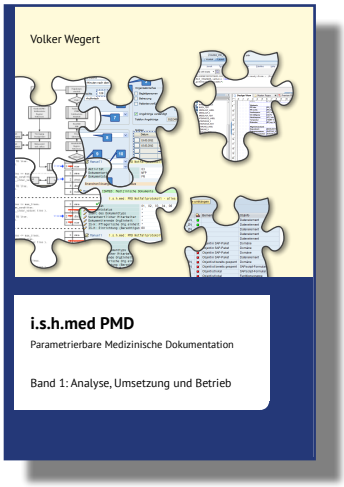


Aus dem Inhalt:

Analyse und Entwurf
- Einsatzbereiche und Prozessübersicht
- Funktionsumfang
- Ablauf der Analyse

Entwicklungsumgebung
- Konzepte und Objekte
- ABAP für Umsteiger
- ABAP Objects

Umsetzung, Betrieb und Weiterentwicklung
- Planung der technischen Umsetzung
- Dokumentarten
- Dokumentationselemente und Dokumenttypen
- Werteversorgung
- Einfache Zusatzprogrammierung
- Gestaltung der Druckausgabe
- Test, Produktivsetzung und Betrieb
- Anpassung und Weiterentwicklung
- Dokumenttypen kopieren und verteilen

Volker Wegert

i.s.h.med PMD
Band 1: Analyse, Umsetzung und Betrieb

Dieses Buch bietet eine umfassende Einführung in den Entwurfs- und Entwicklungsprozess PMD-basierter Anwendungen. Sie erhalten grundlegende Informationen über die Einsatzbereiche, Möglichkeiten und Grenzen der PMD sowie einen Überblick über die Arbeitsschritte, die zur Erstellung eines PMD-Dokumenttyps erforderlich sind. Anhand eines durchgängig verwendeten Beispiels werden die Schritte erläutert, die zur Erarbeitung eines umsetzbaren Entwurfs erforderlich sind. Das Buch enthält eine Einführung in die ABAP-Entwicklungsumgebung für Leser, die noch nie oder nur in geringem Umfang mit den Entwicklungswerkzeugen des SAP NetWeaver-Systems in Berührung gekommen sind. Anschließend wird aus dem Entwurf schrittweise ein einsetzbarer PMD-Dokumenttyp erstellt. Anhand des Beispiels aus dem ersten Teil erlernen Sie die Anlage und Einstellung der verschiedenen Objekte, die Erweiterung der Anwendung durch zusätzliche Programmierung sowie die Erstellung der Druckausgabe. Auf die Schritte, die zur erfolgreichen Inbetriebnahme eines Dokumenttyps erforderlich sind, wird ebenso eingegangen wie auf die Anpassung und Weiterentwicklung bestehender Dokumenttypen. Ein umfangreicher Anhang mit ausführlichem Schlagwortverzeichnis erleichtert die tägliche Arbeit mit der PMD.

621 Seiten, Hardcover, 2014, 69,90 €
ISBN 978-3-8495-9921-8
http://www.pmd-buch.de

Teil I.

Analyse und Entwurf

Dieser Teil des Buchs richtet sich an Fachanwender und Systembetreuer, die an der Erstellung von Dokumentationsanwendungen beteiligt sind.
*Sie erhalten in **Kapitel 1** grundlegende Informationen über die Einsatzbereiche, Möglichkeiten und Grenzen der Parametrierbaren Medizinischen Dokumentation (PMD) in Siemens i.s.h.med. Dabei werden die Vorzüge und Schwächen der strukturierten Dokumentation in Relation zu anderen Lösungsalternativen betrachtet. Gleichzeitig vermittelt das Kapitel einen Überblick über die Arbeitsschritte, die zur Erstellung eines PMD-Dokumenttyps erforderlich sind. Sie erfahren, welche Abhängigkeiten zwischen den Schritten bestehen und über welche Fähigkeiten die handelnden Personen verfügen müssen.*
*In **Kapitel 2** erhalten Sie detaillierte Informationen über den Funktionsumfang der PMD-Dokumenttypen und ihre Integration in die i.s.h.med-Standardanwendungen. Sie lernen weiterhin die Grundelemente der PMD kennen und erfahren, welche gestalterischen Möglichkeiten Ihnen beim Entwurf eines PMD-Dokumenttyps zur Verfügung stehen und welche Rahmenbedingungen Sie dabei zu beachten haben.*
*In **Kapitel 3** schließlich wird der Prozess, der zu einem umsetzungsreifen fachlichen Entwurf eines PMD-Dokumenttyps führt, thematisiert. Hier erfahren Sie anhand eines Beispiels, auf welche Aspekte Sie bei der Analyse achten müssen und lernen einige der typischen Problemstellungen und Fallen des PMD-Entwurfs kennen.*

2.2. Grundfunktionen der PMD

Im folgenden Abschnitt werden die Funktionen, die in jeder PMD-basierten Dokumentationsanwendung zur Verfügung stehen, vorgestellt. Zunächst gilt es aber einen der zentralen Begriffe der PMD-Entwicklung einzuführen.

2.2.1. Dokumenttyp

Jedes Dokument in i.s.h.med gehört zu genau einem Dokumenttyp, auch oft als *Doktyp* abgekürzt. Dieser Typ muss bei der Anlage des Dokuments angegeben werden und kann nachträglich nicht mehr geändert werden. Im Fall von PMD-Dokumenttypen legt der Dokumenttyp die Struktur des Dokuments sowie die zu verwendenden Zusatzprogrammierungen fest. Tatsächlich ist die Erstellung von PMD-Dokumentationsanwendungen in weiten Teilen mit dem Entwurf und der Realisierung von Dokumenttypen gleichzusetzen. Dokumenttypen können ihrerseits versioniert werden, so dass der Schlüssel eines Dokumenttyps nur in Kombination mit der Version eindeutig ist.

Wie in Abschnitt 1.2.1.2 auf Seite 22 bereits angedeutet gibt es zwei Versionen der PMD-Programmierumgebung, die *Generatorversion 1* und *Generatorversion 2* genannt werden. Die Generatorversion 1 wird seit längerer Zeit nicht mehr weiterentwickelt und – je nach Release – auch nicht mehr gepflegt. Alle Informationen im folgenden Abschnitt beziehen sich auf die Generatorversion 2. Welche Generatorversion zum Einsatz kommt, wird neben vielen anderen Einstellungen durch den Dokumenttyp festgelegt.

2.2.2. Dokumentanlage und Prozessintegration

Dokumente können im Rahmen der klinischen Arbeitsabläufe auf vielfältige Weise erzeugt werden. Jederzeit möglich ist die manuelle Anlage eines Dokuments aus dem SAP- oder Benutzermenü, dem klinischen Arbeitsplatz und dem Patientenorganizer. Auch in den OP-Monitor und den Dokumentationsarbeitsplatz ist die Dokumentenanlage bzw. der Aufruf bestehender Dokumente integriert, wobei in diesen Anwendungen durch eine spezifischere Zuordnung der für den jeweiligen Anwendungsfall relevanten Dokumenttypen eine deutlich zielgenauere Benutzerführung möglich ist.

Dokumentanlage bei Leistungserbringung
2 / 6.5 / S. 239

Mit Hilfe einer kleineren Zusatzprogrammierung ist es möglich, ein Dokument automatisch bei der Erbringung einer Leistung anzulegen. Dies ist insbesondere im Kontext der Leistungskommunikation interessant, da im Anschluss an die Dokumentation der erbrachten Leistungen sofort die medizinische Dokumentation ausgelöst

wird. Bei Verwendung der alten Leistungsanforderung war es auch möglich, Dokumente als sogenannte *technische Zusatzdokumente zur Anforderung* bereits bei der Beauftragung der Leistung zu erstellen. Auf diese Weise konnten Daten, die zur Planung und Vorbereitung der Leistungserbringung erforderlich waren, frühzeitig elektronisch erfasst und übermittelt werden. Mit der Einführung des klinischen Auftrags besteht diese Möglichkeit leider nicht mehr, auch wenn verschiedene Projektlösungen kursieren, die die eigentlich nicht kompatiblen Technologien mehr oder weniger gelungen miteinander verknüpfen.

Dokumente können darüber hinaus aus beliebigen Kundenprogrammen angelegt werden. Dazu zählen insbesondere auch Schnittstellenprogramme, die Daten aus externen Subsystemen übernehmen und in Dokumentform ablegen. Solche Programme mussten in früheren Releases in der Regel neu erstellt bzw. durch Kopie aus bestehenden Programmen abgeleitet werden; in Bestandssystemen werden Sie daher noch sehr viele Schnittstellen finden, die auf dieser Technik beruhen. Seit kurzem gibt es mit der **Message Communication Infrastructure** (MCI) eine moderne, komponentenorientierte Schnittstellentechnologie, die in Band 2 genauer vorgestellt wird.

MCI
2 / 6.6 / S. 250

2.2.3. Bezugsdaten

Ein Dokument kann mit einem Patienten, einem Fall oder einer oder mehreren Bewegungen oder Leistungen verknüpft sein. Welche dieser Verknüpfungen in einem konkreten Fall zum Einsatz kommt, hängt von den Einstellungen zur **Bezugsebene** im Dokumenttyp ab. Folgende Einstellungen sind möglich:

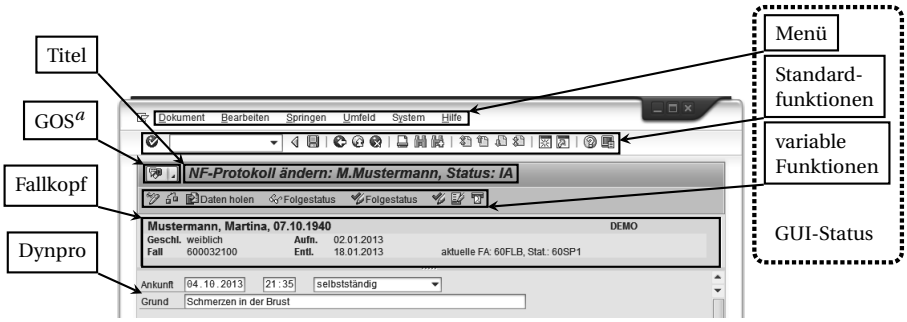
- **Patient** bedeutet, dass das Dokument nur dem Patienten zugeordnet ist, also fallübergreifend gilt;
- **Fall** bedeutet entsprechend, dass das Dokument einem Patienten und einem Fall dieses Patienten zugeordnet ist;
- **nur Fall** entspricht der Einstellung *Fall*, nur dass aus der Anwendung eventuell übergebene spezifischere Bezugsdaten (Bewegungen oder Leistungen) verworfen werden;
- **Bewegung** bedeutet, dass das Dokument je nach Einstellung *genau einer* oder *einer oder mehreren* Bewegungen eines Falls und damit automatisch einem Patienten zugeordnet ist;
- **Leistung** bedeutet, dass das Dokument je nach Einstellung *genau einer* oder *einer oder mehreren* Leistungen und damit automatisch den zugehörigen Bewegungen des Falls und dem Patienten zugeordnet ist.

An dieser Stelle sei angemerkt, dass die Bezugsebene *Patient* in der Praxis sehr viel sparsamer eingesetzt wird als man intuitiv vermuten mag. Das liegt insbesondere daran, dass ein Abschluss eines patientenbezogenen Dokuments problematisch ist, weil damit ein Fortschreiben bei Wiederaufnahme des Patienten erschwert wird. Gleichzeitig ist der Abschluss aber erforderlich, um die Daten vor rückwirkenden Manipulationen zu schützen.

Zur Steuerung der Bezugsdaten gibt es weiterhin das **Unikatskennzeichen**. Wenn dieses Kennzeichen gesetzt ist, stellt das System sicher, dass es auf einer konkreten Bezugsebene – also zu einem Fall, einer Leistung usw. – immer nur ein Dokument dieses Dokumenttyps gibt. Damit kann z. B. verhindert werden, dass ein bestimmtes zentrales Dokument zu einem Fall mehr als einmal angelegt wird oder dass es zu einer Leistung mehr als einen Befund eines bestimmten Dokumenttyps gibt. Umgekehrt ist es mit einer sog. **Zuordnungseinschränkung** möglich, ein Dokument mit maximal einer Bewegung oder Leistung verknüpft werden darf.

Die Bezugsdaten können verwendet werden, um in der Dokumentationsanwendung unterhalb der Drucktastenleiste einen Standard-Kopfdatensatz anzuzeigen. Obwohl dieser Anzeigebereich auch Patienten- und Bewegungsdaten anzeigen kann, ist er als **Fallkopf** bekannt. Gegenüber der manuellen Nachbildung dieser Kopfdaten mit eigenen Strukturen ist der Einsatz des Fallkopfs deutlich zeitsparender möglich. Dafür ist es nicht ohne größeren Aufwand möglich, den Anzeigebereich dokumentspezifisch selbst zu gestalten. Abbildung 2.1 zeigt den Fallkopf und benennt gleichzeitig die anderen Standard-Elemente der SAP GUI.

Fallkopf in Dialogeinstellungen 11.3.3 / S. 298



[a] GOS = Generische Objektdienste

Abbildung 2.1: Kopfbereich des Fensters mit Fallkopf und anderen Elementen

2.2.4. Datenverwaltung

Während der Erstellung eines neuen PMD-Dokumenttyps werden automatisch alle Datenstrukturen (Datenbanktabellen) erstellt, die zur Ablage der Dokumentinhalte

erforderlich sind. Die Struktur der notwendigen Tabellen wird direkt aus dem Aufbau des Dokumenttyps abgeleitet. In Abschnitt 2.3 auf Seite 52 werden die einzelnen Elemente, aus denen sich diese Strukturen zusammensetzen, im Detail besprochen. Alle notwendigen Lese- und Schreibzugriffe werden von der generierten Anwendung automatisch erledigt.

Jede Änderung in der inneren Struktur – zumindest der datenhaltigen Elemente – zieht eine Änderung der Datenbanktabellen nach sich. Solange es sich nur um Erweiterungen handelt, kann dies vom System automatisch erledigt werden; bei Feldänderungen oder komplexeren Umstrukturierungen fällt mehr Aufwand an. In Kapitel 16 auf Seite 535 wird im Detail auf die dazu notwendigen Schritte eingegangen.

Die generierte Dokumentationsanwendung bzw. die Dokumentenverwaltung erhalten bereits eine Sperrlogik, die den gleichzeitigen schreibenden Zugriff auf ein Dokument durch mehrere Benutzer unterbindet. Diese Programmteile können nicht ohne großen Aufwand oder unangenehme Nebenwirkungen angepasst werden – glücklicherweise ist das in der Regel auch nicht erforderlich.

2.2.5. Statusverwaltung und Versionierung

In i.s.h.med werden einige Aspekte der Dokumentenverarbeitung an eine anwendungsunabhängige Basiskomponente delegiert. Wenn Sie sich schon mit der Dokumentenverwaltung in anderen Anwendungen – zum Beispiel in der Materialwirtschaft oder im Projektplanungssystem – beschäftigt haben, werden Ihnen die folgenden Begriffe daher vertraut sein.

Die Dokumentenverwaltung bietet vielfältige Möglichkeiten der Status-, Versions- und Berechtigungssteuerung. Zentrales Steuerelement ist dabei die **Dokumentart**. Sie kategorisiert Dokumente und legt verschiedene Grundeinstellungen fest. Die Vergabe der Dokumentnummer wird über die Dokumentart gesteuert, ebenso wie einige andere Optionen, die im Umfeld der PMD allerdings selten genutzt werden. Weiterhin kann anhand der Dokumentart eine Berechtigungssteuerung realisiert werden. Die Einstellungsmöglichkeiten der Dokumentart werden im Detail in Kapitel 9 auf Seite 253 erläutert. In i.s.h.med wird die Dokumentart eines Dokuments im Dokumenttyp festgelegt. Bei der Anlage wird die Dokumentart in den Schlüssel des Dokuments kopiert.

Durch die Kopie der Dokumentart in den Schlüssel kann es dazu kommen, dass bei nachträglicher Änderung der Dokumentart im Dokumenttyp die tatsächlich angelegten Dokumente eine andere Dokumentart tragen als im Dokumenttyp angegeben. Das kann wiederum zu Folgefehlern führen,

etwa bei der Anlage einer Dokumentversion.

Die Dokumentart legt insbesondere das sogenannte **Statusnetz**, d. h. die möglichen Status eines Dokuments und deren Verknüpfung untereinander fest. Ein Dokument wird immer in einem durch die Dokumentart festgelegten Initialstatus angelegt. Zu jedem weiteren Status innerhalb der Dokumentart kann es maximal sechs Vorgängerstatus geben, man definiert die Statusverknüpfung also gewissermaßen rückwärts. Aus diesen Informationen ermittelt das System dann zur Laufzeit, welches die möglichen Folgestatus für ein Dokument sind. Sobald ein Dokument einen **Freigabestatus** erreicht hat, gilt es als festgeschrieben und kann inhaltlich nicht mehr geändert werden. Es kann allerdings mehrere alternative oder auch sequentielle Freigabestatus geben. So kann zum Beispiel auf einen Status *inhaltlich abgeschlossen* ein zweiter Freigabestatus *gedruckt und versandt* folgen. Abbildung 2.2 zeigt einen Entwurf[1] eines solchen Statusnetzes, wobei die Freigabestatus hier grau schattiert dargestellt werden.

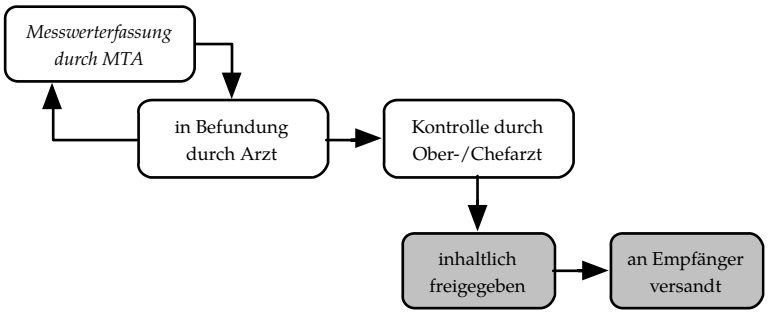


Abbildung 2.2: Beispiel eines Statusnetzes

Sollte eine inhaltliche Korrektur eines freigegebenen Dokuments erforderlich werden, muss eine neue **Version** angelegt werden. Dabei wird eine echte Kopie aller Dokumenteninhalte erzeugt, sodass jederzeit auf den alten Stand zurückgegriffen werden kann (Revisionssicherheit). Diese Kopierfunktion wird von der Dokumentationsanwendung automatisch bereitgestellt. Technisch werden die Versionen eines Dokuments unter der gleichen Dokumentnummer, aber mit einer neuen Versionsnummer (Schlüsselbestandteil) abgelegt. In den meisten Listen wird automatisch die jeweils aktuellste Version angezeigt, ggf. mit einer Navigationsmöglichkeit zu den Vorversionen.

Dokumente werden in i.s.h.med nicht unmittelbar gelöscht, sondern nur mit einem **Löschkennzeichen** versehen. Das Löschkennzeichen kann nur über die entsprechende Funktion in den Dokumentverwaltungsdaten zurückgenommen werden.

Dokumentverwaltungsdaten 2.2.10 / S. 49

[1] Dieser Entwurf wird in Abschnitt 9.1.2 auf Seite 254 noch einmal genauer besprochen.

Teil II.

Entwicklungsumgebung

Dieser Teil des Buchs richtet sich an Systembetreuer und Entwickler, die mit der technischen Umsetzung der im ersten Teil besprochenen Entwürfe beauftragt sind, aber noch nie oder nur in geringem Umfang mit den Entwicklungswerkzeugen des SAP NetWeaver-Systems in Berührung gekommen sind. Die folgenden Kapitel sind ***kein Ersatz für eine systematische Ausbildung****, etwa durch die Teilnahme am Schulungsprogramm der SAP. Sie sollen vielmehr die notwendigsten Strukturinformationen für angehende PMD-Entwickler liefern und erfahrenen ABAP-Entwicklern Hinweise auf die Teile der Entwicklungsumgebung geben, die in der PMD-Entwicklung üblicherweise verwendet werden. Die Kursbezeichnungen, die in diesem Teil des Buchs angeführt werden, beziehen sich auf das offizielle Schulungsprogramm der SAP AG.*
Sie erhalten zunächst in ***Kapitel 4*** *grundlegende Informationen über die Struktur des SAP NetWeaver-Systems und der Entwicklungsumgebung. Hier lernen Sie auch einige der häufig verwendeten Werkzeuge kennen.*
Kapitel 5 *erläutert die Objekttypen der Entwicklungsumgebung, mit denen Sie im Rahmen der PMD-Entwicklung in Kontakt kommen werden.*
Kapitel 6 *wagt den Versuch, Nutzern anderer Programmiersprachen die Sprache ABAP – die Sie zur Entwicklung von PMD-Dokumenttypen verwenden müssen – in möglichst kurzer und praxisnaher Form nahezubringen.*

5.3. Programme

Programmtypen

In der ABAP-Entwicklungsumgebung gibt es aus historischen Gründen eine ganze Reihe verschiedener Programmtypen. Im Folgenden werden nur die Typen erläutert, die für die PMD-Entwicklung erforderlich sind. Entwickler mit Vorkenntnissen in der ABAP-Entwicklung können die anderen Programmtypen natürlich auch nutzen, aber für den Einstieg reichen die hier benannten Typen aus.

⇒ [AB-PTY]

5.3.1. Reports (ausführbare Programme)

Ein Report ist ein direkt ausführbares Programm. Der Name ist für Einsteiger sicherlich maximal verwirrend, denn Reports enthalten nicht notwendigerweise Auswertungs- oder andere Reporting-Funktionen. Es sind einfach Programme, die – anders als die anderen Typen – direkt im Dialog gestartet oder zur Ausführung als Hintergrundjob eingeplant werden können. Sie können beliebige Programmanweisungen enthalten; so ist zum Beispiel der Mandantenrücksetzer `RNUMNDT0` als Report ausgeführt, könnte aber funktional kaum weiter von einer Auswertung entfernt sein: Er löscht konfigurierbar Daten aus einer IS-H-Installation, um sie für den Produktivbetrieb vorzubereiten.

In den meisten Oberflächen der Entwicklungsumgebung ist der Begriff *Report* mittlerweile durch *ausführbares Programm* ersetzt worden, aber die einleitende ABAP-Anweisung lautet nach wie vor `REPORT`, und die meisten ABAP-Entwickler sprechen auch noch von Reports. Abbildung 5.13 zeigt die Eigenschaften eines Reports. Beachten Sie bei der Arbeit immer das Feld *Typ*, denn dieser Eigenschaftsdialog wird auch für die anderen Programmtypen verwendet.

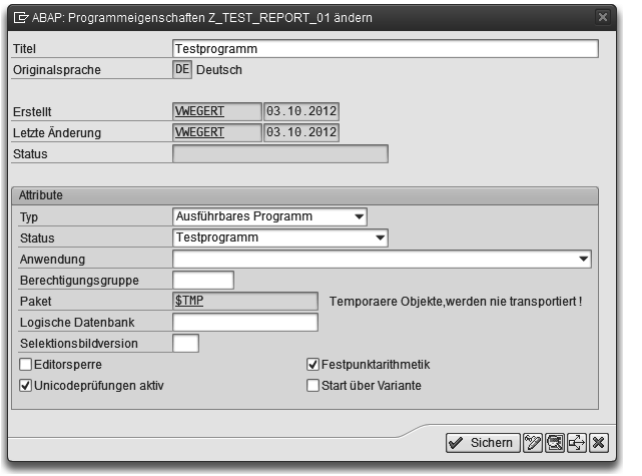


Abbildung 5.13: Eigenschaften eines Reports

Reports pflegen Sie über die ABAP Workbench oder die Transaktion `SE38`. Als Einsteiger in die PMD-Entwicklung müssen Sie zunächst keine Reports selbst schreiben. Erst in den fortgeschritteneren Szenarien in Band 2 benötigen Sie diesen Programmtyp. Reports eignen sich aber gut, um einfache Programmabläufe zu testen oder die in Kapitel 6 auf Seite 181 vorgestellten Sprachkonstrukte auszuprobieren. Abbildung 5.14 zeigt ein solches – definitiv nicht ernst zu nehmendes – Testprogramm.

ABAP Editor: Report Z_TEST_REPORT_01 ändern

Muster Pretty Printer

Report Z_TEST_REPORT_01 aktiv

```
1  *&---------------------------------------------------------------------*
2  *& Report  Z_TEST_REPORT_01
3  *&---------------------------------------------------------------------*
4
5  REPORT z_test_report_01.
6
7  CONSTANTS: co_num_bottles TYPE i VALUE 99.
8
9  DATA: g_counter TYPE i.
10
11 START-OF-SELECTION.
12   g_counter = co_num_bottles.
13   WHILE g_counter > 1.
14     WRITE: / g_counter, 'bottles of beer on the wall,'.
15     WRITE: / g_counter, 'bottles of beer.'.
16     WRITE: / 'Take one down, pass it around,'.
17     SUBTRACT 1 FROM g_counter.
18     WRITE: / g_counter, 'bottles of beer on the wall.'.
19   ENDWHILE.
20
21   WRITE: / 'No more bottles of beer on the wall,'.
22   WRITE: / 'no more bottles of beer.'.
23   WRITE: / 'Go to the store and buy some more,'.
24   g_counter = co_num_bottles.
25   WRITE: / g_counter, 'bottles of beer on the wall,'.
```

Abbildung 5.14: Quelltext eines Reports

Selektionsbilder

⇒ [AB-SEL]

Mit den deklarativen Anweisungen `PARAMETERS` und `SELECT-OPTIONS` kann ein Report mit einem sog. **Selektionsbild** versehen werden. Dieses kann mit den verschiedenen Varianten der Anweisung `SELECTION-SCREEN` weiter ausgestaltet werden. Dabei handelt es sich um ein automatisch generiertes Dynpro, das zur Parametereingabe verwendet wird. Wenn das Programm im Dialog gestartet wird, wird zunächst dieses Selektionsbild angezeigt. Von hier aus kann der Benutzer wählen, ob das Programm mit den eingegebenen Parametern im Dialog gestartet oder als Hintergrundjob eingeplant werden soll. Wenn das Programm kein Selektionsbild enthält, wird es sofort gestartet.

Programme, die Daten ändern, sollten immer ein Selektionsbild enthalten, damit sie nicht versehentlich gestartet werden.

Die auf dem Selektionsbild eingegebenen Werte können als **Selektionsvariante** gespeichert werden. Es können beliebig viele dieser Varianten abgelegt werden. In einer Variante können auch weitere Einstellungen vorgenommen werden: Felder können ausgeblendet oder deaktiviert werden, und für bestimmte Feldtypen sind dynamische Berechnungen (z. B. *Tagesdatum +/- 14 Tage*) einstellbar.

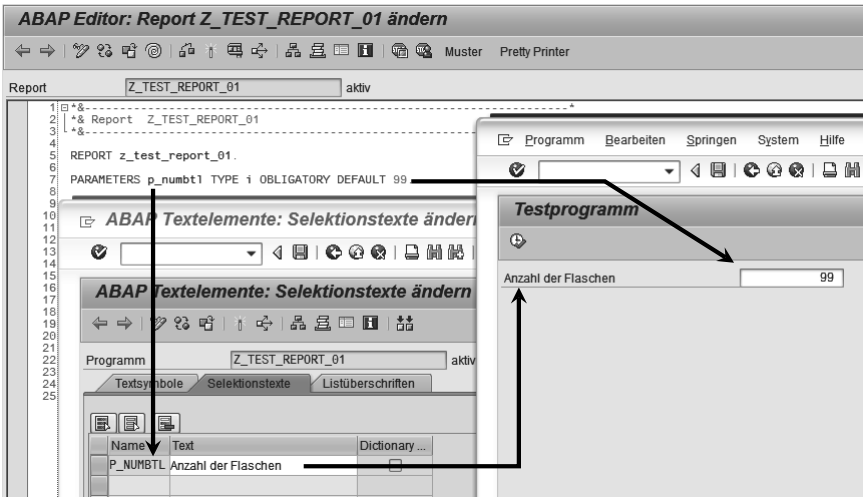


Abbildung 5.15: Selektionsbild eines Reports

Abbildung 5.15 zeigt ein Programm mit der Anweisung `PARAMETERS`, die Pflege der Selektionstexte und das daraus generierte Selektionsbild. Die Selektionstext pflegen Sie über den Menüpunkt Springen > Textelemente > Selektionstexte; sie werden in einer sprachabhängigen Tabelle abgelegt und sind an das Übersetzungssystem angeschlossen.

Programmablauf

Reports werden durch sog. **Zeitpunktanweisungen** gegliedert. Dabei handelt es sich um deklarative ABAP-Anweisungen, die keine eigene Funktion haben, sondern den Programmtext in Blöcke unterteilen und Einsprungpunkte für die Programmausführung bereitstellen. Der Programmablauf ist in der Online-Hilfe ausführlich beschrieben; hier genügt der Hinweis, dass der Programmteil nach `START-OF-SELECTION` nach der Dialog- oder Hintergrundausführung des Programms gestartet wird und somit dem üblichen Verständnis eines Hauptprogramms am nächsten kommt.

⇒ [AB-BLK]

Hintergrundverarbeitung

Prinzipiell kann jedes Programm in der **Hintergrundverarbeitung** gestartet werden. Es hängt dann natürlich von der inhaltlichen Ausgestaltung des Programms ab, ob es in der Hintergrundverarbeitung auch sinnvolle Resultate liefert. Bei einem Auswertungs- oder Datenverarbeitungsprogramm ist damit eher zu rechnen als bei einem Programm, das eine komplexe Dialogoberfläche zur Änderung von Daten bereitstellt. In der Hintergrundverarbeitung können auch Abfolgen von Programmen und komplexe periodische Ausführungsschemata definiert werden. So ist es prinzipiell möglich, ein geeignetes Auswertungsprogramm jeden Montag um 05:30 zu starten und das Ergebnis per Mail an einen Empfänger versenden zu lassen. Mit Hilfe der Fabrikkalender kann sogar festgelegt werden, wie sich das System verhalten soll, wenn der jeweilige Ausführungstag ein Feiertag ist.

⇒ [HP-JOB]

Teil III.

Umsetzung, Betrieb und Weiterentwicklung

Nach der umfangreichen Vorbereitung der vorhergehenden Kapitel befasst sich dieser Teil des Buchs mit den Arbeitsschritten, die aus einem Entwurf einen einsetzbaren PMD-Dokumenttyp werden lassen. Er richtet sich an Systembetreuer und Entwickler, die mit der technischen Umsetzung der im ersten Teil besprochenen Entwürfe beauftragt sind.
Die zur Umsetzung notwendigen technischen Vorüberlegungen werden in ***Kapitel 8*** *erläutert. Anschließend werden Sie durch die Anlage der Dokumentart (****Kapitel 9****), der Dokelemente (****Kapitel 10****) sowie schließlich des Dokumenttyps (****Kapitel 11****) geführt.*
In ***Kapitel 12*** *werden die eingebauten Möglichkeiten, Werte vorzubelegen und zu übernehmen, eingeführt. Da diese Funktionen in der Praxis selten ausreichen, erhalten Sie in* ***Kapitel 13*** *einen Überblick über die wichtigsten Möglichkeiten, die PMD-Anwendung durch zusätzliche Programmierung zu erweitern.* ***Kapitel 14*** *befasst sich mit der Erstellung und Erweiterung der Druckausgabe.*
In ***Kapitel 15*** *werden die Schritte erläutert, die erforderlich sind, um einen fertiggestellten PMD- Dokumenttyp auf die Folgesysteme zu übertragen und schließlich produktiv zu nutzen.* ***Kapitel 16*** *befasst sich mit den Besonderheiten, die bei der Anpassung eines bereits bestehenden Dokumenttyps zu beachten sind. In* ***Kapitel 17*** *lernen Sie die Möglichkeiten kennen, einen Dokumenttyp unverändert weiterzuverwenden oder eine Kopie zur weiteren Anpassung zu erstellen.*

11. Dokumenttypen

Dokumenttypen – oder kurz Doktypen – haben Sie in den einführenden Kapiteln bereits als zentrales Element der PMD und Träger der Struktur eines Dokuments kennengelernt. In diesem Abschnitt lernen Sie die Werkzeuge kennen, mit denen Sie einen Dokumenttypen bearbeiten können. Dabei wird die Definitionsoberfläche für Dokumenttypen zunächst anhand der Erstellung eines neuen Dokumenttyps eingeführt und anschließend in allen Funktionen vorgestellt.

Dokumenttyp 2.2.1 / S. 39

Dokumenttypen werden mit den Transaktionen `N2T6` (Anlage) und `N2T7` (Änderung) bearbeitet; weiterhin gibt es die Transaktion `N2T8` zur Anzeige. Die Kopierfunktionen werden in einem gesonderten Kapitel beschrieben, weil sie nicht ganz trivial zu verwenden sind.

Dokumenttyp kopieren 17 / S. 543

Dokumenttypen werden über einen zehnstelligen alphanumerischen Schlüssel identifiziert und mandanten- und einrichtungsbezogen abgelegt. Dabei gelten die bei der Planung angesprochenen Namensbereiche. Zum Schlüssel gehört weiterhin eine Versionsnummer, die zwar dreistellig definiert ist, aber nur zweistellig verwendet werden kann.

Namensräume und Schlüssel 8.3 / S. 244

Versionierung von Dokumenttypen 16.1 / S. 535

Der ausgelieferte Beispiel-Dokumenttyp mit dem Schlüssel `DEMONFP` befindet sich in einem Installationspaket, das mit der PMD-Transportworkbench installiert werden kann. Das Installationspaket ist seinerseits wiederum in einem BC-Set namens `/PMDBUCH/NFPROT_DEMONFP_V01` verpackt. In den folgenden Kapiteln finden Sie größtenteils Bilder und Beispieltexte, die sich auf Elemente im Kundennamensraum `Z...` beziehen; für den ausgelieferten Dokumenttyp wird für alle Entwicklungsumgebungsobjekte ein reservierter Namensraum und für die Dokelemente das Präfix `DEMONFP` verwendet.

PMD-Transportworkbench 15.4.2 / S. 524

11.1. Einführung in die Definitionsumgebung

Nach der Eingabe der Schlüsseldaten eines Dokumenttyps auf dem Einstiegsbild der Transaktionen `N2T6`, `N2T7` oder `N2T8` erreichen Sie zunächst den in Abbildung 11.1 gezeigten Dialog, in dem allgemeine Eigenschaften des Dokumenttyps festgelegt werden. Dieser Dialog ist für alle Dokumenttypen identisch, gilt also auch für Word

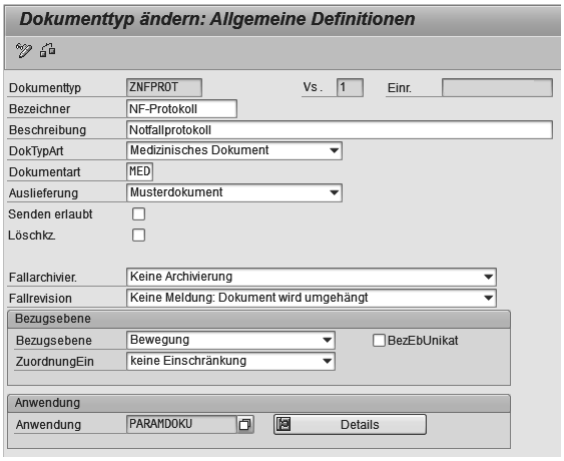


Abbildung 11.1: Allgemeine Eigenschaften jedes Dokumenttyps

Container, Dateiablagen oder andere unstrukturierte Dokumenttypen. Diese verwenden die gleichen Kopfdaten, aber andere Detaileinstellungen und werden hier nicht weiter behandelt.

Die Anwendung wird bei der Anlage des Dokumenttyps angegeben und beim ersten Speichern festgeschrieben. Sie ist – zumindest ab Release 6 EhP 6 – später nicht mehr änderbar. Auch in früheren Releases sollten Sie allerdings nicht versuchen, die Anwendung nachträglich zu ändern.

Nur wenn das Feld *Anwendung* den Wert `PARAMDOKU` enthält, handelt es sich bei dem Dokumenttyp um einen PMD-Dokumenttyp.

Kopfdaten
11.3.1 / S. 293

Die genaue Bedeutung der übrigen Optionen wird im nächsten Kapitel behandelt.

Mit der Drucktaste Details hinter dem Feld *Anwendung* erreichen Sie die in Abbildung 11.2 dargestellte PMD-Definitionsumgebung, in der alle weiteren Arbeiten stattfinden. Diese Drucktaste ist leider mit keiner Funktionstaste verknüpft, so dass Tastaturliebhaber hier immer zur Maus greifen müssen.

Vor Release 6 gab es hier noch mehrere Felder zur Angabe verschiedener Anwendungen für unterschiedliche Aktionen; diese sind korrekterweise in die internen Kopfdaten des Dokumenttyps verlagert worden. Wenn Sie sich noch auf einem solchen System bewegen, achten Sie darauf, die in diesen Releases noch unbeschriftete Drucktaste hinter einem Feld zu betätigen, in dem `PARAMDOKU` steht.

Die Definitionsumgebung ist eine typische Master-Detail-Arbeitsumgebung. Mit einem Doppelklick auf den Knoten in der linken Baumanzeige wählen Sie das Objekt bzw. die Sicht aus, die auf der rechten Seite angezeigt werden sollen. Auf der linken

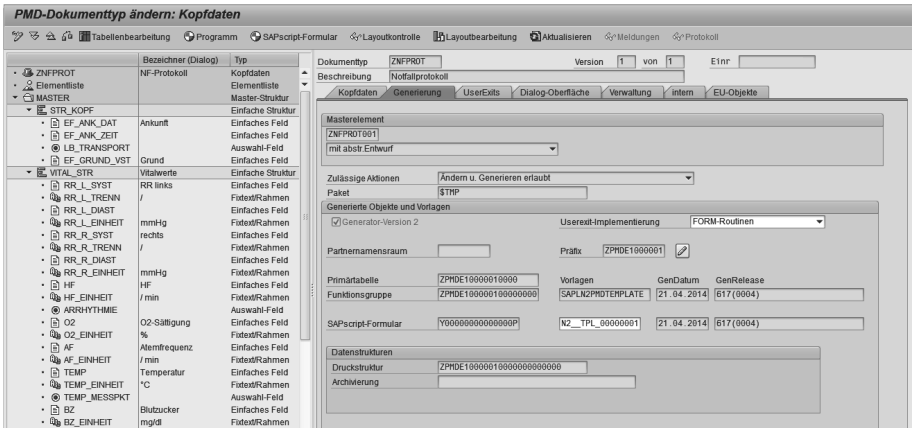


Abbildung 11.2: Gesamtansicht der Doktyp-Definition

Seite finden Sie zwei fixe Einträge, die die Kopfdaten und die Elementliste aufrufen. Darunter werden, beginnend beim Master-Element, alle Dokelemente in ihrer strukturellen Anordnung aufgeführt. Wenn Sie einen neuen Dokumenttyp anlegen, sieht die Sicht etwas anders aus: Der linke Teil ist leer und wird erst gefüllt, nachdem Sie ein Master-Element angegeben und den Dokumenttyp gesichert haben.

In der Baumansicht der Dokumenttyp-Struktur ist es möglich, die Reihenfolge der Spalten mit Ausnahme der führenden Hierarchiespalte zu ändern. Beim Start der Definitionsumgebung wird hinter dem technischen Namen des Felds der Dialogbezeichner angezeigt. Dies ist in der Praxis oft wenig hilfreich, weil der Dialogbezeichner oft ausgeblendet oder abgekürzt wird oder nicht eindeutig ist. Sie können aber einfach die Spalte *Kurzbezeichner (int)* nach vorne ziehen und so die Bezeichnungen der Dokelemente in der Übersicht anzeigen. Leider wird diese Einstellung nicht gespeichert, sondern muss bei jedem Start der Definitionsumgebung erneut vorgenommen werden.

Der Knoten *Elementliste* in der Baumanzeige und die Knoten unterhalb des Master-Elements verweisen auf die gleichen Daten: Sie bearbeiten jeweils die Eigenschaften der Dokelemente in diesem konkreten Dokumenttyp. Diese Zusammenstellung wird auch als **konkreter Entwurf** bezeichnet.

Der konkrete Entwurf entsteht als echte Kopie des abstrakten Entwurfs: Die Dokelement-Struktur wird ausgelesen und in eine neue Struktur überführt, die deutlich mehr Einstellungsmöglichkeiten bietet. Änderungen an den Einstellungen im konkreten Entwurf haben keine Auswirkungen auf die Dokelement-Definitionen.

11.7. Gestaltung der Dialoganwendung

Bei der Gestaltung der Dialoganwendung können Sie drei Werkzeuge bzw. Verfahrensweisen verwenden, die Ihnen im Folgenden vorgestellt werden. Diese Werkzeuge ergänzen sich gegenseitig – je nach der zu erzielenden Änderung kann es sinnvoll sein, mit allen drei Werkzeugen im Wechsel zu arbeiten. Alle verfügbaren Verfahren zu kennen und zum richtigen Zeitpunkt nutzen zu können, ist eine wesentliche Grundvoraussetzung, um einen PMD-Dokumenttyp in kurzer Zeit und mit wenig Frustration erstellen zu können.

11.7.1. Layoutkontrolle

Die Layoutkontrolle gehört nicht zu den vorgenannten drei Verfahren; sie stellt ein übergreifendes Werkzeug zur Simulation der Dialoganwendung dar. Die Generierung der gesamten Dokumentationsanwendung ist relativ aufwändig; deshalb ist es nicht praktikabel, bei jeder kleinen Änderung der Dialoggestaltung die komplette Anwendung neu zu generieren, nur um den überarbeiteten Entwurf zu kontrollieren. Zu diesem Zweck wurde die Layoutkontrolle geschaffen.

Tabellenbearbeitung
11.6.2 / S. 313

Um die Layoutkontrolle verwenden zu können, müssen die zum Dokumenttyp gehörenden Datenbanktabellen generiert und aktiviert worden sein.

Die Layoutkontrolle generiert nur die Oberfläche gemäß der eingestellten Elementeigenschaften und zeigt diese mithilfe eines dynamisch erzeugten Programms an. Sie eignet sich daher sehr gut, um die Positionierung der Dynpro-Elemente wie Bezeichner, Eingabefelder und anderer Dialogelemente zu kontrollieren. Die Positionen und Größen dieser Elemente entsprechen genau denen der später generierten Dialoganwendung.

Als Rahmenprogramm zur Anzeige der ersten Layoutkontrolle dient der Report `RN2VIEW1`. Wenn eine zweite Layoutkontrolle gestartet wird, während die erste noch aktiv ist, wird der Report `RN2LAY_CTRL_nn` verwendet, wobei `nn` ab `02` bis `10` aufwärts zählt. Mehr als zehn gleichzeitige Layoutkontrollen sind nicht möglich. Zum Aufbau der Layoutkontrolle werden jeweils zwei Includes vom System generiert. Für den Report `RN2VIEW1` sind dies `RN2VIEWC` und `IN2LAY_FORM_01`, für die Reports `RN2LAY_CTRL_nn` sind es jeweils `IN2LAY_DATA_nn` und `IN2LAY_FORM_nn`.

Die Layoutkontrolle stellt nur ein funktionsloses Rahmenprogramm zur Verfügung, dessen Fähigkeiten gerade eben ausreichen, die generierten Dynpros anzuzeigen, zwischen ihnen zu wechseln und die Registerkarten-Navigation zu unterstützen. Alle weitergehenden Funktionen werden nicht realisiert. Das bedeutet insbesondere:

- Es werden keine Controls unterstützt. Mehrzeilige Editoren für Langtexte, erweiterte Texte und SAPscript-Texte sowie Linkelemente können deshalb nicht angezeigt werden. Stattdessen wird ein Rahmen angezeigt, der den vorgesehenen Darstellungsbereich markiert. Das Gleiche gilt für Subscreens.

- Es wird weder die Bildmodifikation noch irgendeine andere zusätzlich programmierte Logik ausgeführt. Das bedeutet, dass keine Ausblendungen, aber auch keine dynamische Symbol- und Textversorgung erfolgt. Einige Elemente können nur per Bildmodifikation mit Symbolen versehen werden; diese Elemente werden in der Layoutkontrolle immer ohne Symbole dargestellt.

- Es wird kein Fallkopf angezeigt, selbst wenn dieser aktiviert wurde. Wenn Sie den Fallkopf verwenden möchten, müssen Sie bei der Verwendung der Layoutkontrolle bedenken, dass Sie nicht den gesamten vermeintlich leeren Raum nutzen dürfen, sondern einige Zeilen[1] am Ende jedes Dynpros frei halten müssen, weil das gesamte Dynpro durch den Fallkopf nach unten verschoben wird.

Neben der optischen und inhaltlichen Überprüfung des Dialogentwurfs eignet sich die Layoutkontrolle auch, um Elementkollisionen frühzeitig zu erkennen. Dynpro-Elemente dürfen sich nicht überlagern, weil es ansonsten zu Fehlfunktionen und Abstürzen kommen kann. Wie im Rahmen der Erstellung des Dialogentwurfs bereits erläutert wurde, betrifft dieses Problem insbesondere Bezeichner, die üblicherweise in Proportionalschrift dargestellt werden, für die aber Platz für eine Darstellung in Nichtproportionalschrift reserviert werden muss[2].

Dialogentwurf 2.4 / S. 79

Wenn Sie die Layoutkontrolle über die Drucktaste aus der Definitionsumgebung starten, wird der Bildaufbau ohne weitere Prüfungen ausgeführt. Nur in besonders schweren Fällen von Kollisionen kommt es hierbei zu Fehlermeldungen oder gar Abstürzen. Dafür werden viele Kollisionen nicht erkannt. Bei der Generierung der Dialoganwendung werden daher verschärfte Prüfungen angewandt, die auch über den Menüpunkt Werkzeuge > Layoutkontrolle mit Bildprüfung veranlasst werden können.

Die Bildprüfung wird nicht von der PMD-Entwicklungsumgebung durchgeführt, sondern von einer Basiskomponente, die bei der Generierung von Dynpros verwendet werden muss. Diese Komponente ist nun leider so programmiert, dass sie bei der ersten gefundenen Fehlerkonstellation die gesamte Prüfung abbricht – Sie müssen sich daher leider von Fehlermeldung zu Fehlermeldung vorarbeiten. Außerdem sind die Meldungen dieser Komponente leider nicht immer einfach zu entschlüsseln. Am

[1] siehe Tabelle 2.2 auf Seite 85
[2] siehe Abbildung 2.27 auf Seite 86

Beispiel der in Abbildung 11.13 gezeigten Meldungen sei daher kurz erläutert, wie Sie zur Fehlersuche vorgehen können.

Abbildung 11.13: Typische Fehlermeldung der Bildprüfung

- Die Dynpronummer `0001` in der ersten Zeile deutet auf das Bild hin, auf dem der Fehler aufgetreten ist. Untersuchen Sie jeweils das Bild, dessen Dynpronummer unmittelbar vor der eigentlichen Fehlermeldung genannt wird.
- Der Stern vor dem technischen Feldnamen in der zweiten Zeile deutet darauf hin, dass die Kollision zwischen dem Bezeichner des genannten Felds und einem anderen Dynproelement aufgetreten ist. Wäre das Feld selbst betroffen, würde der Bezeichner ohne Stern angegeben werden[1].
- Um das Element zu identifizieren, gibt es folgende Optionen:
  - Durchsuchen Sie die Detailbilder aller Dialogelemente nach dem Element, das auf der Registerkarte *Dialog* den angegebenen Feldnamen aufweist. Dies ist zeitraubend und deshalb nicht empfehlenswert.
  - Wechseln Sie in der Elementliste auf die Registerkarte *intern: Screen-Infos* und suchen Sie nach der Zeile mit dem angegebenen Feldnamen. Beim Wechsel auf die anderen Registerkarten bleibt die vertikale Listenposition erhalten, sodass Sie das Element leicht identifizieren können. Leider gibt es auch hier keine Suchfunktion, sodass Sie die gesamte Liste durchgehen müssen.
  - Verwenden Sie die Druckfunktion – Dokumenttyp > Drucken bzw. Standard-Drucktaste – zur Ausgabe der Struktur mit den Zusatzinformationen *Dynpro-Feldname*. In der angezeigten Liste können Sie mithilfe der eingebauten Suchfunktion das betroffene Feld lokalisieren. Leider können Sie von hier nicht direkt zu dem betroffenen Element navigieren. Sie können aber den Schlüssel des Dokelements ablesen und – nachdem Sie die Druckausgabe verlassen haben – mithilfe der Markierungsfunktion Bearbeiten > Markieren > Elemente... dieses Element in der Strukturanzeige hervorheben.

[1] Es ist aber immer noch möglich, dass der Bezeichner mit dem von ihm beschriebenen Feld kollidiert.

- Den oder die Kollisionspartner können Sie aus der Meldung nicht entnehmen. Hier müssen Sie mit der Layoutkontrolle ohne Bildprüfung oder der grafischen Layoutbearbeitung auf die Suche nach den Mitverursachern der Kollision gehen.

11.7.2. Manuelle Positionierung

Die grundlegendste Möglichkeit, den Dialog zu gestalten, besteht darin, die Position jedes Elements sowie ggf. seines Bezeichners manuell festzulegen. Dabei können Sie auch festlegen, auf welchem Dynpro oder welcher Registerkarte ein Element erscheinen soll. Sie finden die dazu erforderlichen Einstellungsmöglichkeiten sowohl in der Elementliste auf der Registerkarte *Dialog-Positionen* (vgl. Abbildung 11.3 auf Seite 288) als auch im Detailbild jedes einzelnen Elements.

☞ Beachten Sie bei der Arbeit mit den Detailbildern, dass die Koordinaten- und Größenfelder nicht für alle Elementtypen gleich angeordnet sind. So befindet sich bei Eingabefeldern die Spalte links, rechts daneben die Breite und darunter die Zeile, während etwa bei Checkboxen die Zeile links und die Spalte rechts daneben steht (vgl. Abbildung 11.9 auf Seite 305). Diese unglückliche Anordnung führt leider auch bei erfahrenen Anwendern oft zu Fehlbedienungen.

Das Koordinatensystem der Dynpros beginnt links oben mit Zeile 1 und Spalte 1. Es werden grundsätzlich die absoluten Positionen der Elemente auf dem Dynpro angegeben. Dies gilt insbesondere für Elemente innerhalb eines Rahmens oder einer Struktur – es wird immer die absolute Koordinate angegeben und nicht etwa die relative Position zum umgebenden Rahmen. Bei Feldern, die auf Registerkarten dargestellt werden, wird die Position relativ zur oberen linken Ecke der Karte angegeben.

Die manuelle Positionierung aller Dialogelemente erfordert zur Gestaltung eines kompletten Dokumenttyps sehr viel Vorstellungskraft oder einen sehr präzise bemaßten Entwurf und ist daher für Einsteiger nicht empfehlenswert. Zur Korrektur einzelner Positionierungen ist es aber die schnellste Variante. Einige Einstellungen lassen sich gar nur so vornehmen, sodass Sie sich mit dieser Arbeitsweise unbedingt vertraut machen sollten. Auch zum Verständnis der automatischen Positionsberechnung ist eine gewisse Übung in der manuellen Positionierung hilfreich.

Positionsberechnung 11.7.4 / S. 331

Für bestimmte Aufgaben kann es sogar sehr sinnvoll sein, die Eigenschaften der Dokelemente direkt zu bearbeiten. So können Sie zum Beispiel die Ausrichtung von Elementen an einem einheitlichen Raster (vgl. Abbildung 2.26 auf Seite 84) leicht

realisieren, indem Sie einfach bei allen betroffenen Elementen die gleiche X-Position eintragen.

Eine weitere Aufgabe, bei der die Arbeit in der Elementliste empfehlenswert ist, ist die Sortierung von Tabellenspalten[1]. Dabei können Sie zu einem kleinen Trick greifen und sich in einer Tabellenkalkulation eine aufsteigende Reihe untereinander angeordneter Zahlen erzeugen, diese in die Zwischenablage kopieren und in die Elementliste einfügen (Abbildung 11.14).

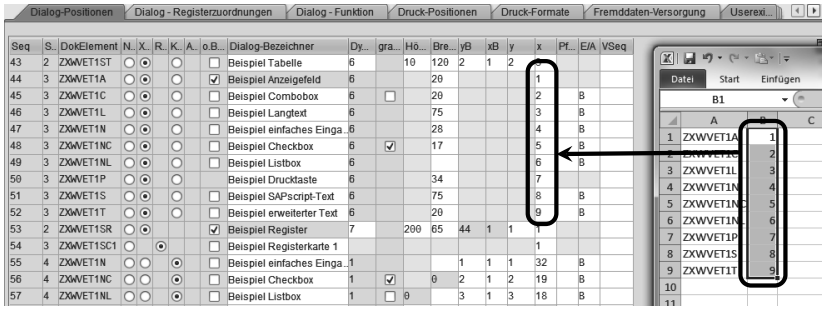


Abbildung 11.14: Generierung der Spaltenpositionen mit einer Tabellenkalkulation

11.7.3. Grafische Layoutbearbeitung

Die grafische Layoutbearbeitung (GLB, Abbildung 11.15) soll einen PMD-Entwickler in die Lage versetzen, schnell und komfortabel Änderungen am Dialogentwurf eines Dokumenttyps vorzunehmen. Leider bleibt dieses Werkzeug an vielen Stellen hinter seinem Anspruch zurück: Die Bedienung ist sehr gewöhnungsbedürftig und die Anwendung selbst fehleranfällig. Dennoch ist die grafische Layoutbearbeitung für die meisten Zwecke der schnellste und bequemste Weg, einen Dialogentwurf zu realisieren.

Aufgrund einer technischen Beschränkung kann die Grafische Layoutbearbeitung nicht die maximale Dynpro-Größe von 255 Spalten und 200 Zeilen ausnutzen, sondern nur einen Arbeitsbereich von 248 Spalten und 100 Zeilen anzeigen. Jenseits dieser Grenzen wird einfach nur ein grauer Block ohne Bearbeitungsmöglichkeit dargestellt.

Im Gegensatz zur Layoutkontrolle ist es für die Nutzung der grafischen Layoutbearbeitung nicht erforderlich, die Datenbanktabellen zu generieren. In der Praxis ist dies allerdings kein Vorteil: Aufgrund der fehlenden Kollisionsprüfung müssen

[1] Zur Erinnerung: Ohne fest angegebene Spaltenreihenfolge ist die Reihenfolge der Spalten von technischen Interna der Dokumenttyp-Anlage abhängig, im Zweifelsfall also unbrauchbar.

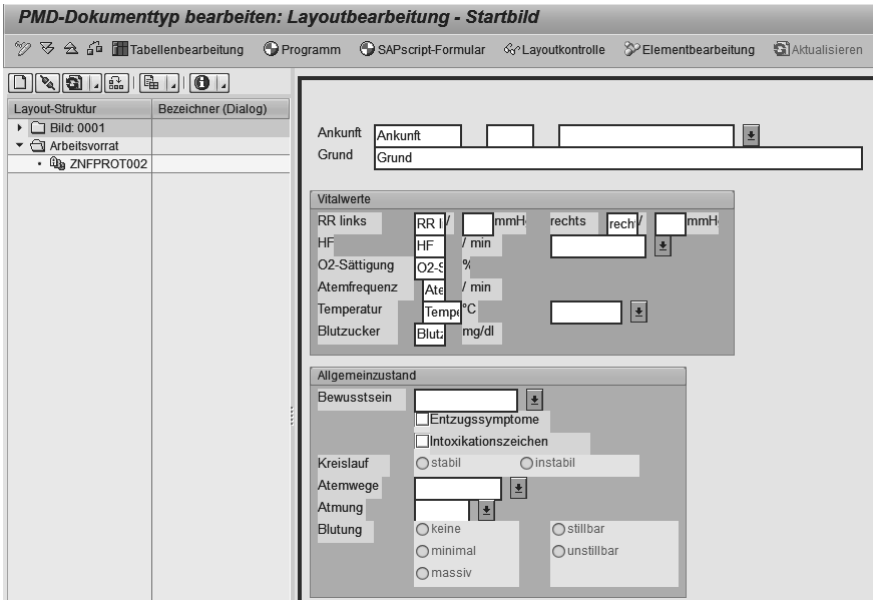


Abbildung 11.15: Grafische Layoutbearbeitung (Übersicht)

Sie auch während der Arbeit mit der GLB immer wieder die Layoutkontrolle mit Bildprüfung verwenden, und für diese benötigen Sie die Datenbanktabellen.

11.7.3.1. Installation

Die Komponenten der grafischen Layoutbearbeitung sind leider nicht wie z. B. die Plantafel in die SAP GUI-Installation integriert. Es gibt zwar ein in i.s.h.med integriertes System zur automatischen Installation von Desktop-Komponenten, das aber auf modernen Betriebssystemen mit eingeschränkten Berechtigungen erfahrungsgemäß nicht zuverlässig funktioniert. Sie müssen diese Programmteile daher auf jedem Rechner, auf dem Sie die grafische Layoutbearbeitung nutzen möchten, manuell nachinstallieren. Da es sich dabei aber in der Regel nur um wenige Entwickler-Arbeitsplätze handeln wird, hält sich der Aufwand in Grenzen.

Starten Sie zur Installation das Programm `RN2LN210`, z. B. mit der Transaktion `SA38` bzw. System > Dienste > Reporting. Wählen Sie aus der Liste nacheinander die folgenden Einträge aus und laden Sie die jeweilige Datei mit der Drucktaste Install bzw. F7 herunter.

- `ISHMEDGLBPAINT.SETUP`
- `ISHMEDGLB_BUT.SETUP`

- `ISHMEDGLB_CBOX.SETUP`
- `ISHMEDGLB_RBOX.SETUP`

Jede dieser Dateien ist ein selbstentpackendes Archiv, das die Installationsdateien enthält. Versehen Sie daher jede Datei mit der Endung `.exe` und führen Sie sie aus, um an das Installationspaket zu gelangen. Führen Sie anschließend die Installation mit lokaler Administratorberechtigung durch.

Die vier Pakete enthalten gleichnamige Dateien. Sie dürfen die Pakete daher nicht in das gleiche Verzeichnis entpacken, weil Sie ansonsten die Inhalte eines Pakets mit denen des folgenden Pakets teilweise überschreiben würden.

Aus dieser Anleitung ist implizit ersichtlich, dass die grafische Layoutbearbeitung nur innerhalb der SAP GUI für Windows zur Verfügung steht; andere Plattformen werden nicht unterstützt.

11.7.3.2. Aufruf

Die grafische Layoutbearbeitung erreichen Sie aus der Dokumenttypdefinition mit der Drucktaste [Layoutbearbeitung]. Mit der Drucktaste [Elementbearbeitung] wechseln Sie wieder zurück in die Elementlisten-Darstellung.

Mit der Standard-Drucktaste [⟲] bzw. [F3] verlassen Sie die Dokumenttypdefinition – unabhängig davon, ob Sie gerade im Layout- oder im Elementbearbeitungsmodus sind. Sie können also nicht *Zurück* in die Elementliste springen – eine sehr beliebte Fehlbedienung.

Ebenso wie im Elementbearbeitungsmodus ist der Arbeitsbereich der grafischen Layoutbearbeitung zweigeteilt (vgl. Abbildung 11.15). Der linke Teilbereich besteht aus einer Baumanzeige mit verschiedenen Funktionen und einem zusätzlichen Anzeigebereich (im Bild nicht dargestellt), der rechte Teil stellt die eigentliche Layoutbearbeitung dar.

11.7.3.3. Grafische Elementbearbeitung

In der Layoutbearbeitung werden die Elemente und Bezeichner in einer groben Annäherung ihrer endgültigen Anzeige abgebildet. Die Nummer des aktuell angezeigten Bilds finden Sie im Fenstertitel; um ein anderes Bild zu bearbeiten, doppelklicken Sie auf das entsprechende Ordnersymbol in der Baumanzeige auf der linken Seite.

13. Einfache Zusatzprogrammierung

Sie haben im vorhergehenden Kapitel die rein deklarativen Möglichkeiten, Dokumentinhalte bereitzustellen, kennengelernt. Angesichts des doch recht übersichtlichen Funktionsumfangs ist spätestens jetzt klar, dass selbst für einfachste Erweiterungswünsche wie eine Berechnung des Body Mass Index (BMI) auf andere Werkzeuge zurückgegriffen werden muss.

In diesem Kapitel erhalten Sie einen Überblick über die Programmierschnittstellen der generierten Dokumentationsanwendung. Dabei wird zunächst nur die Dialogumgebung betrachtet; der Druckausgabe ist aufgrund der verschiedenen verfügbaren Technologien ein eigenes Kapitel gewidmet.

Spätestens ab diesem Punkt benötigen Sie zumindest ABAP-Grundkenntnisse, um den Ausführungen folgen zu können. Lesen Sie falls notwendig die Einführung in Kapitel 6 auf Seite 181.

Sie können die Dokumentationsanwendung erweitern, indem Sie an bestimmten vorgegebenen Stellen zusätzliche Programmteile hinzufügen oder bestehende Programmteile ersetzen. Dazu erstellen Sie so genannte **User Exits**.

Die Bezeichnung *user exit* ist gleich in mehrerlei Hinsicht verwirrend: Die zusätzlichen Programmteile werden vollkommen benutzerunabhängig ausgeführt – es wäre also sinnvoller, von *customer exits* zu sprechen. Dieser Begriff ist allerdings bereits durch eine Vorläufertechnologie des heutigen Erweiterungskonzepts belegt. Wie Sie in Abschnitt 13.2 auf Seite 361 feststellen werden, sind die User Exits der PMD mit diesem Konzept nicht verwandt.

User Exits treten in zwei Gruppen auf:

- Die **dokumentbezogenen User Exits** werden in den Kopfdaten eines Dokumenttyps eingetragen. Sie existieren einmal für jeden Dokumenttyp.
- Die **elementbezogenen User Exits** werden in den Detaildaten einzelner Dokelemente innerhalb des Dokumenttyps eingetragen. Sie können daher innerhalb eines Dokumenttyps mehrfach und in verschiedenen Ausprägungen existieren. In der Regel beeinflussen sie nur das Element, bei dem sie eingetragen sind.

Dieses Kapitel beginnt mit einer Darstellung der internen Abläufe einer PMD-Dialoganwendung, wobei Sie die verschiedenen User Exits im Anwendungszusammenhang kennenlernen. Anschließend lernen Sie die technischen Grundkonzepte und die wichtigsten Methoden der Programmierschnittstellen für die Dialogumgebung kennen.

13.1. Programmablauf und Erweiterungsmöglichkeiten

Abbildung 13.1 zeigt – grob vereinfacht – den Ablauf der PMD-Dokumentbearbeitung. Je nach Aufrufart des Dokuments wird zunächst eine leicht unterschiedliche Initialisierungsroutine durchlaufen; dann wird das Dokument dem Benutzer angezeigt. Der Kontrollfluss wechselt jetzt in Form einer Schleife so lange zwischen System und Benutzer, bis der Benutzer das Dokument verlässt. In diesem Fall wird normalerweise noch ein weiterer abschließender Programmblock durchlaufen, in dem sich ein dokumentbezogener User Exit *Dokument-Ende* befindet. Dieser generelle Ablauf bildet das in Abschnitt 4.1 auf Seite 131 erläuterte Konzept der Dynpro-Umgebung ab.

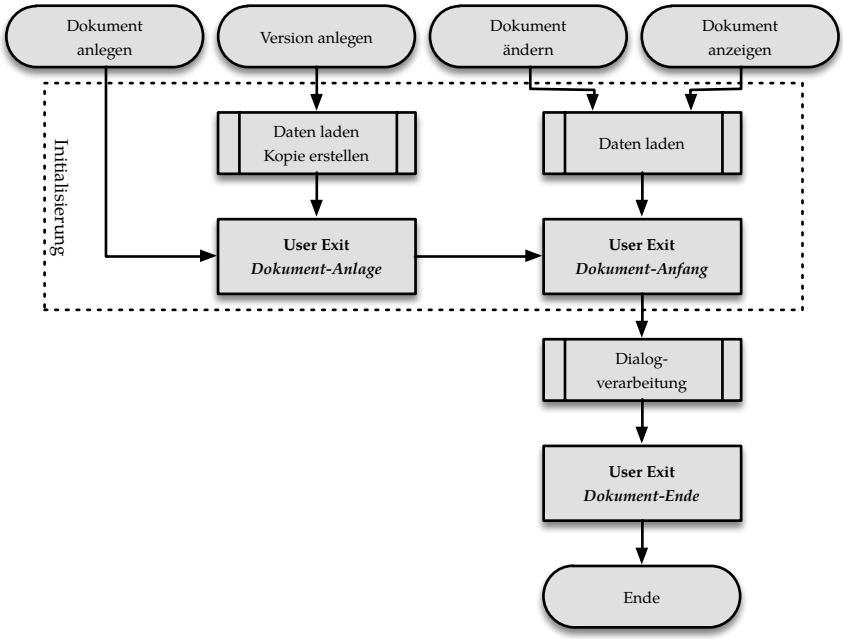


Abbildung 13.1: Übersichtsdarstellung des Dialogablaufs

Es ist möglich, das Dokument so zu verlassen, dass der abschließende Programmteil nicht durchlaufen wird!

Dies kann etwa dadurch erfolgen, dass der Benutzer mit dem Präfix `/n` im Transaktionscodefeld der SAP GUI die Transaktion abbricht und eine neue Transaktion startet, das Fenster schließt oder sich komplett abmeldet. In diesen Fällen wird überhaupt kein Programmteil des abgebrochenen Programms mehr durchlaufen. Sie dürfen sich bei der Entwicklung also **nicht** darauf verlassen, dass der abschließende Block in jedem Fall durchlaufen wird.

Der Initialisierungsblock ist ebenfalls in Abbildung 13.1 enthalten. Bei der Anlage eines neuen Dokuments oder einer Version eines bestehenden Dokuments wird zunächst der dokumentbezogene User Exit *Dokument-Anlage* durchlaufen; bei der Erstellung einer Dokumentversion werden vorher die Daten der Vorversion einkopiert. Beim Aufruf eines bestehenden Dokuments wird dieser User Exit nicht durchlaufen, stattdessen werden einfach die Dokumentdaten geladen. Anschließend wird für alle Aufrufarten der dokumentbezogene User Exit *Dokument-Anfang* durchlaufen.

Beachten Sie, dass bei der Anlage eines neuen Dokuments oder einer neuen Version der User Exit *Dokument-Anlage* **vor** dem User Exit *Dokument-Anfang* durchlaufen wird.

Diese Eigenschaft der PMD führt gerade bei Einsteigern oft zu Fehlern in der technischen Ausführung der Erweiterungen. Ein typisches Problem wird im Zusammenhang mit Katalogdatenzugriffen beschrieben.

Katalogdaten 13.4.7 / S. 410

Nachdem die Initialisierung der Dokumentationsanwendung abgeschlossen ist, beginnt die eingangs erwähnte Schleife zur Dokumentbearbeitung. Der Ablauf dieses Programmteils ist in Abbildung 13.2 dargestellt[1]. Zunächst wird in der sogenannten PBO-Verarbeitung[2] die Oberfläche zur Anzeige vorbereitet:

- Zunächst wird der dokumentbezogene User Exit *PBO* aufgerufen.
- Dann wird die PBO-Verarbeitung eventueller Linkbausteine, Registerkarten oder Subscreens – in dieser Reihenfolge – ausgeführt.
- Dann erfolgt die Versorgung der Anzeigefelder, durch die je nach Einstellung ein elementbezogener User Exit *Werteversorgung* aufgerufen wird.
- Abschließend wird die dynamische Bildgestaltung durch die Ausführung der elementbezogenen User Exits *Bildmodifikation* realisiert.

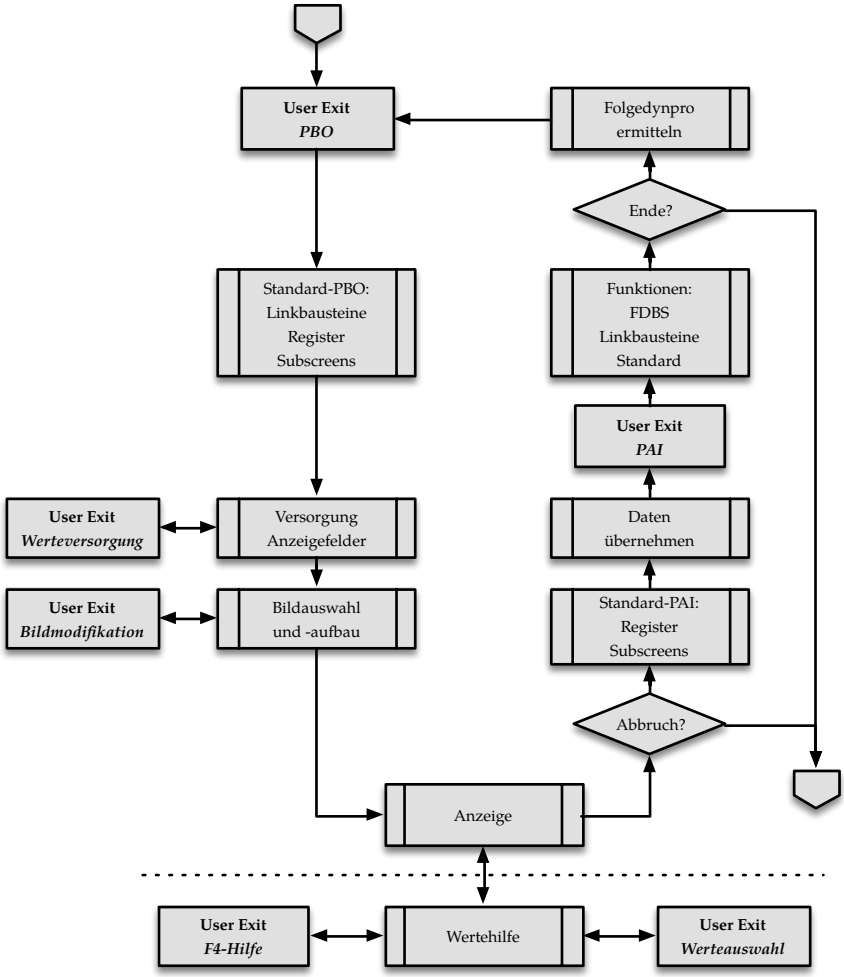


Abbildung 13.2: Bearbeitungsschleife der Dialoganwendung

Nachdem diese Verarbeitung abgeschlossen ist, wird das Dynpro samt Daten zur SAP GUI übertragen und dem Benutzer angezeigt.

Wenn der Benutzer eine Funktion auslöst, werden die eingegebenen Daten von der SAP GUI zusammengestellt und an den Server übertragen. In der sogenannten PAI-Verarbeitung[1] werden folgende Schritte ausgeführt:

[1] In den Abbildungen und den folgenden Erläuterungen fehlen der Übersicht halber einige interne Verarbeitungsschritte, die für das Verständnis hier nicht vonnöten sind.

[2] *process before output* = Verarbeitung vor der Dynpro-Ausgabe

[1] *process after input* = Verarbeitung nach der Dynpro-Eingabe

13.4.2. Patientendaten

Die API-Methode `api__get_npat` liefert die grundlegenden Patientendaten in Form einer `NPAT`-Struktur:

```
FORM z_mein_userexit RAISING cx_dynamic_check.

  DATA: ls_npat TYPE npat.

  ls_npat = <pub__service>->api__get_npat().

* ...

ENDFORM.
```

Wenn Sie nur die Patienten**nummer** benötigen, ist es schneller, diese mit `api__get_ndoc` aus den Dokumentverwaltungsdaten zu entnehmen.

Abgesehen vom Geschlecht und dem Geburtsdatum sowie den Namensfeldern, die in Tabelle 13.7 zusammengefasst sind, gibt es nur wenige Felder in der Tabelle `NPAT`, die von allgemeinem Interesse sind.

Feldname	Inhalt
`GSCHL`	Geschlecht (`1` = m, `2` = w, `3` = unbekannt)
`NNAME`	Nachname
`VNAME`	Vorname
`TITEL`	Titel (z. B. *Prof. Dr. med.*)
`NAMZU`	Namenszusatz (z. B. *Graf*)
`VORSW`	Vorsatzwort (z. B. *von*)
`GBDAT`	Geburtsdatum
`GBNAM`	Geburtsname

Tabelle 13.7: wichtige Felder der Tabelle `NPAT`

Wenn Sie den Patientennamen in zusammengesetzter Form benötigen, können Sie dazu den Funktionsbaustein `ISH_NPAT_CONCATENATE` verwenden. Auch diesem Baustein können Sie entweder die gesamte `NPAT`-Struktur oder nur die Patientennummer übergeben; achten Sie darauf, den Parameter `SS_READ_NPAT` entsprechend zu setzen.

```
FORM z_mein_userexit RAISING cx_dynamic_check.

  DATA: ls_npat         TYPE npat,
        l_patient_name  TYPE c LENGTH 160.

  ls_npat = <pub__service>->api__get_npat().

* Name zusammensetzen
  CALL FUNCTION 'ISH_NPAT_CONCATENATE'
    EXPORTING
      ss_patnr = ls_npat-patnr
```

```
12        ss_npat  = ls_npat
13      IMPORTING
14        ss_pname = l_patient_name.
15    ENDIF.
16  * Name: Dr. med. von Müller, Maximilian
17
18  * ...
19
20  ENDFORM.
```

Der Baustein `ISH_NPAT_CONCATENATE` setzt allerdings leider fix den Vornamen des Patienten hinter den Nachnamen – also *Dr. med. von Müller, Maximilian* – und eignet sich deshalb nur begrenzt zur Verwendung innerhalb von Dokumenttypen. Alternativ können Sie den Namen natürlich selbst zusammensetzen. Beachten Sie dabei die korrekte Reihenfolge der Namensbestandteile und vernachlässigen Sie insbesondere Namenszusatz und Vorsatzwort nicht, da es ansonsten zu unangenehmen und völlig unnötigen Rückfragen erfahrungsgemäß anspruchsvoller Patienten kommen kann. Die korrekte Reihenfolge ist im folgenden Beispiel angegeben:

```
1   FORM z_mein_userexit RAISING cx_dynamic_check.
2
3     DATA: ls_npat         TYPE npat,
4           l_patient_name  TYPE c LENGTH 160.
5
6     ls_npat = <pub__service>->api__get_npat().
7
8   * Name zusammensetzen
9     CONCATENATE ls_npat-titel  " Prof. Dr. med.
10                ls_npat-vname  " Maximilian
11                ls_npat-namzu  " Graf
12                ls_npat-vorsw  " von
13                ls_npat-nname  " Müllerstein
14      INTO l_patient_name SEPARATED BY space.
15    CONDENSE l_patient_name.
16
17  * ...
18
19  ENDFORM.
```

Die API-Methode `api__get_npat` liefert nur die wichtigsten Patientendaten. Unter Umständen benötigen Sie noch die einrichtungsbezogenen Patientendaten in der Struktur `NPAE` oder die Adressen des Patienten oder der verschiedenen Personen, die v. A. aus historischen Gründen in den Patientenstammdaten hinterlegt werden (Arbeitgeber und Angehörige). Um diese Daten zu erhalten, können Sie den Funktionsbaustein `ISH_READ_NPAT` verwenden[1]. Der Aufruf erfolgt analog zu den bisher vorgestellten Funktionsbausteinen und wird hier deshalb nicht noch einmal abgedruckt.

[1] Nichts anderes macht letztendlich die API-Methode, nur wird von den vielen Rückgabewerten nur die `NPAT`-Struktur an den Aufrufer zurückgegeben.

13.4.3. Falldaten

Die API-Methode `api__get_nfal` liefert die grundlegenden Falldaten in Form einer `NFAL`-Struktur, sofern das Dokument mit einem Fall verknüpft ist. Wenn das Dokument nur mit Patientenbezug angelegt wurde, bleibt die Struktur leer.

```
FORM z_mein_userexit RAISING cx_dynamic_check.

  DATA: ls_nfal TYPE nfal.

  ls_nfal = <pub__service>->api__get_nfal().

* ...

ENDFORM.
```

Die Methode verwendet einen in IS-H eingebauten Puffermechanismus. Falls die Falldaten in einem parallelen Modus oder durch einen anderen Benutzer geändert werden, kann die Methode deshalb veraltete Daten liefern.

Wenn Sie nur die Fall**nummer** benötigen, ist es in der Regel schneller, diese mit `api__get_ndoc` aus den Dokumentverwaltungsdaten zu entnehmen.

Tabelle 13.8 zeigt eine Auswahl der in der Praxis häufig verwendeten Felder der Tabelle `NFAL`.

Feldname	Inhalt
FALAR	Fallart (1 = stat., 2 = amb., 3 = teilstat.)
ABRKZ	Abrechnungskennzeichen
KZTXT	Bemerkungstext
PATGEW	Gewicht bei Aufnahme
GWEIN	Einheit Gewicht
PATGRO	Größe bei Aufnahme
GREIN	Einheit Größe
RESPI	Beatmungsdauer in Stunden

Tabelle 13.8: wichtige Felder der Tabelle `NFAL`

Beachten Sie, dass die Behandlungskategorie bewegungsbezogen abgelegt wird, weil Sie sich zeitraumabhängig ändern kann. Beachten Sie weiterhin, dass das Anfangs- und Enddatum des Falls (`BEGDT` und `ENDDT`) für stationäre Fälle **nicht** dem Aufnahme- und Entlassdatum entsprechen – diese Daten müssen Sie über die Bewegungen ermitteln.

Bewegungen 13.4.4 / S. 404

Im Zusammenhang mit den Falldaten benötigen Sie auch gelegentlich die **zugeordneten Personen**. Dabei handelt es sich um Geschäftspartner, die dem Fall in einer Reihe von vorgegebenen Rollen zugeordnet werden. Diese Zuordnungen werden

in der Tabelle NFPZ abgelegt und umfassen insbesondere den Hausarzt und den einweisenden Arzt.

Die Fall-Personen-Zuordnung ist zeitabhängig – es kann im zeitlichen Verlauf mehr als einen Hausarzt zu einem Fall geben. Es ist auch nicht völlig ausgeschlossen, dass es zu einem Zeitpunkt mehrere aktive Hausärzte zu einem Fall gibt.

Um die Fall-Personen-Zuordnungen zu lesen, können Sie den Funktionsbaustein ISH_READ_NFPZ verwenden. Die Geschäftspartnerdaten samt Adresse können Sie dann wieder anhand des Felds PERNR mit dem Funktionsbaustein ISH_READ_NGPA ermitteln, wie in Abschnitt 13.4.1 auf Seite 396 bereits für den verantwortlichen Mitarbeiter gezeigt.

13.4.4. Bewegungen

In vielen Fällen werden Sie die Bewegungsdaten zum Fall benötigen:

- Für stationäre Patienten müssen Sie in der Regel Aufnahme- respektive Entlassdatum ermitteln.
- Auf hausinternen Befunden soll oft die interne Station vermerkt sein, um den Befund zustellen zu können.
- Wenn ein Dokument mit Bezug zu einer Bewegung – etwa einer Operation – angelegt wird, sollen Daten dieser Bewegung übernommen werden.

Tabelle 13.9 gibt einen Überblick über die wichtigsten Felder der Bewegungsdaten. Da Sie die Daten in unterschiedlichen Strukturen mit jeweils leicht unterschiedlichen Feldnamen erhalten, ist die Tabelle hier etwas komplexer aufgebaut.

Zum Zugriff auf die Bewegungsdaten gibt es keine API-Methoden. Sie können entweder die offiziellen BAPI-Funktionsbausteine, die internen Adapterbausteine für den Bewegungszugriff in der Funktionsgruppe NBE2 oder die internen Lesebausteine der Funktionsgruppe NBE1 verwenden.

Viele der üblicherweise benötigten Daten können Sie auch mit dem Funktionsbaustein ISH_N2_MEDDOC_GETADMINDATA ermitteln. Beachten Sie dazu die Parameter EXT_DOCDATA_MOVE und EXT_DOCDATA_MOVE_OUT.

Die Verwendung von BAPIs hat den Vorteil, dass Sie mit freigegebenen und gewarteten Standardschnittstellen arbeiten; dafür ist die Programmlaufzeit durch eine Vielzahl von Prüfungen und Datenumsetzungen sehr viel länger. Tabelle 13.10 gibt

Feldname		Inhalt
intern	BAPI	
BEWTY	MOVEMNT_CTGRY	Bewegungstyp (Festwerte)
BWART	MOVEMNT_TYPE	Bewegungsart (Customizing)
BWIDT	MOVEMNT_DATE	Ist-Datum
BWIZT	MOVEMNT_TIME	Ist-Zeit
PLANB	STATUS_IND	Plankennzeichen (leer = Ist, P = Plan, W = Warteliste)
BWPDT	MOVEMNT_PDATE	Plan-Datum
BWPZT	MOVMENT_PTIME	Plan-Zeit
BWEDT	MOVEMNT_ENDDATE	Ende-Datum
BWEZT	MOVEMNT_ENDTIME	Ende-Zeit
PLANE	STATUS_IND_END	Plankennzeichen Enddatum (leer = Ist, P = Plan)
TELNR	PHONENO	Telefonnummer Patient im Haus
KZTXT	MOVEMNT_COMMENT	Bemerkung (LGTXT = X → Langtext!)
ORGFA	DEPARTMENT	Fachabteilung
ORGPF	NURS_TREAT_OU	Station/Ambulanz
ZIMMR	ROOM	Raum / Zimmer
BETT	BED	Bett
BEKAT	TREATMCATEGORY	Behandlungskategorie

Tabelle 13.9: wichtige Felder der Tabelle NBEW

einen Anhaltspunkt, in welchen Dimensionen sich der Mehraufwand bewegt, jeweils bezogen auf einen ungepufferten Zugriff mit den direkten Lesebausteinen. Für häufige Lesezugriffe ist es daher durchaus sinnvoll und zulässig, die direkten Lesebausteine zu verwenden.

Zugriffsweg	Faktor	
	gepuffert	ungepuffert
NBE1-Bausteine	0,6	1
NBE2-Bausteine	25	35
BAPI-Aufruf lokal	195	
BAPI-Aufruf in neuem Modus	215	

Tabelle 13.10: Geschwindigkeitvergleich Bewegungszugriffe

Wenn Sie die BAPI-Funktionsbausteine verwenden, sollten Sie sie immer mit dem Zusatz DESTINATION 'NONE' aufrufen.

BAPIs setzen aus Sicherheitsgründen bestimmte interne Puffer zurück und können dadurch verschiedene Standardprogramme – insbesondere den Klinischen Arbeitsplatz oder den OP-Arbeitsplatz – nachhaltig stören, wenn sie im gleichen internen Modus aufgerufen werden. Kurzdumps, die auf ein fehlendes oder beschädigtes *environment* hinweisen, werden oft durch dieses Problem verursacht. Durch den Zusatz DESTINATION 'NONE' wird der Baustein in einem eigenen internen Modus gestartet und erhält damit seinen eigenen Pufferbereich.

```
29  *     Fehler melden
30        l_error = lr_exception->get_text( ).
31        CONCATENATE: 'Fehler bei der Ermittlung von Größe und Gewicht:' l_error
32          INTO l_message SEPARATED BY space.
33        <pub__service>->api__list_error_text( i_text = l_message ).
34    ENDTRY.
35
36  ENDFORM.
37
38  * ... andere Routinen analog ...
```

Bei dieser Art der Fehlerbehandlung müssen Sie allerdings darauf achten, dass die Routinen darauf vorbereitet sind, dass eine der vorhergehenden Routinen abgebrochen ist. Die Routine zur Berechnung des BMI muss also damit umgehen können, dass die vorhergehende Routine zur Ermittlung von Größe und Gewicht abgebrochen ist und darf sich nicht darauf verlassen, dass Größe und Gewicht immer zur Verfügung stehen werden.

13.9. Anwendung im Beispieldokumenttyp

Konzept
3.4 / S. 113

In diesem Kapitel werden die im Konzept des bereits mehrfach verwendeten Beispieldokumenttyps festgehaltenen Anforderungen, die zusätzliche User Exits erfordern, umgesetzt. Dazu ist es sinnvoll, die Anforderungen aus einer technischen Perspektive nochmals genau zu betrachten und zu entscheiden, wo – sprich mit welchem User Exit – sie umgesetzt werden. So entsteht gleichzeitig eine Checkliste, anhand der der Bearbeitungsstand kontrolliert werden kann.

PMD-Transportworkbench
15.4.2 / S. 524

Der ausgelieferte Beispiel-Dokumenttyp mit dem Schlüssel `DEMONFP` befindet sich in einem Installationspaket, das mit der PMD-Transportworkbench installiert werden kann. Das Installationspaket ist seinerseits wiederum in einem BC-Set namens `/PMDBUCH/NFPROT_DEMONFP_V01` verpackt.

Notfallprotokoll

Das Dokument wird für jeden Besuch in der Notfallambulanz neu erstellt und mit der Bewegung verknüpft.	nicht relevant
Bei der Anlage werden das Bewegungsdatum und die Bewegungszeit als *Ankunftsdatum* `ANKUNFT_DAT` und *Ankunftszeit* `ANKUNFT_ZEIT` übernommen.	Dies erfolgt in einer eigenen Routine, die aus dem User Exit *Dokument-Anlage* aufgerufen wird.
Diese Felder sind Pflichtfelder beim Speichern des Dokuments.	Dies kann durch Parametrierung (ohne User Exit) abgebildet werden.

Als *Transportart* `TRANSPORT` stehen die Werte *selbstständig, Rettungsdienst sitzend* und *Rettungsdienst liegend* zur Verfügung.	nicht relevant
Der *Grund der Vorstellung* `GRUND_VORST` ist ein Pflichtfeld bei der Freigabe des Dokuments; beim Speichern erfolgt eine Warnmeldung, wenn kein Grund angegeben wurde.	Die Warnung muss programmiert werden; das erfolgt im elementbezogenen User Exit *Werte-Prüfung*. Der Übersichtlichkeit halber wird hier auch gleich die Fehlermeldung programmiert.
Vitalparameter	
Die Feldpaare *RR links systolisch* `RR_L_SYST` und *diastolisch* `RR_L_DIAST` und *RR rechts systolisch* `RR_R_SYST` und *diastolisch* `RR_R_DIAST` werden jeweils entweder komplett oder gar nicht ausgefüllt. Wenn in einem der beiden Feldpaare nur der systolische oder nur der diastolische Wert eingegeben wurde, erfolgt eine Fehlermeldung beim Sichern.	Die Prüfung wird im User Exit *Dokument prüfen* programmiert.
Zur Angabe einer *Arrhythmie* `ARRHYTHMIE` stehen die Werte *rhythmisch* und *arrhythmisch* zur Verfügung.	nicht relevant
Der *Messpunkt der Körpertemperatur* `TEMP_MESSPKT` kann die Werte *aurikular, axillär, oral* oder *rektal* annehmen.	nicht relevant
Allgemeinzustand	
Der *Bewusstseinszustand* `BEWUSSTSEIN` kann mit den Werten *wach, desorientiert, somnolent* oder *komatös* beschrieben werden.	nicht relevant
Der *Zustand der Atemwege* `ATEMWEGE` kann die Werte *frei, teilverlegt* oder *intubiert* annehmen.	nicht relevant
Die *Atmung* `ATMUNG` kann mit den Werten *ruhig, schnell, Zyanose* oder *beatmet* beschrieben werden.	nicht relevant
Wenn der *Zustand der Atemwege* `ATEMWEGE` von einem anderen Wert auf *intubiert* geändert wird, wird in das Feld *Atmung* `ATMUNG` einmalig der Wert *beatmet* eingetragen und eine Meldung ausgegeben. Das Feld kann aber manuell wieder auf einen der anderen Werte geschaltet werden und behält diesen dann auch bei.	Das Feld `ATEMWEGE` muss dafür aktiviert werden und einen elementbezogenen User Exit *Funktion* erhalten.

Die Radiobuttons zur Angabe der *Stillbarkeit der Blutung* `BLT_STILLBAR` werden ausgeblendet, wenn *keine Blutung* in `BLT_STAERKE` ausgewählt wurde.	Das Feld `BLT_STAERKE` muss aktiviert werden, das Feld `BLT_STILLBAR` erhält einen User Exit *Bildmodifikation*.
Organisatorisches	
Die *Telefonnummer der Angehörigen* `TEL_ANG` wird nicht im Dokument gespeichert, sondern dynamisch aus den Patientendaten (Feld `NPAT-ANTE1`) ermittelt und beim Öffnen des Dokuments automatisch aktualisiert.	Das Feld ist ein Anzeigefeld, das mit einem User Exit *Werteversorgung* versorgt werden muss.
Um die Daten nachtragen zu können, wird über eine Drucktaste `TEL_ANG_BTN` ein Absprung in die Patientendaten realisiert. Dazu wird eine einrichtungsbezogen konfigurierbare Variante des Clinical Process Builders aufgerufen (Funktionsbaustein `ISH_NV2000_CALL`).	Dazu wird ein User Exit *Funktion* zur Drucktaste programmiert.
Vor dem Aufruf der Patientendaten wird ggf. das Dokument gesichert.	Dazu muss nur das Kennzeichen *Sicherungsabfrage* der Drucktaste `TEL_ANG_BTN` gesetzt werden; zusätzliche Programmierung ist nicht erforderlich.
Bei der Rückkehr aus den Patientendaten wird die Telefonnummer aktualisiert.	Das passiert im PBO nach der Funktionsausführung automatisch und muss nicht weiter programmiert werden.
Notizen	
Die Tabelle der *Notizen* `NOTIZEN_TAB` ist nicht direkt änderbar. Alle Spalten sind auf nur Ausgabe geschaltet, es werden keine freien Zeilen zur Eingabe vorgeblendet.	nicht relevant, reine Parametrierung
Die Standardfunktionen zur Tabellenbearbeitung werden bis auf *Zeile löschen* deaktiviert.	Dies erfolgt in einer Routine, die aus dem dokumentbezogenen User Exit *PBO* aufgerufen wird.
Eine neue Notiz wird angelegt, indem der Text in das Feld `NOTIZ_EINGAB` eingegeben wird und mit der Drucktaste `NOTIZ_BTN` in die Tabelle übernommen wird. Dabei werden Datum und Uhrzeit der Eingabe in die jeweiligen Spalten `NOTIZ_DATUM` und `NOTIZ_ZEIT` übertragen.	Dies wird in einem User Exit *Funktion* zur Drucktaste `NOTIZ_BTN` programmiert.

Die Tabelle NOTIZEN_TAB wird automatisch nach Datum und Uhrzeit aufsteigend sortiert.

Da die Tabelle nur über die Eingabefunktion bearbeitet werden kann, sollte dies im Rahmen des User Exit *Funktion* zur Drucktaste NOTIZ_BTN programmiert werden.

In den folgenden Abschnitten werden die einzelnen User Exits vorgestellt. Die Abschnitte stehen in keiner besonderen Reihenfolge; die Programmierung kann in beliebiger Abfolge umgesetzt werden, solange die Routinen nicht aufeinander aufbauen. In diesen Beispielen werden zur besseren Lesbarkeit keine Nachrichten aus Nachrichtenklassen, sondern nur reine Textnachrichten verwendet.

13.9.1. Vorbelegung bei Dokumentanlage

Bei der Anlage des Dokuments sollen Bewegungsdaten übernommen werden; dies erledigt eine separate Routine Z_ANKUNFT_VORBELEGEN. Bei der Realisierung dieser Routine ist zu beachten, dass der User Exit *Dokument-Anlage* auch bei der Anlage einer Version durchlaufen wird. In diesem Fall soll der aus der Vorversion übernommene Wert in der Regel nicht überschrieben werden.

```
1  *---------------------------------------------------------------------*
2  * User-Exit Dokument-Anlage
3  *---------------------------------------------------------------------*
4  FORM z_dok_anlage RAISING cx_dynamic_check.
5
6    PERFORM z_ankunft_vorbelegen.
7
8  ENDFORM.                    "z_dok_anlage
9
10 *&---------------------------------------------------------------------*
11 *&      Form  z_ankunft_vorbelegen
12 *&---------------------------------------------------------------------*
13 *       Ankunftsdatum und -zeit vorbelegen
14 *---------------------------------------------------------------------*
15 FORM z_ankunft_vorbelegen RAISING cx_dynamic_check.
16
17   DATA: ls_ndoc TYPE ndoc,
18         ls_nbew TYPE nbew.
19
20 * nur bei Neuanlage eines Dokuments, nicht bei Anlage einer Version!
21   IF <pub__service>->api__get_procmode( ) = n2f_meth_insert.
22 *   Dokumentdaten und daraus Bezugsbewegung lesen
23     ls_ndoc = <pub__service>->api__get_ndoc( ).
24     CALL FUNCTION 'ISH_NBEWTAB_GET_LFDNR'
25       EXPORTING
26         ss_einri  = ls_ndoc-einri
27         ss_falnr  = ls_ndoc-falnr
28         ss_lfdnr  = ls_ndoc-lfdbew
29       IMPORTING
```

```
30          ss_nbew   = ls_nbew
31        EXCEPTIONS
32          not_found = 1.
33      IF sy-subrc <> 0.
34        <pub__service>->api__list_error_text(
35          i_text = 'Bezugsbewegung kann nicht gelesen werden.' ).
36      ELSE.
37  *     Daten übertragen
38        <pub__service>->api__set_value( i_alias = 'ANKUNFT_DAT'
39                                        i_value = ls_nbew-bwidt ).
40        <pub__service>->api__set_value( i_alias = 'ANKUNFT_ZEIT'
41                                        i_value = ls_nbew-bwizt ).
42      ENDIF.
43    ENDIF. " procmode = n2f_meth_insert
44
45  ENDFORM.                    "z_ankunft_vorbelegen
```

13.9.2. Deaktivierung der Tabellenfunktionen

Auch diese Aufgabe erledigt eine separate Routine `Z_TAB_FUNKT_DEAKTIVIEREN`, die aus dem User Exit *PBO* aufgerufen wird. Die hier angegebenen Funktionscodes können Sie dem GUI-Status entnehmen.

GUI-Status austauschen 13.6.4 / S. 433

```
1   *---------------------------------------------------------------------*
2   * User-Exit Dokument-PBO
3   *---------------------------------------------------------------------*
4   FORM z_dok_pbo RAISING cx_dynamic_check.
5
6     PERFORM z_tab_funkt_deaktivieren.
7
8   ENDFORM.                    "z_dok_pbo
9
10  *&---------------------------------------------------------------------*
11  *&      Form  z_tab_funkt_deaktivieren
12  *&---------------------------------------------------------------------*
13  *       Tabellenbearbeitungsfunktionen deaktivieren
14  *---------------------------------------------------------------------*
15  FORM z_tab_funkt_deaktivieren RAISING cx_dynamic_check.
16
17    <pub__service>->api__disable_function( 'MKAL' ).     " Alle markieren
18    <pub__service>->api__disable_function( 'MKLO' ).     " Alle Mark. löschen
19    <pub__service>->api__disable_function( 'MOVC' ).     " Ausschneiden
20    <pub__service>->api__disable_function( 'COPC' ).     " Kopieren
21    <pub__service>->api__disable_function( 'INSC' ).     " Einsetzen
22    <pub__service>->api__disable_function( 'NEWL' ).     " Neue Einträge
23    <pub__service>->api__disable_function( 'INSL' ).     " Zeile einfügen
24    <pub__service>->api__disable_function( 'DELE' ).     " Zeilen löschen
25    <pub__service>->api__disable_function( 'SORTUP' ).   " Aufwärts sortieren
26    <pub__service>->api__disable_function( 'SORTDOWN' ). " Absteigend sortieren
27
28  ENDFORM.                    "z_tab_funkt_deaktivieren
```

14.3.4. Überarbeitung des generierten Formulars

Erfahrungsgemäß werden Sie mit den Parametrierungsmöglichkeiten in der PMD-Entwicklungsumgebung nicht alle an den Ausdruck gestellten Anforderungen erfüllen können. Dem Autor ist kein Dokumenttyp bekannt, dessen Druckausgabe ausschließlich mit einem generierten Formular realisiert wurde – wohl auch, weil es ab einem gewissen Punkt unverhältnismäßig viel Aufwand ist, die verbleibenden Anpassungen durch permanente Nachjustierung der Vorlage und Einstellungen vorzunehmen.

Um das Formular manuell weiter zu bearbeiten, sollten Sie zunächst eine Kopie anlegen. Öffnen Sie dazu das generierten Formular (mit der Transaktion `SE71` oder durch Doppelklick auf den Formularnamen in den Kopfdaten des Dokumenttyps); achten Sie darauf, dass Sie im Änderungsmodus sind, wählen Sie Formular > Sichern als... und geben Sie einen neuen Formularnamen an. Hier können Sie sich an den Namenskonventionen für andere PMD-Elemente wie Includes orientieren.

Namenskonventionen Includes 13.2.1.1 / S. 362

Vergessen Sie nicht, das Formular mit dem gleichnamigen Menüpunkt zu aktivieren – sowohl nach der erstmaligen Anlage als auch nach jeder Änderung.

Tragen Sie anschließend den Namen des kopierten Formulars in den Kopfdaten des Dokumenttyps auf der Registerkarte *UserExits* in das Feld *eigenes Formular* ein (vgl. Abbildung 13.3 auf Seite 363). Diese Änderung wird sofort wirksam; Sie brauchen dazu das Programm nicht neu zu generieren.

Ändern Sie nicht das generierte Formular. Bei der Weiterentwicklung eines Dokumenttyps kann es sinnvoll sein, das Formular erneut zu generieren. Dabei können manuelle Anpassungen beschädigt oder komplett überschrieben werden.

Jetzt können Sie das Formular mit allen durch die SAPscript-Technologie verfügbaren Mitteln weiter bearbeiten. Ein Beispiel zu dieser Vorgehensweise finden Sie in Abschnitt 14.3.7.

14.3.5. Ausgabe zusätzlicher Daten

Wie in Abschnitt 14.1 auf Seite 468 bereits erläutert stehen dem Formular zur Ausgabe alle globalen Variablen des Druckprogramms zur Verfügung. Wenn Sie zusätzliche Daten ausgeben möchten, müssen Sie also globale Variablen definieren und im User Exit *Druck-Anfang* mit den zu druckenden Werten versorgen.

globale Variablen 13.2.2.2 / S. 368

Es hat sich bewährt, die zusätzlich auszugebenden Felder in einer eigenen Struktur im ABAP Dictionary zusammenzufassen. Dadurch bleibt nicht nur das Daten-Include übersichtlich, sondern Sie erleichtern sich auch einen eventuellen späteren Umstieg auf andere Ausgabetechnologien. Abbildung 14.9 zeigt eine solche Struktur.

Im ausgelieferten Beispiel-Dokumenttyp wird eine analog aufgebaute Struktur namens `/PMDBUCH/NFPROT_ADDDATA` verwendet.

Abbildung 14.9: Datenstruktur zur Ausgabe von Zusatzdaten

In vielen bestehenden Dokumenttypen werden Sie statt einer Struktur eine ganze Reihe von Variablen vorfinden, die zum Druck verwendet werden. Diese Vorgehensweise ist durchaus auch praktikabel – sie führt nur zu etwas unübersichtlicherem Programmtext und potentiell zu Mehraufwand bei einem Wechsel der Drucktechnologie. Bei Neuentwicklungen lohnt es sich daher, einige Minuten in die Anlage einer genau definierten Menge zu druckender Zusatzdaten zu investieren und diese Daten in einer Struktur zu kapseln.

Es kann auch sinnvoll sein, zwei Strukturen vorzusehen – eine Struktur, die in allen Dokumenttypen verwendet wird und allgemeine Felder wie Patienten-, Fall- und Klinikdaten enthält und eine zweite Struktur, die nur die für den Dokumenttyp spezifischen Daten enthält. Die allgemeine Struktur können Sie z. B. mit einem Funktionsbaustein füllen und durch dessen Wiederverwendung den Entwicklungsaufwand für mehrere Dokumenttypen senken.

Wenn Sie nicht mit zwei getrennten Strukturen arbeiten möchten, können Sie eine gemeinsame Struktur für alle Dokumenttypen anlegen und diese in die dokumenttypspezifischen Strukturen inkludieren. Alternativ können Sie per Konvention vereinbaren, dass jede Zusatzdatenstruktur bestimmte Felder mit immer gleichen Namen enthält, um zumindest die Vorlagen wiederverwendbar zu gestalten.

Strukturinklusion
5.2.4 / S. 151

Im Daten-Include legen Sie anschließend eine globale Variable an, die auf der Struktur basiert. Zu der in Abbildung 14.9 gezeigten Abbildung könnte dies etwa mit der folgenden Anweisung geschehen:

```
DATA: zs_druck_daten TYPE zishmed_pmd_nfprot_v01_data.
```

Beachten Sie die Namensbereiche bei der Anlage der Variablen – der Name muss mit `Y` oder `Z` beginnen.

Im User Exit *Druck-Anfang* könnten Sie diese Struktur dann wie folgt befüllen:

```
*---------------------------------------------------------------------*
*       User Exit Druck-Anfang
*---------------------------------------------------------------------*
FORM z_druck_anfang RAISING cx_dynamic_check.

  PERFORM z_druck_daten_versorgen
    CHANGING zs_druck_daten.

ENDFORM.                    "z_druck_anfang

*&---------------------------------------------------------------------*
*&      Form  z_druck_daten_versorgen
*&---------------------------------------------------------------------*
*       Zusatzdaten zum Druck bereitstellen
*---------------------------------------------------------------------*
FORM z_druck_daten_versorgen
  CHANGING cs_druck_daten TYPE zishmed_pmd_znfprot_add_data
  RAISING cx_dynamic_check.

  DATA: ls_npat TYPE npat,
        ls_nfal TYPE nfal.

  CLEAR cs_druck_daten.

* Patientendaten bereitstellen
  ls_npat = <pub__service>->api__get_npat( ).
  cs_druck_daten-patnr      = ls_npat-patnr.
  cs_druck_daten-gebdat_pat = ls_npat-gbdat.
  CONCATENATE ls_npat-titel  " Prof. Dr. med.
              ls_npat-vname  " Maximilian
              ls_npat-namzu  " Graf
              ls_npat-vorsw  " von
              ls_npat-nname  " Müllerstein
      INTO cs_druck_daten-name_pat SEPARATED BY space.
  CONDENSE cs_druck_daten-name_pat.
  CALL FUNCTION 'ISH_CONVERT_SEX_OUTPUT'
    EXPORTING
      ss_gschl  = ls_npat-gschl
    IMPORTING
      ss_gschle = cs_druck_daten-geschl_pat.

* Falldaten bereitstellen
  ls_nfal = <pub__service>->api__get_nfal( ).
  cs_druck_daten-falnr = ls_nfal-falnr.
  CALL FUNCTION 'ISH_CONVERT_CASETYPE_OUTPUT'
```

```
46      EXPORTING
47        ss_einri = ls_nfal-einri
48        ss_falai = ls_nfal-falar
49      IMPORTING
50        ss_fatxt = cs_druck_daten-falar_text.
51
52  ENDFORM.                    "z_druck_daten_versorgen
```

Abbildung 14.10 zeigt beispielhaft die Verwendung dieser Daten in einem Druckformular.

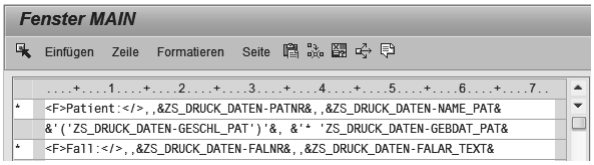


Abbildung 14.10: Verwendung von Zusatzdaten im SAPscript-Formular

Falls Sie mit getrennten Strukturen für allgemeine und dokumenttypspezifische Daten arbeiten oder Namenskonventionen für Standardfelder etabliert haben, können Sie derartige Textpassagen bereits in das Vorlageformular eintragen. Bei der Generierung des Formulars im Dokumenttyp können Sie dann auf ein Formular mit Klinik-Kopf und Patientendaten zurückgreifen.

14.3.6. Verwendung des User Exits Druck-Ausgabe

Da einige Aspekte der Druckausgabe bei der Verwendung von SAPscript vom Druckprogramm gesteuert werden, ist es möglich, sie mithilfe des User Exits *Druck-Ausgabe* zu beeinflussen. Die Steuerungsmöglichkeit beschränkt sich dabei darauf, welche Elemente des `MAIN`-Fensters gedruckt werden und in welcher Reihenfolge dies geschehen soll. Dafür stehen zwei API-Methoden zur Verfügung.

Mit der Methode `api__writeable_elements` können Sie eine Liste der Elemente erhalten, die im Zuge der Druckausgabe angesteuert werden können.

```
1  FORM z_mein_userexit RAISING cx_dynamic_check.
2
3    DATA: lt_aliases TYPE n2_alias_tab.
4
5    lt_aliases = <pub__service>->api__writeable_elements( ).
6
7  * ...
8
9  ENDFORM.
```

Die Methode liefert die Alias der Elemente in der Reihenfolge, in der die Standard-Druckverarbeitung die Elemente ausgeben würde. Es werden nur die druckbaren

Elemente übergeben, also z. B. keine Drucktasten. Unter diesen Elementen werden auch nur diejenigen angegeben, die im `MAIN`-Fenster ausgegeben werden, da nur diese durch das Druckprogramm steuerbar sind.

Die Methode liefert alle prinzipiell druckbaren Elemente – also auch diejenigen, die in der Parametrierung mit dem Kennzeichen *nicht drucken* versehen sind. Diese müssen Sie in der Programmierung ggf. wieder herausfiltern.

Mit Hilfe der Methode `api__write_elements` können Sie die Druckausgabe von Elementen veranlassen. Die Methoden übernimmt eine Tabelle mit den Alias der zu druckenden Elemente.

```
1  FORM z_mein_userexit RAISING cx_dynamic_check.
2
3    DATA: lt_aliases TYPE n2_alias_tab.
4
5  * ...
6
7    <pub__service>->api__write_elements( i_aliases = lt_aliases ).
8
9  ENDFORM.
```

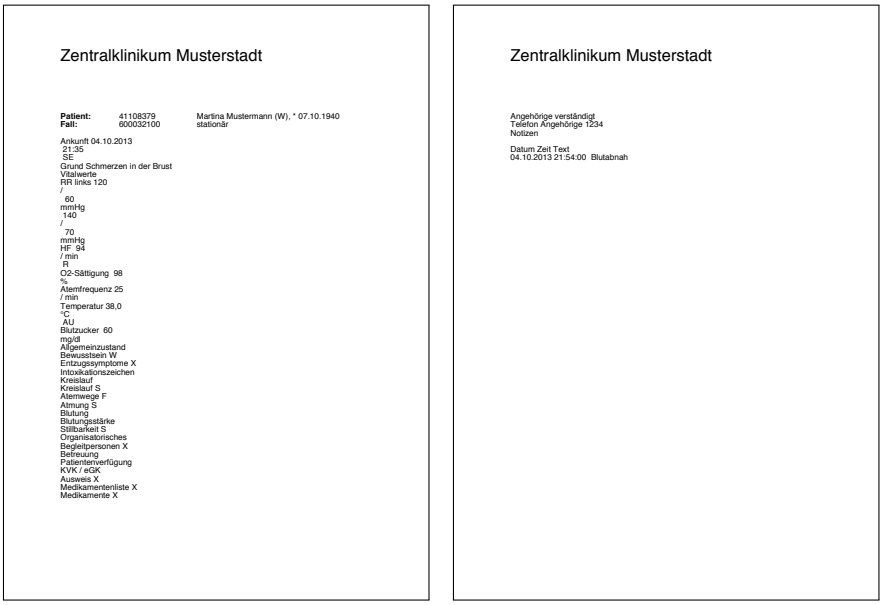
Die übergebene Tabelle kann einen oder mehrere Alias enthalten; wenn sie leer ist, werden alle Elemente gedruckt. Die Elemente werden in der Reihenfolge gedruckt, in der sie in der Tabelle stehen.

Tabellen können nur als gesamtes Element gedruckt werde. Es ist nicht möglich, einzelne Tabellenspalten zum Druck anzusteuern.

14.3.7. SAPscript-Ausgabe des Beispieldokumenttyps

Um das Beispieldokument zu drucken, wird ein Vorlageformular verwendet, das bereits einen rudimentären Klinik-Kopf sowie die in Abschnitt 14.3.5 gezeigte Ausgabe der Zusatzdaten enthält; der Dokumenttyp enthält die dort angegebenen Erweiterungen, um die Zusatzdaten zur Verfügung zu stellen. Neben den dafür erforderlichen Anpassungen enthält das Vorlageformular vor allem ein Standardformat, das auf der serifenlosen Proportionalschrift `HELVE` beruht und Tabulatoren bei 5 und 7 cm enthält. Abbildung 14.11 zeigt das daraus generierte Formular, das in dieser Form sicherlich noch nicht verwendbar ist.

Für den ausgelieferten Beispiel-Dokumenttyp wird das Vorlageformular `/PMDBUCH/NFP_TPL` verwendet; das generierte SAPscript-Formular wird ebenfalls unter dem Namen `/PMDBUCH/NFP_GEN` ausgeliefert.


Zentralklinikum Musterstadt

Patient: 41108379 Martina Mustermann (W), * 07.10.1940
Fall: 600032100 stationär

Ankunft 04.10.2013
21:35
SE
Grund Schmerzen in der Brust
Vitalwerte
RR links 120
/
60
mmHg
140
/
70
mmHg
HF 94
/ min
R
O2-Sättigung 98
%
Atemfrequenz 25
/ min
Temperatur 38,0
°C
AU
Blutzucker 60
mg/dl
Allgemeinzustand
Bewusstsein W
Entzugssymptome X
Intoxikationszeichen
Kreislauf
Kreislauf S
Atemwege F
Atmung S
Blutung
Blutungsstärke
Stillbarkeit S
Organisatorisches
Begleitpersonen X
Betreuung
Patientenverfügung
KVK / eGK
Ausweis X
Medikamentenliste X
Medikamente X

Zentralklinikum Musterstadt

Angehörige verständigt
Telefon Angehörige 1234
Notizen

Datum Zeit Text
04.10.2013 21:54:00 Blutabnah

Abbildung 14.11: Generierte SAPscript-Ausgabe mit Standardeinstellungen

Um die Ausgabe zu verbessern, kann im Rahmen der Parametrierung etwa wie folgt vorgegangen werden:

- Die Kopfdatenstruktur `STR_KOPF` wird zur Ausgabe der Überschrift *Notfallprotokoll* verwendet; hierfür ist es sinnvoll, ein eigenes Absatzformat in der Vorlage vorzusehen. Die restlichen Strukturen werden zur Ausgabe von Zwischenüberschriften verwendet und dafür fett gedruckt, unterstrichen und mit einer Leerzeile vom vorhergehenden Inhalt abgesetzt.

- Die Bezeichner der Werte werden ggf. vervollständigt und um den Text `:„` ergänzt. Dadurch werden die Werte untereinander ausgerichtet. Gleichzeitig wird das Absatzformat der Werte, deren Bezeichner jetzt mit einen Tabulator endet, auf = gestellt. Die so entstehende Langzeile stellt sicher, dass der Wert unmittelbar auf den Tabulator folgt und nicht noch durch ein Leerzeichen eingerückt wird.

- Die Auswahlfelder werden auf die Darstellung *nur Text* umgestellt; das gilt auch für die Checkboxen.

- Die für den Dialog notwendigen Fixtexte `KREISL_BEZ` und `BLT_BEZ` werden nicht ausgedruckt.

- Die Absatzformate der Einheiten, die vom System mit ∗ vorbelegt wurden, werden entfernt. Dadurch werden die Einheiten unmittelbar hinter die Werte gedruckt. Auf die gleiche Weise werden die Ankunftsdaten sowie die mehrteiligen Blutdruck-Angaben in jeweils einer Zeile zusammengefasst. Auch die Angaben zur Arrhythmie und zum Messpunkt der Temperatur werden so hinter die Messwerte bzw. Einheiten gesetzt.
- Der Bezeichner der Tabelle `NOTIZEN_TAB` wird nicht gedruckt; um eine unerwünschte Leerzeile zwischen der durch die Struktur gedruckten Überschrift und dem Tabelleninhalt zu vermeiden, wird das Absatzformat des Bezeichners auf `/*` geändert.
- Das Eingabefeld `NOTIZ_EINGAB` zur Erfassung neuer Notizen wird nicht gedruckt.

Abbildung 14.12 zeigt das Ergebnis dieser Anpassungen. Diese Version ist schon wesentlich besser lesbar, allerdings an einer Stelle noch inkonsistent gegenüber der Dialoganwendung: Die Radiobuttons zur Angabe der *Stillbarkeit der Blutung* (Element `BLT_STILLBAR`) werden im Dialog ausgeblendet, wenn *keine Blutung* in `BLT_STAERKE` ausgewählt wurde. Im Druck wird das Element `BLT_STILLBAR` derzeit allerdings immer ausgegeben, was zu der etwas widersprüchlichen Aussage *keine Blutung, diese war aber stillbar* führt. Um dieses Problem zu beheben, bietet sich ein User Exit *Druck-Ausgabe* an:

```
1  *---------------------------------------------------------------------*
2  *       User Exit Druck-Ausgabe
3  *---------------------------------------------------------------------*
4  FORM z_druck_ausgabe RAISING cx_dynamic_check.
5
6    DATA: lt_elements TYPE n2_alias_tab.
7
8    lt_elements = <pub__service>->api__writeable_elements( ).
9
10 * in der Parametrierung ausgeschaltete Elemente aus der Liste entfernen
11   DELETE lt_elements WHERE table_line = 'KREISL_BEZ'.
12   DELETE lt_elements WHERE table_line = 'BLT_BEZ'.
13
14 * Stillbarkeit der Blutung nicht ausgeben, wenn keine Blutung
15 * (siehe auch Bildmodifikation, Routine z_elem_blt_stillbar_bmod)
16   IF <pub__service>->api__value_is_initial( i_alias = 'BLT_STAERKE' ) = true.
17     DELETE lt_elements WHERE table_line = 'BLT_STILLBAR'.
18   ENDIF.
19
20   <pub__service>->api__write_elements( lt_elements ).
21
22 ENDFORM.
```

Zentralklinikum Musterstadt

Patient: 41108379 Martina Mustermann (W), * 07.10.1940
Fall: 600032100 stationär

Notfallprotokoll

Ankunft: 04.10.2013 21:35 selbstständig
Grund: Schmerzen in der Brust

Vitalwerte
RR links: 120 / 60 mmHg
RR rechts: 140 / 70 mmHg
Herzfrequenz: 94 / min rhythmisch
O2-Sättigung: 98 %
Atemfrequenz: 25 / min
Temperatur: 38,0 °C aurikular
Blutzucker: 60 mg/dl

Allgemeinzustand
Bewusstsein: wach
Entzugssymptome: ja
Intoxikationszeichen: nein
Kreislauf: stabil
Atemwege: frei
Atmung: schnell
Blutungsstärke: keine
Stillbarkeit: stillbar

Organisatorisches
Begleitpersonen: ja
Betreuung: nein
Patientenverfügung: nein
KVK / eGK: nein
Ausweis: ja
Medikamentenliste: ja
Medikamente: ja
Telefon Angehörige: 1234

Notizen
Datum Zeit Text
04.10.2013 21:54:00 Blutabnah

Abbildung 14.12: Generierte SAPscript-Ausgabe mit angepassten Einstellungen

Nach dieser inhaltlichen Korrektur sind noch einige Aspekte der Ausgabe anzupassen. Dies kann aber leider nicht mehr durch Parametrierung erfolgen, sodass zu diesem Zeitpunkt das generierte Formular kopiert und angepasst werden muss.

Das manuell angepasste SAPscript-Formular zum Druck des Beispiel-Dokumenttyps wird unter dem Namen `/PMDBUCH/NFP_MAN` ausgeliefert.

Folgende Anpassungen werden fast ausschließlich im `MAIN`-Fenster durchgeführt:

- Zu Beginn des Fensters werden die aus dem Vorlageformular übernommenen Zusatzdaten unter die generierte Überschrift verschoben.
- Die Transportart wird in Klammern gesetzt, um sie besser von der davor ausgegebenen Uhrzeit abzusetzen. Da hier der Text ausgegeben wird, muss das Element `VAL_TRANSPORTDECC` angepasst werden. Die Klammern werden innerhalb des Textsymbols gesetzt: `&'('RN2TEXT-TEXT(C)')'&`. Dies hat den Vorteil, dass keine leeren Klammern zurückbleiben, wenn einmal ein Text nicht ausgegeben wird.
- Die Aufbereitungsoption `(*)`, die bei der Generierung automatisch eingefügt wird, wird bei den Elementen zur Ausgabe der Messwerte durch `(C)` ersetzt. Die Option `C` steht für *condense*; sie entfernt führende Leerzeichen, die etwa bei der Ausgabe zweistelliger Werte in dreistelligen Feldern entstehen und die für ein ungleichmäßiges Druckbild verantwortlich sind.

Darüber hinaus muss die Formatierung der Tabelle manuell vorgenommen werden. Dazu wird ein spezielles Absatzformat eingefügt, das zwei Tabulatoren bei 2,5 cm und 4 cm enthält, um die Uhrzeit und den Text auszurichten. Die Tabulatoren reichen in diesem Fall allerdings nicht aus, da ein langer Text unter Umständen einen Umbruch verursachen und in der nächsten Zeile dann unterhalb des Datums und der Uhrzeit erscheinen würde. Dies kann mit einem relativ einfachen Trick unterbunden werden: Das Absatzformat erhält einen Einzug von 4 cm und davon abweichend einen Einzug von -4 cm für die erste Zeile. Durch diesen sog. *hängenden Einzug* erscheint der Text in der Folgezeile an der gleichen Position wie in der ersten Zeile.

Anschließend muss noch der Text im Formularfenster wie in Abbildung 14.13 gezeigt angepasst werden. In diesem Beispiel wurde auf die Ausgabe der Überschriften verzichtet, da die Bedeutung der Spalten intuitiv erfasst werden kann. Beachten Sie

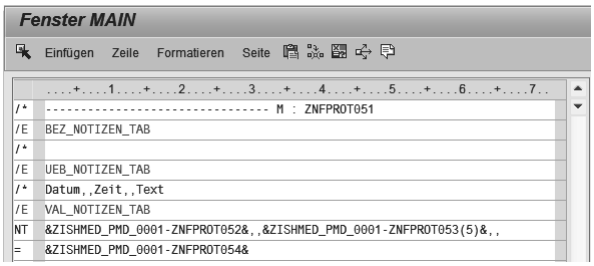


Abbildung 14.13: Anpassung der Tabellenausgabe

die Langzeile zur Ausgabe des Texts – ohne diese Kennzeichnung würde wiederum ein Leerzeichen zwischen dem Tabulator am Ende der vorhergehenden Zeile und dem Text eingefügt, was bei einem Umbruch zu einem Versatz zwischen der ersten Zeile und der Folgezeile führen würde.

Abbildung 14.14 zeigt das Ergebnis dieser Anpassungen. Mit ein wenig Übung lassen sich mit überschaubarem Aufwand bereits recht akzeptable Resultate erzeugen.

Zentralklinikum Musterstadt

Notfallprotokoll

Patient:	41108379	Martina Mustermann (W), * 07.10.1940
Fall:	600032100	stationär

Ankunft:	04.10.2013 21:35 (selbstständig)
Grund:	Schmerzen in der Brust

Vitalwerte

RR links:	120 / 60 mmHg
RR rechts:	140 / 70 mmHg
Herzfrequenz:	94 / min rhythmisch
O2-Sättigung:	98 %
Atemfrequenz:	25 / min
Temperatur:	38,0 °C aurikular
Blutzucker:	60 mg/dl

Allgemeinzustand

Bewusstsein:	wach
Entzugssymptome:	ja
Intoxikationszeichen:	nein
Kreislauf:	stabil
Atemwege:	frei
Atmung:	schnell
Blutungsstärke:	keine

Organisatorisches

Begleitpersonen:	ja
Betreuung:	nein
Patientenverfügung:	nein
KVK / eGK:	nein
Ausweis:	ja
Medikamentenliste:	ja
Medikamente:	ja
Angehörige verständigt:	nein
Telefon Angehörige:	1234

Notizen

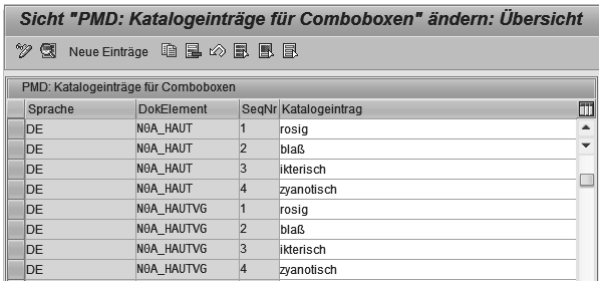
04.10.2013	21:54	Blutabnahme für Standard-Labor erfolgt
20.10.2013	10:55	langer langer langer langer langer langer langer langer Text
20.10.2013	10:55	mmmm

15.2. Eingabehilfen

Es gibt eine Reihe von Möglichkeiten, die Nutzung von Dokumenttypen durch geeignete Eingabehilfen zu unterstützen. Um den Einsatz einer neuen Dokumentationsanwendung zu erleichtern, sollten Sie daher zur Produktivsetzung sicherstellen, dass alle benötigten Comboboxwerte und Textbausteine vorhanden sind. Darüber hinaus kann es sinnvoll sein, den Arbeitsablauf durch die Verwendung besonderer Kennzeichen für Dokumente besser abzubilden.

15.2.1. Pflege der Comboboxwerte

Es gibt mehrere Möglichkeiten, die Pflegeoberfläche für Comboboxwerte aufzurufen. Sie können die Transaktion `N2KA` oder die Drucktaste [Werte] oben rechts im Detailbild der Combobox in der Dokumenttypdefinition verwenden. Diese Einstiegspunkte führen in die gleiche Listenansicht (Abbildung 15.2). In dieser Liste werden alle Comboboxwerte nach Sprache und Dokelement geordnet angezeigt.



Sprache	DokElement	SeqNr	Katalogeintrag
DE	N0A_HAUT	1	rosig
DE	N0A_HAUT	2	blaß
DE	N0A_HAUT	3	ikterisch
DE	N0A_HAUT	4	zyanotisch
DE	N0A_HAUTVG	1	rosig
DE	N0A_HAUTVG	2	blaß
DE	N0A_HAUTVG	3	ikterisch
DE	N0A_HAUTVG	4	zyanotisch

Abbildung 15.2: Listenansicht der Comboboxwerte

Die Comboboxwerte können nur in Abhängigkeit vom Dokelement, nicht aber vom Dokumenttyp gepflegt werden.

Zusatzprogrammierung Comboboxen
13.6.2 / S. 426

Wenn Sie ein und dasselbe Combobox-Dokelement in mehreren Dokumenttypen verwenden, gelten die in der Tabelle eingetragenen Werte für alle Dokumenttypen. Dies kann allerdings durch Zusatzprogrammierung beeinflusst werden.

Verwenden Sie die über [Auswahl] > [Nach Inhalten...] erreichbare Filterfunktion, um nur die Werte eines einzelnen Dokelements anzuzeigen. Alternativ können Sie die Drucktaste [Positionieren...] unterhalb der Liste verwenden, um zu dem gewünschten Element zu navigieren. Wenn Sie die Liste aus der Dokumenttypdefinition aufrufen, sollten Sie zuvor den Dokelement-Schlüssel in die Zwischenablage kopieren, um den Filter bzw. die Positionierung leichter ausführen zu können.

Abbildung 15.3 zeigt die Daten eines einzelnen Comboboxwerts. Neben der Sprache und dem Dokelement-Schlüssel wird der Wert durch einen Schlüssel identifiziert, der im Dialog als *Reihenfolge* oder *SeqNr* bezeichnet wird, allerdings nicht die spätere Anzeigereihenfolge der Werte wiedergibt. Dieser Wert muss nur eindeutig sein.

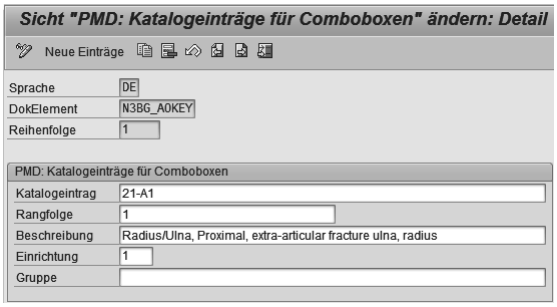


Abbildung 15.3: Detailansicht eines Comboboxwerts

Das Feld *Katalogeintrag* enthält den eigentlichen Wert, der in der Liste angezeigt und in die Combobox übernommen wird. Das Feld *Rangfolge* legt die primäre Sortierung fest; alle Einträge mit gleicher Rangfolge werden nach dem Katalogeintrag sortiert.

Das Feld *Einrichtung* ist optional. Sie können die Comboboxwerte einrichtungsbezogen ablegen oder übergreifend für alle Einrichtungen pflegen, indem Sie keine Einrichtung angeben. Es ist ohne weitere Vorkehrungen allerdings kaum sinnvoll, für ein konkretes Element diese beiden Vorgehensweisen zu mischen: Der Standard zeigt alle Werte an, die entweder der aktuellen Einrichtung oder keiner Einrichtung zugeordnet sind – es wird also eine gemeinsame Liste aus den einrichtungsübergreifenden Werten und den Werten der aktuellen Einrichtung erstellt. Dies ist nur in Einzelfällen und bei sehr sorgfältiger Pflege der Werte sinnvoll. Das Verhalten kann durch einen User Exit zur *Werteauswahl* allerdings geändert werden; weitere Informationen dazu finden Sie in Abschnitt 13.6.2 auf Seite 426.

Das Feld *Beschreibung* können Sie zur Dokumentation des Werts verwenden; der hier eingegebene Text wird weder in der Dialoganwendung verwendet noch an einen eventuell vorhandenen User Exit *Werteauswahl* weitergegeben. Mit dem Feld *Gruppe* hingegen können Sie Einträge logisch gruppieren, um sie später im User Exit leichter verarbeiten zu können. Ohne zusätzliche Programmierung ist dieses Feld wirkungslos.

Comboboxwerte werden als mandantenbezogene Einträge einer Steuertabelle im Entwicklungssystem gepflegt und mit dem normalen Transportverfahren, also mithilfe eines Customizing-Transportauftrags in die Folgesysteme übertragen.

15.2.2. Pflege von Textbausteinen und Sprungmarken

Textbausteine und Sprungmarken
2.3.1.6 / S. 62

Textbausteine und Sprungmarken haben Sie bei der Vorstellung von Langtextfeldern bereits kennengelernt. Textbausteine können der Einrichtung, der fachlichen oder dokumentierenden Organisationseinheit oder einem einzelnen Benutzer zugeordnet sein; Sprungmarken werden immer einrichtungsbezogen gepflegt.

Wenn ein Benutzer in mehreren Einrichtungen tätig ist, ist es nicht unproblematisch, in seinen Textbausteinen Sprungmarken zu verwenden – je nach aktueller Einrichtung könnten diese Sprungmarken nicht gefunden werden oder unterschiedliche Inhalte bereitstellen.

Neben der individuellen Pflege durch den jeweiligen Benutzer ist es möglich, Textbausteine und Sprungmarken mit der Transaktion `N2TBS` zentral zu administrieren. Dazu benötigen Sie die Berechtigung `N_2TBS`, die die Pflege der Textbausteine für die o. g. Bezugsebenen erlaubt.

15.2.2.1. Textbausteine

Die Pflegeoberfläche stellt die Textbausteine von jeweils zwei Bezugsebenen nebeneinander dar; auf dem in Abbildung 15.4 gezeigten Einstiegsbild legen Sie fest, welche Bezugsebenen das sein sollen. Bei den hier und an einigen anderen Stellen angezeigten *Textbaustein-Sets* handelt es sich um eine nicht vollständig ausgeführte Produktfunktion, die Sie nicht für Kundenprojekte einsetzen sollten.

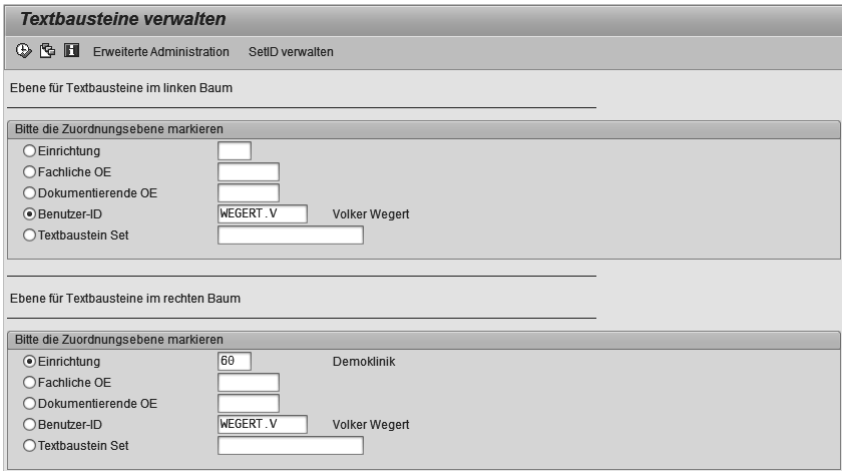


Abbildung 15.4: Einstiegsbild der Textbaustein-Administration

Abbildung 15.5 zeigt die Administrationsoberfläche. Im oberen Bereich des Bilds werden die Textbausteine der ausgewählten Bezugsebenen angezeigt, der untere Teil

zeigt den gerade ausgewählten Textbaustein an. Neben den Drucktasten oberhalb der Baumanzeige können Sie die Textbausteine mittels Drag&Drop innerhalb der Baumstrukturen verschieben. Textbausteine und Kategorien in der Baumanzeige verfügen weiterhin über ein Kontextmenü, mit dem Texte und Schlüssel geändert werden können.

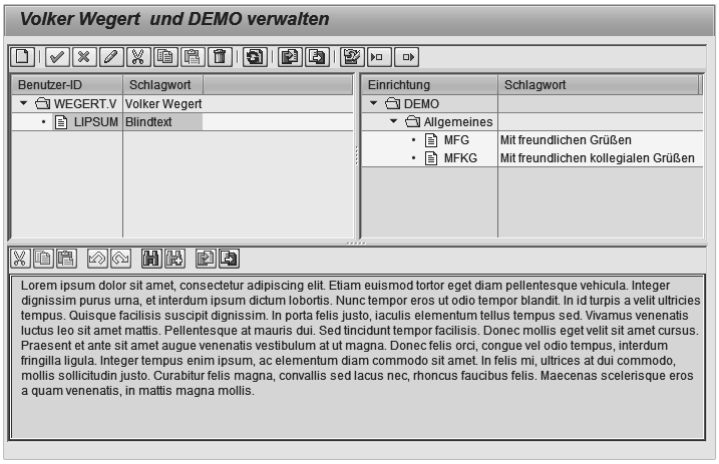


Abbildung 15.5: Baumanzeige der Textbausteine

Die Administrationsoberfläche kann sich in zwei Modi befinden. Nach dem Start können Sie zwischen verschiedenen Bausteinen navigieren und die meisten Optionen der Drucktastenleiste nutzen; der Texteditor im unteren Bereich der Oberfläche ist nicht eingabebereit. Mit der Drucktaste [✎] können Sie einen bestehenden Textbaustein ändern; dazu wird der Texteditor aktiviert. Diesen Modus können Sie nur mit den Drucktasten [✔] und [✖] wieder verlassen.

Mit der Drucktaste [🗋] können Sie einen neuen Textbaustein anlegen; dazu öffnet sich das gleiche Dialogfenster, mit dem auch jeder andere Benutzer einen Textbaustein erfassen kann. Beachten Sie, dass der Textbaustein in der Bezugsebene angelegt wird, in deren Baumstruktur sich der aktuell ausgewählte Textbaustein befindet (Abbildung 15.6).

Mit den Drucktasten [✂], [⧉] und [📋] können Sie komplette Textbausteine ausschneiden, kopieren und innerhalb der Baumstrukturen wieder einfügen. Mit der Drucktaste [🗑] kann ein Textbaustein gelöscht werden.

Vor der Löschung erfolgt keine Sicherheitsabfrage, und es gibt keine Funktion, um den gelöschten Baustein wiederherzustellen.

Mit der Drucktaste [⇲] können Sie Textbausteine in eine XML-Datei exportieren. Dabei wird jeweils der ausgewählte Eintrag in der Baumanzeige berücksichtigt,

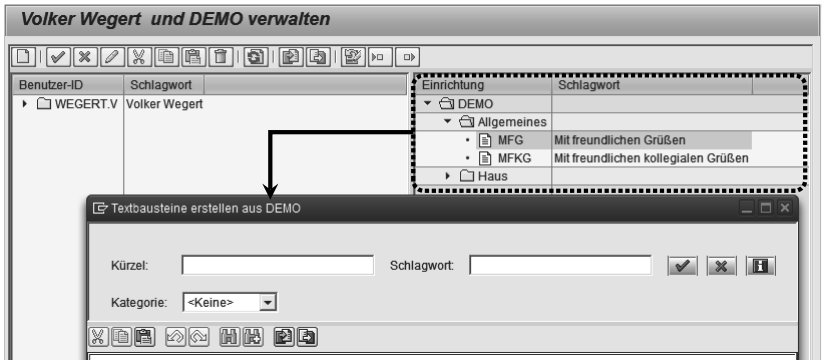


Abbildung 15.6: Anlage eines Textbausteins

sodass Sie einen einzelnen Textbaustein, eine gesamte Kategorie oder alle Kategorien und Textbausteine einer Bezugsebene exportieren können. Eine Mehrfachselektion einzelner Textbausteine ist nicht möglich.

Mit der Drucktaste [ ] können Sie die Textbausteine aus einer so erstellten Datei wieder importieren. Dies ist auch die einzige Möglichkeit, Textbausteine zwischen verschiedenen Systemen zu transportieren, da sie nicht an das reguläre Transportsystem angeschlossen sind.

Mit der Drucktaste [Erweiterte Administration] auf dem Einstiegsbild (Abbildung 15.4) der Transaktion `N2TBS` erreichen Sie das in Abbildung 15.7 gezeigte Werkzeug, mit dem Sie den gesamten Textbausteinbestand einer Bezugsebene kopieren, verschieben oder löschen können.

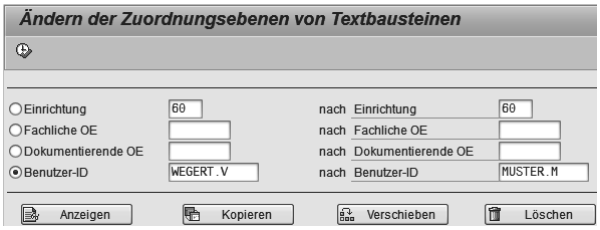


Abbildung 15.7: Erweiterte Textbaustein-Administration

15.2.2.2. Sprungmarken

Sprungmarken werden üblicherweise innerhalb von Textbausteinen verwendet und teilweise innerhalb der gleichen Pflegeoberfläche bearbeitet. Wie bereits erläutert, bestehen Sprungmarken aus einer Kennung im Text, die von den speziellen Klammern <| . . . |> umschlossen wird. Es gibt vier Arten von Sprungmarken, die aber in

der Dokumentation und Benutzerführung nicht konsistent voneinander abgegrenzt werden – insofern ist die folgende Klassifizierung nur der Versuch, die vorhandenen Funktionen strukturiert zu erläutern.

Einfache Sprungmarken dienen nur der Navigation und haben darüber hinaus keine weitere Funktion. Jede Sprungmarke, die nicht zu einer der folgenden drei Arten gehört, ist implizit eine solche Navigationshilfe.

Sprungmarken, deren Kennung mit dem Zeichen `|` beginnen, verweisen auf einen Textbaustein mit dem angegebenen Kürzel. Beim Erreichen einer solchen Sprungmarke wird der zugehörige Textbaustein gesucht und – falls möglich – anstelle der Sprungmarke eingefügt. Eine Sprungmarke `<||TBNAME|>` würde also zum Beispiel durch den Inhalt des Textbausteins `TBNAME` ersetzt werden.

Es gibt Sprungmarken, die automatisch durch einen vom System ermittelten Wert ersetzt werden. Diese Sprungmarken unterscheiden sich aus Sicht des Verwenders nicht von den einfachen Navigationshilfen – es ist ausschließlich die Verwendung einer bestimmten Kennung, die die automatische Ersetzung auslöst. Tabelle 15.2 zeigt die systemseitig vordefinierten Sprungmarken; Sie können durch eine Implementierung des Business Add-In `ISHMED_LTE_FILL_TAG` weitere Sprungmarken hinzufügen[1].

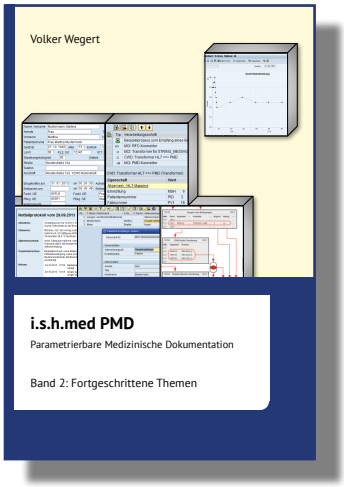
Business Add-In 5.4.2 / S. 169

Sprungmarke	Wert
`<|IS-H*MED_TIME|>`	aktuelle Uhrzeit im Format HH:MM
`<|IS-H*MED_DATE|>`	aktuelles Datum im Format TT.MM.JJJJ
`<|IS-H*MED_PATNR|>`	Patientennummer
`<|IS-H*MED_FALLNR|>`	Fallnummer
`<|IS-H*MED_LNRLS|>`	laufende Nummer der Leistung
`<|IS-H*MED_PAT-NNAME|>`	Nachname des Patienten
`<|IS-H*MED_PAT-VNAME|>`	Vorname des Patienten
`<|IS-H*MED_PAT-TITEL|>`	Titel des Patienten
`<|IS-H*MED_PAT-GBDAT|>`	Geburtsdatum des Patienten im Format *TT.MM.JJJJ
`<|IS-H*MED_VMA-TFLN|>`	Titel, Vor- und Nachname des verantw. Mitarbeiters
`<|IS-H*MED_VMA-FLN|>`	Vor- und Nachname des verantw. Mitarbeiters
`<|IS-H*MED_VMA-TLN|>`	Titel und Nachname des verantw. Mitarbeiters
`<|IS-H*MED_VMA-LN|>`	Nachname des verantw. Mitarbeiters

Tabelle 15.2: Vordefinierte Sprungmarken mit Wert-Ersetzung

Da die Entscheidung, ob die Sprungmarke ersetzt werden soll, dynamisch innerhalb der BAdI-Implementierung getroffen wird, ist es prinzipiell nicht möglich, systemseitig eine Liste der Sprungmarken mit automatischer Ersetzung zu erstellen – im Zweifelsfall müssen Sie den vorhandenen Programmtext analysieren.

[1] Die in Tabelle 15.2 angegebenen Standard-Sprungmarken stammen letztendlich auch nur aus der Default-Implementierung dieses BAdIs.



Aus dem Inhalt:

Wiederverwendbare Standard-Bausteine
Linkbausteine, Fremddatenbausteine

Fortgeschrittene User Exit-Programmierung
Objektorientierung, Alias-Konstanten, Generische Programmierung

Fortgeschrittene Dialogprogrammierung
Standarddialoge und eigene Popup-Fenster, Listenauswahl, Kontextmenüs, Rechtschreibprüfung, Volltextsuche, Docking-Container und amodale Dialoge, ...

Dokumentanzeige und -ausgabe
Standard-Kontextdaten, Orgmittelsteuerung, Versandsteuerung, HTML-Ausgabe mit XSLT

Dokumenttyp flexibler gestalten
Steuertabellen, Personalisierung, Text vorbelegen, Inhaltsvorlagen, Berechtigungsprüfung, Protokollierung, ...

Arbeit mit Dokumenten
Dokumentverwaltungsdaten, Zugriff auf Dokumentinhalte, Dokumente anlegen und aufrufen, HL7-Kommunikation mit MCI, ...

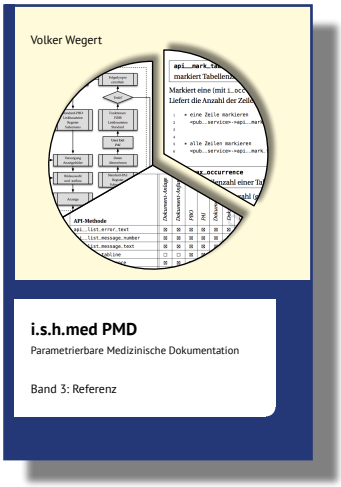
Technische Informationen
Namensvergabe, Umstellung von Generatorversion 1

Volker Wegert

i.s.h.med PMD
Band 2: Fortgeschrittene Themen

Dieses Buch setzt auf den in Band 1 vermittelten Grundlagen auf und vertieft eine Reihe inhaltlich und technisch anspruchsvollerer Themen. Sie lernen wiederverwendbare Standard-Bausteine und nützliche Komponenten der Entwicklungsumgebung kennen. Die fortgeschrittene User Exit- und Dialogprogrammierung wird ebenso vertieft wie verschiedene Erweiterungen der Druckausgabe. Dazu zählt insbesondere auch der Einsatz der Versandsteuerung und der Orgmittelsteuerung. Das Buch geht auch auf den externen Zugriff auf Dokumentdaten — etwa aus anderen Dokumenten oder eigenen Anwendungen — sowie auf den Umgang mit Dokumentverwaltungsdaten ein. Ein Kapitel beschreibt ausführlich die Entwicklung eigener Linkbausteine. Das Buch schließt mit technischen Hintergrund-informationen zur Namensvergabe der generierten Objekte und zur Umstellung von Dokumenttypen, die noch mit Generatorversion 1 erstellt wurden.

354 Seiten, Hardcover, 2014, 69,90 €
ISBN 978-3-7323-0101-0
http://www.pmd-buch.de



Aus dem Inhalt:

- Funktionen der Dokumenttypbearbeitung
- Einstellungen der Dokumentart
- Dokelementtypen mit Kennzeichen und Größenangaben
- Fremddatenbausteine
- Dialogerweiterungen: User Exits, Ablauf und Verarbeitungsmodi
- Druckausgabe: User Exits, Ablauf und Anwendungen
- API-Referenz: Variablen und Methoden
- Benutzereinstellungen und Berechtigungen
- Generische Programmierung
- Technische Objektnamen
- Beispiel-Dokumenttypen

Volker Wegert

i.s.h.med PMD
Band 3: Referenz

Dieses Buch stellt eine Referenz zur täglichen Arbeit mit der PMD dar. Hier finden Sie in komprimierter Form die wichtigsten Informationen, die Sie zur Erstellung und Anpassung von Dokumenttypen benötigen. Auf die Einstellungen der Dokumentart und der Dokelementtypen wird ebenso eingegangen wie auf die Abläufe der Dialog- und Druckausgabe und die verfügbaren User Exits. Zentraler Bestandteil der Referenz ist eine Übersicht über die verfügbaren API-Variablen und -Methoden mit Beschreibung und Anwendungsbeispielen. In diesen Buch sind weiterhin die wichtigsten Berechtigungsobjekte und Benutzerparameter, die Strukturen und Konstanten zur generischen Programmierung sowie häufig benötigte Transaktionen, Tabellen und Programme aufgeführt.

65 Seiten, Paperback, 2014, 9,90 €
ISBN 978-3-7323-0104-1
http://www.pmd-buch.de