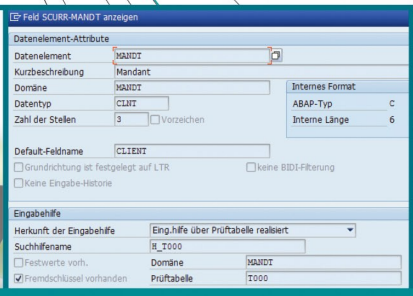
SAP PRESS



SAP®

Der technische Einstieg

Für Ausbildung und Selbststudium

- Grundwissen für Anpassung, Entwicklung und Administration
- So finden Sie sich erfolgreich im SAP-Umfeld zurecht
- Der Wegweiser für Studenten, Junior-Berater und Quereinsteiger

2., aktualisierte und erweiterte Auflage

Reinhold Plota
Waldemar Fix

Rheinwerk Publishing

Kapitel 9
Programme starten

In diesem Kapitel ...

... lernen Sie:

- wie Sie Programme als Dialog oder im Hintergrund starten
- was eine Variante ist und wozu sie da ist
- welche Möglichkeiten der Fehleranalyse Sie bei Programmabbrüchen haben

Programme können grob in *Online- oder Dialogprogramme* und *Batch-Programme* unterteilt werden. Dialogprogramme dienen vor allem der Erfassung oder Abfrage von Daten. Batch-Programme verarbeiten gespeicherte Daten und ermöglichen eine performantere Massendatenverarbeitung. SAP unterscheidet auch systemintern zwischen Dialogprozessen und Batch-Prozessen. Im ersten Fall kann das SAP-System in den Dialog mit dem Anwender treten und im zweiten Fall nicht.

9.1 Programme im Dialog starten

Dialogprogramme

Im bisherigen Verlauf des Buches haben wir die Programme immer nur im Dialog gestartet. Zu diesen Dialoganwendungen gehören auch die Werkzeuge des ABAP Dictionarys oder der Anwendungsentwicklung. Dialogprogramme werden über einen Transaktionscode gestartet. Wenn Sie Dialogprogramme über einen Menübaum starten, geschieht dies ebenfalls über einen Transaktionscode und nicht über den Programmnamen (siehe Abschnitt 2.3.1, »Verknüpfung im SAP-Logon«).

Transaktionscode

Transaktionscodes geben Sie im kleinen Kommandofeld links in der Systemfunktionsleiste ein. Dabei müssen Sie die Ebene berücksichtigen, aus der Sie über einen Transaktionscode eine Dialoganwendung starten wollen. Vom Prinzip her starten Sie Dialoganwendungen über das Kommandofeld immer vom Grundbild von SAP Easy Access aus. Befinden Sie sich nicht im Grundbild, müssen Sie vor den Transaktionscode den Steuerbefehl `/n` oder

/o setzen, also z. B. /nSE80 oder /oSE80. Im ersten Fall kehren Sie von der Befehlsfolge her zunächst zum Grundbild zurück – dies bewirkt der Befehl /n –, und dann wird die Transaktion gestartet; im zweiten Fall geschieht das Gleiche, nur wird dabei vorher ein neuer Modus geöffnet (zu den Steuerbefehlen siehe Tabelle 2.2 in Abschnitt 2.3.2, »Kommandofeld«).

9.2 Programme im Hintergrund starten

Job

Wenn Sie ein Programm für eine Hintergrundverarbeitung (Batch) starten wollen, müssen Sie dem SAP-System zuvor einen Auftrag dafür erteilen. Dieser Auftrag wird in der Datenverarbeitung *Job* genannt. Der Begriff *Batch* stammt noch aus der frühen Zeit der Datenverarbeitung und war ein Synonym für *Stapelverarbeitung*. Zu dieser Zeit gab es noch kein Multiprocessing; die Programme konnten also nur nacheinander laufen.

Step

Jobs stellen eine Klammer für darunterliegende *Steps* dar. Bei der Definition eines Jobs legen Sie lediglich den Namen des Jobs fest und seine Startbedingung. Welches Programm oder Programme dabei ausgeführt werden sollen, legen Sie in den Steps fest.

9.2.1 Jobs über Transaktion SM36 definieren

Job anlegen

Jobs definieren Sie in Transaktion SM36. Nachdem Sie Transaktion SM36 aufgerufen haben, legen Sie als Erstes den Namen des Jobs fest (siehe Abbildung 9.1). Die anderen Einstellungen, wie in den Feldern **Jobklasse** und **Ausführungsziel**, enthalten von der Administration vorgegebene Default-Einstellungen, die Sie so übernehmen können.

Jobklasse

Über die Jobklasse wird die Priorität eines Jobs gesteuert. Sie variiert zwischen:

- **A**: hoch
- **B**: mittel
- **C**: niedrig

Die Grundeinstellung sollte **C** (niedrig) sein. Je höher die Priorität, umso bevorzugter wird der Job vom SAP-System behandelt. Dies geht aber zu Lasten anderer Jobs.

Im Feld **Ausführungsziel** geben Sie an, in welchem SAP-System der Job ausgeführt werden soll. Dies ist in den meisten Fällen das SAP-System, in dem Sie sich angemeldet haben, sodass Sie hier nichts eintragen müssen.

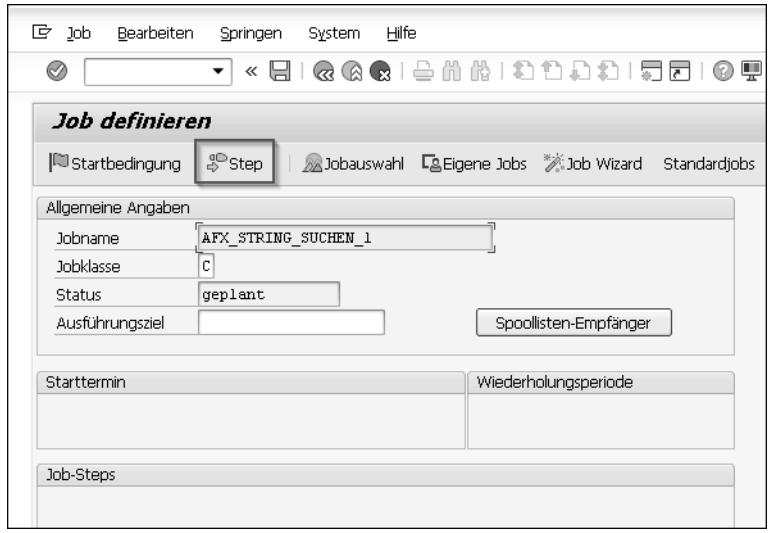


Abbildung 9.1 Transaktion SM36 – Job definieren

Als Nächstes legen Sie die Steps fest, die unter dem Job ausgeführt werden sollen. Ein Job kann ein bis mehrere Steps enthalten. Dies hängt von der von Ihnen festgelegten Verarbeitungslogik ab. Wenn Sie mehrere Steps definieren, werden sie nacheinander ausgeführt, und nicht zur gleichen Zeit. Sie müssen sich daher also Gedanken über die Reihenfolge der Steps machen.

Step anlegen

Um einen Step zu definieren, klicken Sie auf den Button Step (siehe Abbildung 9.1). Es öffnet sich ein Fenster, in dem Sie Angaben dazu machen, was ausgeführt werden soll (siehe Abbildung 9.2). Hierbei wählen Sie zwischen:

- ABAP-Programm
- Externes Kommando
- Externes Programm

Die beiden letzten Buttons gehören eher zur Systemadministration und werden hier daher vernachlässigt. Wenn Sie das Fenster öffnen, ist der Bereich **ABAP-Programm** für Ihre Eingaben bereit. Die drei Buttons in Abbildung 9.2 bewirken lediglich einen Sprung in den betreffenden Bereich des Fensters.

AFX_CODE_SCANNER

Als Beispiel für ein auszuführendes Programm in diesem Step haben wir das Programm AFX_CODE_SCANNER gewählt. Dieses Programm ist ein Utility, das Programmcode nach einem vorgegebenen String durchsucht. Das Ergebnis der Suche wird entweder im Dialog auf dem Bildschirm angezeigt, wenn Sie das Programm im Dialog gestartet haben, oder in eine *Spool-Datei*

eingetragen. Wo und was das Programm AFX_CODE_SCANNER suchen soll, wird in einer Variante zu dem Programm vorgegeben. Was Varianten sind, erläutern wir in Abschnitt 9.3, »Variante erzeugen«.

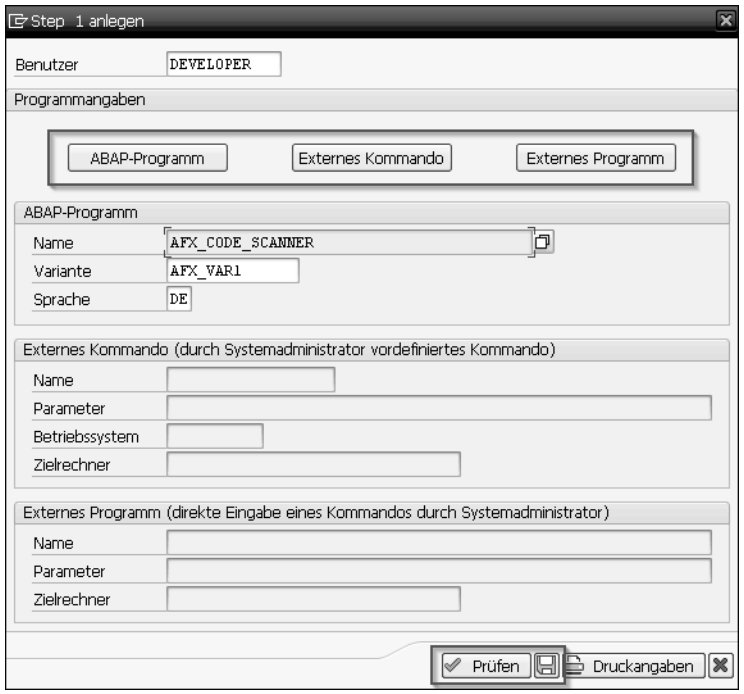


Abbildung 9.2 Transaktion SM36 – Step festlegen

Im Feld **Name** tragen Sie den Programmnamen ein. Bei der Definition von Steps sind die den Programmen zugeordneten Transaktionscodes bedeutungslos. Auf Groß- und Kleinschreibung müssen Sie nicht achten. Danach wählen Sie im Feld **Variante** die betreffende Variante zu dem Programm aus (wenn es mehrere Varianten gibt). Es gibt Programme, zu denen es keine Varianten gibt; dann tragen Sie in das Feld **Variante** nichts ein. Der Sprachschlüssel ist aus der Systemanmeldung vorbelegt.

Weitere Angaben müssen Sie nicht machen. Wenn Sie meinen, dass Ihre Angaben korrekt sind, können Sie unten in Abbildung 9.2 gleich auf das Icon (**Sichern**) klicken; ansonsten lassen Sie zuvor über den Button Prüfen die Richtigkeit der Angaben überprüfen. Damit ist gemeint, ob das SAP-System mit diesen Angaben das Programm starten kann.

Steplistenüberblick

Anschließend verzweigt das System in das Fenster **Steplistenüberblick** (siehe Abbildung 9.3). Hier werden alle von Ihnen angelegten Steps aufgelistet, und zwar in der Reihenfolge, in der Sie sie angelegt haben, der erste Step

zuoberst. In dieser Reihenfolge werden die Steps auch ausgeführt. In diesem Überblick können Sie die Reihenfolge der Steps nachträglich ändern. Setzen Sie z. B. im in Abbildung 9.3 gezeigten Fenster den Cursor auf die **Nr. 2** des zweiten Steps, und klicken Sie anschließend auf das Icon (**Markieren**). Danach markieren Sie die **Nr. 1** des ersten Steps und klicken auf das Icon (**Verschieben**). Die Reihenfolge der beiden Steps wird getauscht.

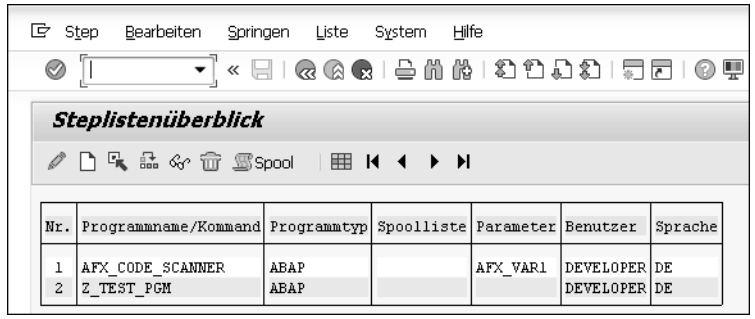


Abbildung 9.3 Transaktion SM36 – Steplistenüberblick

Mit dem Icon (**Anlegen**) legen Sie einen neuen Step an, und mit dem Icon (**Ändern**) können Sie einen bereits vorhandenen Step bearbeiten. Das Icon (**Anzeigen**) zeigt den Step an, und das Icon (**Löschen**) löscht den markierten Step.

Aus Transaktion SM37 (Jobübersicht) können Sie in Transaktion SM36 (Batch-Anforderung) verzweigen, um sich die Definition eines ausgeführten Jobs und dessen Steps noch einmal anzeigen zu lassen. Transaktion SM37 bietet eine Übersicht der im System vorhandenen Jobs und deren Status. Nach dem Rückverzweigen in Transaktion SM36 können Sie sich mit dem Button Spool die Einträge im *Spool* zum ausgeführten Step anschauen. Im Spool befinden sich temporäre Dateien, in denen Programmausgaben, die eigentlich für einen Drucker vorgesehen sind, gespeichert werden. Von der Systemadministration wird dieses Verfahren als Standardausgabe festgelegt. Diese temporären Dateien werden nach einer gewissen Aufbewahrungsfrist gelöscht.

Status »geplant«

Wenn Sie alle Steps zu Ihrem Job definiert haben, drücken Sie entweder die F3-Taste, oder Sie klicken auf das Icon (**Zurück**). Damit kehren Sie zum Ausgangsfenster Ihrer Jobdefinition zurück. Beachten Sie, dass der Status des Jobs immer noch wie in Abbildung 9.1 auf **geplant** steht.

Jobstartbedingung

Als Letztes legen Sie die Startbedingung für den Job fest. Klicken Sie dazu in Abbildung 9.1 auf den Button Startbedingung. Es öffnet sich ein Fenster, in dem Sie die Startbedingung des Jobs festlegen (siehe Abbildung 9.4).

Abbildung 9.4 Transaktion SM36 – Startbedingungen für einen Job

Für die Startbedingungen haben Sie die folgenden Auswahlmöglichkeiten:

- **Sofort** – Der Job wird nach der Freigabe sofort gestartet.
- **Datum/Uhrzeit** – Der Job wird zu einem festgelegten Zeitpunkt gestartet.
- **Nach Job** – Der Job wird nach der Beendigung eines anderen benannten Jobs gestartet.
- **Nach Ereignis** – Der Job wird nach dem Auslösen eines Events im System gestartet.
- **Bei Betriebsart** – Der Job wird mit dem Beginn einer Betriebsart, z. B. Normalbetrieb oder Nachtbetrieb, gestartet.

Für Sie als Einsteiger sollten nur die ersten drei Startbedingungen relevant sein. Klicken Sie im in Abbildung 9.4 gezeigten Fenster für die gewünschte Startbedingung auf den jeweiligen Button. Über den Button **Datum/Uhrzeit** geben Sie ein Datum und die Uhrzeit an, wann der Job starten soll. Über **Nach Job** geben Sie einen Job an, nach dessen Beendigung Ihr Job starten soll. Für diese Startbedingung geben wir weiter unten noch ein Beispiel. Wenn Sie Ihre Startbedingung festgelegt haben, klicken Sie auf (**Sichern**) zum Speichern der Startbedingung.

Status freigegeben

Ein Job erhält dann den Status **freigegeben**, wenn Sie den Job im Ausgangsbild von Transaktion SM36 (siehe Abbildung 9.1) speichern. Gleichzeitig wird der Job systemseitig eingeplant, d. h., er steht nun in der *Job-Queue* in einer Reihe mit anderen eingeplanten Jobs. Dies setzt voraus, dass Sie zuvor alle notwendigen Angaben vorgenommen haben. Der Status des Jobs ändert sich danach von **geplant** in **freigegeben**. Sie erhalten in der Statuszeile einen entsprechenden Hinweis (siehe Abbildung 9.5).

Abbildung 9.5 Transaktion SM36 – Hinweis in der Statuszeile

Job wird ein zweites Mal eingeplant

Beachten Sie, dass, wenn Sie den Job ein zweites Mal sichern – vielleicht, weil Sie sich nicht sicher sind, ob Sie es bereits getan hatten –, der Job ein zweites Mal eingeplant wird, mit den gleichen Bedingungen, die Sie bereits festgelegt hatten. Das führt möglicherweise zu unerwünschten Ergebnissen, vor allem, wenn die Startbedingung **Sofort** lautet. Sie können den Job in Transaktion SM36 (Batch-Anforderung) auch im Status **geplant** belassen und ihn später in Transaktion SM37 (Jobübersicht) mit Freigeben freigeben (siehe Abbildung 9.12).

9.2.2 Jobs über den Dialog generieren

Job im Dialog anlegen

Neben der Definition von Jobs über Transaktion SM36 (Batch-Anforderung) können Sie sich Jobs auch über den Dialog generieren lassen. Das ist natürlich nur bei den Programmen sinnvoll, die für eine Hintergrundverarbeitung möglich sind und die zu Programmbeginn Selektionskriterien für die eigene Programmsteuerung erwarten. Dabei handelt es sich um Standard-Dynpros, die im Dialog als normales Eingabefenster auf dem Bildschirm erscheinen. Wechseln Sie mit Transaktion SE38 in den ABAP Editor. Im Feld **Programm** geben Sie den Programmnamen ein. Wir bleiben bei dem Programm `AFX_CODE_SCANNER` (siehe Abbildung 9.6).

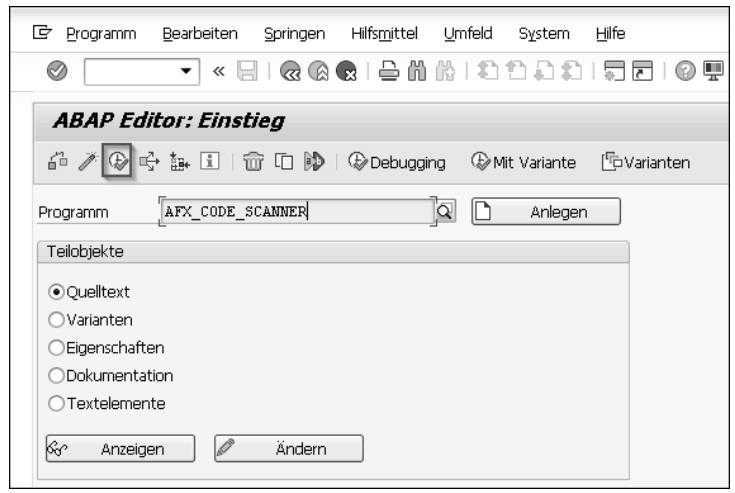


Abbildung 9.6 Transaktion SE38 – Programm starten

Starten Sie das Programm über das Icon (**Ausführen**). Es erscheint ein Standard-Dynpro, in dem Sie die Suchkriterien für das Programm AFX_CODE_SCANNER eingeben (siehe Abbildung 9.7). Nachdem Sie die Suchkriterien erfasst haben, drücken Sie F9 oder wählen den Menüpfad **Programm • Im Hintergrund ausführen**.

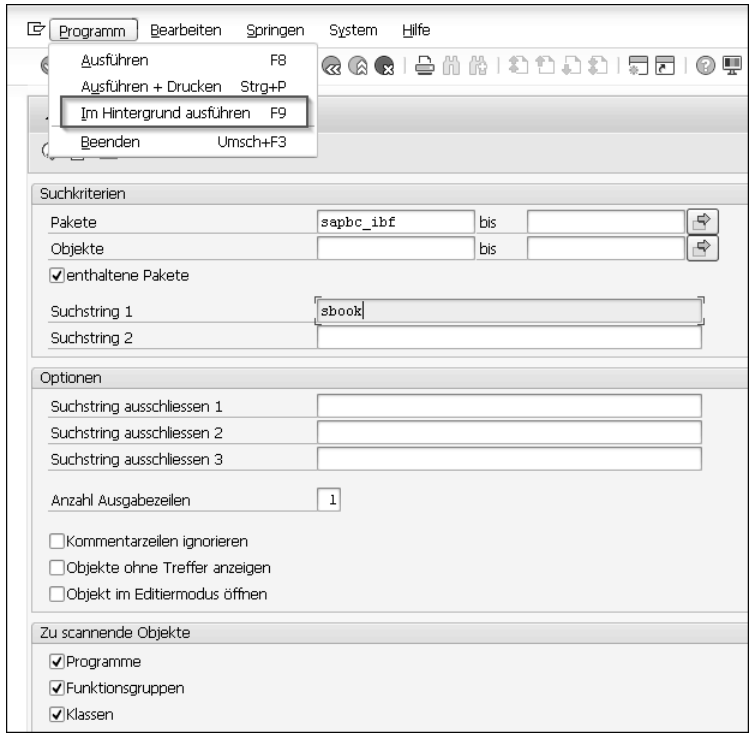


Abbildung 9.7 Transaktion SM38 – weitere Verarbeitung im Hintergrund ausführen

Standardausgabegerät

Für Programme, die im Dialog laufen, ist das Standardausgabegerät der Bildschirm. Bei einer Hintergrundverarbeitung entfällt diese Option. Das Standardausgabegerät ist dann entweder ein Drucker oder die Ausgabe in eine Spool-Datei. Letzteres ist in der Praxis der Standard. Dennoch erscheint nach dem Drücken von F9 die Abfrage nach dem Standardausgabegerät. Wählen Sie wie in Abbildung 9.8 im Punkt **Eigenschaften** die Ausgabe in den SAP-Spool.

Die weiteren Optionen wären **Sofort** oder **Datum und Uhrzeit**. Allerdings müssten Sie dann im Feld **Ausgabegerät** einen gültigen Drucker eintragen. Bestätigten Sie Ihre Auswahl mit (**Weiter**). Danach fragt das SAP-System sofort nach den Startbedingungen (siehe Abbildung 9.4). Das SAP-System verlangt in diesem Fall keine Stepdefinition. Bei einer Stepdefinition müssen Sie Angaben zum Programm und zur Variante machen. Beides haben Sie in Transaktion SE38 (ABAP Editor) bereits definiert. Sie haben ein Programm ausgewählt und die Selektion festgelegt, mit der das Programm arbeiten soll. Dem SAP-System sind also Programm und Variante bekannt, und es verwendet diese Information automatisch als Stepdefinition. Wenn Sie als Startbedingung **Sofort** gewählt haben, wird die Hintergrundverarbeitung sofort gestartet.

Abbildung 9.8 Ausgabegerät für Hintergrundverarbeitung auswählen

9.2.3 Startbedingung »Nach Job«

Startbedingung »Nach Job«

In der Praxis kommt es häufig vor, dass Jobs miteinander verkettet werden, nicht nur in der Produktivumgebung, sondern auch bei diversen Testläufen. Dabei müssen die verketteten Jobs nicht unbedingt voneinander abhängig sein. Wenn z. B. im Rahmen von Qualitätstests größere Jobläufe stattfinden, weiß man in der Regel nur die ungefähre Laufzeit eines Jobs. Wenn Sie mit dem Jobsteuerungsteam abgesprochen haben, dass Ihr Job B nach Beendigung von Job A starten darf, haben Sie ein Problem bei der Einplanung: Welche Startbedingung soll für Job B gewählt werden? Die Startbedingung **Sofort** kann nicht ausgewählt werden. Bei der Startbedingung gemäß **Datum/Uhrzeit** können Sie Glück haben oder auch nicht – vielleicht wäre Job A genau dann beendet, wenn kurz danach Job B starten soll, oder Job B startet zu früh, während Job A noch läuft, oder aber Job B startet zu spät, während Job A seit längerem beendet ist. Dann wäre wertvolle Zeit verschenkt. Um diesem Problem zu entgehen, gibt es die Startbedingung **Nach Job**.

Für diesen Zweck wählen Sie die Startbedingung **Nach Job** aus (siehe Abbildung 9.9). Der Job, nach dem unser Job starten soll, heißt Z_TEST_PGM_JOB_1. Diesen vorherigen Job tragen wir unter **Name** ein. Den Start unseres Jobs machen wir nicht vom Endestatus des vorherigen Jobs abhängig. Wenn Sie im Feld **Start statusabhängig** einen Haken setzen, startet Ihr Job nur dann,

wenn der vorherige Job erfolgreich beendet wurde. Bricht der vorherige Job ab, wird Ihr Job nicht gestartet. Kehren Sie mit F3 oder mit dem Icon (**Zurück**) zum Ausgangsbild von Transaktion SM36 (Batch-Anforderung) zurück, und sichern Sie den Job. Er erhält dadurch den Status **freigegeben**.

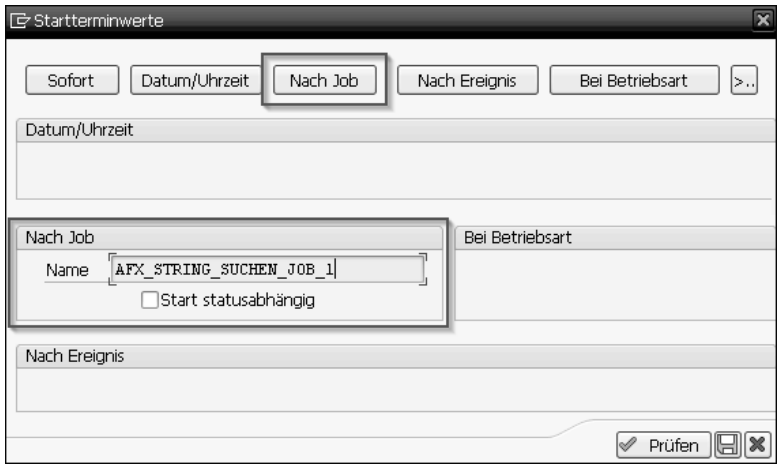


Abbildung 9.9 Transaktion SM36 – Startbedingung »Nach Job«

9.2.4 Jobübersicht in Transaktion SM37

Jobs auswählen

Wechseln Sie nun zu Transaktion SM37 (Jobübersicht). Als Erstes wählen Sie einen Job aus. Ihr **Benutzername** ist voreingestellt (siehe Abbildung 9.10).

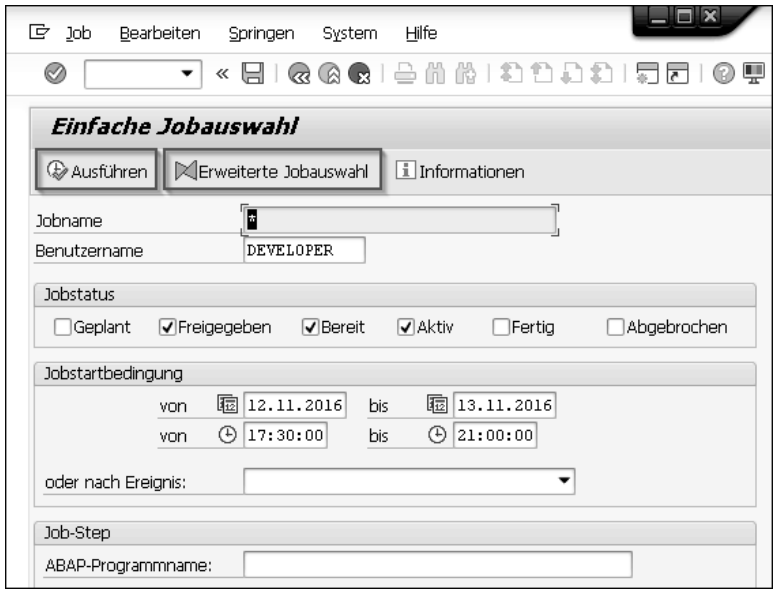


Abbildung 9.10 Transaktion SM37 – Jobauswahl

Im Feld **Jobname** sind Wildcards (*) erlaubt. Sie können die Auswahl mit den folgenden Eigenschaften weiter einschränken:

- Jobstatus
- Jobstartbedingung
- Job-Step

Im Bereich **Job-Step** können Sie festlegen, dass Sie sich nur für Steps interessieren, bei denen ein bestimmtes Programm gestartet wurde. Klicken Sie anschließend auf den Button Ausführen.

Jobübersicht

Als nächstes Bild erscheint die eigentliche Jobübersicht. Hier gibt es anwendungsbezogene Funktions-Icons, deren Bedeutung bekannt sein sollte (siehe Abbildung 9.11). Die einzelnen Funktionen finden Sie auch im Menüpunkt **Job** in Abbildung 9.12. Dieser Menüpunkt enthält andere Knoten als der Menüpunkt **Job** in Abbildung 9.10.

Abbildung 9.11 Jobübersicht – anwendungsbezogene Funktions-Icons

Die Bedeutungen der wichtigsten Icons haben wir in Tabelle 9.1 aufgelistet.

Icon	Tastenkombination	Bedeutung
Aktualisieren	F8	**Aktualisieren** der Anzeige. Dies bietet sich vor allem an, wenn Sie den Status eines Jobs oder das Laufzeitverhalten eines Jobs beobachten wollen.
Freigeben	Strg + ⇧ + F10	**Freigeben** eines Jobs, der sich im Status **geplant** befindet.
STOP	Strg + F1	**Abbrechen** eines aktiven Jobs. Das ist mitunter notwendig.
	⇧ + F2	**Löschen** eines Jobs. Der Job muss zuvor beendet oder abgebrochen worden sein.
Spool	Strg + ⇧ + F8	**Anzeigen** von Spool-Einträgen.
Job-Log	Strg + ⇧ + F11	**Job-Log anzeigen**

Tabelle 9.1 Jobübersicht – Bedeutung der wichtigsten anwendungsbezogenen Funktions-Icons

Icon	Tastenkombination	Bedeutung
Step	Strg + ⇧ + F9	**Verzweigen** in Transaktion SM36 und **Anzeigen** des Steplistenüberblicks.
Job-Details	F2	**Verzweigen** in das Ausgangsbild von Transaktion SM36.
AppServers	F7	**Verzweigen** in eine Übersicht der Applikationsserver. Sie können in Transaktion SM36 (Batch-Anforderung) festlegen, auf welchem Applikationsserver der Job laufen soll.

Tabelle 9.1 Jobübersicht – Bedeutung der wichtigsten anwendungsbezogenen Funktions-Icons (Forts.)

Wichtig ist, dass Sie einen Job in der Jobübersicht mit einem Haken markieren, bevor Sie auf eines dieser Icons klicken. Es darf nur ein Job markiert sein, weil das SAP-System sonst nicht weiß, welcher Job tatsächlich gemeint ist. Ausgenommen hiervon sind die Buttons Aktualisieren und AppServers. Die hinter dem Button AppServers stehenden Buttons dienen dem Layout der Bildschirmanzeige. Probieren Sie es aus!

Die Anzahl der angezeigten Jobs hängt von Ihrer Jobauswahl ab (siehe Abbildung 9.12). Zunächst werden Sie Ihren relevanten Job unter den aufgelisteten Jobs suchen.

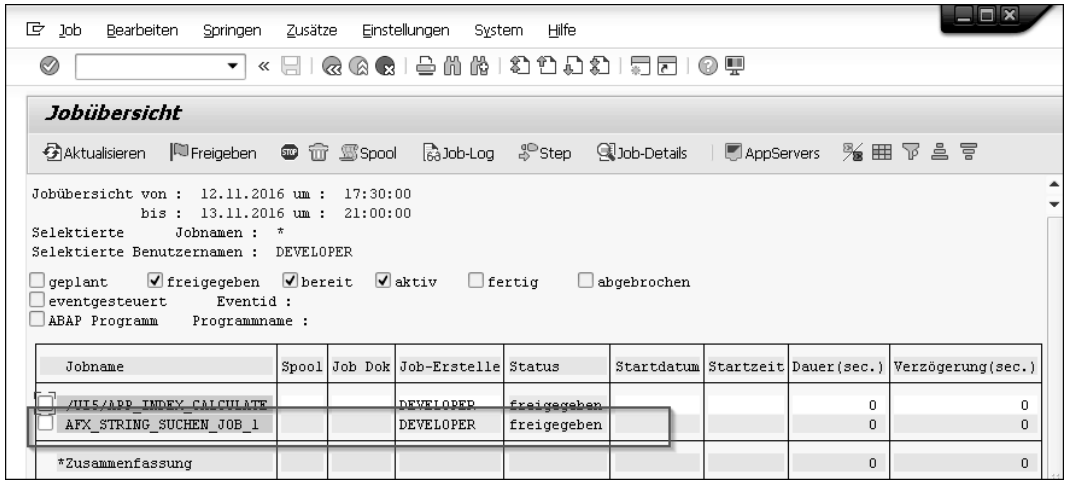


Abbildung 9.12 Transaktion SM37 – gefilterte Jobübersicht

In Abbildung 9.12 ist dies einfach, denn es ist der zweite Job. Bei sehr vielen aufgelisteten Jobs kann die Suche allerdings mühselig werden, sodass Sie möglicherweise weitere Einschränkungen vornehmen müssen. Unser Beispieljob befindet sich im Status **freigegeben**. Dies bedeutet, dass er gestartet werden kann.

Ein Job kann die in Tabelle 9.2 aufgelisteten Status annehmen. **Jobstatusübersicht**

Status	Beschreibung
geplant	Der Job wurde dem Jobverarbeitungssystem übergeben, er ist in diesem Status allerdings noch nicht ausführbar. Dazu muss er freigegeben werden. Wenn alle Startbedingungen für einen Job erfüllt sind, kann er durch den Button Freigeben (siehe Abbildung 9.12) freigegeben werden.
freigegeben	Der Job ist eingeplant und bereit, gestartet zu werden, die Startbedingungen sind erfüllt. Wann genau der Job gestartet wird, entscheidet der *Job-Scheduler*, das Job-Verwaltungsprogramm.
bereit	Der Job wurde in die Job-Queue eingestellt, konnte allerdings noch nicht gestartet werden, weil ein dafür notwendiger systeminterner Workprozess noch nicht frei ist.
aktiv	Job läuft.
fertig	Job ist beendet.
abgebrochen	Job wurde durch einen Verarbeitungsfehler oder willentlich abgebrochen.

Tabelle 9.2 Status, die ein Job annehmen kann

In Abbildung 9.12 sehen Sie nur den ersten der beiden verketteten Jobs, AFX_STRING_SUCHEN_JOB_1, obwohl wir bei der Jobauswahl in Abbildung 9.10 im Feld **Jobname** keine Einschränkung vorgenommen haben. Um auch den zweiten Job sehen zu können, tragen Sie dort im Feld **Oder nach Ereignis** eine Wildcard (*) ein.

Alternativ können Sie auch auf den Button Erweiterte Jobauswahl klicken. Im Bild der Jobauswahl klappt im unteren Bereich ein Subfenster mit fünf Registerkarten auf. Tragen Sie auf der ersten Registerkarte **Startbedingung** im Feld **ODER Start nach Job** eine Wildcard (*) ein (siehe Abbildung 9.13).

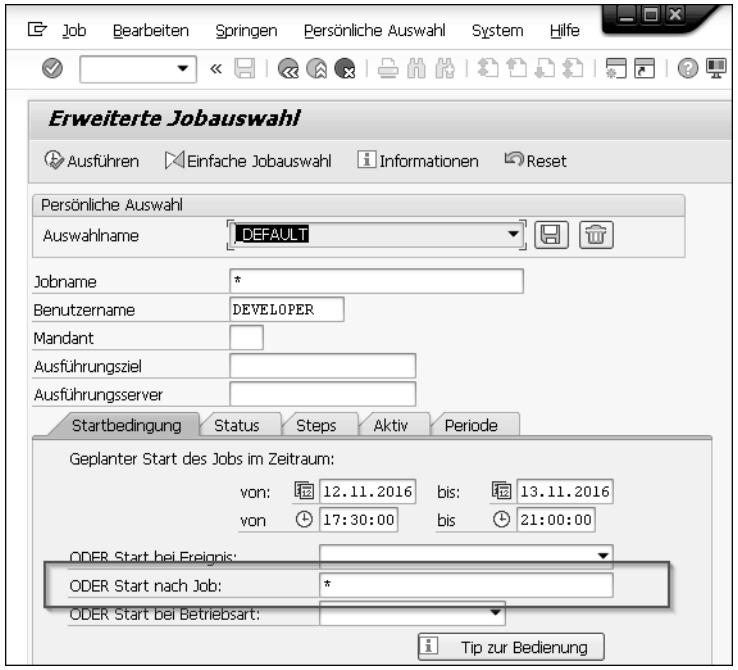


Abbildung 9.13 Transaktion SM37 – erweiterte Jobauswahl

Klicken Sie anschließend auf den Button Ausführen. In der Jobübersicht ist nun auch der zweite abhängige Job zu sehen (siehe Abbildung 9.14).

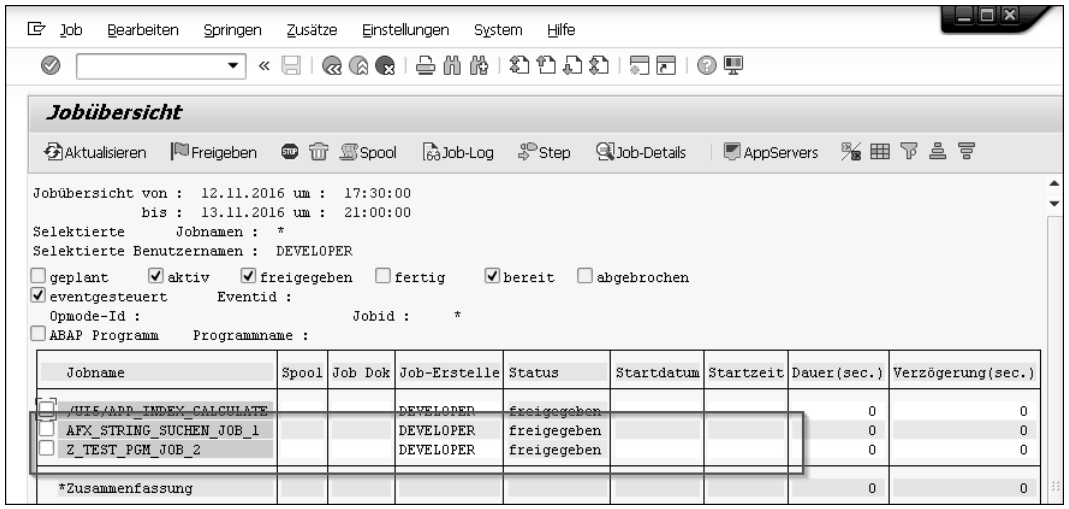


Abbildung 9.14 Jobübersicht – Vorgänger- und Nachfolgerjob sichtbar

Vorgänger- und Nachfolgerjob

Markieren Sie nun den ersten Job, indem Sie vor dem ersten Job im kleinen Kästchen einen Haken setzen, und klicken Sie auf den Button Job-Details,

oder führen Sie auf dem ersten Job einen Doppelklick aus. Sie verzweigen anschließend in das Ausgangsbild von Transaktion SM36 (Batch-Anforderung), siehe Abbildung 9.15.

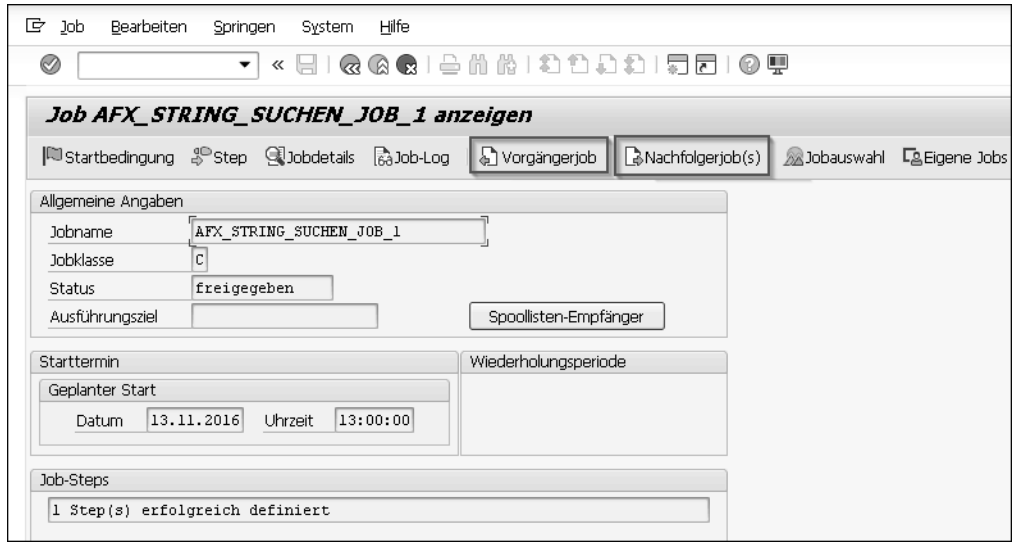


Abbildung 9.15 Transaktion SM36 – Button für die Anzeige von Vorgängerjob und Nachfolgerjob

Klicken Sie in diesem Bild auf den Button Nachfolgerjob(s). Es wird Ihnen danach der Nachfolgerjob `Z_TEST_PGM_JOB_2` zum ersten Job angezeigt (siehe Abbildung 9.16).

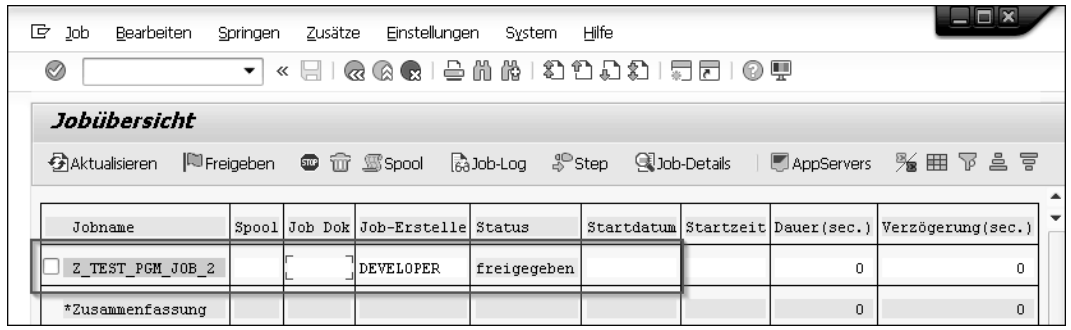


Abbildung 9.16 Anzeige des Nachfolgerjobs zum Job AFX_STRING_SUCHEN_JOB_1

Genauso können Sie auch mit dem zweiten Job verfahren, um sich den Jobvorgänger anzeigen zu lassen. In Transaktion SM36 (Batch-Anforderung) klicken Sie dann auf den Button Vorgängerjob. Als Ergebnis wird Ihnen der Job angezeigt, von dem der zweite Job mit seiner Startbedingung abhängig ist.

Auf diese Weise können Sie im Nachhinein die Reihenfolge und Abhängigkeiten von verketteten Jobs überprüfen.

Variante überprüfen

Auf einem ähnlichen Weg lässt sich auch nachträglich überprüfen, mit welcher Variante ein Job ausgeführt wurde. Markieren Sie in unserem Beispiel in Transaktion SM37 (Jobübersicht) den ersten Job (der zweite Job hat keine Variante), und klicken Sie dann auf den Button Step. Sie verzweigen danach in Transaktion SM36 und dort in die Stepübersicht des ersten Jobs. Setzen Sie den Mauszeiger auf den ersten Step, um ihn zu markieren (hier muss kein Haken gesetzt werden), und rufen Sie den Menüpfad **Springen • Variante** auf.

Ihnen wird anschließend die gültige Variante zum Step mit den zur Laufzeit eingetragenen Werten angezeigt. Diese Überprüfung ist immer dann wertvoll, wenn z. B. im produktiven Betrieb ein Verarbeitungsjob ordnungsgemäß durchgelaufen ist, der Fachbereich hinterher aber reklamiert, dass der Job verkehrte Ergebnisse geliefert hätte. Möglicherweise liegt dies daran, dass die Variante nicht korrekt war.

Spool-Ausgabe

Wenn in Transaktion SM37 (Jobübersicht) bei einem Job in der Spalte **Spool** das Symbol vorhanden ist, hat der Job eine Ausgabe in eine Spool-Datei erzeugt (siehe Abbildung 9.17).

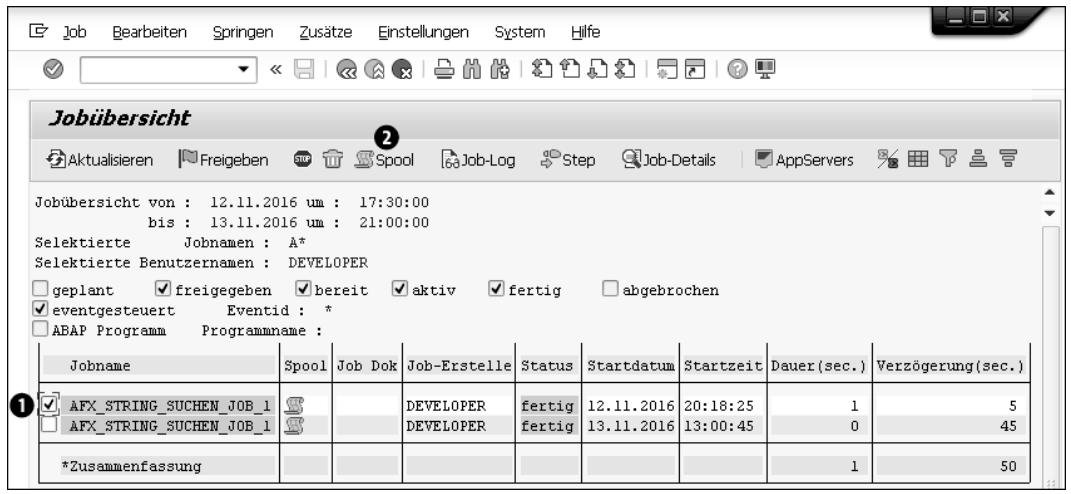


Abbildung 9.17 Job mit einer Ausgabe in die Spool-Datei

Markieren Sie diesen Job ❶, und klicken Sie dann auf den Button Spool ❷. Ihnen wird eine Übersicht der Spool-Einträge des Jobs angezeigt. Wenn sich ein Job aus mehreren Steps zusammensetzt und jeder Step eine eigene Ausgabe erzeugt, würden hier mehrere separate Spool-Einträge angezeigt. In unserem Beispiel ist es nur ein Spool-Eintrag (siehe Abbildung 9.18).

Abbildung 9.18 Übersicht über die erzeugten Spool-Ausgaben eines Jobs

Um sich die tatsächliche Ausgabe anzuschauen, klicken Sie auf das Symbol in der Spalte **Ty** in Abbildung 9.18. Das Suchergebnis des Programms AFX_CODE_SCANNER wird angezeigt (siehe Abbildung 9.19).

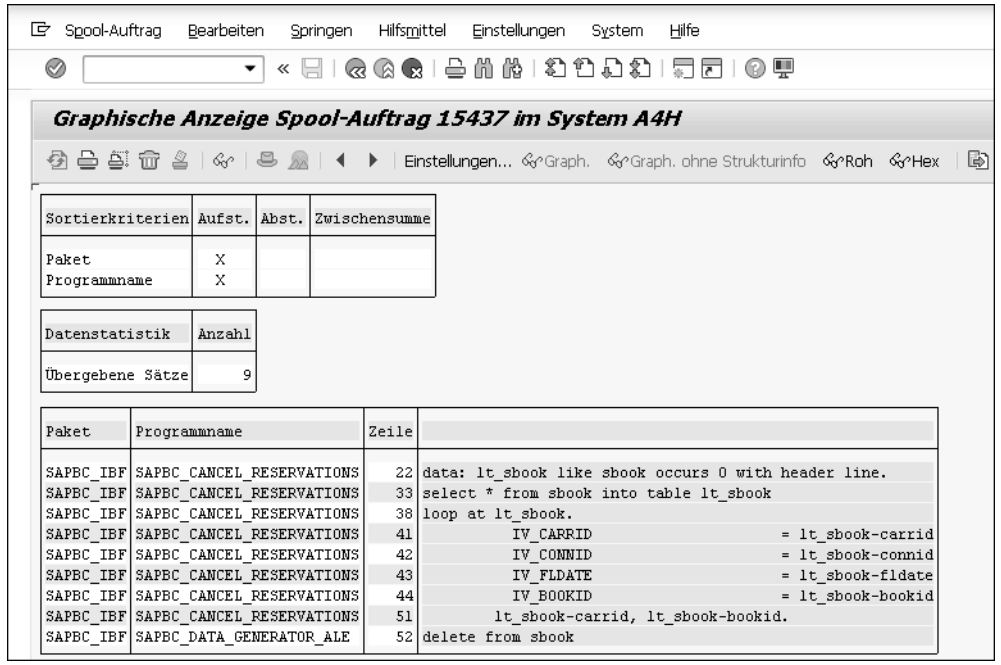


Abbildung 9.19 Ergebnisanzeige des Jobs aus der Spool-Datei

Jobverwaltungstabellen

Wie in einem SAP-System üblich, werden alle Informationen in Datenbanktabellen gespeichert, so auch bei den Jobs. Wechseln Sie in Transaktion SE11 (Pflege ABAP Dictionary) oder SE16 (Data Browser), und lassen Sie sich die Tabellen auflisten, die mit TBTC* beginnen. Hier finden Sie die Jobverwaltungstabellen. In zwei von diesen Tabellen sind die Jobs und ihre Steps eingetragen:

- TBTCO – Zustandsübersicht über alle im System vorhandenen Jobs und deren Status
- TBTCP – Übersicht über die Steps zu den Jobs

Vielleicht schreiben Sie einmal ein eigenes Jobverwaltungsprogramm. Diese Tabellen bieten Ihnen die notwendigen Informationen.

9.3 Variante erzeugen

In den vorangegangenen Seiten haben wir bereits vielfach auf Varianten hingewiesen. Daran erkennen Sie bereits, welche Bedeutung Varianten haben. Mit einer Variante werden einem Programm Vorgaben für die Batch-Verarbeitung mitgegeben. Dies können z. B. Werte für die Datenselektion oder Parameter für die Steuerung innerhalb eines Programms sein. Letzten Endes hängt es davon ab, was ein Programm leisten soll und inwieweit ein Programmierer eine Parametrisierung des Programms vorgesehen hat.

Selektionsbild

In der ABAP-Programmierung werden solche Vorgaben im Rahmen von *Selektionsbildern* umgesetzt. Selektionsbilder werden zu Beginn eines Programms aufgerufen, sodass einem Programm vor seiner eigentlichen Abarbeitung mitgeteilt werden kann, welche Daten zu selektieren sind, und unter Umständen, wie es sich bei bestimmten Bedingungen verhalten soll. Ein typisches Beispiel wäre ein Testschalter. Ist er gesetzt, soll das Programm keine Daten in die Datenbank schreiben.

Variante

Ein Selektionsbild allein macht noch keine Variante aus. Erst die Zuordnung von Werten für die Datenselektion oder für die vorgesehenen Parameter ergibt im Zusammenhang mit einem Selektionsbild eine Variante. Der Begriff *Variante* beschreibt, in welcher Variation oder in welcher Art und Weise ein Programm ausgeführt werden soll. Es kann z. B. die Datenselektion variieren, oder einmal ist ein Testschalter gesetzt und einmal nicht. Diese verschiedenen Variationen oder Einstellungen können für ein Programm gespeichert werden, und jede einzelne Variation wird Variante genannt. Varianten erhalten einen Namen und sind an ein Programm gekoppelt.

Wir haben in Abschnitt 9.2, »Programme im Hintergrund starten«, bereits mit einer Variante gearbeitet. Das dort verwendete Programm AFX_CODE_SCANNER hat ein sehr umfangreiches Selektionsbild. Für eine einfache Suchaufforderung hatten wir die Variante AFX_VAR1 angelegt und verwendet.

Variante anlegen

Um eine Variante zu erzeugen, wechseln Sie in den ABAP Editor (Transaktion SE38). Im Feld **Programm** tragen Sie den Namen des Programms ein,

für das eine Variante angelegt werden soll. Es gibt zwei Wege, um eine Variante anzulegen. Für den ersten Weg starten Sie das Programm über das Icon (**Ausführen**). Es erscheint das Selektionsbild zu diesem Programm (siehe Abbildung 9.20). Wir bleiben beim Beispiel AFX_CODE_SCANNER.

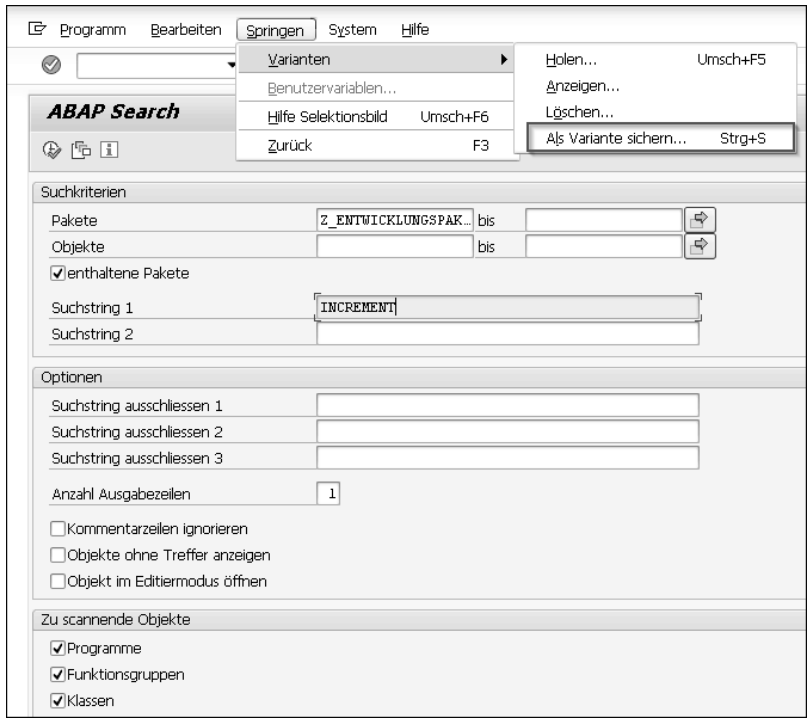


Abbildung 9.20 Transaktion SE38 – Selektion als Variante sichern

Variantenattribute

Geben Sie eine neue Suchbedingung ein. Klicken Sie anschließend auf (**Sichern**) zum Speichern der Eingaben, oder wählen Sie den Menüpfad **Springen • Varianten • Als Variante sichern** (siehe Abbildung 9.20). Das Ergebnis ist dasselbe: Es erscheint ein Bild zur Pflege der **Variantenattribute** (siehe Abbildung 9.21).

Vergeben Sie im Feld **Variantenname** einen Namen für die Variante. Im Feld **Bedeutung** tragen Sie einen erläuternden Text ein. Im Bildbereich **Objekte des Selektionsbildes** sind alle Selektionsfelder (**Typ**: **S**) und Parameter (**Typ**: **P**) des Selektionsbildes aufgelistet. **Typ S** steht für »Selektion«. Das bedeutet, dass Sie bei einem Feld einen Bereich von/bis eingeben können. **Typ P** steht für »Parameter«. In einem Parameter können Sie nur einen Wert angeben.

Des Weiteren können Sie nur für diese Variante die Eigenschaften der einzelnen Felder verändern, indem Sie in den jeweiligen Spalten und pro Feld einen Haken setzen oder nicht.

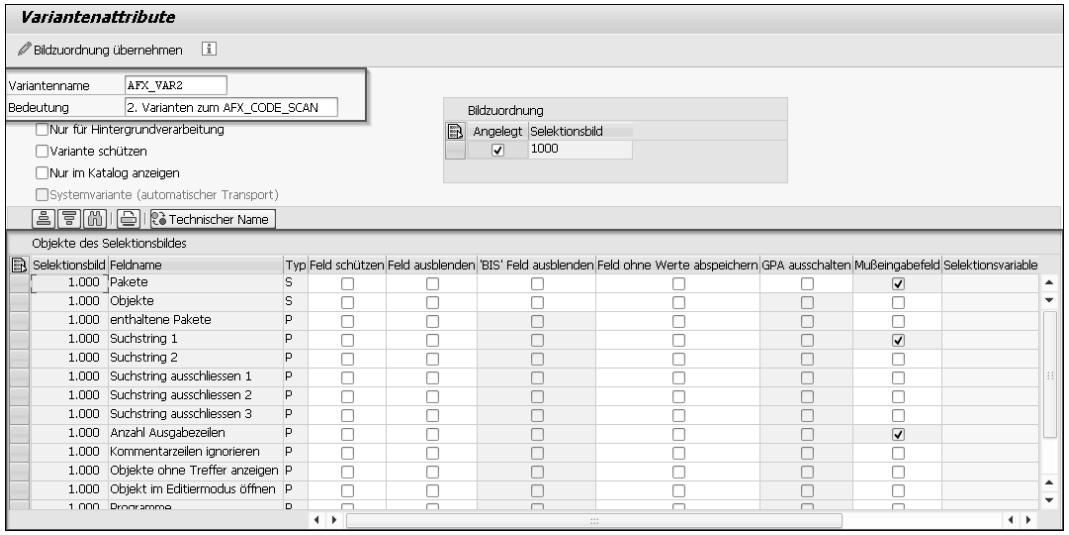


Abbildung 9.21 Bild zur Erfassung von Variantenattributen

Die Voreinstellung entspricht der Deklaration im Programm. Speichern Sie Ihre Eingaben. Das SAP-System wechselt zurück zur Selektionsmaske des Programms AFX_CODE_SCANNER. In der Statuszeile erscheint der Hinweis, dass die Variante gesichert wurde (siehe Abbildung 9.22). Diese Variante können Sie nun in einem Jobstep einsetzen.

Abbildung 9.22 Transaktion SE38 – Hinweis

Der zweite Weg, um eine Variante anzulegen, führt zurück zum Einstiegsbild von Transaktion SE38. Geben Sie den Namen des Programms ein, und klicken Sie auf den Button Varianten (siehe Abbildung 9.23).

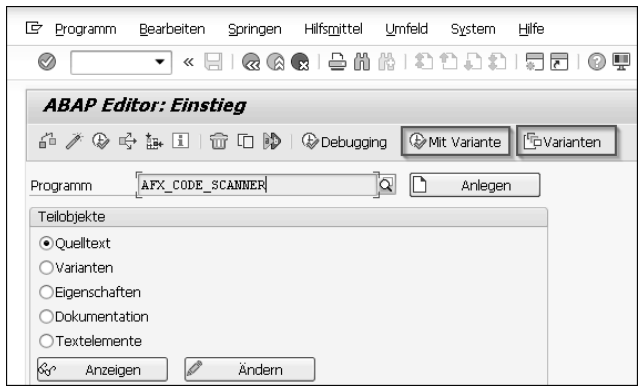


Abbildung 9.23 Transaktion SE38 – Einstiegsbild

Es erscheint ein neues Fenster, in dem Sie den Namen der neuen Variante eingeben. Klicken Sie auf den Button Anlegen (siehe Abbildung 9.24).

Abbildung 9.24 Transaktion SE38 – Variante anlegen

Es erscheint das Selektionsbild des Programms AFX_CODE_SCANNER. Das weitere Vorgehen deckt sich mit den oben beschriebenen Schritten, allerdings mit einem kleinen Unterschied: Da das SAP-System weiß, dass Sie eine Variante anlegen möchten, erscheint diesmal im Selektionsbild oben der Button Attribute. Wenn Sie darauf klicken, gelangen Sie zur Pflege der Variantenattribute.

Vorhandene Varianten können Sie anzeigen oder ändern, indem Sie in Abbildung 9.24 auf die entsprechenden Buttons klicken. Befinden Sie sich bereits im Selektionsbild, können Sie in der anwendungsbezogenen Icon-Leiste auf das Icon klicken, um eine Variante auszuwählen.

Variantenwerte ändern

Den Wertinhalt einer Variante können Sie nachträglich ändern. Bedenken Sie dabei aber, dass andere Kollegen möglicherweise die gleichen Varianten benutzen. Sie sollten daher Änderungen an Varianten, die von anderen ebenfalls genutzt werden, bekannt geben. Das gilt erst recht für den produktiven Betrieb. Varianten haben in der Produktion eine hohe Bedeutung, weil darüber letztendlich die inhaltliche Verarbeitung festgelegt wird.

9.4 Tabelle TVARVC

In vielen Varianten gibt es immer die gleichen Parameter. Daraus kann man schließen, dass diese Parameter eine übergreifende Bedeutung haben. Dazu gehört z. B. der Verarbeitungsstichtag. Um sicherzustellen, dass alle Anwendungen einer Verarbeitung denselben Stichtag verwenden, gibt es die Möglichkeit, diesen Parameter zentral zu verwalten. Dies geschieht über Tabelle TVARVC.

Transaktion STVARV

Um den Aufbau von Tabelle TVARVC zu verstehen, starten Sie am besten die Pflegetransaktion STVARV (siehe Abbildung 9.25).

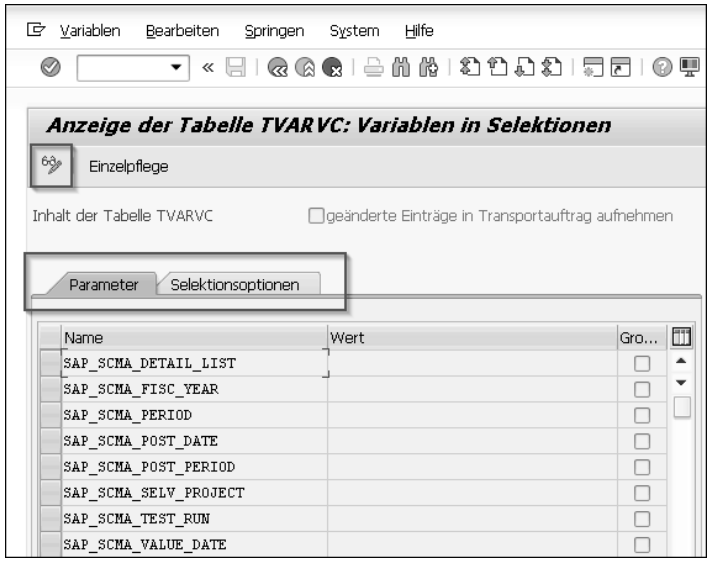


Abbildung 9.25 Transaktion STVARV – Einstiegsbild

In dieser Transaktion können Sie sowohl Parameter als auch Selektionsoptionen pflegen. Um einen neuen Parameter einzutragen, wechseln Sie auf die Registerkarte **Parameter** und klicken in der anwendungsbezogenen Icon-Leiste auf das Icon (**Anzeigen/Ändern**). Die anwendungsbezogene Icon-Leiste wird dadurch erweitert (siehe Abbildung 9.26).

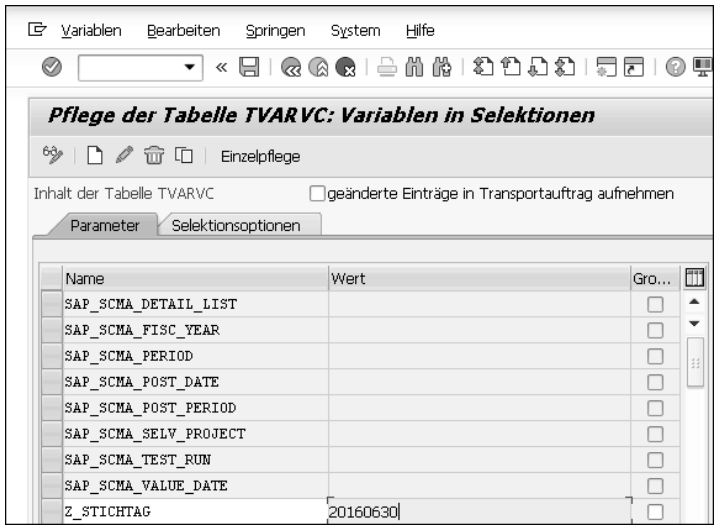


Abbildung 9.26 Transaktion STVARV – neuen Parameter eintragen

Klicken Sie auf das weiße Blatt (**Anlegen**) für einen Neueintrag. Die nächste freie Zeile unter den bereits vorhandenen Einträgen wird für eine Eingabe freigegeben. Tragen Sie in der Spalte **Name** den Parameternamen und in der Spalte **Wert** den Parameterwert ein.

Der Name des Parameters muss nicht mit dem Parameternamen in der Variante übereinstimmen. Wenn Sie als Parameterwert ein Datum eingeben, muss dies im Format JJJJMMTT geschehen (Jahr, Monat, Tag – ohne einen trennenden Punkt), ansonsten wird das Datum später in der Variante nicht korrekt angezeigt. Sichern Sie Ihre Eingaben.

Um eine Selektionsoption zu erfassen, wechseln Sie nun auf die Registerkarte **Selektionsoptionen**. Die expandierte anwendungsbezogene Icon-Leiste ist bereits vorhanden. Klicken Sie wiederum auf das Icon (**Anlegen**), ❶ in Abbildung 9.27, für einen Neueintrag. Wie bei den Parametern wird die nächste freie Zeile unterhalb der bereits vorhandenen Einträge zur Eingabe freigegeben.

Abbildung 9.27 Transaktion STVARV – neue Selektionsoption erfassen

Selektionsoption

Als Erstes vergeben Sie in der Spalte **Name** einen Namen für die Selektionsoption. In der Spalte **Option** ❷ wählen Sie die Parameter für die Selektionsoption aus. Mit den dort möglichen Operatoren können Sie z. B. Folgendes ausdrücken:

- ein Intervall abdecken
- nur einen Einzelwert oder mehrere Einzelwerte zulassen
- einen Einzelwert oder mehrere Einzelwerte ausschließen
- Werte sollen größer/gleich oder kleiner/gleich einem Referenzwert sein
- Werte sollen einem bestimmten Muster entsprechen

Bei einem Intervall geben Sie die untere und obere Abgrenzung an. Für die folgenden Operatoren geben Sie den Referenzwert in der Spalte **Untere Grenze** an:

- Einzelwert
- Größer oder gleich
- Größer
- Kleiner oder gleich
- Kleiner
- Ungleich

Wenn Sie mehrere Einzelwerte ein- oder ausschließen wollen, klicken Sie in der Spalte **Mehrfachselektion** ❸ auf den Button . Es öffnet sich ein Selektionsbild, in dem Sie auf der Registerkarte **Einzelwerte selektieren** mehrere Einzelwerte eintragen können (siehe Abbildung 9.28).

Abbildung 9.28 Transaktion STVARV – Einzelwerte eintragen

Bei den folgenden Operatoren geben Sie in der Spalte **Untere Grenze** ein Referenzmuster an, das entweder zutreffen oder ausgeschlossen werden soll:

- Muster
- Muster ausschließen

Ein Muster haben Sie immer dann, wenn Sie Wildcards (*) benutzen. In den verbleibenden Spalten in Abbildung 9.27 können Sie die Bedingung **Groß-/Kleinschreibung** einschließen und die Intervallprüfung ein- oder ausschalten.

Variante mit TVARVC verknüpfen

Der Eintrag in die Tabelle TVARVC für sich allein bewirkt noch nichts; Sie müssen außerdem eine Verknüpfung zwischen einer Variante und dem Tabelleneintrag herstellen. Wir nutzen dazu ein kleines Beispielprogramm, das die Bestellungen aus Kapitel 5, »ABAP-Dictionary-Objekte«, gemäß einer Datenselektion auflisten soll. Dazu verwenden wir eine Variante, in der der Stichtag für die Datenselektion festgelegt wird. Den Stichtag haben wir in Tabelle TVARVC unter dem Parameter Z_STICHTAG eingetragen.

Zu dem Beispielprogramm legen wir eine neue Variante an. Dazu rufen Sie das Programm über Transaktion SE38 (ABAP Editor) auf und klicken auf den Button Attribute (siehe Abbildung 9.29).

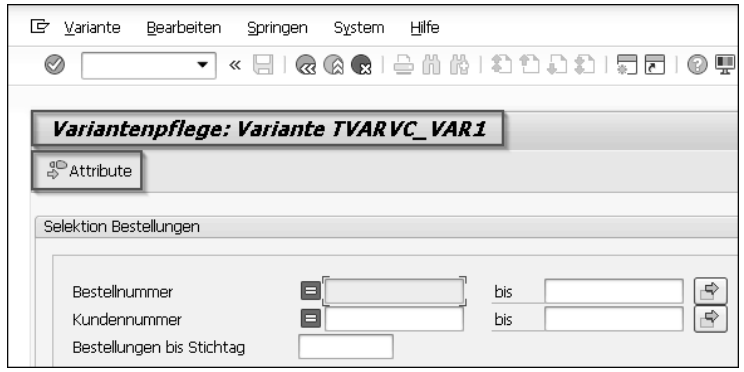


Abbildung 9.29 Transaktion SE38 – Variantenpflege zu einem Programm

Es öffnet sich die Pflege der Variantenattribute. Der Stichtag zu den Bestellungen bezieht sich auf den Parameter P_DATE aus dem Beispielprogramm. In Abbildung 9.30 ist dies die letzte Zeile.

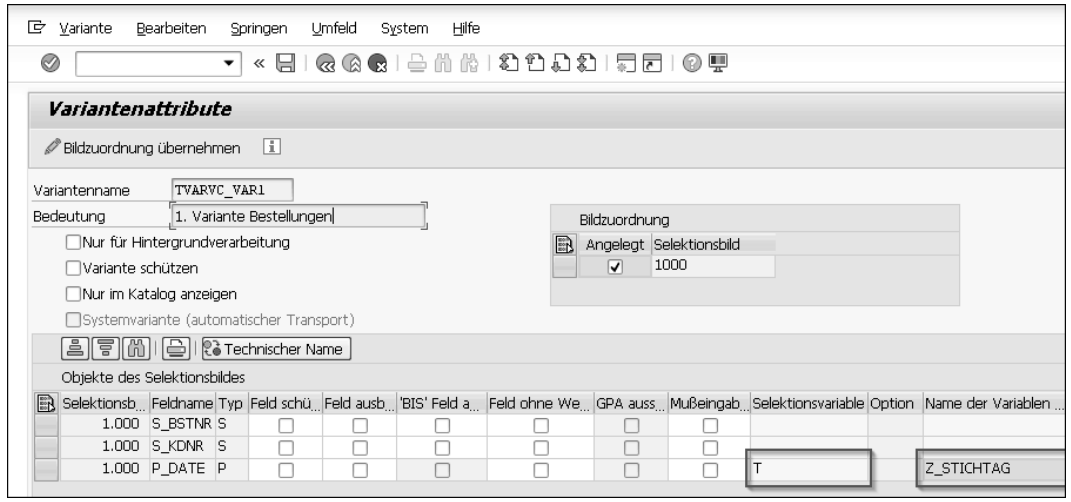


Abbildung 9.30 Pflege der Variantenattribute

Es gibt in der Zeile zwei Spalten, die für die Verknüpfung mit der Tabelle TVARVC von Bedeutung sind:

- Selektionsvariable
- Name der Variablen (Eingabe nur per F4)

Wenn Sie zuerst in die Spalte **Selektionsvariable** der Zeile zum Parameter P_DATE klicken, können Sie mit [⧉] ein Auswahlfenster öffnen (siehe Abbildung 9.31).

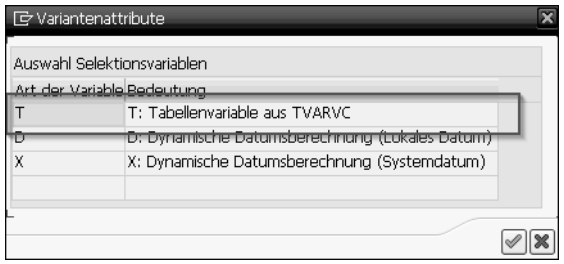


Abbildung 9.31 Auswahlfenster zur Typbestimmung der Selektionsvariablen

Auswahlwert T

Um den Bezug zur Tabelle TVARVC herzustellen, wählen Sie aus diesem Fenster den ersten Eintrag **T** aus. Danach wechseln Sie zur Spalte **Name der Variablen (Eingabe nur per F4)**. Auch hier gibt es ein Auswahlfenster, in dem die Parameter aus Tabelle TVARVC aufgelistet sind (siehe Abbildung 9.32).

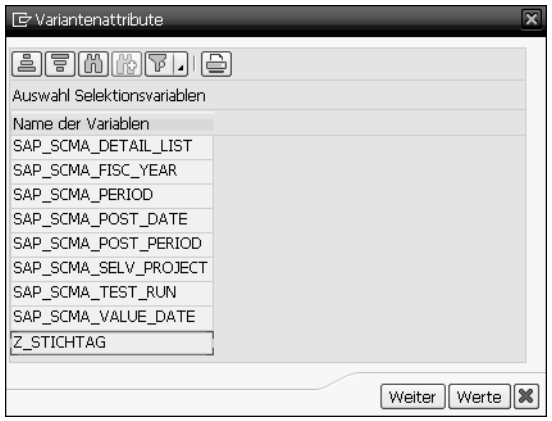


Abbildung 9.32 Parameterauswahl aus Tabelle TVARVC

Der zuvor über Transaktion STVARV (Pflege Selektionsvariablen) angelegte Parameter Z_STICHTAG wird mitaufgeführt. Durch die Auswahl dieses Parameters wird die Verknüpfung zwischen dem Parameter P_DATE aus dem Programm und dem Parameter Z_STICHTAG aus Tabelle TVARVC hergestellt.

Nach dem Sichern der Eingaben wechseln Sie wieder zurück in das Selektionsbild der neuen Variante. Sie sehen nun, dass der Wert des Parameters aus Tabelle TVARVC im Selektionsbild als Vorgabewert angezeigt wird und fester Bestandteil dieser Variante ist (siehe Abbildung 9.33).

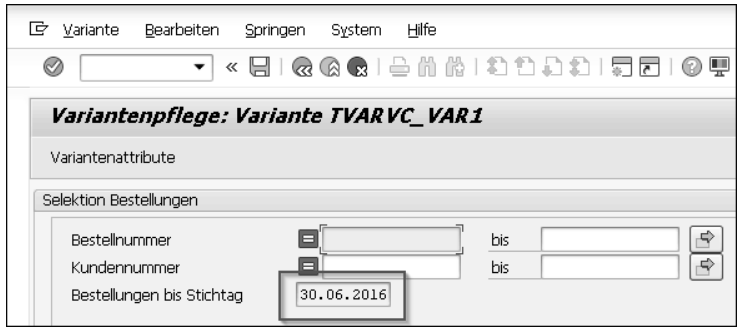


Abbildung 9.33 Variante mit Auswahlwert aus Tabelle TVARVC

Eine voreingestellte Mengenoperation, wie es bei den Selektionsparametern möglich ist, gibt es bei den Einzelparametern nicht. Die Abfrage von kleiner/gleich einem Stichtag müssen Sie im Programm selbst codieren. Nachdem Sie die Variante gesichert haben, können Sie sie beim Einplanen eines Jobs verwenden (siehe Abbildung 9.34).

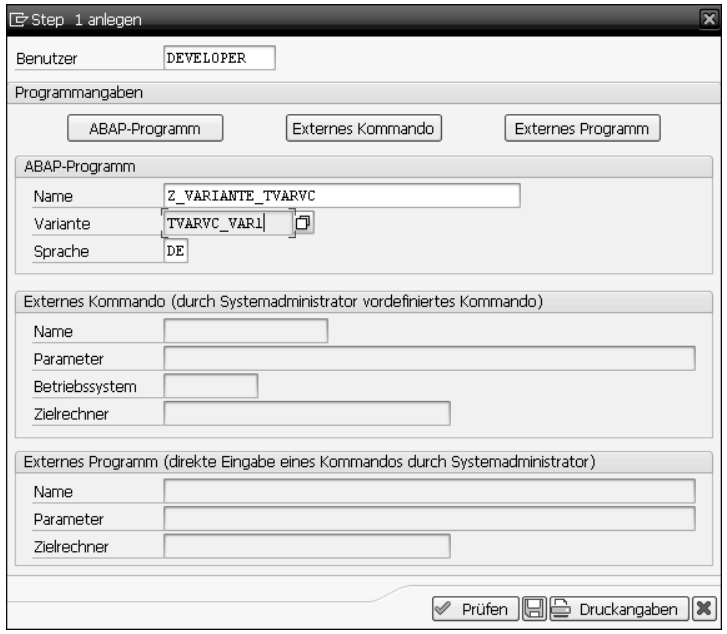


Abbildung 9.34 Transaktion SM36 – Festlegung des Beispielprogramms mit einer neuen Variante der Stepdefinition

Beim Umgang mit Tabelle TVARVC sollten Sie sorgsam sein, denn Sie haben keinen exklusiven Zugriff auf diese Tabelle. Es greifen sehr viele Anwendungen auf diese Tabelle zu. Bevor Sie bei einem bereits vorhandenen Parameter oder bei einer Selektionsoption Werte ändern, sollten Sie sich vergewissern, dass dies keine Auswirkungen auf andere Anwendungen hat.

9.5 Programme, Funktionsbausteine und Methoden testen

Entwicklertest

Bevor Programme, Funktionsbausteine oder Klassen mit Methoden für den fachlichen Qualitätstest und später für den produktiven Betrieb übergeben werden, sollten sie vom Entwickler getestet werden. Dieses Vorgehen wird *Entwicklertest* genannt.

Ein ausführbares Programm zu testen, ist von der Vorstellung her noch einfach, denn ein ausführbares Programm sollte für sich allein aufgerufen werden können, sonst wurde ein grundsätzlicher Fehler gemacht. Funktionsbausteine und Methoden werden allerdings zur Ausführung aufgerufen und sind somit nicht ohne weiteres für sich allein ausführbar. Dennoch können Funktionsbausteine und Methoden auf eine ähnliche Weise getestet werden wie ausführbare Programme.

Testen des Programms

In der Entwicklungsumgebung dieser Repository-Objekte – Transaktionen SE80 (Object Navigator), SE38 (ABAP Editor), SE37 (Jobübersicht) und SE24 (Class Builder) – gibt es das Icon (**Testen**). Hierüber können Programme wie auch Funktionsbausteine und Methoden für einen Testlauf aufgerufen werden. Alternativ können Sie auch F8 drücken. Ausführbare Programme haben in den allermeisten Fällen ein Selektionsbild, in dem Sie verschiedene Testläufe anhand unterschiedlicher Datenselektionen oder Parametereinstellungen durchführen können. So ein Testlauf unterscheidet sich vom Prinzip her nicht wesentlich von einem späteren produktiven Einsatz.

Wenn Sie Funktionsbausteine und Methoden über das Icon testen (siehe Abbildung 9.35), simuliert das SAP-System den Aufruf eines Funktionsbausteins oder einer Methode.

Beide Entwicklungsobjekte verfügen über eine Parameterleiste mit Eingabe- und Ausgabeparametern. Wenn Sie also einen Funktionsbaustein oder eine Methode testen wollen, sollten Sie beiden Objekten Eingabewerte übergeben, weil sonst ein funktionaler Test unzulänglich wäre. Genauso möchten Sie die Rückgabe von Ergebnissen über die Ausgabeparameter kontrollieren.

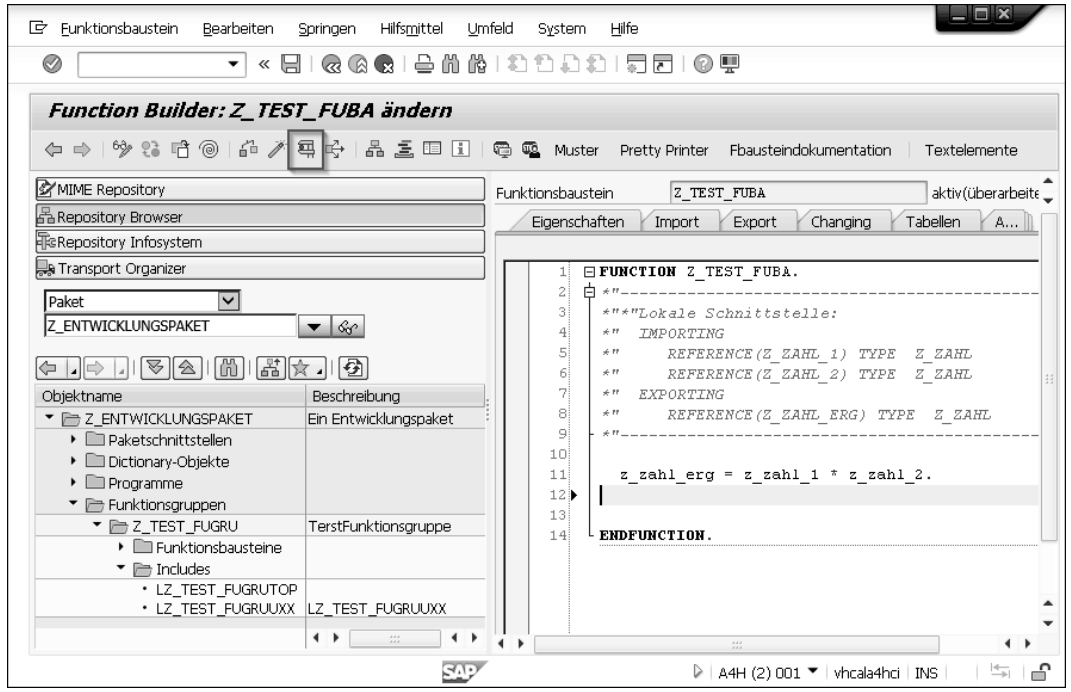


Abbildung 9.35 Testaufruf eines Funktionsbausteins

Funktionsbausteintest

Die Entwicklungsumgebung simuliert den Aufruf, weiß aber selbst nicht, welche Parameterwerte beim Aufruf gelten sollen. Zu diesem Zweck generiert das SAP-System ein einfaches Eingabebild, in dem Sie die Werte für die Eingabeparameter eingeben. Abbildung 9.36 zeigt dieses Eingabebild für Funktionsbausteine.

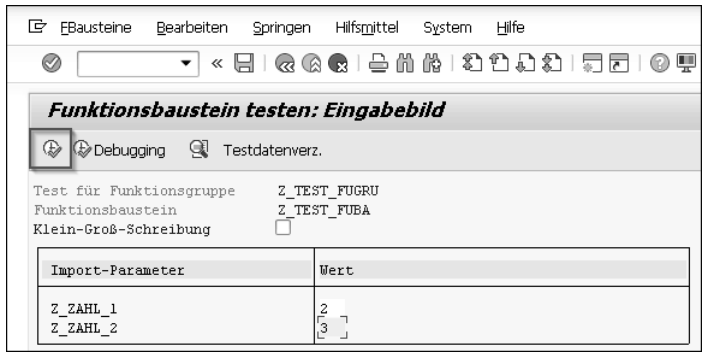


Abbildung 9.36 Generierte Eingabemaske für Eingabeparameter

Anhand der Definition des Funktionsbausteins weiß das SAP-System, welche Eingabe- und Ausgabeparameter für den Funktionsbaustein festgelegt

wurden und welche Parameter möglicherweise änderbar sind (CHANGING-Parameter). Nachdem Sie die Werte für die Eingabeparameter eingetragen haben, starten Sie die Ausführung über das Icon (**Ausführen**). Der Funktionsbaustein wird nun ausgeführt. Nach dem Durchlauf werden die Werte der Ausgabeparameter angezeigt (siehe Abbildung 9.37).

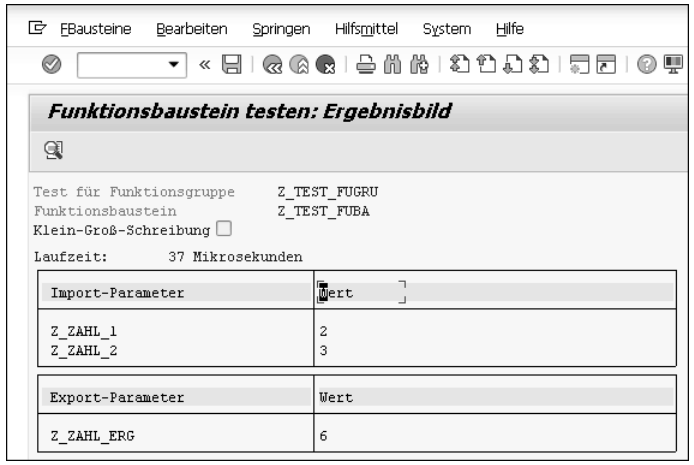


Abbildung 9.37 Anzeige der Ausgabeparameter

Test der Methode

Zum Testen von Methoden klicken Sie auf das Icon (**Testen**), oder Sie drücken F8. Anders als bei den Funktionsbausteinen generiert das SAP-System in diesem Fall zunächst eine Instanz der Klasse. Anschließend zeigt es Ihnen die Instanz in ihrer Ausprägung an. In Abbildung 9.38 hat das SAP-System als Beispiel eine Instanz der Klasse CL_EX_BAPITEXT_SCUSTOMER aus der Flugbuchungs-Schulungsanwendung von SAP erzeugt.

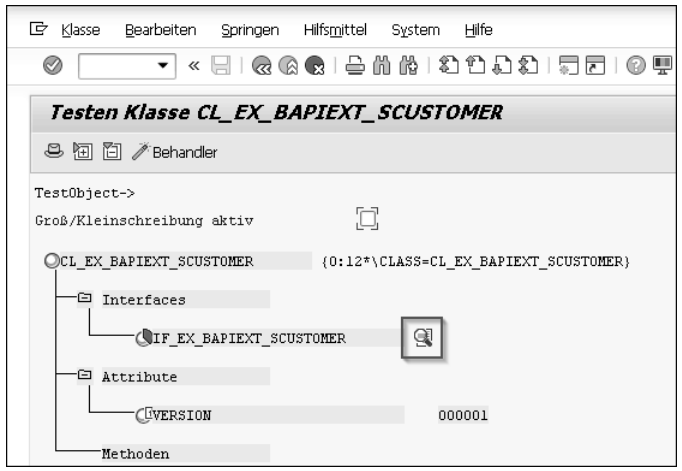


Abbildung 9.38 Instanzbildung für den Test einer Klasse

In diesem Objekt ist ein Interface eingebunden sowie Attribute und Methoden. Neben dem Interface IF_EX_BAPITEXT_SCUSTOMER befindet sich das Icon , auf das Sie klicken können, um die Definition zu sehen. Hier gibt es die Methoden CREATEFROMDATA_EXIT2 und CREATEFROMDATA_EXIT1 (siehe Abbildung 9.39).

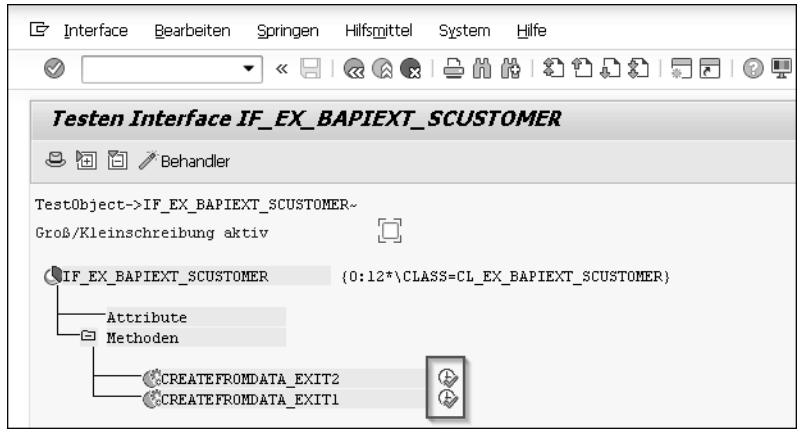


Abbildung 9.39 Aufklappen des Interface

Bei beiden Methoden finden Sie das Icon (**Ausführen**), ❶ in Abbildung 9.40, das in SAP standardmäßig symbolisiert, dass Sie darüber etwas ausführen können. Nach dem Klicken auf das Icon (**Ausführen**) bei der ersten Methode sehen Sie, dass es für diese Methode drei Importparameter gibt:

- CUSTOMER_DATA
- TEST_RUN
- EXTENSION_IN

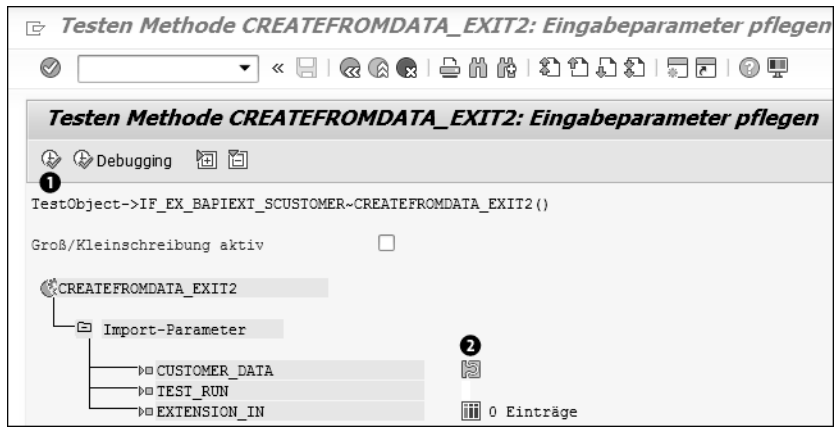


Abbildung 9.40 Importparameter in einer Klassen-Testumgebung

Testdaten eingeben

Sie können nun für alle drei Parametertypen Testdaten eingeben. Klicken Sie z. B. bei CUSTOMER_DATA auf das Icon (**Detailsicht**) ❷. Es werden Ihnen anschließend die Felder der Struktur angezeigt, in denen Sie Werte erfassen können (siehe Abbildung 9.41).

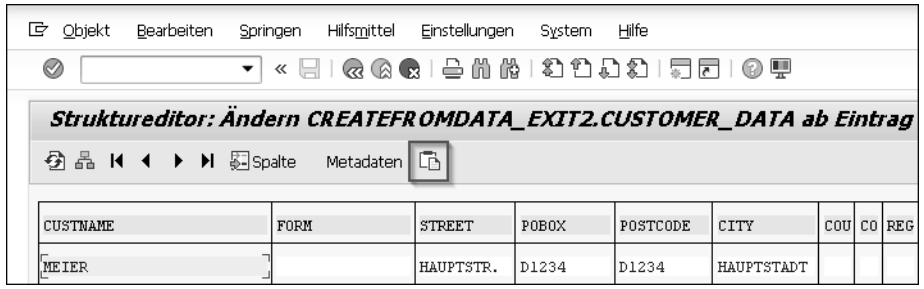


Abbildung 9.41 Testdaten für eine Struktur eintragen

Werte anschauen

Über das Icon (**Einsetzen**) werden die Testdaten gespeichert. Mit F3 kehren Sie zum vorherigen Bild zurück. Für die Importparameter können Sie anschließend ebenfalls Testdaten erfassen. Für die Testdaten bei der Tabelle klicken Sie dazu auf das Icon (**Liste**). Bei einem einfachen Eingabefeld können Sie Daten direkt eintragen. Nachdem Sie die Testdaten erfasst haben, klicken Sie auf das Icon (**Ausführen**) in Abbildung 9.40, um die Methode auszuführen. Als Ergebnis werden Ihnen die Exportparameter der Methode mit ihren Inhalten angezeigt (siehe Abbildung 9.42). In diesem Beispiel wäre es eine Tabelle, allerdings ohne Einträge. Diese Testdatenkonstellation hat also nicht zum gewünschten Ergebnis geführt.

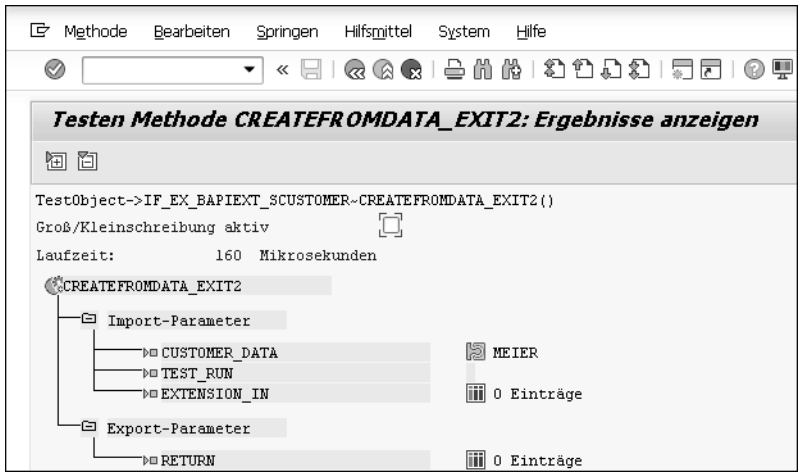


Abbildung 9.42 Ergebnisanzeige nach Testlauf der Methode

Debugger

Sie können dem ungewünschten Ergebnis tiefer auf den Grund gehen, indem Sie den Debugger einschalten. Mit einem Debugger können Sie Schritt für Schritt die Ausführung der codierten Befehle verfolgen. Ein Programm oder das Coding eines Funktionsbausteins oder einer Methode wird dabei wie in einem Editor dargestellt. Beim Testen von Funktionsbausteinen oder Methoden rufen Sie den Debugger über den Button Debugging in dem in Abbildung 9.40 gezeigten Bild auf.

Bei einem Programm können Sie den Debugger entweder aus einem Dynpro-Bild heraus über den Befehl /h im Kommandofeld in der Menüleiste aufrufen; bei dem nächsten auszuführenden Programmbefehl wird dann der Debugger aufgerufen, oder Sie setzen gezielt einen Breakpoint an der Stelle des Codings, ab der Sie den weiteren Verlauf der Programmausführung kontrollieren möchten. Ist im Programmablauf die Stelle mit dem gesetzten Breakpoint erreicht, wird automatisch der Debugger gestartet.

Debugger

Den Debugger können Sie nur im Online-Betrieb benutzen. Wenn Sie ein Programm über einen Job gestartet haben, werden die Breakpoints ignoriert, und der Debugger wird nicht aufgerufen. Sie müssen sich dann mit der Ausgabe von Nachrichten behelfen, über die Sie z. B. Zwischenergebnisse ausgeben. Diese Nachrichten erscheinen dann im Laufprotokoll. Wenn Sie also ein Programm für die Hintergrundverarbeitung testen wollen, starten Sie es in Transaktion SE38 (ABAP Editor) im Online-Betrieb mit F8.

Eine ausführliche Beschreibung des Debuggers finden Sie in der SAP-Hilfe unter *http://s-prs.de/v433203*.

9.6 Prozessketten

Prozess

Prozessketten stellen eine andere Möglichkeit der Hintergrundverarbeitung dar. SAP definiert einen Prozess als »einen Vorgang innerhalb oder außerhalb eines SAP-Systems mit definiertem Anfang und Ende« (siehe SAP-Hilfe unter *http://s-prs.de/v433204*). Ein ABAP-Programm wird in diesem Sinne auch als Prozess verstanden.

Das Einrichten und Verwalten von Prozessketten geschieht über Transaktion RSPC (Prozesskettenpflege). In SAP NetWeaver 7.5 ist diese Transaktion in Transaktion RSA1 (Data Warehouse Workbench) integriert (siehe Abbildung 9.43).

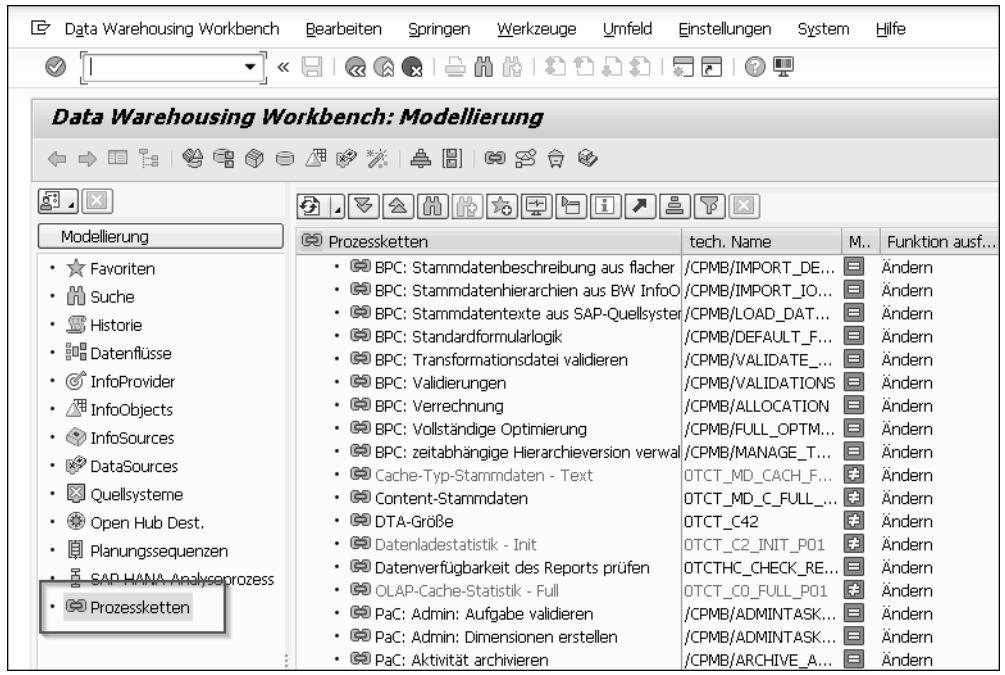


Abbildung 9.43 Transaktion RSPC in Transaktion RSA1 in SAP NetWeaver 7.5

Prozessarten

Prozesse werden nach den folgenden Typen unterschieden:

- Startprozess
- Anwendungsprozess
- Sammelprozess

Ein *Startprozess* definiert den Start einer Prozesskette. Der Startprozess enthält keine weitere Funktionalität, sondern er ist lediglich als Auslöser zu betrachten. Er darf in einer Kette nur einmal vorkommen und darf keinen Vorgängerprozess haben.

Anwendungsprozesse enthalten Arbeitsprozesse, die automatisiert ablaufen sollen, dies können z. B. ABAP-Programme sein. *Sammelprozesse* fassen mehrere Kettenstränge zu einem Kettenstrang zusammen. Die einzelnen Prozesse sind durch *Events* (Ereignisse) miteinander verbunden. Für einen Prozess gibt es einen *Vorgänger-* und einen *Nachfolgerprozess*, sofern er kein Startprozess und nicht der letzte Prozess einer Kette ist.

Prozessmerkmale

Ein Prozess enthält die folgenden charakterisierenden Merkmale:

- **Prozesstyp**
  Der Prozesstyp beschreibt die Art des Prozesses. Darüber wird die Aufgabe eines Prozesses bestimmt.

- **Prozessvariante**
  Der Name einer Prozessvariante ist gleichzeitig der Name des Prozesses in der Prozesskette. Dieser Typ von Variante ist nicht gleichzusetzen mit dem Typ der Variante eines Selektionsbildes (siehe Abschnitt 9.3, »Variante erzeugen«). Über eine Prozessvariante wird ein Prozess konfiguriert. Erst zusammen mit dem Prozesstyp ist eine Prozessvariante eindeutig.
- **Prozessinstanz**
  Eine Prozessinstanz bestimmt die Ausprägung eines Prozesses. Sie enthält u. a. Informationen, die einem Folgeprozess übergeben werden können. Die Prozessinstanz wird von der Prozesskettenverwaltung gesichert. Protokolle werden unter der Prozessinstanz angelegt.

Metakette

Prozessketten können in andere Prozessketten integriert und darüber gestartet werden. Die Prozesskette, die eine andere Prozesskette integriert, wird als *Metakette* bezeichnet.

Startbedingung

Gestartet wird eine Prozesskette immer über den Startprozess. Der Start einer Prozesskette ist gleichbedeutend mit dem Start eines Jobs. Daher legen Sie für die Prozesskette die gleichen Startbedingungen fest wie für einen Job (siehe Abschnitt 9.2, »Programme im Hintergrund starten«). Zu den Startbedingungen einer Prozesskette gelangen Sie, indem Sie auf einem Startprozess einen Doppelklick ausführen. Dadurch wird die Variante des Startprozesses aufgerufen. Im Bereich **Einplanungsoptionen** ist der Radiobutton **direkte Einplanung** aktiviert (siehe Abbildung 9.44). Daneben gibt es einen Button mit der etwas irreführenden Bezeichnung **Selektionen ändern**. Wenn Sie auf diesen Button klicken, gelangen Sie zu den Startbedingungen. Jeder folgende Prozess wird über einen eigenen Job gestartet. Diese Jobs werden durch das Auslösen von Events aktiviert und müssen nicht extra eingeplant werden.

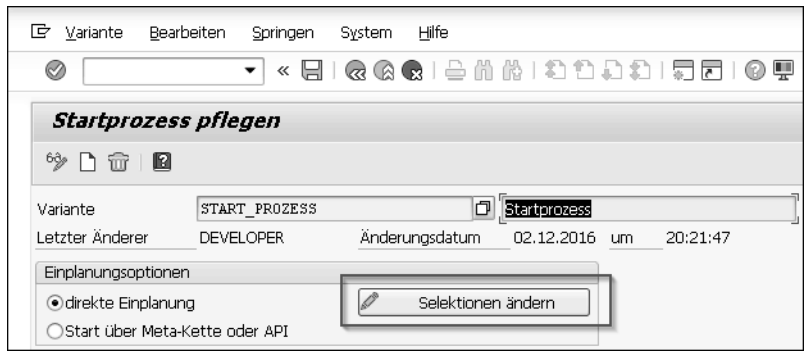


Abbildung 9.44 Startprozess – Variantenpflege

Status

Nach dem Durchlaufen einer Prozesskette sind die einzelnen Prozesse in Transaktion RSPC (Prozesskettenpflege) farblich markiert. Darüber wird signalisiert, mit welchem Status ein Prozess beendet wurde (siehe Abbildung 9.45). Grün bedeutet ein fehlerfreies Ende des Prozesses. Bei Warnungen wird ein Prozess gelb markiert und bei einem Abbruch rot.

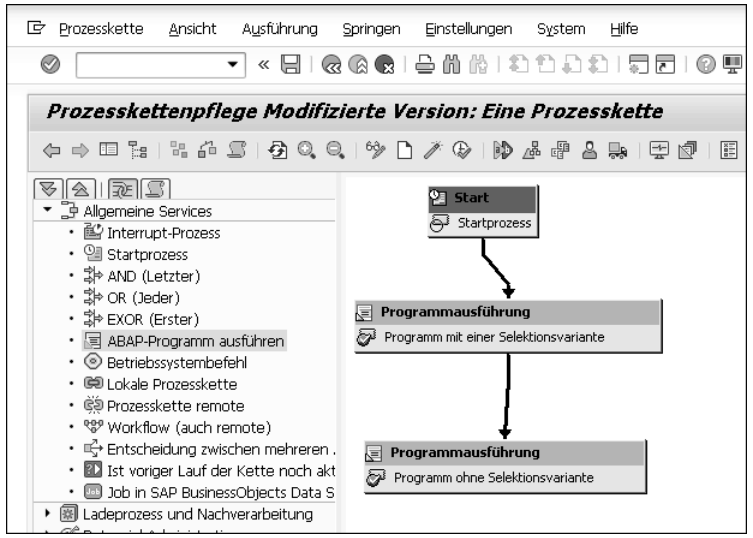


Abbildung 9.45 Erfolgreicher Durchlauf einer Prozesskette

Prozessmeldungen

Wenn Sie mit der rechten Maustaste auf einen Prozess klicken, können Sie sich zu dem Prozess die vorhandenen Meldungen anschauen (siehe Abbildung 9.46).

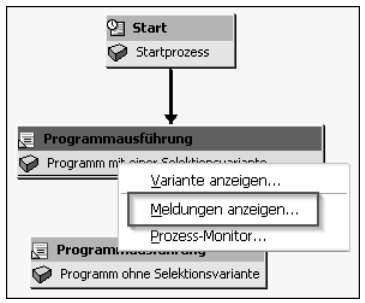


Abbildung 9.46 Kontextmenü zu einem Prozess über rechte Maustaste

Sammelprozess

Sie haben beim Design von Prozessketten die Möglichkeit, den Verarbeitungsablauf mit Bedingungen zu verknüpfen. Dazu dienen die Sammelprozesse. Ein Sammelprozess fasst mehrere Kettenstränge zu einem Kettenstrang zusammen. Beim Anlegen eines Sammelprozesses müssen Sie

lediglich eine Kurzbeschreibung eingeben. Die Prozessvariante mit dem Namen generiert das SAP-System automatisch. Es gibt drei Typen von Sammelprozessen:

- **AND (Letzter)**
  Alle Vorgängerprozesse, die mit dem Sammelprozess verbunden sind, müssen beendet sein, bevor die Kette fortgeführt wird.
- **OR (Jeder)**
  Die Prozesskette wird jedes Mal vom Sammelprozess aus fortgeführt, wenn jeweils ein Vorgängerprozess das entsprechenden Event ausgelöst hat.
- **EXOR (Erster)**
  Die Prozesskette wird komplett fortgeführt, wenn das erste Event von einem der Vorgängerprozesse erfolgreich ausgelöst wurde.

Alle Vorgängerprozesse zu einem Sammelprozess lösen das gleiche Event aus. Abbildung 9.47 zeigt ein einfaches Beispiel für einen Sammelprozess (siehe Markierung) mit einer AND-Verknüpfung. Eine solche Verknüpfung könnte sich z. B. bei einer Verbuchung anbieten. Die Verbuchung (Update auf der Datenbank) soll erst dann starten, wenn alle notwendigen vorangehenden Prozesse beendet sind.

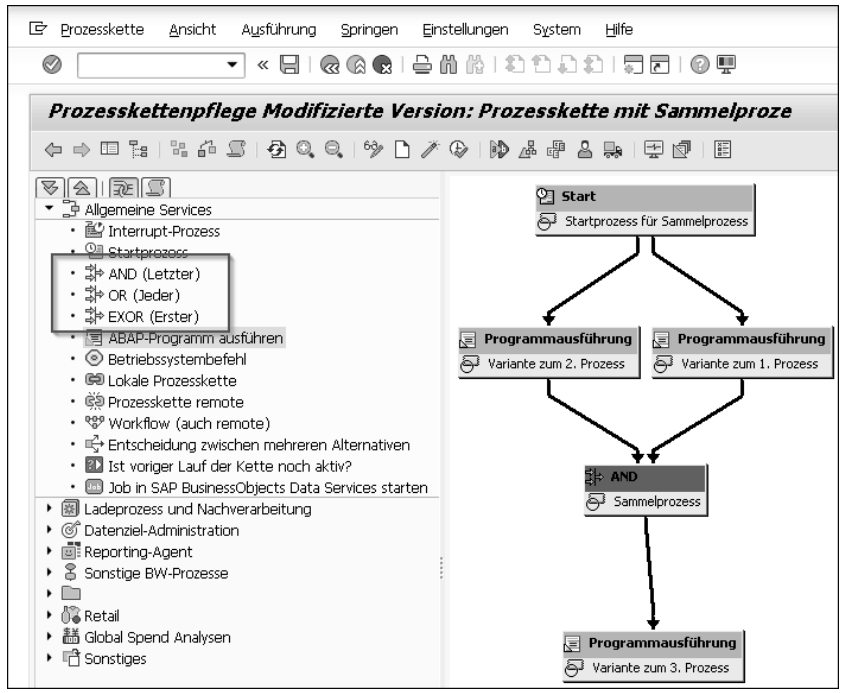


Abbildung 9.47 Transaktion RSPC – Beispiel für einen Sammelprozess

In diesem Beispiel startet der Startprozess zwei Anwendungsprozesse, abhängig davon, wann im Applikationsserver entsprechende Workprozesse frei sind. Beide Anwendungsprozesse sind mit dem Sammelprozess über eine AND-Verknüpfung verbunden. Dieser Sammelprozess wartet so lange, bis beide Vorgängerprozesse das Event für den Sammelprozess erfolgreich ausgelöst haben. Erst dann löst der Sammelprozess selbst das Event für den letzten Anwendungsprozess aus.

Prozessketten werden in der Praxis häufig angewendet. Der Vorteil von Prozessketten liegt in der grafischen Darstellung der Abhängigkeiten zwischen den einzelnen Prozessen. Ein weiterer Vorteil ist die Automatisierung der Abläufe. Allerdings wird bei komplexen Prozessketten der Vorteil der grafischen Darstellung minimiert, wenn der Monitor für die Darstellung nicht ausreicht.

9.7 Fehleranalyse bei Programmabbruch

Trotz umfangreicher technischer und fachlicher Tests kommt es vor, dass ein Programm abbricht. Dies ist nicht schön, passiert aber immer wieder. Die Ursachen dafür können vielfältig sein, z. B.:

- mangelhafte Entwicklerqualität
- mangelhafte Datenqualität
- fachliche oder technische Notwendigkeit

Ein Zeichen mangelhafter Entwicklungsqualität wäre z. B. eine Division durch null. Dies führt grundsätzlich zu einem Abbruch. In einigen Unternehmen gehört es zu den Programmierrichtlinien, vor einer programmierten Division den Divisor auf ungleich null abzufragen.

Beispiel

Ein Beispiel für mangelhafte Datenqualität wäre das folgende: In einem Programm gibt es für vorhandene Artikel in einem Lager ein numerisches Mengenfeld. Die Daten dafür werden vor dem Programmlauf in einem Microsoft-Excel-Sheet erfasst und nachfolgend als **.csv**-Datei in das Programm eingelesen. Bei den Artikeln, die aktuell nicht im Lager vorhanden sind, soll als Menge null eingetragen werden. Nicht alle Erfasser halten sich an diese Regel, sondern manche lassen in solchen Fällen das Feld leer. Beim Übertrag eines leeren Excel-Feldes in ein numerisches ABAP-Feld kommt es zum Abbruch, weil das leere Excel-Feld keinen numerischen Inhalt hat, sondern einen alphanumerischen, in diesem Fall Blanks. Alphanumerische Werte können nicht in ein numerisches Feld übertragen werden. Dies sollte im Programm vorher geprüft werden.

Ungewollte und gewollte Abbrüche

Während die ersten beiden Beispiele *ungewollte Abbrüche* sind, gibt es auch den *gewollten Abbruch*. So bedingt z. B. die Zusammenstellung einer Jobkette eine bestimmte logische Reihenfolge der Verarbeitung. Dabei sind die Programme von den Ergebnissen der zuvor gelaufenen Programme abhängig. Die Programme prüfen zu Beginn, ob alle notwendigen Daten für die Verarbeitung vorhanden sind. Ist dies nicht der Fall, wird mit einer entsprechenden Fehlermeldung abgebrochen.

Dump

Bei ungewollten Abbrüchen und bei Abbrüchen, bei denen eine Meldung vom Typ **X** oder der ABAP-Befehl `raise exception type ...` ausgegeben wurde, erzeugt das SAP-System einen *Dump*. Ein Dump ist ein kontextbezogener Speicherauszug, der den Speicherinhalt zum Zeitpunkt des Abbruchs wiedergibt. Zusätzlich zeigt der SAP-Dump die Stelle im Coding, an der der Abbruch passierte. Bei einer Division durch null wäre der Fehler schnell gefunden; wir zeigen Ihnen gleich ein Beispiel dafür. Es gibt aber auch Abbrüche, die von einer zentralen Fehlerroutine ausgelöst wurden. Der Dump zeigt dann auf die zentrale Fehlerroutine. Die Ursache für den Abbruch liegt aber ganz woanders. In solch einem Fall den ursächlichen Fehler zu finden, kann eine Herausforderung sein.

Als Beispiel für einen Dump nutzen wir ein ganz einfaches Programm, das lediglich eine Division durch null enthält. Das Programm wird über einen Job gestartet. Die Jobübersicht in Transaktion SM37 zeigt den Programmabbruch an. Der Job selbst wurde ebenfalls abgebrochen (siehe Abbildung 9.48).

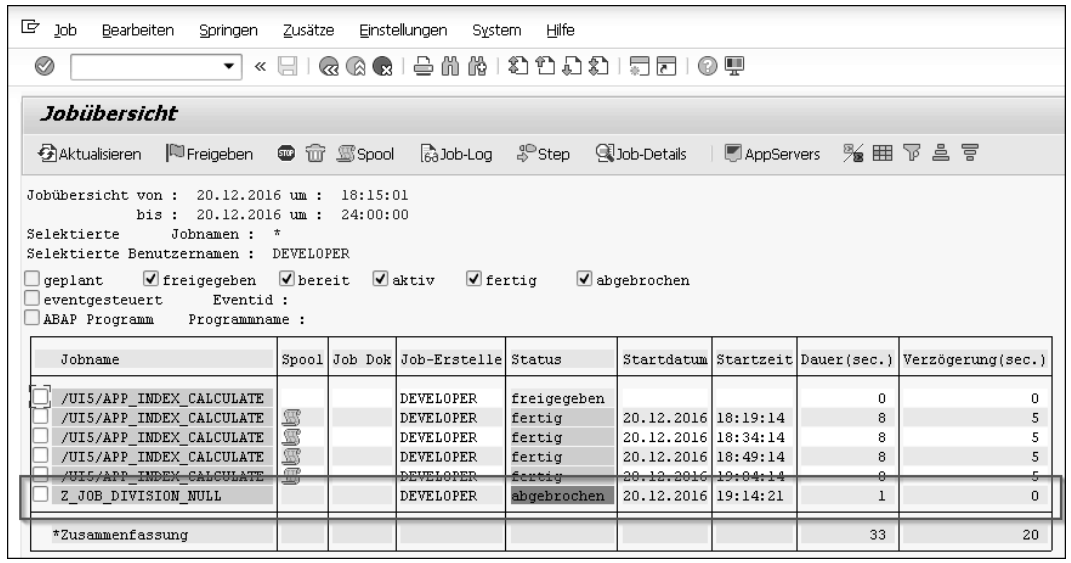


Abbildung 9.48 Transaktion SM37 – Job abgebrochen

ST22 Wenn Sie ins Job-Log schauen, wird Ihnen auch der Grund für den Abbruch angezeigt und dass Sie für nähere Informationen Transaktion ST22 aufrufen sollten (siehe Abbildung 9.49).

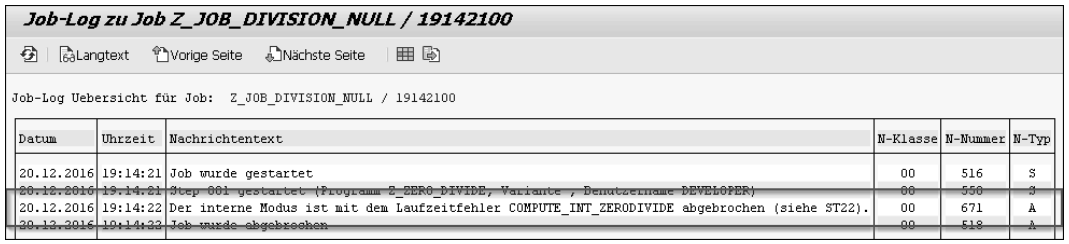


Abbildung 9.49 Transaktion SM37 – Job-Log zum abgebrochenen Job

Über die Transaktion können Sie eine Selektion auf vorhandene Laufzeitfehler vornehmen. Im Bereich **Eigene Auswahl** in Transaktion ST22 (ABAP-Dump-Analyse) ist Ihr Benutzer im Feld **Benutzer** voreingestellt (siehe Abbildung 9.50). Sie können dort aber auch einen anderen Benutzer eintragen.

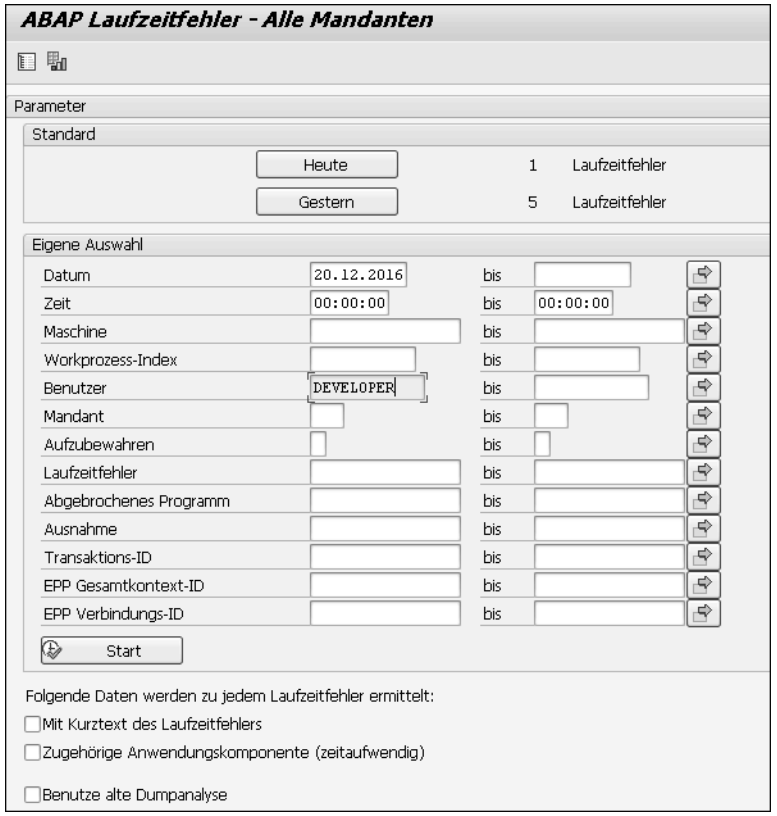


Abbildung 9.50 Transaktion ST22 – Selektionsauswahl und Anzeige von Laufzeitfehlern

Klicken Sie auf den Button Start, um Ihre eigenen Laufzeitfehler aufzurufen. Alternativ können Sie auch auf die Buttons Heute und Gestern klicken. Es werden Ihnen dann alle Laufzeitfehler, die sich heute oder gestern ereignet haben, angezeigt.

Dump aufrufen

Es gibt zwei Möglichkeiten, um den Dump für diesen Abbruch aufzurufen:

- Sie führen im Job-Log einen Doppelklick auf dem Fehlerhinweis aus (siehe Abbildung 9.49).
- Sie führen in Transaktion ST22 einen Doppelklick auf dem angezeigten Laufzeitfehler aus (siehe Abbildung 9.51).

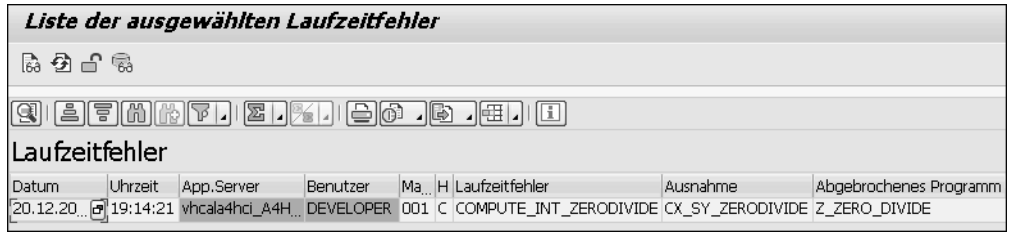


Abbildung 9.51 Transaktion ST22 – Anzeige des Laufzeitfehlers

Der Dump gibt Ihnen gleich zu Beginn einen Hinweis darauf, was passiert ist (siehe Abbildung 9.52).

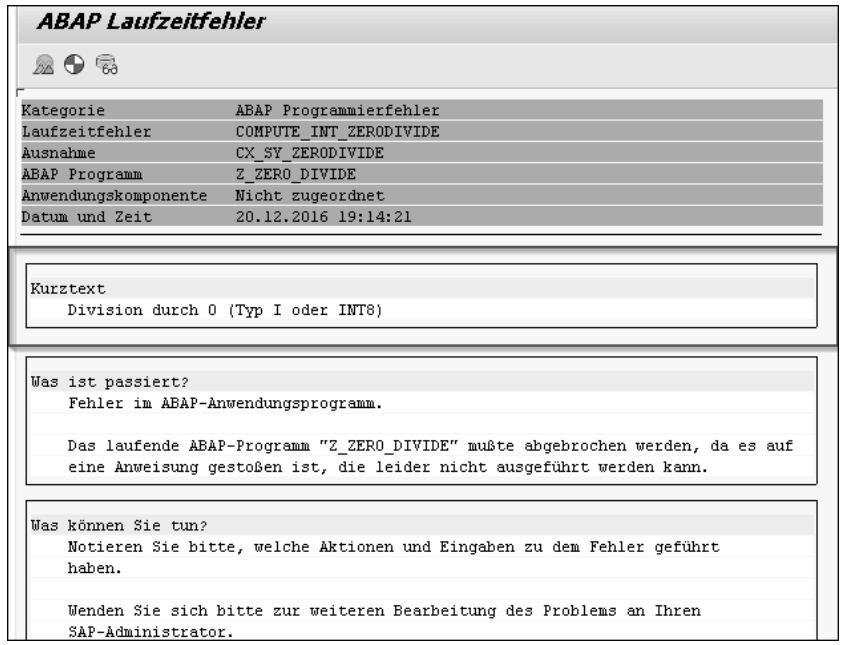


Abbildung 9.52 Dump mit Fehlerhinweis

Wenn Sie im Dump weiter nach unten scrollen, gelangen Sie zu der Stelle, an der Ihnen der Dump den Ort im Coding anzeigt, an der der Abbruch ausgelöst wurde (siehe Abbildung 9.53). Die verursachende Coding-Zeile ist links durch die Zeichen >>>>> markiert.

ABAP Laufzeitfehler

Kategorie	ABAP Programmierfehler
Laufzeitfehler	COMPUTE_INT_ZERODIVIDE
Ausnahme	CX_SY_ZERODIVIDE
ABAP Programm	Z_ZERO_DIVIDE
Anwendungskomponente	Nicht zugeordnet
Datum und Zeit	20.12.2016 19:14:21

Informationen zur Abbruchstelle

Der Abbruch trat im ABAP-Programm bzw. Include "Z_ZERO_DIVIDE" auf, und zwar in "START-OF-SELECTION". Das Hauptprogramm war "Z_ZERO_DIVIDE".

Im Quelltext befindet sich die Abbruchstelle in Zeile 12 des Programms bzw. Includes "Z_ZERO_DIVIDE".
Das Programm "Z_ZERO_DIVIDE" wurde in einem Hintergrund-Job gestartet.

Jobname..... Z_JOB_DIVISION_NULL
Jobinitiator... DEVELOPER
Jobnummer..... 19142100

Ausschnitt Quelltext

Zeile	Quelltext
1	*&---*&
2	*& Report Z_ZERO_DIVIDE
3	*&---*&
4	*&
5	*&---*&
6	REPORT Z_ZERO_DIVIDE.
7	
8	data: lv_num1 type int4 value 20,
9	lv_num2 type int4 value 0,
10	lv_erg type int4 value 0.
11	
>>>>>	lv_erg = lv_num1 / lv_num2.

Abbildung 9.53 Abbruchstelle im Programm

Wenn Sie im Dump noch weiter nach unten scrollen, bekommen Sie einen Speicherauszug mit den zum Zeitpunkt des Abbruchs vorhandenen Werten. Allerdings nutzt Ihnen der Auszug nur dann etwas, wenn die Abbruchstelle auch tatsächlich die ursächliche Stelle im Programm war. Bei einer zentralen Fehlerroutine wird Ihnen der Speicherinhalt im Umfeld dieser Routine angezeigt.

SM21 – SAP-SysLog

Der Abbruch wird auch im SAP-SysLog (Transaktion SM21), angezeigt. Im SAP-SysLog werden Systemmeldungen eingetragen, dazu gehören auch Programmabbrüche. In Abbildung 9.54 sehen Sie den Programmabbruch.

SysLog-Meldungen

SysLog der Instanz vhcala4hci_A4H_00

Datum	TIME	Instanz	Typ	Prozess-Nr	Mdt	Benutzer	Prio.	MeldungsID	Meldungstext
23.12.2016	14:04:58	vhcala4hci_A4H_00	BTC	011	001	DEVELOPER		Q02	Stop Workp. 11, Pid 12719
23.12.2016	14:19:58	vhcala4hci_A4H_00	BTC	015	001	DEVELOPER		Q02	Stop Workp. 15, Pid 12721
23.12.2016	14:34:58	vhcala4hci_A4H_00	BTC	013	001	DEVELOPER		Q02	Stop Workp. 13, Pid 11718
23.12.2016	14:49:58	vhcala4hci_A4H_00	BTC	011	001	DEVELOPER		Q02	Stop Workp. 11, Pid 13499
23.12.2016	15:04:40	vhcala4hci_A4H_00	DIA	005	001	DEVELOPER		AB0	Laufzeitfehler "COMPUTE_INT_ZERODIVIDE" aufgetreten.
23.12.2016	15:04:40	vhcala4hci_A4H_00	DIA	005	001	DEVELOPER		AB1	> Kurzdump "161223 150440 vhcala4hci_A4H_00 DEVELOPER" erstellt.
23.12.2016	15:04:58	vhcala4hci_A4H_00	BTC	012	001	DEVELOPER		Q02	Stop Workp. 12, Pid 11717

Abbildung 9.54 Transaktion SM21 – SAP-SysLog mit Programmabbruch

SAP-Standard-protokollierung

SAP bietet in der Programmiersprache ABAP Funktionsbausteine für das Protokollieren von Meldungen an, die ein Programm ausgibt. Wir haben dafür ein zweites Beispiel; es handelt sich ebenfalls um eine Division durch null, die aber durch das Programm protokolliert wird. Abbildung 9.55 zeigt Ihnen einen Programmausschnitt, in dem diese Funktionsbausteine verwendet werden. Der Meldungstyp ist diesmal **E** (Error), der nicht zu einem Abbruch führt (siehe Abbildung 9.55).

Report Z_ZERO_DIVIDE_SLG1 aktiv

```
19    try.
20      lv_erg = lv_num1 / lv_num2.
21  *     if lv_num2 = 0.
22
23      catch cx_sy_zerodivide.
24  * Protokoll im ApplicationLog eröffnen
25        lv_log-extnumber = 'PGM_ZERO_DIVIDE_SLG1'.
26        lv_log-object    = 'NOTIF'.
27        lv_log-subobject = 'ZERODIVIDE'.
28        lv_log-aldate    = sy-datum.
29        lv_log-altime    = sy-uzeit.
30        lv_log-aluser    = sy-uname.
31        lv_log-alprog    = sy-calld.
32
33        call function 'BAL_LOG_CREATE'
34          exporting
35            i_s_log      = lv_log
36          importing
37            e_log_handle = lv_appl_log_handle.
38  * Meldung
39        lv_msg-msgty = 'E'.
40        lv_msg-msgid = 'Z_NACHRICHTKLASSE'.
41        lv_msg-msgno = '900'.
42  * Meldung dem Protokoll hinzufügen
43        call function 'BAL_LOG_MSG_ADD'
44          exporting
45            i_log_handle = lv_appl_log_handle
46            i_s_msg      = lv_msg
47          exceptions
48            others  = 1.
49  * Log in der Datenbank sichern
50        call function 'BAL_DB_SAVE'
51          exporting
52            i_save_all       = 'X'
53          exceptions
54            log_not_found    = 1
55            save_not_allowed = 2
56            numbering_error =  3
57            others           = 4.
58
59  *     raise exception type cx_sy_zerodivide.
60    endtry.
```

Abbildung 9.55 Fehlerprotokollierung über die Funktionsbausteine BAL_...

Die SAP-Standardprotokollierung ist mit den folgenden drei Transaktionen verbunden:

- Transaktion SLG0 – Identifier für eigene Logs
- Transaktion SLG1 – Anwendungslogs
- Transaktion SLG2 – Löschen von Anwendungslogs

SLG0 In Transaktion SLG0 können Sie ein Meldungsobjekt festlegen und diesem Objekt untergeordnete Objekte zuordnen. In unserem Beispiel haben wir das Objekt `NOTIF` mit dem Objekttext »Meldungen« genutzt und dazu ein Unterobjekt `ZERODIVIDE` eingetragen.

SLG1 Im Programm nehmen wir in den Zeilen 26 und 27 Bezug darauf (siehe Abbildung 9.55). Zusätzlich wird im Programm der *externe Identifier* `PGM_ZERO_DIVIDE_SLG1` definiert (Zeile 25). Mit diesen drei Identifiers können Sie in Transaktion SLG1 die protokollierten Anwendungsmeldungen selektieren. Wenn Sie Transaktion SLG1 aufrufen, wird Ihnen eine Selektion über die Protokolle angeboten (siehe Abbildung 9.56).

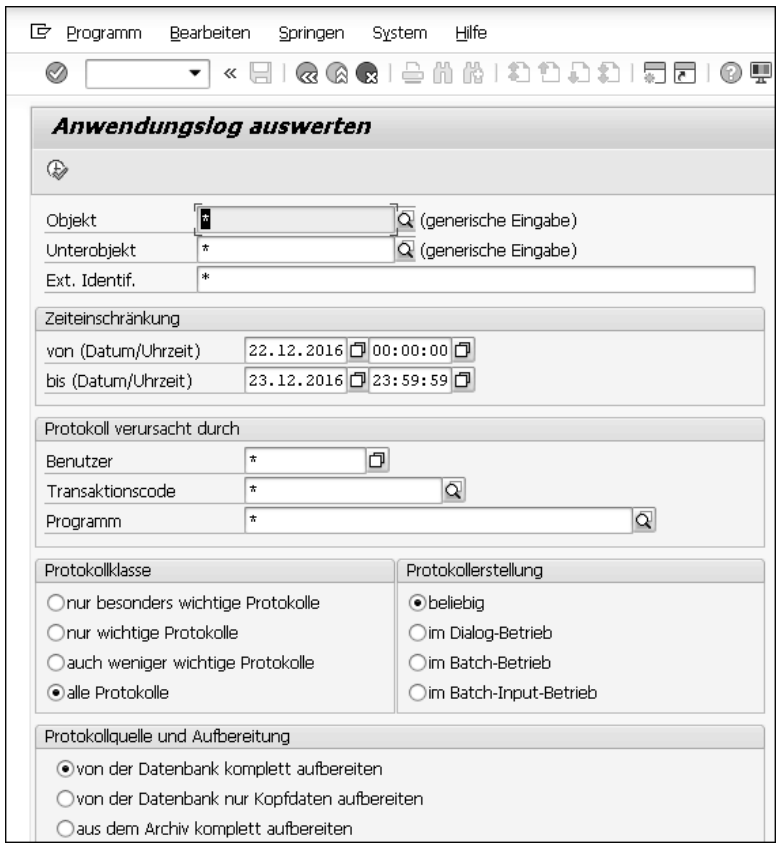


Abbildung 9.56 Transaktion SLG1 – Selektion der Protokolle

Als Erstes können Sie nach dem externen Identifier, dem Objekt und/oder dem Unterobjekt selektieren. Danach können Sie weitere Einschränkungen vornehmen, z. B. nach:

- dem Zeitpunkt der Protokollierung
- dem Verursacher des Protokolleintrags
- der Protokollklasse und Protokollerstellung
- der Protokollquelle und Aufbereitung

In der Praxis werden Sie am ehesten eine Selektion über die drei Identifiers, über die Zeiteinschränkung oder über den Protokollverursacher vornehmen. Nach dem Selektionseintrag klicken Sie auf das Icon (**Ausführen**). Sie gelangen danach zur Anzeige der Meldungen. Im oberen Teil des Dialogs stehen die Meldungsköpfe ❶ und im unteren Teil die zugehörigen konkreten Meldungen (❷ in Abbildung 9.57).

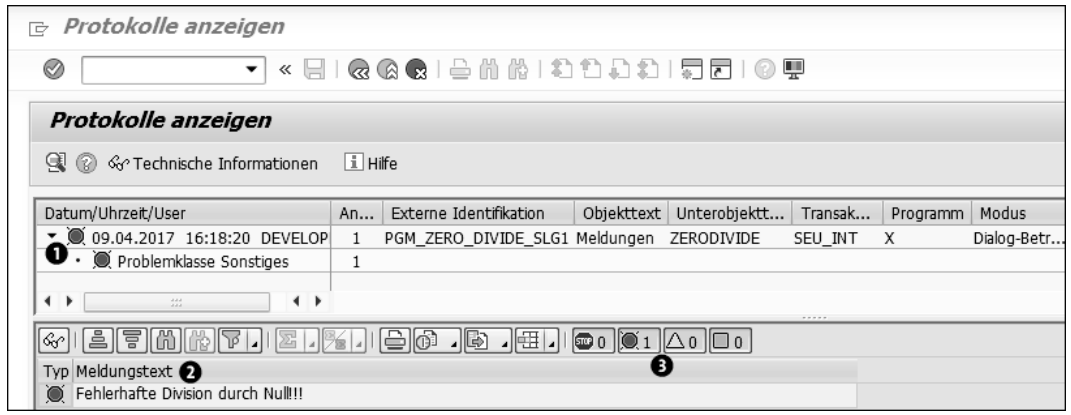


Abbildung 9.57 Transaktion SLG1 – Anwendungsprotokolle

In Abbildung 9.57 sehen Sie unseren protokollierten Fehlerhinweis aus dem Programmausschnitt in Abbildung 9.55. Zur Anzeige der konkreten Meldungen gelangen Sie, indem Sie auf dem kleinen Pfeil neben dem Meldungskopf einen Doppelklick ausführen; der Meldungskopf wird aufgeblättert.

Führen Sie anschließend auf der angezeigten Problemklasse einen Doppelklick aus; danach werden Ihnen die zugehörigen Meldungen angezeigt, in diesem Beispiel unsere Fehlermeldung. In der Praxis werden Sie dort, je nach Selektion, sehr viele Meldungen sehen, die alle unterschiedlichen Typs sein werden ❸.

Typ	Beschreibung
0	Meldung als Information
0	Meldung als Warnung
1	Meldung zu einem Fehler
STOP 0	Meldung zu einem Programmabbruch

Tabelle 9.3 Meldungstypen

In manchen Fällen bekommen Sie noch eine genauere Information zu einer Meldung, wenn Sie dazu auf den Button Technische Informationen in der anwendungsbezogenen Icon-Leiste klicken. Der genaue Informationsgehalt ist vom Meldungsinhalt abhängig und kann variieren.

Tipps zur Fehlersuche

Wenn es zu einem Programmabbruch kommt, schauen Sie sich als Erstes den Dump dazu an, der Ihnen vielleicht einen ersten Hinweis liefert. Als Nächstes werfen Sie einen Blick auf die Protokollierung in Transaktion SLG1 (Anwendungslogs). Bei einer ordnungsgemäßen Protokollierung wird Ihnen dazu ein Hinweis angezeigt, wie weit das Programm mit seiner Verarbeitung gekommen ist.

Wenn das abgebrochene Programm über einen Job gestartet wurde, schauen Sie in Transaktion SM37 (Jobübersicht) in das Job-Log. Vielleicht finden Sie dort einen Hinweis. Wenn das Programm Spool-Ausgaben gemacht hat, schauen Sie sich diese an. Versuchen Sie herauszufinden, welche Datensätze sich gerade in der Verarbeitung befanden.

Verwendungsnachweis

Wenn der Fehler in einem Funktionsbaustein oder in einer Methode auftrat, hilft Ihnen unter Umständen der *Verwendungsnachweis* weiter. Der Verwendungsnachweis listet Ihnen die Repository-Objekte auf, in denen z. B. ein Funktionsbaustein oder eine Methode aufgerufen wird. Das gilt auch für Programme, Komponenten oder die Objekte des ABAP Dictionarys wie beispielsweise eine Tabelle, ein Tabellenfeld, eine Struktur, ein Datenfeld oder ein Domäne.

Wir zeigen Ihnen ein Beispiel für den Verwendungsnachweis eines Funktionsbausteins. Dazu rufen Sie den Funktionsbaustein in Transaktion SE37 (Jobübersicht) oder SE80 (Object Navigator) auf. Markieren Sie danach den Namen des Funktionsbausteins ❶, und klicken Sie in der anwendungsbezogenen Icon-Leiste auf das Icon (❷ in Abbildung 9.58).

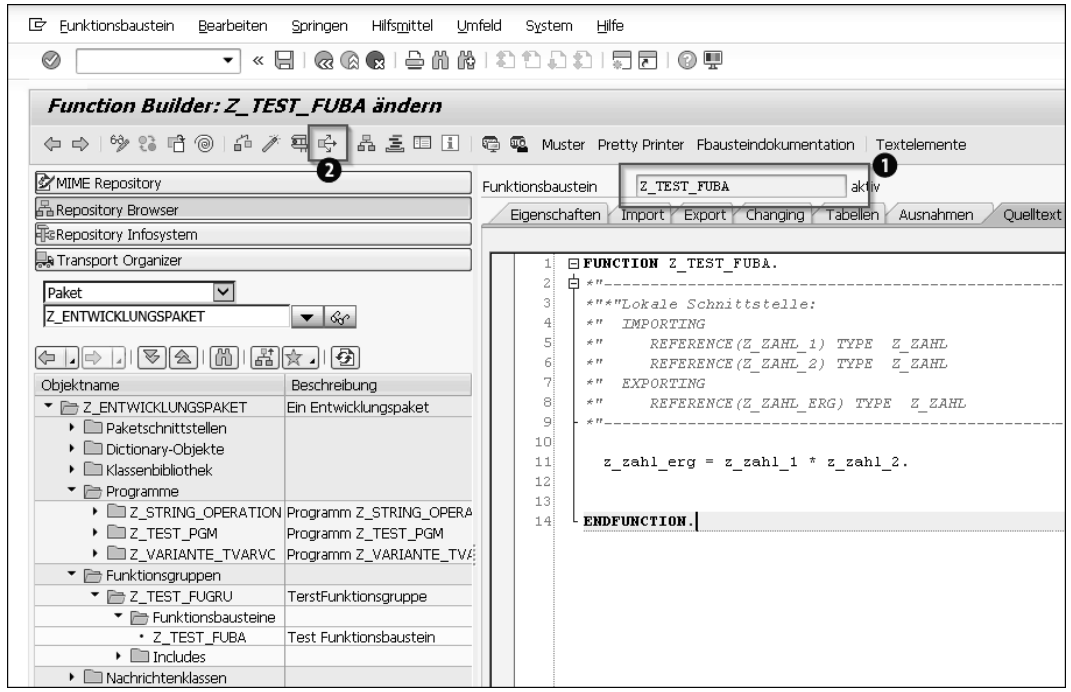


Abbildung 9.58 Verwendungsnachweis für einen Funktionsbaustein

Es erscheint ein Subfenster, in dem Sie den Suchbereich für den Verwendungsnachweis festlegen. Im Zweifel markieren Sie alles über das Icon (**Alles markieren**), siehe Abbildung 9.59. Mit dem Klicken auf (**Weiter**) beginnt die Suche.

Abbildung 9.59 Suchbereich für den Verwendungsnachweis festlegen

Wenn der Verwendungsnachweis nichts gefunden hat, wird in der Statuszeile eine entsprechende Meldung ausgegeben; ansonsten wird Ihnen das Suchergebnis aufgelistet (siehe Abbildung 9.60).

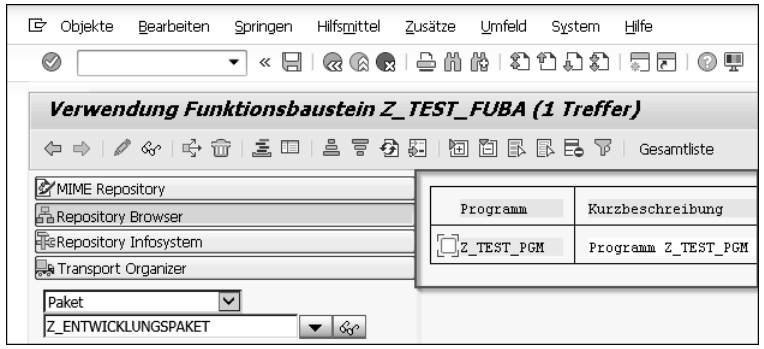


Abbildung 9.60 Suchergebnis des Verwendungsnachweises

Das Ergebnis ist in diesem Fall überschaubar. Der Funktionsbaustein wird lediglich im Testprogramm `Z_TEST_PGM` verwendet.

Möglicherweise gibt das Anwendungsprotokoll in Transaktion SLG1 bei einer Meldung die Nachrichtenklasse und die Fehlernummer aus. Dann könnte es sich anbieten, einen Verwendungsnachweis über die Fehlernummer vorzunehmen. Dazu wechseln Sie in Transaktion SE91 und rufen die Nachrichtenklasse auf. Markieren Sie dann die Fehlernummer, und rufen Sie, wie oben beschrieben, den Verwendungsnachweis auf.

Fehlernummer und Nachrichtenklasse

Achten Sie bei den Fundstellen darauf, dass die gefundene Fehlernummer auch zu der Nachrichtenklasse gehört. Eine Fehlernummer ist nur zusammen mit der Nachrichtenklasse eindeutig. Aber so finden Sie unter Umständen die Stelle, an der die Fehlermeldung generiert wurde.

Umfeldermittlung

Während der Verwendungsnachweis die Stellen sucht, an denen Ihr Repository-Objekt verwendet wird, ist die *Umfeldermittlung* das Gegenteil davon. Sie zeigt Ihnen an, welche Referenzen Ihr Objekt aufruft, die nicht Teil Ihres Objekts sind. Bei diesen *Außenreferenzen* handelt es sich um Referenzen, die nicht in Ihrem Objekt definiert sind. Um die Umfeldermittlung aufzurufen, wechseln Sie in Transaktion SE80 (Object Navigator) in das Repository-Infosystem. Dort wählen Sie einen Objekttyp aus, z. B. eine Methode, und klicken mit der rechten Maustaste darauf. Über das Kontextmenü wählen Sie die **Umfeldermittlung** aus (siehe Abbildung 9.61). Danach spezifizieren Sie das Repository-Objekt und starten die Umfeldermittlung.

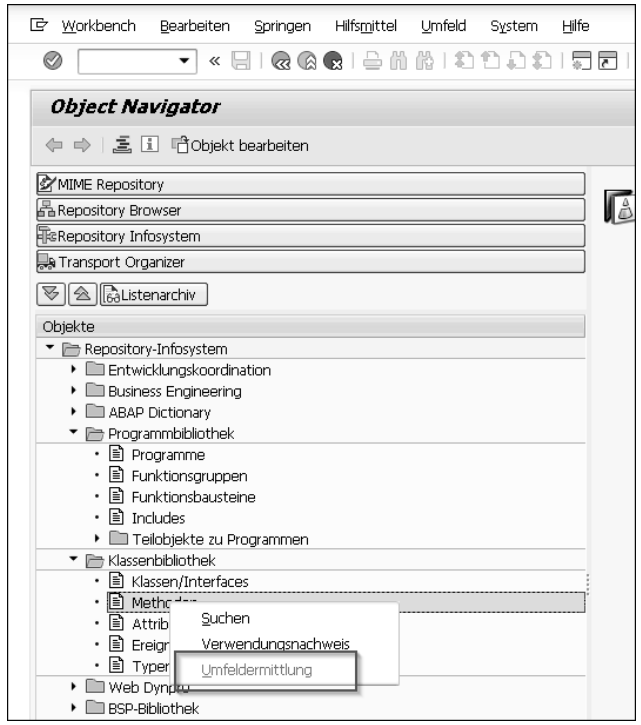


Abbildung 9.61 Transaktion SE80 – Umfeldermittlung

F1-Hilfe – technische Information

Für eine weitergehende Analyse ist es manchmal hilfreich zu wissen, welches Tabellenfeld mit einem Dynpro-Feld verbunden ist. Setzen Sie dazu den Cursor in das Dynpro-Feld, das Sie interessiert, und rufen Sie die F1-Hilfe auf. In dem Fenster können Sie das Icon (**Technische Informationen**) zum Aufruf von technischen Informationen nutzen (siehe Abbildung 9.62).

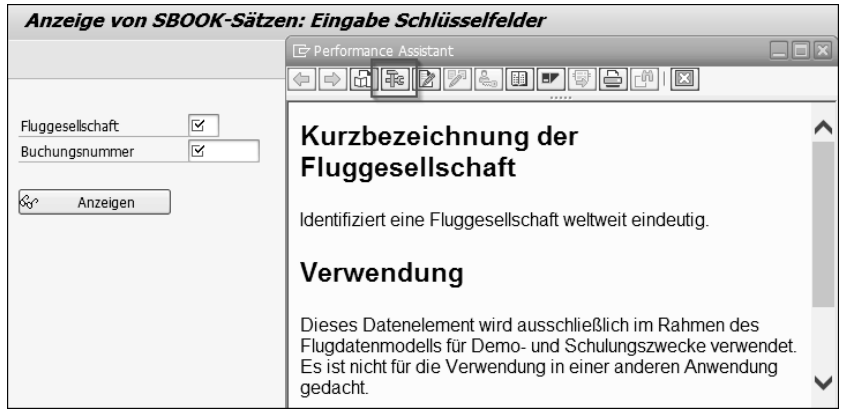


Abbildung 9.62 F1-Hilfe – Aufruf von technischen Informationen

Wenn Sie auf dieses Icon klicken, werden Ihnen zu diesem Dynpro-Feld die technischen Informationen angezeigt, u. a. der Name des Tabellenfeldes und die zugehörige Tabelle (siehe Abbildung 9.63).

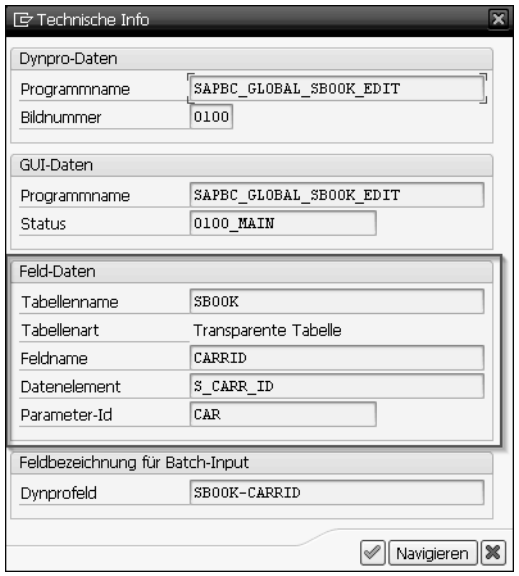


Abbildung 9.63 Technische Informationen zu einem Dynpro-Feld

Sie sehen, dass das Dynpro-Feld mit dem Tabellenfeld `CARRID` verbunden ist und dieses Feld zur Tabelle `SBOOK` gehört. Diese Information könnte dann von Interesse sein, wenn für ein solches Tabellenfeld ein Wertebereich hinterlegt ist. Das Protokoll aus Transaktion SLG1 (Anwendungslogs) zeigt Ihnen vielleicht in einer Meldung einen Wert an, der nicht in diesen Wertebereich passt. Dies wäre ein Zeichen von mangelhafter Datenqualität, die geklärt werden müsste. Dieser Weg der Analyse ist nur so lange hilfreich, wie ein Dynpro-Feld direkt mit einem Tabellenfeld verknüpft ist; dies ist aber nicht immer der Fall.

Einleitung

Für Neueinsteiger in die Produktwelt eines Standardsoftwareanbieters ist es zu Beginn nicht immer einfach, den geeigneten Einstiegspunkt zu finden, der einem quasi als Anker dient, von dem aus die weitere Erkundungstour gestartet werden kann. Die Suche nach dem Startpunkt hängt auch überwiegend von der Rolle im Berufsfeld ab, die dabei eingenommen wird. Entweder haben Sie für sich schon die Rolle definiert oder erst einmal den Rahmen, in welche Richtung es gehen soll. So können Sie sich z. B. in der Rolle eines SAP-Entwicklers, SAP-Administrators oder SAP-Anwenders befinden und suchen für sich den geeigneten Einstieg. Hatten Sie bis jetzt keine Berührungspunkte mit SAP-Anwendungen, sollten Sie erst das große Ganze verstehen, um dann für sich einen Schwerpunkt zu setzen.

Zielgruppe

Wir möchten all diejenigen ansprechen, die als Einsteiger mit der Entwicklung, Steuerung und Verwaltung von Software in einem SAP-System zu tun haben oder sich dafür interessieren. Dabei greifen wir die Themen auf, von denen wir sicher sind, dass sie praxisrelevant sind. Dazu gehören z. B. der Umgang mit dem ABAP Dictionary, die Entwicklung von Repository-Objekten, das Customizing, das Einrichten und Starten von Jobs, die Vergabe von Berechtigungen, Features von SAP Business Warehouse und vieles mehr. Die Themen bauen meistens aufeinander auf, sodass wir auch Themen aufgenommen haben – wie beispielsweise Kundenerweiterungen oder SAP HANA –, die einen gewissen Wissensstand voraussetzen. Diesen Wissensstand bauen Sie aber immer in den vorangehenden Kapiteln auf.

Ziel des Buches

Wenn Sie sich nach dem Lesen dieses Buches die typische Einsteigerfrage, »Und wie mache ich das?«, selbst beantworten können oder zumindest wissen, wie Sie die noch fehlende Information beschaffen, haben wir als Autoren unser Ziel erreicht. SAP bietet eine Fülle von Themen, die in eingehender Tiefe behandelt werden können. Wollte man dem gerecht werden, müsste es ein Buch in mehreren Bänden geben. Für einen schnellen und soliden Überblick über die vielen Themen wäre das zu umfangreich. Somit gibt es in diesem Buch Lücken. Wir hoffen aber, bei noch offenen Fragen Ihr Interesse geweckt zu haben, und möchten Sie dazu ermuntern, eigenständig Recherche zu betreiben. Denkanstöße, diverse Quellen und das Vorgehen dazu liefert das Buch in jedem Fall.

Ein SAP-System kann auf Java oder ABAP basieren. Wir gehen in diesem Buch von einem ABAP-Applikationsserver aus. Diese Konstellation finden Sie in der Praxis immer noch am häufigsten vor. Die Programmiersprache ABAP erläutern wir jedoch bewusst nicht, weil es schon viele Einstiegsbücher zu ABAP gibt.

Zum Aufbau des Buches

Für diejenigen, die mit der SAP-Welt noch nicht vertraut sind, möchten wir in **Kapitel 1**, »Die Welt von SAP«, zunächst SAP und dessen Produkte vorstellen. Danach geben wir Ihnen einen Überblick über die Architektur eines SAP-Systems und gehen auf die Bedeutung einer angeschlossenen Datenbank ein. Dieses Grundwissen wird Sie in den folgenden Kapiteln immer wieder begleiten.

Kapitel 2, »Der Einstieg ins System«, führt Sie in den Umgang mit einem SAP-System ein. Wir erläutern Ihnen, was unter SAP GUI und SAP-Logon zu verstehen ist und welche Einstellmöglichkeiten Sie im SAP GUI haben. Die Standardeinstiegsoberfläche in ein SAP-System ist immer SAP Easy Access. Wir machen Sie in diesem Kapitel mit der Oberfläche von SAP Easy Access vertraut. Welches sind die wichtigsten Funktions-Icons, was sind Favoriten, wie erhalten Sie Informationen über das System, wie können Sie das SAP-Menü um ein selbst gestaltetes Menü erweitern, und was hat es mit der Benutzeroberfläche SAP Fiori auf sich?

Mandanten sind in einem SAP-System der oberste Ordnungsbegriff. Wir zeigen Ihnen in **Kapitel 3**, »Mandanten«, über welchen Transaktionscode Mandanten gepflegt werden und welche Bedeutung die Zuordnung von Mandantenrollen für eine SAP-Systemlandschaft hat. Ein SAP-System verfügt über eine ausgereifte Benutzerverwaltung. Daneben können Sie sehr detailliert Berechtigungen vergeben. **Kapitel 4**, »SAP-Berechtigungen«, bietet genügend Stoff für ein eigenes Buch. Wir wollen dieses Thema jedoch nicht ausklammern und zeigen Ihnen anhand eines einfachen Beispiels, wie Sie für einen Benutzer Berechtigungen einrichten können.

Kapitel 5, »ABAP-Dictionary-Objekte«, behandelt das ABAP Dictionary. Das Kapitel ist sehr umfangreich, womit zum Ausdruck kommt, welch große Bedeutung das ABAP Dictionary für ein SAP-System hat. Das ABAP Dictionary schafft die Grundlage einer einheitlichen Datenbasis und ist eng mit der Entwicklung von ABAP-Programmen verwoben.

Kapitel 6, »SAP-Entwicklungsobjekte«, stellt Ihnen die Entwicklungsobjekte in einem SAP-System vor. Wir haben uns auf die wichtigsten Entwicklungsobjekte beschränkt. Wie wir zu Beginn der Einleitung zu diesem Buch erwähnt haben, erläutern wir Ihnen nicht die Programmiersprache ABAP. Daher gehen wir bei den Entwicklungsobjekten auch nicht auf die unterschiedlichen Entwicklungstechniken ein. Für den technischen Umgang mit einem SAP-System sollten Ihnen die Entwicklungsobjekte aber geläufig sein. Außerdem sollte Ihnen bekannt sein, wie der Namensraum in einem SAP-System definiert ist und wie Sie Zugang zur Entwicklungsumgebung erhalten.

In **Kapitel 7**, »Transporte zwischen SAP-Systemen«, erklären wir das Transportmanagement in einer dreistufigen SAP-Landschaft. Wir stellen verschiedene Arten von Transporten vor und gehen darauf ein, wie Sie sie anlegen und verwalten.

SAP verfügt über eine Versionsverwaltung, mit der Sie die Historie von Entwicklungsobjekten und Datenbankinhalten verfolgen können. Allerdings müssen dafür gewisse Voraussetzungen gegeben sein. Wir zeigen Ihnen in **Kapitel 8**, »Versionsverwaltung« wie Sie die spezifische Versionsverwaltung aufrufen können.

Wir haben **Kapitel 9**, »Programme starten«, genannt. Das hört sich sehr einfach an, und so, als ob es darüber nicht viel zu schreiben gebe. Aber auch dieses Kapitel ist umfangreich geworden. Sie haben verschiedene Möglichkeiten, um ein Programm zu starten, sei es in einem Dialog oder im Hintergrund. Wir gehen hier auch auf die Definition und Überwachung von Jobs und Prozessketten ein, außerdem auf Varianten, auf die Tabelle TVARVC und Tests. Als Letztes zeigen wir Ihnen Möglichkeiten der Fehleranalyse nach einem Programmabbruch.

In **Kapitel 10**, »Prozesse«, geht es um Prozesse, die Bestandteil der Architektur eines ABAP-Applikationsservers sind. In ihnen läuft Programmcode ab. Es gibt unterschiedliche Typen von Prozessen, denen jeweils spezifische Aufgaben zugeteilt sind. Für die Definition einer Parallelverarbeitung ist dieses Wissen unabdingbar.

Kapitel 11, »Customizing«, dreht sich um das Customizing, das in einem SAP-System von großer Bedeutung ist. SAP bietet Ihnen ein Verfahren, um eigenes Customizing zu entwickeln und es in das Standard-Customizing zu integrieren. An einem Beispiel zeigen wir Ihnen, wie Sie dabei vorgehen.

In **Kapitel 12**, »Kundeneigene Erweiterungen«, stellen wir verschiedene Möglichkeiten von Erweiterungen der SAP-Standardsoftware vor. Dabei beschreiben wir verschiedene Typen wie User-Exits, Business Transaction Events und Business Add-Ins (BAdIs).

In **Kapitel 13**, »Business Application Programming Interface«, zeigen wir Ihnen Möglichkeiten zum externen Aufruf von Funktionen der SAP-Standardsoftware auf, mit denen Daten übertragen werden: Business Application Programming Interfaces (BAPIs).

In **Kapitel 14**, »SAP Business Warehouse und Data Warehousing Workbench«, stellen wir Ihnen kurz SAP Business Warehouse (SAP BW) vor und beschreiben dessen wichtigsten Prozesse. Wir gehen dabei auch auf die Modellierung in der Data Warehousing Workbench, auf die InfoObjects und den InfoProvider ein.

In **Kapitel 15**, »Daten auswerten«, beschreiben wir, wie Sie die Daten, die innerhalb des SAP-Systems erzeugt und gespeichert werden, auswerten können.

In **Kapitel 16**, »SAP Business Workflow«, stellen wir Ihnen das Konzept und die Architektur von SAP Business Workflow vor, ein Werkzeug, mit dem Sie Arbeitsabläufe im SAP-System steuern und dadurch vereinfachen können.

In **Kapitel 17**, »SAP HANA – eine nähere Betrachtung«, durchleuchten wir SAP HANA und das dahinterstehende Konzept. Dabei gehen wir auch auf die damit verbundenen Themen In-Memory-Computing und Big Data ein.

In **Kapitel 18**, »SAP Support Portal«, gehen wir auf das Angebot des SAP Service Marketplace ein und beschreiben seine Anwendung.

Im **Anhang** finden Sie zum Nachschlagen ein Glossar, eine Liste mit Abkürzungen sowie Literaturtipps.

[»] In Kästen, die mit dem Hinweissymbol gekennzeichnet sind, finden Sie Informationen zu weiterführenden Themen oder wichtigen Inhalten, die Sie sich merken sollten.

[+] Die mit dem Tippsymbol gekennzeichneten Kästen geben spezielle Empfehlungen, die Ihnen die Arbeit erleichtern können.

[!] In den Kästen, die mit diesem Icon markiert sind, erklären wir Ihnen, wo sie besonders achtsam sein sollten.

Außerdem finden Sie auf der Website zum Buch, *www.sap-press.de/4788*, unter **Materialien** weitere hilfreiche Informationen, z. B. zur mehrsprachigen Entwicklung, zur Layoutanpassung und zur Performanceanalyse.

Danksagung

Wir möchten uns an dieser Stelle bei Christian Boxberger, Matthias Weber, Siu-Tung Cheung und Christoph Blenckner für ihre Anmerkungen und die konstruktive Kritik zu den einzelnen Kapiteln bedanken. Des Weiteren gilt unser Dank Steffi Spiesecke – ohne ihre vor Jahren geäußerte Aufforderung, »Kannst du nicht mal einen Einführungsleitfaden schreiben?«, wäre dieses Buch wohl nie entstanden. Auch möchten wir besonders unseren Ehefrauen und Kindern für ihre Geduld und ihren Verzicht auf unser Familienleben während des Schreibens dieses Buches danken. An dieser Stelle bedanken wir uns auch gerne bei unseren Kunden; schließlich lernt man am besten durch *Learning by Doing* in Projekten. Des Weiteren gilt unser Dank den Lektoren des Rheinwerk Verlags: Kerstin Billen, Janina Karrasch und Stefan Thißen – für ihre Unterstützung und ihr Entgegenkommen.

Auf einen Blick

Inhalt