CHECK 77

B

BAdI 19, 20, 97, 114
 aktivieren 25, 31
 ARC_MM_EBAN_CHECK 241
 ARC_MM_EBAN_PRECHECK 240
 ARC_MM_EBAN_WRITE 241
 ARC_MM_EINA_CHECK 241
 ARC_MM_EINA_WRITE 242
 ARC_MM_EKKO_CHECK 67, 242
 ARC_MM_EKKO_WRITE 67, 72, 243
 ARC_MM_INVBEL_CHECK 260
 ARC_MM_INVBEL_WRITE 261
 ARC_MM_MATBEL_CHECK 259
 ARC_MM_MATBEL_WRITE 260
 ARC_MM_REBEL_CHECK 278
 ARC_MM_REBEL_WRITE 278
 BADI_MATERIAL_OD 280, 281
 BADI_MATERIAL_REF 184, 280
 BADI_MM_MATNR 283
 deaktivieren 25, 31
 Definition 20
 Dokumentation 22, 28
 filterabhängiges 21
 Implementierung 20, 22
 klassisches 20, 26
 MB_ACCOUNTING_DISTRIBUTE 256
 MB_CHECK_LINE_BADI 252
 MB_DOCUMENT_BADI 250
 MB_GOODSMOVEMENT_DCI 256
 MB_INSMK_WIP_CHANGE 252
 MB_MIGO_BADI 25, 97, 123, 251
 MB_MIGO_ITEM_BADI 130, 251
 MB_RESERVATION_BADI 138, 259
 ME_BAPI_PO_CUST 221
 ME_BAPI_PR_CUST 217
 ME_CHDOC_ACTIVE 234
 ME_CHECK_ALL_ITEMS 222
 ME_CIP_ALLOW_CHANGE 234
 ME_COMMITMENT_PLAN 232
 ME_COMMITMENT_RETURN 233
 ME_DEFINE_CALCTYPE 230
 ME_GUI_PO_CUST 25, 33, 35, 49,
 56, 223
 ME_HOLD_PO 20, 222
 ME_PO_PRICING_CUST 231

- ME_POHIST_DISP_CUST* 27, 222
ME_PROCESS_OUT_CUST 227
ME_PROCESS_PO_CUST 33, 36, 62, 216, 224
ME_PROCESS_REQ_CUST 34
ME_PURCHDOC_POSTED 223
ME_RELEASE_CREATE 228
ME_REQ_HEADER_TEXT 218
ME_REQ_OI_EXT 233
ME_REQ_POSTED 218
ME_TAX_FROM_ADDRESS 225
mehrfach nutzbares 21
MEOUT_BAPI_CUST 227
MM EDI_DESADV_IN 238
MMSRV_SM_BAPI_CUST 243
MMSRV_SM_MAIN 244
MMSRV_SM_NOTIFY 244
MRM_BLOCKREASON_DELETE_CUST 272
MRM_DOWNPAYMENT 271
MRM_ERS_HDAT_MODIFY 264
MRM_ERS_IDAT_MODIFY 264
MRM_HEADER_CHECK 265
MRM_HEADER_DEFAULT 266
MRM_INVOICE_UPDATE 270
MRM_ITEM_CUSTFIELDS 151, 265
MRM_MRIS_HDAT_MODIFY 266
MRM_MRIS_IDAT_MODIFY 266
MRM_MRKO_HDAT_MODIFY 267
MRM_PARTNER_CHECK 271
MRM_PAYMENT_TERMS 267
MRM_RELEASE_CHECK 267
MRM_RETENTIONS 272
MRM_TOLERANCE_GROUP 268
MRM_TRANSACT_DEFAULT 268
MRM_UDC_DISTRIBUTE 268
MRM_VARIANCE_TYPE 269
MRM_WT_SPLIT_UPDATE 269
neues 26
SAP-internes 21
SMOD_MRLB001 228
WRF_MRM_ASSIGN_TEST 270
WRF_PREPAY_INVOICE 270
 BAdI Builder 71
BADI_MATERIAL_OD 280, 281
BADI_MATERIAL_REF 184, 280
BADI_MM_MATNR 283
 BAPI
 BAPI_CONTRACT_CHANGE 227
 BAPI_CONTRACT_CREATE 227
 BAPI_PO_CHANGE 221, 224
 BAPI_PO_CREATE1 221, 224
 BAPI_PR_CHANGE 217
 BAPI_PR_CREATE 217
 BAPI_PR_GETDETAIL 217
 BAPI_REQUISITION_CREATE 220
 BAPI_SAG_CHANGE 227
 BAPI_SAG_CREATE 227
 BAPI_SERVICE_CHANGE 243
 BAPI_SERVICE_CREATE 243
 BAPI_USER_GET_DETAIL 88
 BAPIRET2 128, 131
 BASO0001 245
 Beistellkomponente 257
 Belegnummer 48
 Belegposition 42
 Belegübersicht 82
 Benutzergruppe 79
 Benutzerstamm 88
 Berechtigungsobjekt
 S_TCODE 77
 Bericht 79
 RGUGBR00 205
 RGUGBR28 196
 Bestandsfindung 255
 Bestandsführung 97, 250
 Bestellanforderung, Selektionsvariante
 34, 75, 82, 216
 Bestellung 20, 33, 82, 221
 Belegübersicht 234
 Bestelldatum 133
 Bestellentwicklung 31, 222
 Bestellnummer 43
 Eingabeüberprüfung 36, 65
 Merken 20
 Nachrichtensammler 65
 Selektionsvariante 75
 Bewertung 145, 261
 Bezugsnebenkosten 268
 Bezugsquellenfindung 235
 Bilanzbewertung 262
 BKPF 195
 BLAREL 239
 Boolesche Klasse 195, 204
 BSEG 195
 Buchhaltungsbeleg 193
 Buchungskreis 174
 Buchungsvorgang, WRX 146
 Business Add-in → BAdI

C

C_EXIT_PARAM_CLASS 201
 C_EXIT_PARAM_FIELD 201, 208
 C_EXIT_PARAM_NONE 201
 CALL FUNCTION ... IN UPDATE
 TASK 123
 CHAIN 170
 Changing-Parameter 177, 189
 Chargenstamm 254
 CHECK 66
 CHECK_HEADER 128
 CMOD 16, 19, 91, 132, 179, 180
 COBL_MRM 156
 COMMIT WORK 48
 CUSTOM_DATA_TRANSFER 164

D

DATA_MODIFY 139
 Daten, globale 39, 173
 Datenaustausch 154
 Datenbank 46, 153
 Datenbanktabelle 42, 46
 Datenfeld, OK-CODE 40
 Datenkommunikation 164
 Datentyp 52
 Definition 20
 DELFOR 239
 DELINS 239
 DELVRY01 238
 DESADV 238, 239, 238
 Dictionary 41
 Dienstleistungsabwicklung 87, 243
 Direkte Typeingabe 111
 Disposition, plangesteuerte 187
 Dispositionslosgröße 187
 DRSEG_CI 155
 Dynpro 35, 49, 98, 114, 152

E

Einbehalt 272
 Eingabeüberprüfung 97, 127
 Eingangsrechnung 147
 Einkauf 33
 Einkaufsbeleg, archivieren 68
 Einkaufsinfosatz 241

Einteilung 134
 Einteilungsdaten 66
 Enjoy-Bestellung 33
 Erfassungsblatt 93
 ERS-Verfahren 264
 Erweiterung 16, 33
 BASO0001 245
 IQSM0007 255
 LIFO0040 261
 LMEK0001 231
 LMEK0002 232
 LMELA002 257
 LMELA010 239, 257
 LMEQR001 235
 LMEXF001 232
 LMR1M001 273
 LMR1M002 263
 LMR1M003 274
 LMR1M004 274
 LMR1M005 275
 LMR1M006 275
 LWBON001 240
 LWSUS001 235
 M06B0001 218
 M06B0002 219
 M06B0003 219
 M06B0004 220
 M06B0005 219
 M06E0004 225
 M06E0005 225
 MB_CF001 253
 MBCF0002 253
 MBCF0005 253
 MBCF0006 257
 MBCF0007 259
 MBCF0009 254
 MBCFC003 254
 MBCFC004 255
 ME590001 226
 MEETA001 230
 MEFLD004 258
 MEQUERY1 234
 MEREQ001 34, 220
 MEVME001 258
 MGA00001 177, 282
 MGA00002 282
 MGA00003 283
 MGW00002 282
 MM06E001 239
 MM06E003 236

- MM06E004 226
 - MM06E005 33, 236
 - MM06E007 15, 226
 - MM06E008 229
 - MM06E009 237
 - MM06E010 237
 - MM06E011 221
 - MM06L001 238
 - MM08R001 279
 - MM08R002 273, 279
 - MMDA0001 237
 - MMFAB001 229
 - MRFLB001 230
 - MRMH0001 276
 - MRMH0002 276
 - MRMH0003 277
 - MRMN0001 277
 - NIWE0000 262
 - NIWE0001 262
 - NIWE0002 263
 - NIWE0003 263
 - RMVKON00 275
 - SRVDET 245
 - SRVEDIT 245
 - SRVESI 246
 - SRVESKN 246
 - SRVESLL 90, 246
 - SRVESSR 247
 - SRVEUSCR 247
 - SRVKNTTP 248
 - SRVLIMIT 248
 - SRVMAIL1 248
 - SRVMSTLV 249
 - SRVREL 249
 - SRVSEL 249
 - XMBF0001 255
 - Erweiterungsimplementierung 28, 29
 - Erweiterungskategorie 155, 180
 - Erweiterungspunkt 26
 - expliziter 27
 - impliziter 27
 - Erweiterungsspot 26, 27
 - ES_BADI_INVOICE_UPDATE 270
 - ES_BADI_ME_BAPI 217, 221, 227
 - ES_BADI_ME_POHIST 27, 222
 - ES_BADI_MRM_DOWN-PAYMENT 271
 - ES_BADI_MRM_PARTNER 271
 - ES_BADI_MRM_RETENTION 272
 - ES_COMMITMENT_PLAN 232
 - MB_GOODSMOUMENT 252, 256
 - ME_PROCESS_OUT 227
 - MRM_BLOCKREASON_DELETE 272
 - Exit-Routine 206
 - Substitution 207
 - Validierung 199
 - Expertenmodus 197
 - Expliziter Erweiterungspunkt 27
 - EXPORT ... COMPRESSION ON 125
 - EXPORT ... TO DATA BUFFER 124
 - Exportparameter 163
- ## F
-
- Fehlermeldung 65, 181
 - Feld
 - BEXCLUDE 204
 - Definition 38
 - Eigenschaft 41
 - kundeneigenes 97
 - SGTXT 251, 253, 274
 - Symbol 212
 - Feld-Feld-Zuweisung, Substitution 206
 - Feldstatus 54
 - Form, get_exit_titles 200, 208
 - Freie Abgrenzung 78
 - Freigabestrategie 225
 - Freigabeverfahren 218
 - Funktionsbaustein 34, 42, 45, 69, 72, 98
 - BAPI_MATERIAL_SAVEDATA 280
 - BAPI_USER_GET_DETAIL 88
 - CONVERSION_EXIT_MATN1_INPUT 283
 - CONVERSION_EXIT_MATN1_OUTPUT 283
 - MATERIAL_NUMBER_GET 282
 - Funktionsgruppe 34, 98
- ## G
-
- Geschäftslogik 56
 - Geschäftspartner 271
 - Global Unique Identifier (GUID) 123
 - Globale Daten 39, 173
 - Grundliste 82

H

Handle 73
 Hintergrundprüfung 157
 Höchstbestand 187
 HOLD_DATA_DELETE 123
 HOLD_DATA_LOAD 123, 126
 HOLD_DATA_SAVE 123

I

I_CLASS_ID 114
 IDoc
 Basistyp DELVRY01 238
 IDOC_INPUT_SRVMAS 244
 Verarbeitung 238
 Implementierende Klasse 30, 51, 53
 Implementierung 20, 22, 99, 161
 Impliziter Erweiterungspunkt 27
 IMPORT ... ACCEPTING TRUNCATION 127
 IMPORT ... FROM DATA BUFFER 124
 Importdaten 226
 Importparameter 45, 48
 IN UPDATE TASK 48
 Include
 LMEVIEWSF01 40
 MM_MESSAGES_MAC 65
 Z-Include 18
 InfoSet 79
 INSERT 125
 Instanz 165
 Instanzattribut 58, 113
 Interface 22, 28, 57, 162, 165
 IF_PURCHASE_ORDER_ITEM_MM 57
 IF_PURCHASE_ORDER_MM 57, 62
 Interne Tabelle 44
 Inventurbeleg 260
 INVOICE_DATA_GET 163, 166
 IQSM0007 255

K

Karteireiter 35
 Klasse 162
 Boolesche 195, 204
 CL_EXITHANDLER 165
 implementierende 30, 51, 53
 Klassen-Interface 22

Klassisches BAdI 20, 26
 Kommunikation 151
 Komponente 16, 17
 Konfigurationsdaten 234
 Konsignationsabrechnung 267
 Konsignationsabwicklung 275
 Konstante 110, 206
 C_EXIT_PARAM_CLASS 201
 C_EXIT_PARAM_FIELD 201, 208
 C_EXIT_PARAM_NONE 201
 Konstanter Wert, Substitution 206
 Kontenfindung 146
 Kontenplan 146
 Kontierung 87, 145, 261
 Kontierungsdaten 66
 Kontierungsfeld 152
 Kontoart 155
 Kontomodifikationskonstante 146,
 148, 263
 Kontrakt 227
 Kopfdaten 39
 Kostenstelle 88
 Kundeneigenes Feld 97
 Kundennamensraum 154, 173

L

Lagerort 130, 254
 Laufzeitverhalten 31
 Leistungsblatt 245
 Leistungserfassungsblatt 87, 245
 Leistungspezifikation 245
 Leistungsstamm 244
 Leistungsverzeichnis 249
 Leistungszeile 87, 91
 Lieferant 132
 Lieferantenadresse 237
 Lieferantenbeurteilung 238
 Lieferavis 239
 Lieferdatum 132
 Lieferplan 134, 227
 Lieferplaneinteilung 134
 LIFO, Bewertung 261
 LIFO0040 261
 Limitprüfung 248
 LMEKO001 231
 LMEKO002 232
 LMELA002 257
 LMELA010 239, 257
 LMEQR001 235

LMEVIEWSF01 40
 LMEXF001 232
 LMR1M001 273
 LMR1M002 263
 LMR1M003 274
 LMR1M004 274
 LMR1M005 275
 LMR1M006 275
 Logistik-Informationssystem 240
 Logistik-Rechnungsprüfung 151, 170,
 197, 211, 264
 Lohnbearbeitung 234, 257
 LOOP AT SCREEN 104
 Löschkennzeichen 83
 Löschrprogramm 72
 LWBON001 240
 LWSUS001 235

M

M06B0001 218
 M06B0002 219
 M06B0003 219
 M06B0004 220
 M06B0005 219
 M06E0004 225
 M06E0005 225
 Makro
 MMPUR_DYNAMIC_CAST 57
 MMPUR_MESSAGE_FORCED 66
 MMPUR_METAFIELD 65
 MARC 185
 Marktpreisanalyse 262
 Material 182
 Materialbeleg 100, 250, 259
 Materialbelegnummer 101
 Materialnummer 178
 Materialstamm, Anlage 177, 181, 280
 Materialstammdaten, Prüfung 178
 Materialstammpflege, eigene Fehler-
 meldung 177
 Materialstatus 184, 186
 Materialwirtschaft 33
 MB_ACCOUNTING_DISTRIBUTE 256
 MB_CF001 253
 MB_CHECK_LINE_BADI 252
 MB_DOCUMENT_BADI 250
 MB_GOODSMOVEMENT 252, 256
 MB_GOODSMOVEMENT_DCI 256
 MB_INSMK_WIP_CHANGE 252
 MB_MIGO_BADI 25, 97, 123, 251
 MB_MIGO_ITEM_BADI 130, 251
 MB_RESERVATION_BADI 138, 259
 MB21 138
 MB22 138
 MBCF0002 253
 MBCF0005 253
 MBCF0006 257
 MBCF0007 259
 MBCF0009 254
 MBCFC003 254
 MBCFC004 255
 ME_BAPI_PO_CUST 221
 ME_BAPI_PR_CUST 217
 ME_CHDOC_ACTIVE 234
 ME_CHECK_ALL_ITEMS 222
 ME_CIP_ALLOW_CHANGE 234
 ME_COMMITMENT_PLAN 232
 ME_COMMITMENT_RETURN 233
 ME_DEFINE_CALCTYPE 230
 ME_GUI_PO_CUST 25, 33, 223
 ME_HOLD_PO 20, 222
 ME_PO_PRICING_CUST 231
 ME_POHIST_DISP_CUST 27, 222
 ME_PROCESS_OUT 227
 ME_PROCESS_OUT_CUST 227
 ME_PROCESS_PO_CUST 33, 62, 224
 ME_PROCESS_REQ_CUST 34, 216
 ME_PURCHDOC_POSTED 223
 ME_RELEASE_CREATE 228
 ME_REQ_HEADER_TEXT 218
 ME_REQ_OI_EXT 233
 ME_REQ_POSTED 218
 ME_TAX_FROM_ADDRESS 225
 ME23N 49
 ME31K 232
 ME31L 232
 ME59 226
 ME590001 226
 ME84 229
 MEETA001 230
 MEFLD004 258
 Mehrfachkontierung 90
 Meldungsprotokoll 131
 Mengenabweichung 170
 MEOUT_BAPI_CUST 227
 MEQUERY1 234
 MEREQ001 34, 220
 Metafeld 49, 52, 53
 Methode 22, 23

- CHECK 37, 67
 CHECK_HEADER 127, 128
 CHECK_ITEM 127, 129
 CLOSE 37, 63
 CREATE_MATERIAL 184
 CUSTOM_DATA_GET 153, 164
 CUSTOM_DATA_TRANSFER 154, 164
 CUSTOMDATA_MODIFY 153, 162
 DATA_CHECK 139
 DATA_MODIFY 139, 141
 DELETE 68, 72, 73
 EXECUTE 36
 FIELDSELECTION_HEADER 50, 55
 FIELDSELECTION_ITEM 50, 56
 GET_DATA 57
 GET_INSTANCE_FOR_
 SUBSCREENS 165
 HOLD_DATA_DELETE 123, 127
 HOLD_DATA_LOAD 123
 HOLD_DATA_SAVE 123, 125
 INIT 99, 112
 INITIALIZE 36, 63
 INVALIDATE 66
 INVOICE_DATA_GET 154, 163
 INVOICE_DATA_TRANSFER 153, 162
 ITEM_MODIFY 130
 LINE_DELETE 101
 LINE_MODIFY 100, 118
 MAP_DYNPRO_FIELDS 35, 49, 52
 MODE_SET 99, 112
 OPEN 36, 64
 PAI_DETAIL 100, 118
 PAI_HEADER 100
 PBO_DETAIL 100, 117
 PBO_HEADER 100
 POST 37, 64
 POST_DOCUMENT 101, 121
 PROCESS_ACCOUNT 37
 PROCESS_HEADER 36
 PROCESS_ITEM 36, 63
 PROCESS_SCHEDULE 36
 RESET 99, 112
 STATUS_AND_HEADER 99, 123
 SUBSCRIBE 35, 49, 52
 TABPAGE_LABEL_SET 153, 164
 TRANSPORT_FROM_DYNP 36
 TRANSPORT_FROM_MODEL 35
 TRANSPORT_TO_DYNP 35
 TRANSPORT_TO_MODEL 36
 WRITE 68
 MEVME001 258
 MGA00001 282
 MGA00002 282
 MGA00003 283
 MGW00002 282
 MIGO 97, 99, 251
 Merken 116, 123
 Referenzbeleg 111
 MIGO_CLASS_ID 110
 MIR4 157
 MIR6 157
 MIR7 197
 MIRA 157
 MIRO 151, 153, 197
 ML81N 93
 ML86 246, 248
 ML87 246, 248
 MLS5 249
 MM EDI DESADV_IN 238
 MM_INVBEL 260
 MM_MATBEL 259
 MM01 183
 MM06E001 239
 MM06E003 236
 MM06E004 226
 MM06E005 33, 236
 MM06E007 15, 226
 MM06E008 229
 MM06E009 237
 MM06E010 237
 MM06E011 221
 MM06L001 238
 MM08R001 279
 MM08R002 273, 279
 MMDA0001 237
 MMFAB001 229
 MMMFD 52
 MMPUR_METAFIELD 52, 173
 MMSRV_SM_BAPI_CUST 243
 MMSRV_SM_MAIN 244
 MMSRV_SM_NOTIFY 244
 Modifikation 27
 MODULE 165
 MODULE ... ON CHAIN-REQUEST 170
 Modulpool 157
 MOVE-CORRESPONDING 117
 MRBP 157
 MRFLB001 230
 MRM_BLOCKREASON_DELETE 272
 MRM_BLOCKREASON_DELETE_
 CUST 272
 MRM_DOWNPAYMENT 271

MRM_ERS_HDAT_MODIFY 264
 MRM_ERS_IDAT_MODIFY 264
 MRM_HEADER_CHECK 265
 MRM_HEADER_DEFAULT 266
 MRM_INVOICE_UPDATE 270
 MRM_ITEM_CUSTFIELDS 151, 265
 MRM_MRIS_HDAT_MODIFY 266
 MRM_MRIS_IDAT_MODIFY 266
 MRM_MRKO_HDAT_MODIFY 267
 MRM_PARTNER_CHECK 271
 MRM_PAYMENT_TERMS 267
 MRM_RELEASE_CHECK 267
 MRM_RETENTIONS 272
 MRM_TOLERANCE_GROUP 268
 MRM_TRANSACT_DEFAULT 268
 MRM_UDC_DISTRIBUTE 268
 MRM_VARIANCE_TYPE 269
 MRM_WT_SPLIT_UPDATE 269
 MRMH0001 276
 MRMH0002 276
 MRMH0003 277
 MRMN0001 277
 MSEG 101
 MSEG-ZEILE 122

N

Nachricht 195
 BLAREL 239
 DELFOR 239
 DELINS 239
 DESADV 239
 ORDCHG 239
 ORDERS 239
 ORDRSP 239
 REQOTE 239
 Nachrichtenklasse ändern 196
 Nebenkosten 264
 Niederstwertabgleich 261
 NIWE0000 262
 NIWE0001 262
 NIWE0002 263
 NIWE0003 263
 Nullzeile vorschlagen 137
 Nummernkreis 219, 236, 274

O

OB28 194, 196

OB28 194
 Objekt 58
 Obligo 221
 Funktion 232
 Plan 232
 OBYC 146
 OK-CODE 40
 ORDCHG 239
 ORDERS 239
 ORDRSP 239
 OXK3 152

P

PAI_HEADER 100
 Parameter
 CHANGING 46
 Schnittstelle 23, 28
 Parameterart
 C_EXIT_PARAM_CLASS 201
 C_EXIT_PARAM_FIELD 201, 208
 C_EXIT_PARAM_NONE 201
 Partnerrolle 271
 PBO_HEADER 100, 114
 Pipeline
 Abrechnung 267
 Abwicklung 275
 Positionsdaten 38, 101
 Positionsnummer 168
 Positionstext 130, 140, 196, 237, 251,
 253, 274
 Preisabweichung 170
 Preisfindung 222, 230
 PROCESS AFTER INPUT 154
 PROCESS BEFORE OUTPUT 154
 PROCESS_ACCOUNT 66
 PROCESS_HEADER 66
 PROCESS_ITEM 66
 PROCESS_SCHEDULE 66
 Programm
 MMREO050N 284
 MMREO110N 284
 RGGBR000 200
 RGGBS000 207
 RGUGBR00 196
 RM06BD70 241
 RM06BV70 241
 RM06BW70 241
 RM06ED47 72, 243
 RM06ED70 72, 243

RM06EFLB 230
RM06EW47 67, 242, 243
RM06EW70 67, 242
RM06ID47 242
RM06IW47 242
RM07IARCS 261
RM07IDELS 261
RM07MADES 260
RM07MARCS 260
SAPLMR1M 212
 Programmtabelle 160
 Projekt 16, 18
 Projektkontierung 88
 Projektstrukturplan 88
 Prüfung 195
 PSP-Element 88

Q

Query 79
 Query Painter 79

R

Rahmenvertrag 227
 RAWSTRING 124, 127
 RBDRSEG 156
 RBMA 156
 READ TABLE 183
 Rechnung 171
 Rechnungsbeleg 151, 152
 Kopf 162
 Modifikation 151
 Position 162
 Rechnungsplanabrechnung 266
 Rechnungsprüfung 264, 278
 Referenz 58, 99
 Referenzschlüssel 205
 Report
 RM06EFAB 229
 RMMR1MRB 277
 Report → Bericht
 Repository Browser 103, 158
 REQOTE 239
 Reservierung 138, 258
 Retourenposition 233
 Return-Tabelle 128
 RGGBR000 200
 RGGBS000 207

RGUGBR00 205
 RM06BD47 241
 RM06BV70 241
 RM06BW47 241
 RM06BW70 241
 RM06ED47 72, 243
 RM06ED70 72, 243
 RM06EFAB 229
 RM06EFLB 230
 RM06EW47 67, 242, 243
 RM06EW70 67, 242, 243
 RM06ID47 242
 RM06IW47 242
 RM07IARCS 261
 RM07IDELS 261
 RM07MADES 260
 RM07MARCS 260
 RMMR1MRB 277
 RMVKON00 267, 275, 276
 RSEG 156

S

Sachkontenreiter 151
 Sachkonto 140, 141
 SAP Enhancement Framework 26
 SAP Query 79
 Schlüsselfeld 186
 Schnittstelle 162, 165
 Schreibprogramm 67
 Schritt 195
 Screen Painter 103, 158
 SCREEN-NAME 169
 SE11 38, 101, 124, 180
 SE18 20, 28
 SE19 20, 22, 28, 109, 161, 187
 SE24 55
 SE80 39, 103
 Selektionsvariante 78
 Seriennummer 255
 SGTXT 251, 253, 274
 Sichtbarkeit 111
 Signatur 23
 SMOD 16
 SMOD_MRFLB001 228
 SQ01 79, 80
 SQ02 79
 SRVDET 245
 SRVEDIT 245
 SRVESI 246

Index

SRVESKN 246
SRVESLL 246
SRVSSR 247
SRVEUSCR 247
SRVKNTTP 248
SRVLIMIT 248
SRVMAIL1 248
SRVMSTLV 249
SRVREL 249
SRVSEL 249
STATUS_AND_HEADER 123
Struktur 105
 CI_DRSEG_CUST 153
 DRSEG 153
 DRSEG_CI 153, 155
Strukturerweiterung 69
Subscreen 103, 151, 158, 161, 184
Substitution 193, 204
 Exit 206
 Exit-Routine 207
 Feld-Feld-Zuweisung 206
 konstanter Wert 206
 Ursprungsbeleg 210

T

T80D 207
Tabelle
 Arbeitsbereich 160
 BKPF 195
 BSEG 195
 CI_MMMH1 179
 GB01 204
 interne 73, 183, 200, 212
 MARC 185
 MSEG 101
 T80D 200, 207
 YDRSEG 212
Tabellenparameter 46, 107
Table Control Wizard 152, 158, 166
TABLE_LINE 141
Tabstrip 35
Textfeld 101
Textsymbol 164
Toleranzgrenze 134, 170
Toleranzprüfung 151, 170, 273
Toleranzschlüssel 172
Transaktion
 AC03 244
 AOBJ 67
 CMOD 16, 19, 91, 132, 179, 180

MB01 256
MB21 138, 259
MB22 138, 259
ME21N 93
ME23N 49
ME31K 232
ME31L 232
ME59 226
ME84 229
MIGO 97, 110, 111, 116, 123,
 251, 256
MIR4 157
MIR6 157, 272
MIR7 197, 265
MIRA 157, 266
MIRO 151, 153, 197, 265
ML81N 93
ML86 246, 248
ML87 246, 248
MM01 183, 280
MM41 280, 281
MRBP 157
MRBR 268, 273
MRF1 262
MRF3 262
MRIS 266
MRKO 267
MRNB 271
MRRL 271
OB28 194, 196
OBBH 204
OBYC 146
OXK3 152
SE11 38, 101, 124, 180
SE18 20, 28
SE19 20, 22, 28, 109, 141, 161, 187
SE24 55
SE80 39, 103
SE91 198
SMOD 16
SQ01 79, 80
SQ02 79
Typ 26, 111
Typeingabe, direkte 111
Typgruppe 53

U

Überlieferungsmenge 134, 135
Unbegrenzte Abweichung 175
Unterlieferungsmenge 135

- Upcast 57
 Ursprungsbeleg, Substitution 210
 User-Exit 15
 aktivieren 19
 deaktivieren 19
 EXIT_RM06EFAB_001 229
 EXIT_RM06EFLB_001 230
 EXIT_RM06LBAT_001 238
 EXIT_RM06LBEW_001 238
 EXIT_RMMR1MRB_* 277
 EXIT_RMVKON00_* 276
 EXIT_SAPLBASO_* 245
 EXIT_SAPLEBND_* 219, 225, 249
 EXIT_SAPLEBNE_001 220
 EXIT_SAPLEBNF_* 218, 225
 EXIT_SAPLEINL_001 230
 EXIT_SAPLEINM_* 239
 EXIT_SAPLEINR_* 132, 134, 257, 258
 EXIT_SAPLIE01_007 255
 EXIT_SAPLKONT_011 147, 264
 EXIT_SAPLLIFS_* 262
 EXIT_SAPLMBMB_001 253
 EXIT_SAPLMDBF_* 256
 EXIT_SAPLME59_001 226
 EXIT_SAPLMEKO_* 231
 EXIT_SAPLMELO_001 238
 EXIT_SAPLMEQR_001 235
 EXIT_SAPLMEQUERY_* 76, 79,
 83, 235
 EXIT_SAPLMEREQ_* 220
 EXIT_SAPLMEXF_001 232
 EXIT_SAPLMG02_* 282
 EXIT_SAPLMG72_* 283
 EXIT_SAPLMGMU_001 177, 183, 282
 EXIT_SAPLMGNK_003 283
 EXIT_SAPLMIGO_001 257
 EXIT_SAPMLSK_001 87, 88, 246
 EXIT_SAPMLSL_001 248
 EXIT_SAPMLLSP_* 90, 91, 92, 245,
 247, 250
 EXIT_SAPMLLSR_* 93, 246, 247
 EXIT_SAPMLLST_001 249
 EXIT_SAPMLLSX_* 246, 248, 249
 EXIT_SAPLMMDA_001 237
 EXIT_SAPLMR1M_* 173, 273
 EXIT_SAPLMRM_BAPI_001 275
 EXIT_SAPLMRMC_* 173, 273, 275
 EXIT_SAPLMRME_003 274
 EXIT_SAPLMRMH_* 276
 EXIT_SAPLMRMN_* 277
 EXIT_SAPLMRMP_* 173, 273, 274
 EXIT_SAPLNIW0_* 262
 EXIT_SAPLNIW1_* 263
 EXIT_SAPLNIW3_* 263
 EXIT_SAPLNIWE_* 262
 EXIT_SAPLOMCV_* 283
 EXIT_SAPLWN08_001 240
 EXIT_SAPLWN12_001 240
 EXIT_SAPLWN35_001 240
 EXIT_SAPLWSUS_001 235
 EXIT_SAPM07DR_* 254
 EXIT_SAPMM06B_001 219
 EXIT_SAPMM06E_* 17, 221, 226,
 229, 236
 EXIT_SAPMM06L_001 238
 EXIT_SAPMM07M_* 253, 254, 257
 EXIT_SAPMM07R_001 259
- ## V
-
- Validierung 193
 Ändern der Nachrichtenklasse 196
 Exit-Routine 199
 Variable, B_RESULT 202
 Verarbeitung, IDoc 238
 Verbuchung 46, 121, 122
 Verbuchungsabbruch 108
 Verbuchungsbaustein 46, 99, 107
 Verrechnungskonto 145
 Verteilungskennzeichen 90
 Voraussetzung 195, 203
 Vorerfassung 197
 Vorgabewert 97
 Vorschlagsmenge 134, 136
 Vorwärtsdeklaration 53
- ## W
-
- Warenausgang 99
 Warenbegleitschein 253
 Warenbewegung 97
 Wareneingang 99, 132, 135, 256
 Wareneingangsbearbeitungszeit 186
 Warenempfänger 140, 142
 Warengruppe 172
 WE/RE-Kontenfindung 263
 WE/RE-Konto 145
 Wertkontrakt 229

Wertübergabe 46, 107
WE-Sperrbestand 132
WRF_MRM_ASSIGN_TEST 270
WRF_PREPAY_INVOICE 270
WRX 146

X

XMBF0001 255
XML-Rechnung 275
XSTRING 124

Y

YDRSEG 212

Z

Zeitpunkt 194
Zusammengesetzte Erweiterungs-
implementierung 29
Zuweisungsoperator 58