Kapitel 7

Benutzerdialoge

Überblick über das Kapitel

Den Kapitelüberblick entnehmen Sie bitte dem Überblick der einzelnen Lektionen zu diesem Kapitel.

Lernziele des Kapitels

Am Ende dieses Kapitels können Sie

- Eigenschaften und Stärken des Dynpros aufzählen
- einfache Dynpros mit Ein-/Ausgabefeldern sowie Drucktasten realisieren und zu einer Dialoganwendung verknüpfen
- die programminterne Verarbeitung bei Dynpro-Aufrufen erläutern und implementieren
- die Eigenschaften und Nutzungsszenarien des ABAP Web Dynpros aufzählen
- die Programmier- und Laufzeitarchitektur des ABAP Web Dynpros grob erläutern
- einfache Web-Dynpro-Anwendungen mit Ein-/Ausgabefeldern und Drucktasten realisieren
- Eigenschaften und Stärken des Selektionsbilds aufzählen
- Abgrenzungsmöglichkeiten auf dem Selektionsbild realisieren
- Eigenschaften und Stärken der ABAP-Liste aufzählen
- Listen- und Spaltenüberschriften realisieren
- mehrsprachige Listen implementieren
- die ereignisgesteuerte Abarbeitung eines ausführbaren ABAP-Programms beschreiben
- die wichtigsten Grundereignisse aufzählen und ihren Zweck erläutern
- das SAP Grid Control (*SAP List Viewer*) zur Anzeige einer internen Tabelle auf einem Dynpro verwenden

Inhalt des Kapitels

Lektion: Dynpro

Überblick über die Lektion

In dieser Lektion lernen Sie die Gestaltung und Programmierung einfacher Dynpros mit Ein-/Ausgabefeldern und Drucktasten.

Lernziele der Lektion

Am Ende dieser Lektion können Sie

- Eigenschaften und Stärken des Dynpros aufzählen
- einfache Dynpros mit Ein-/Ausgabefeldern sowie Drucktasten realisieren und zu einer Dialoganwendung verknüpfen
- die programminterne Verarbeitung bei Dynpro-Aufrufen erläutern und implementieren

Unternehmensszenario

Sie sollen ein Programm entwickeln, das Datenerfassung und Datenausgabe auf Bildschirmbildern ermöglicht. Ferner sollen auf den Bildschirmbildern Drucktasten angeboten werden, über die entsprechende Funktionen Ihres Programms angesprochen werden können.

Das Dynpro

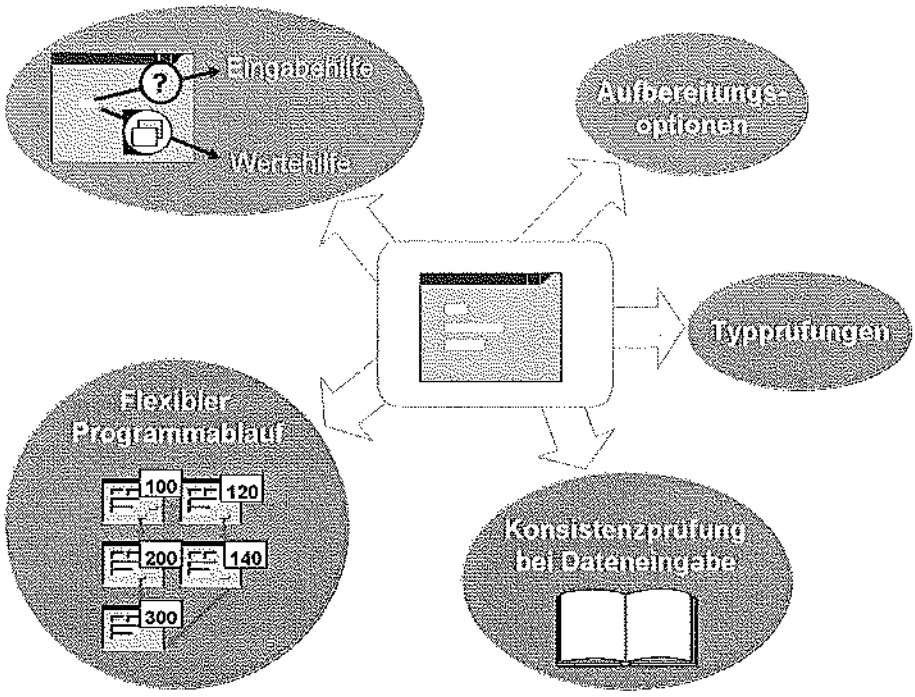


Abbildung 168: Eigenschaften von Dynpros

Ein Dynpro besteht nicht nur aus seinem Layout mit Ein-/Ausgabefeldern, Drucktasten und anderen Bildelementen, sondern auch einer Verarbeitungslogik (Quelltextabschnitte, die als Vor-/Nachbereitung der Dynproanzeige abgearbeitet werden).

Durch die Integration des *ABAP Dictionary* werden für Dynpro-Eingabefelder automatische Konsistenzprüfungen zur Verfügung gestellt. Dies beinhaltet neben der Typprüfung auch Fremdschlüsselprüfungen und Prüfungen gegen Festwerte. Für diese Prüfungen werden automatisch Informationen aus dem *ABAP Dictionary* herangezogen.

Diese Prüfungen können durch programmspezifische Prüfungen ergänzt werden. Für Dynpros stehen Techniken zur Verfügung, die die Reihenfolge der Prüfungen steuern und bei auftretenden Fehlern die gewünschten Felder wieder eingabebereit zur Verfügung stellen.

Das Layout kann sehr flexibel gestaltet werden, und zwar mit Eingabefeldern, Ausgabefeldern, Auswahlknöpfen, Ankreuzfeldern und vor allem Drucktasten, über die entsprechende Funktionen des Programms ausgeführt werden können.

Für das Dynpro stehen dieselben Aufbereitungsoptionen zur Verfügung wie für die Liste und das Selektionsbild: Festpunktzahlen und Datum werden nach den Einstellungen in den Benutzerfestwerten aufbereitet, die Zeit nach hh:mm:ss, Beträge nach dem Inhalt eines Währungsfelds und physikalische Größen (Längen, Gewichte, ...) nach dem Inhalt eines Einheitenfelds.

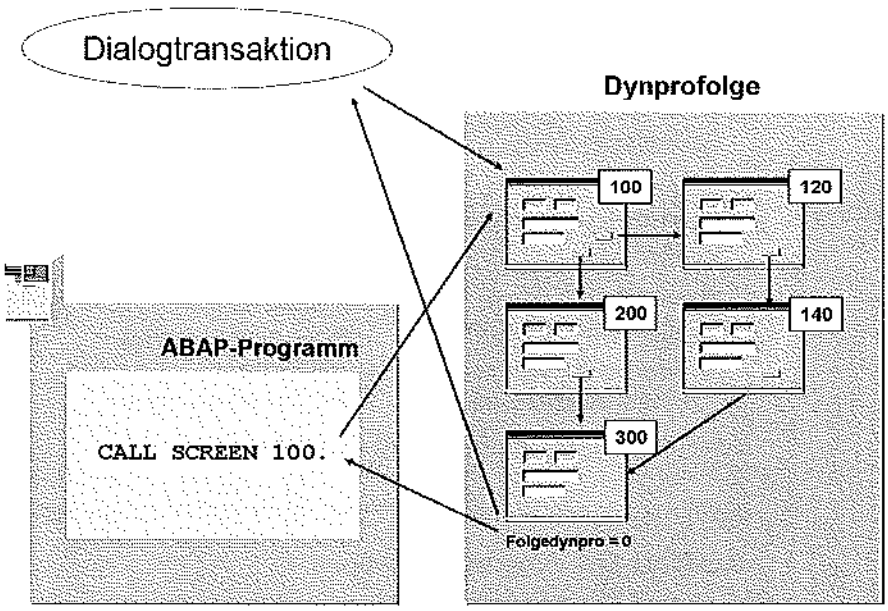


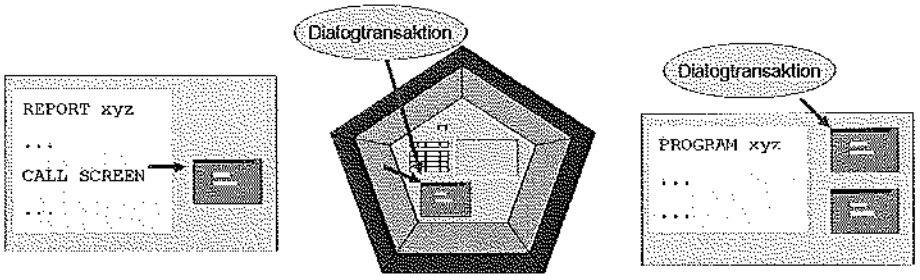
Abbildung 169: Dynprofolge

Prinzipiell bestehen zwei Möglichkeiten, um eine Dynprofolge zu starten:

1. Indem Sie aus einem Verarbeitungsblock Ihres Programms das erste Dynpro aufrufen (Anweisung CALL SCREEN).
2. Indem Sie eine Transaktion anlegen, die auf das Programm und das erste Dynpro verweist (Dialogtransaktion).

Nach dem Prozessieren eines Dynpros wird das statisch oder dynamisch festgelegte Folgedynpro verarbeitet. Das formale Folgedynpro 0 lässt die Verarbeitung zur Aufrufstelle zurück verzweigen bzw. beendet die Dialogtransaktion.

Dynpros –
gibt es in folgenden Programmtypen:



Ausführbare Programme (Reports)

- Anzeige von Daten (statt Liste oder zusätzlich)
- Anzeigen/Ändern von Details zu einer Liste

Funktionsgruppe

- Dialogtransaktion
- Dynpros/Dynprofolgen gekapselt in Funktionsbausteinen

Modulpool

- Dialogtransaktionen

Abbildung 170: Dynpros und Programmtypen

Dynpros können in folgenden Programmtypen angelegt werden:

Ausführbares Programm (Report)

Dynpros werden in ausführbaren Programmen eingesetzt, um entweder zusätzlich zur Listausgabe Daten anzuzeigen oder um die Listausgabe komplett zu ersetzen. Außerdem besteht die Möglichkeit, Dynpros für das Erfassen und Ändern von Daten auf der Liste einzusetzen. Aus Gründen der Wiederverwendbarkeit und der Datenkapselung wird empfohlen, Dynpros nicht mehr direkt in den Reports anzulegen, sondern Dynpros in Funktionsgruppen zu verwenden.

Funktionsgruppe

Dynpros in Funktionsgruppen lassen sich zum einen über Dialogtransaktionen ansprechen. Zum anderen besteht die Möglichkeit, solche Dynpros über die Anweisung `CALL SCREEN` aus dem Quelltext eines Funktionsbausteins heraus zu starten. Dadurch kann ein Dynpro oder eine Dynprofolge sehr einfach und mit definierter Schnittstelle zur Wiederverwendung bereitgestellt werden. Funktionsgruppen sind der empfohlene Programmtyp für das Anlegen von Dynpros.

Modulpool

Dynpros in Modulpools lassen sich nur über Dialogtransaktionen ansprechen. Anders als bei Dynpros in Funktionsgruppen besteht keine Möglichkeit, sie zu kapseln und mit einer sauberen Schnittstelle nach außen zu versehen.

Deshalb sollen nach Möglichkeit keine neuen Modulpools angelegt werden. Stattdessen sollen immer Funktionsgruppen verwendet werden, auch wenn ein Ansprechen der Dynpros über Funktionsbausteine (noch) nicht vorgesehen ist.

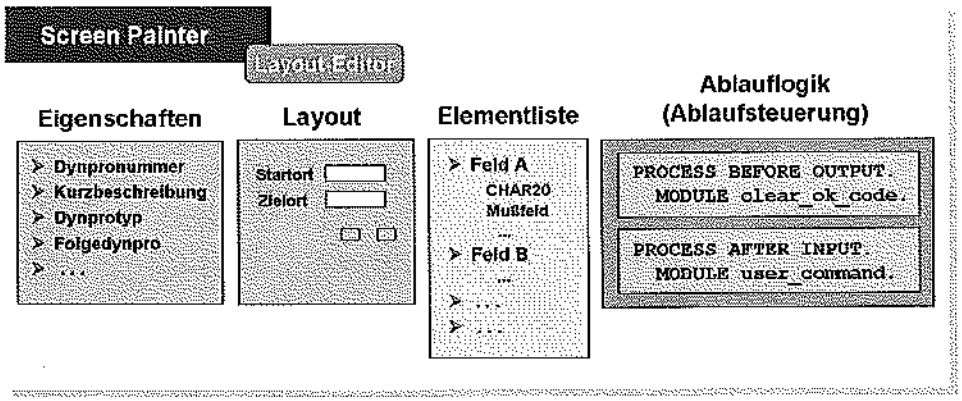


Abbildung 171: Komponenten eines Dynpros

Zu einem Dynpro gehören folgende Komponenten:

Eigenschaften (Attribute)

Sie enthalten u. a. eine vierstellige Nummer als **Dynprokennung**, einen Kurztext, Informationen über den Dynprotyp (z. B. *Normal* für Verwendung der vollen Bildschirmgröße) sowie die Angabe des **Default-Folgedynpros**.

Layout

Auf Ihrem Dynpro können Sie Ein-/Ausgabefelder, Texte, Drucktasten und andere Elemente platzieren. Solche Elemente, heißen **Bild-** oder **Dynpro-Elemente**.

Elementliste

Sie ist die Auflistung aller **Dynpro-Elemente samt deren Eigenschaften**, wie z. B. Position, Größe und Datentyp.

Ablaufsteuerung (Ablauflogik)

Die Ablauflogik eines Dynpros besteht aus dem **PBO** (Process Before Output) und dem **PAI** (Process After Input). PBO enthält Verweise auf Verarbeitungsblöcke (**PBO-Module**), die als Vorbereitung der Dynproanzeige (z. B. Datenselektion) abgearbeitet werden, bevor das Dynpro gesendet wird. PAI enthält Verweise auf Verarbeitungsblöcke (**PAI-Module**), die als Reaktion auf Benutzereingaben und -aktionen (z. B. Daten sichern) abgearbeitet werden.

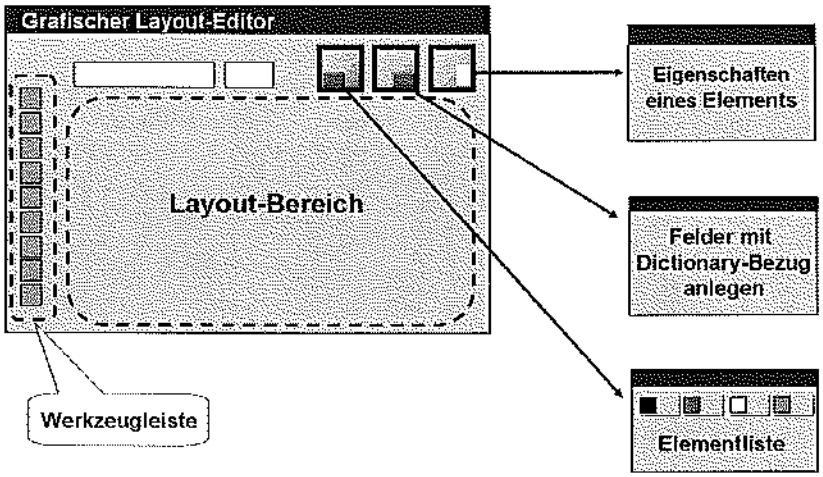


Abbildung 172: Der grafische Layout-Editor

Mit dem grafischen *Layout-Editor* können Sie den Aufbau des Dynpros gestalten (Werkzeugleiste). Außerdem haben Sie folgende Pflegefunktionen über die drei in der obigen Grafik dargestellten Drucktasten:

Elementeigenschaften pflegen

In einem Dialogfenster werden alle Attribute zum markierten Bildschirmelement zur Pflege angezeigt.

Dictionary- oder Programmfelder holen

Über ein Dialogfenster können Sie Dynprofelder mit Bezug auf Dictionary-Felder oder programmintern definierte Felder anlegen.

Elementliste anzeigen

Sie zeigt alle vorhandenen Dynpro-Elemente samt zugehörigen Attributen zur Pflege an.

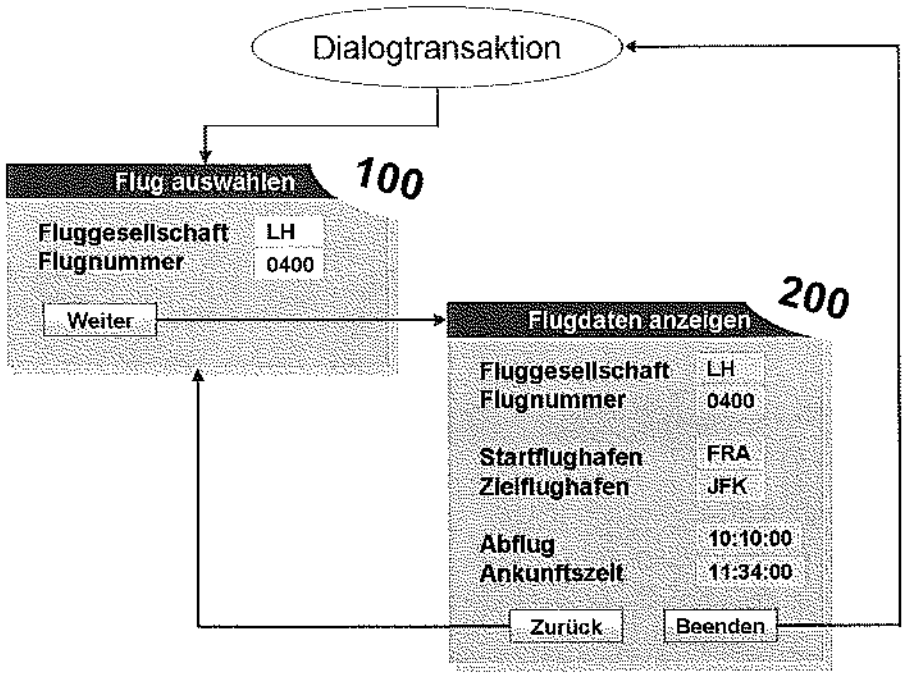


Abbildung 173: Anwendungsbeispiel

In den folgenden Abschnitten wird in mehreren Stufen ein Programm entwickelt, mit dem die Daten zu einer einzelnen Flugverbindung angezeigt werden können.

Das Programm besteht aus einer Folge von zwei Dynpros. Auf dem ersten Dynpro wählt der Benutzer die Flugverbindung aus, auf dem zweiten Dynpro werden die Details zur Flugverbindung angezeigt.

Die Dynprofolge wird über eine Dialogtransaktion gestartet. Betätigt der Benutzer auf dem ersten Dynpro die Drucktaste *Weiter*, werden die Eingaben verarbeitet, die Daten von der Datenbank gelesen und auf das zweite Dynpro navigiert.

Auf dem zweiten Dynpro werden die Daten zur Flugverbindung angezeigt. Betätigt der Benutzer die Drucktaste *Zurück*, so gelangt er auf das erste Dynpro zurück und kann eine andere Flugverbindung auswählen.

Betätigt der Benutzer die Drucktaste *Beenden*, wird die Dynprofolge verlassen, in diesem Fall also die Dialogtransaktion beendet.

Dynpros, Drucktasten und Navigation

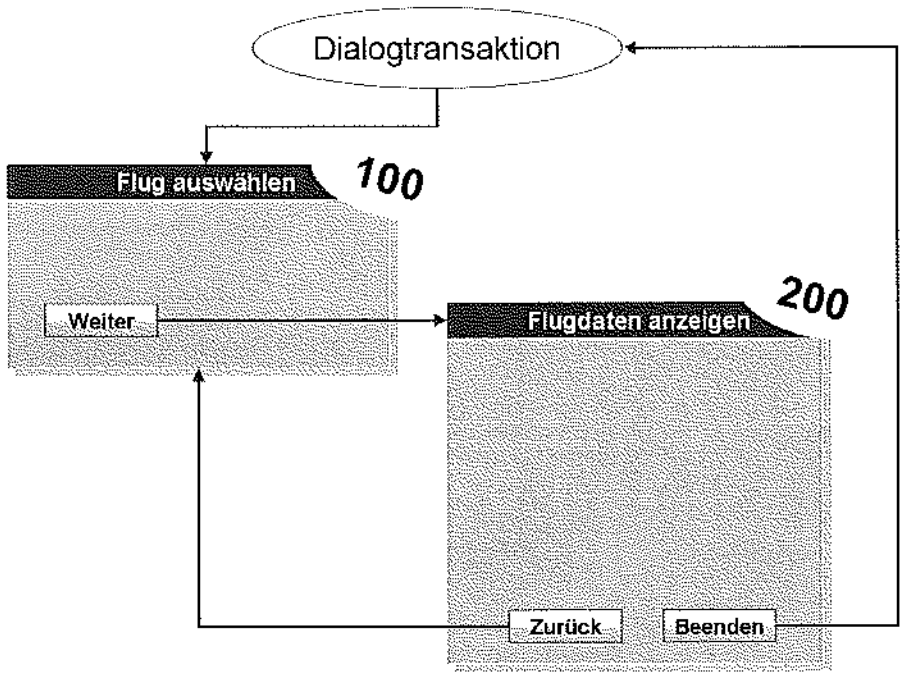


Abbildung 174: Realisierungsstufe 1: Dynpros mit Drucktasten erstellen und Navigation

In der ersten Stufe unseres Beispiels sollen zwei Dynpros angelegt werden, auf denen sich Drucktasten befinden. Das Programm soll so implementiert werden, dass nach Betätigen einer Drucktaste zum jeweils anderen Dynpro navigiert bzw. die Dynprofolge beendet wird. Die Dynprofolge soll über eine Dialogtransaktion gestartet werden.

Dieser Abschnitt befasst sich daher mit Folgendem:

- Erstellen von Dynpros
- Ablauflogik eines Dynpros mit den Ereignisblöcken PBO und PAI
- PBO- und PAI-Module als Verarbeitungsblöcke der entsprechenden Ereignisse
- Implementierung von Drucktasten und Auswerten der Benutzeraktion
- Anlegen von Dialogtransaktionen

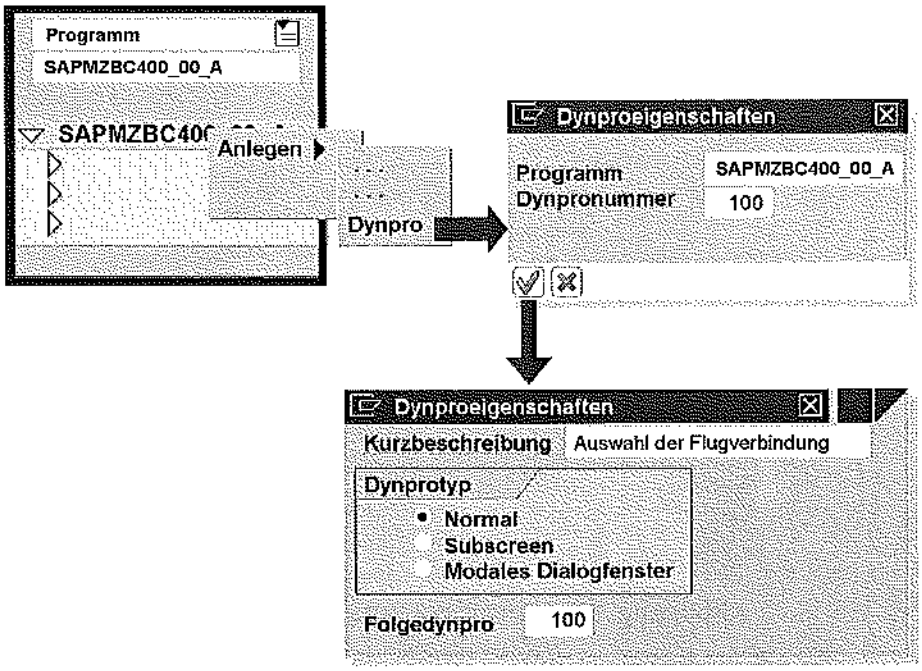


Abbildung 175: Dynpro erstellen

Sie haben mehrere Möglichkeiten, ein Dynpro anzulegen:

Über den Object Navigator

Auf der Objektliste im Navigationsbereich können Sie über das Kontextmenü zu Ihrem Programm ein neues Programmobjekt (*Dynpro*) anlegen (siehe obige Grafik).

Per Vorwärtsnavigation vom ABAP-Editor aus

Aus dem *ABAP-Editor* heraus legen Sie ein Dynpro über Doppelklick auf die Dynpronummer an (in der Anweisung CALL SCREEN nnnn). Dabei verzweigen Sie ebenfalls automatisch in den *Screen Painter* zur Pflege des neuen Dynpros.

Bei der Dynproerstellung werden Sie vom System zunächst zur Pflege der **Dynproeigenschaften** geführt. Hier geben Sie einen **Kurztext** ein, wählen dann als **Dynprotyp** *Normal* und geben die Nummer des **Folgedynpros** an.

Die Angabe des Folgedynpros `0` bewirkt, dass nach kompletter Abarbeitung des Dynpros die Verarbeitung unmittelbar hinter der Dynproaufrufstelle fortgesetzt wird.

Achtung: Da Null der Initialwert des Feldes ist, wird ihre Anzeige bei späterer Auflistung der Dynproattribute unterdrückt.

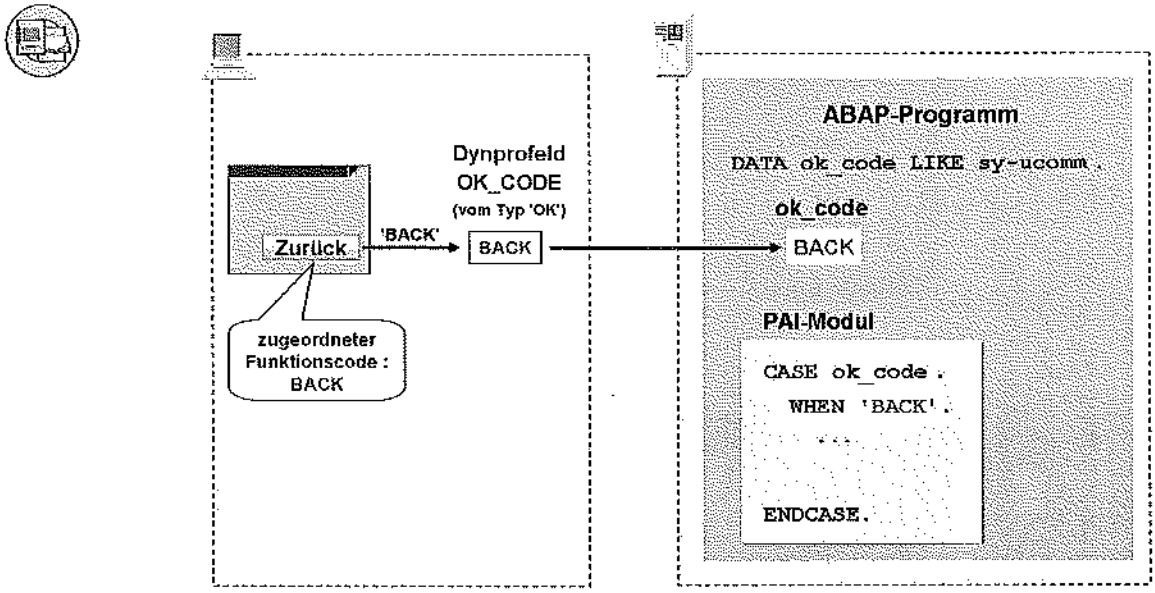


Abbildung 176: Ablauf bei Betätigung einer Drucktaste

Betätigt der Anwender eine Drucktaste, so überträgt das Laufzeitsystem den dieser Drucktaste zugeordneten **Funktionscode** in ein spezielles Dynprofeld (vom Typ 'OK'). Üblicherweise wird dieses Dynprofeld **OK_CODE** genannt.

Der Inhalt dieses speziellen Dynprofeldes wird dann bei Vorhandensein eines namensgleichen programminternen Datenobjekts automatisch dorthin transportiert. Anschließend wird die PAI-Abarbeitung angestoßen, in der Sie über den ins Programm übergebenen Funktionscode die Benutzeraktion erfahren und die entsprechende Verarbeitung durchführen.

Wie Sie Drucktasten definieren, das spezielle Dynprofeld vom Typ 'OK' nutzen, programmintern ein gleichnamiges Datenobjekt deklarieren und im PAI auf die jeweilige Benutzeraktion reagieren, erfahren Sie im Folgenden.

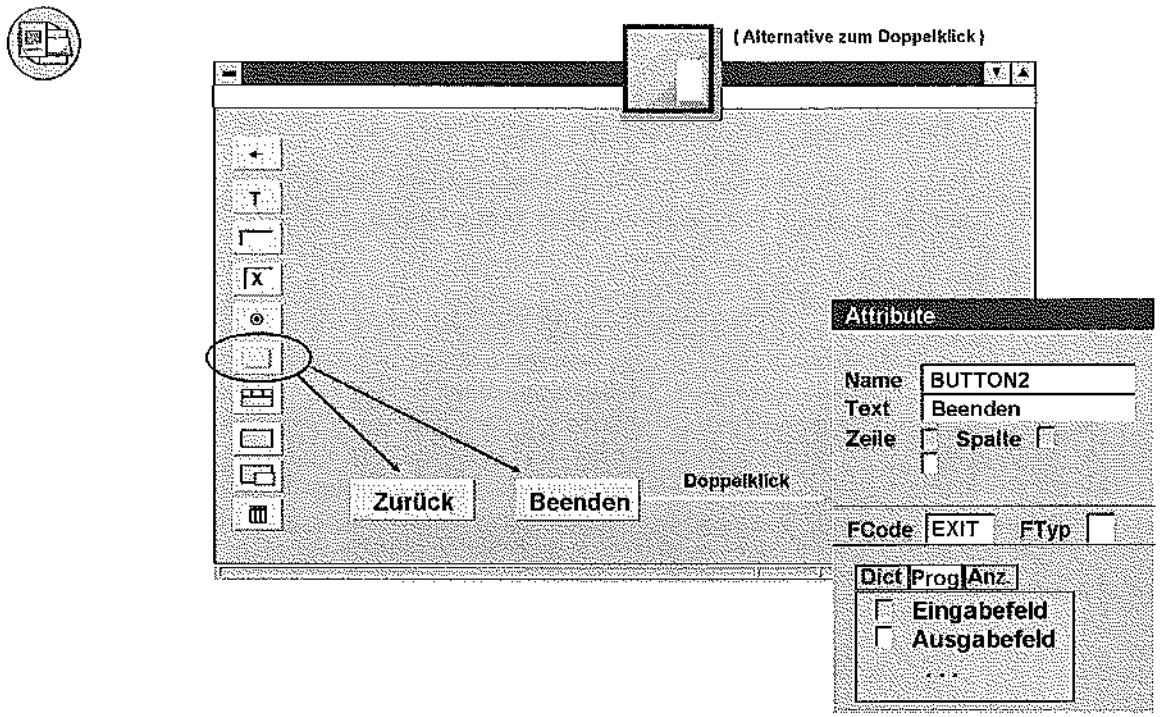


Abbildung 177: Drucktaste definieren / Funktionscode zuordnen

Obige Abbildung zeigt, wie Sie im grafischen *Layout-Editor* Drucktasten definieren. Jeder Drucktaste müssen Sie einen Namen und einen Funktionscode zuordnen. Dies können Sie über die Pflege der Feldeigenschaften realisieren.

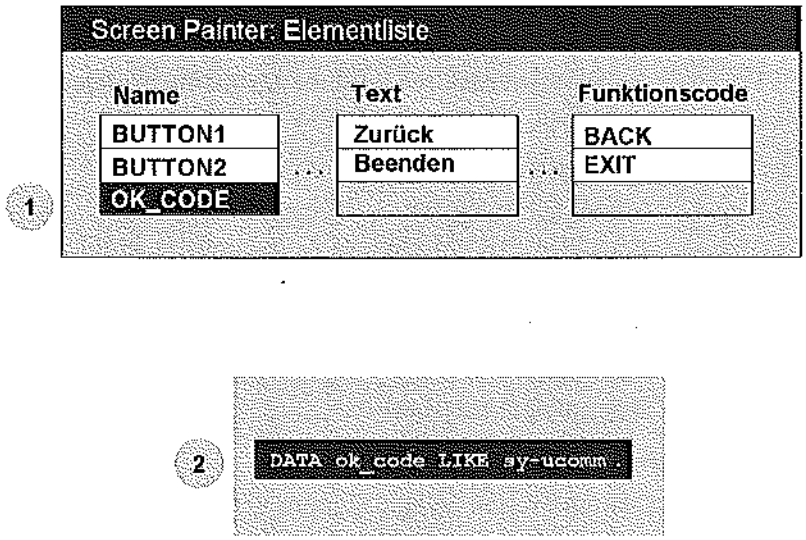


Abbildung 178: Realisierung des Funktionscode-Transports

Das spezielle Dynprofeld, über das der jeweilige Funktionscode zum Programm transportiert wird, heißt Befehlsfeld und ist standardmäßig auf dem Dynpro vorhanden. Um es nutzen zu können, müssen Sie es mit einem Namen versehen. Üblicherweise wird als Name **OK_CODE** verwendet.

Das programminterne gleichnamige Datenobjekt deklarieren Sie mit Typisierung auf das Systemfeld **sy-ucomm** (siehe obige Grafik).

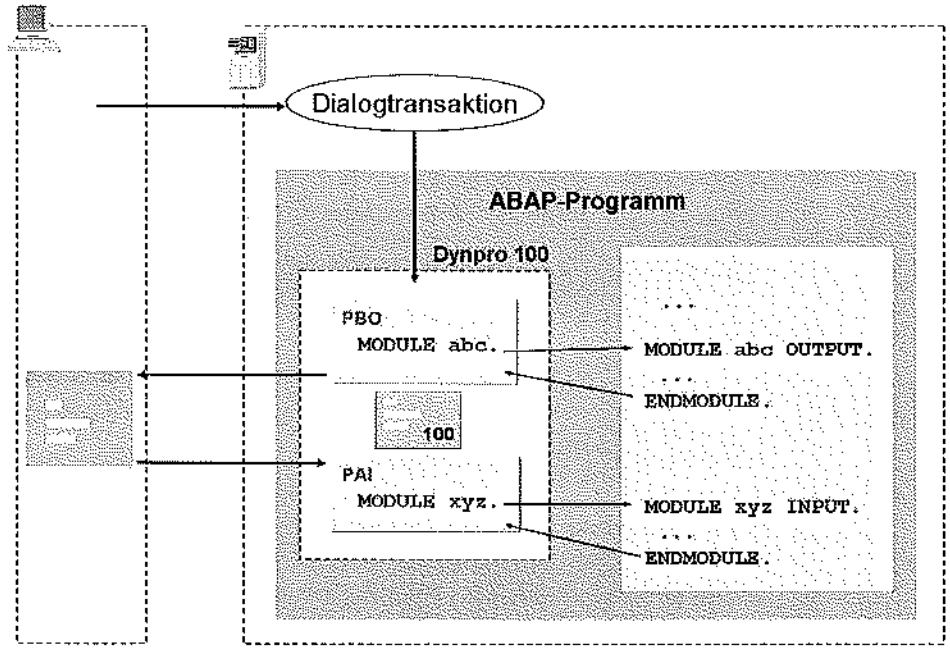


Abbildung 179: Laufzeitarchitektur der Dynpro-Ablaufsteuerung

Über die Dialogtransaktion wird die **Dynproprozessierung** angestoßen. Dazu gehören folgende (automatisch ablaufende) Schritte:

PBO-Abarbeitung

Als Vorbereitung für die Dynproanzeige werden die im PBO-Block genannten PBO-Module in Auflistungsreihenfolge nacheinander abgearbeitet.

Feldtransport vom Programm zum Dynpro

Nach PBO-Abarbeitung werden die anzuzeigenden Daten in die Dynprofelder transportiert.

Dynproanzeige

Das mit Werten belegte Dynpro wird zum Präsentationsserver (SAPGui) gesendet und dem Benutzer angezeigt.

Feldtransport vom Dynpro zum Programm

Jede Benutzeraktion auf dem Dynpro stößt den Transport der Dynprofeldinhalte zurück ins Programm an.

PAI-Abarbeitung

Als Reaktion auf die Benutzeraktion werden die in PAI aufgelisteten Module nacheinander abgearbeitet.

Module sind Quelltextblöcke ohne Schnittstelle. Sie werden durch die ABAP-Anweisungen MODULE und ENDMODULE eingefasst. Für jede Teilfunktion im PBO bzw. PAI sollte ein entsprechendes Modul implementiert werden.

Achtung: Die **Ablauflogik** des Dynpros (PBO/PAI) enthält lediglich **Verweise auf Module**, die im Programm implementiert sind und aus ABAP-Anweisungen bestehen. ABAP-Quelltext darf **nicht** direkt in der Ablauflogik abgelegt werden.

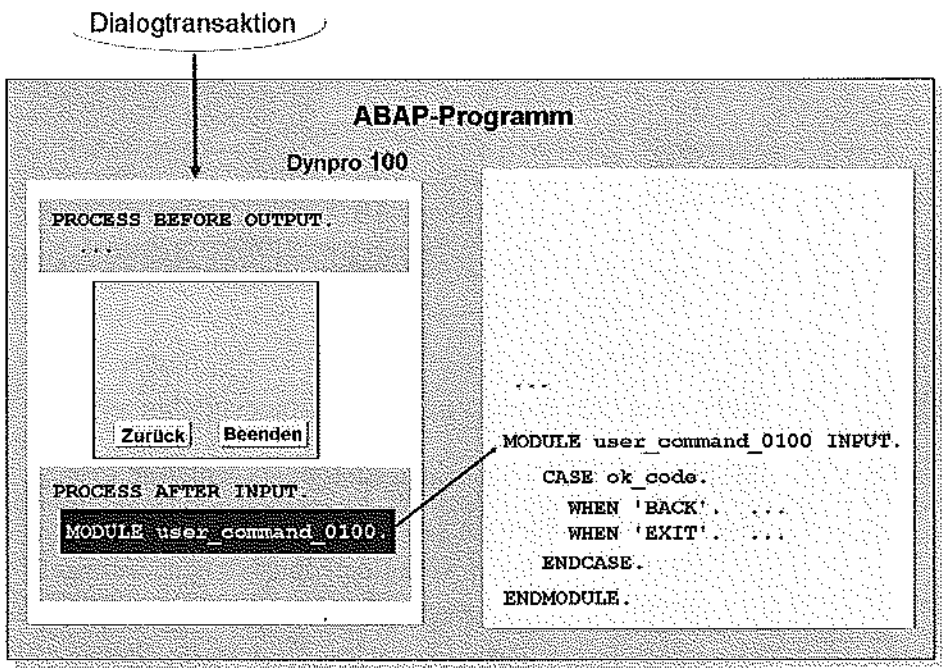


Abbildung 180: Auswertung der Funktionscodes zu PAI

Obige Grafik zeigt die Reaktion des Programms auf die Benutzeraktion. Dabei wird der entsprechende, unmittelbar vor PAI-Abarbeitung ins OK_CODE-Feld übertragene Funktionscode ausgewertet.

Üblicherweise wird das Modul mit der **Hauptverarbeitung im PAI** **USER_COMMAND_nnnn** genannt, wobei nnnn für die Dynpronummer steht.

Hinweis: In einem Modul können Sie auf **alle globalen Datenobjekte** des Programms zugreifen. Innerhalb eines Moduls definierte Variablen sind stets **global**!

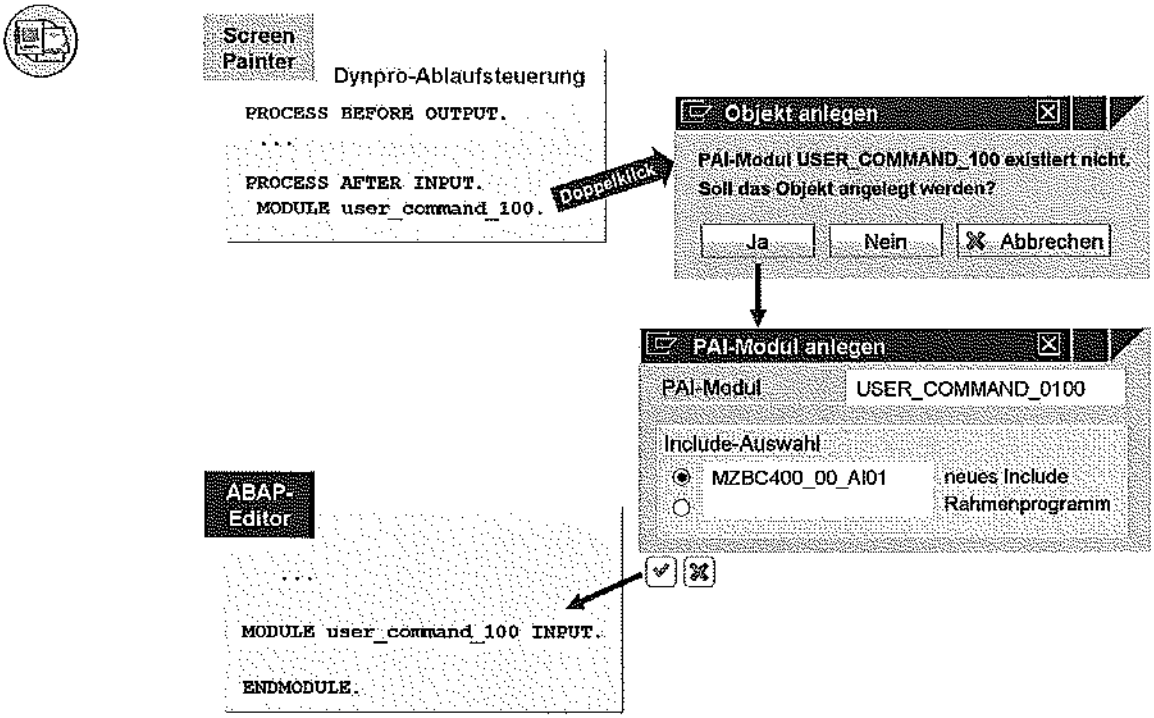


Abbildung 181: Module über Vorwärtsnavigation anlegen

Sie haben zwei Möglichkeiten, ein Modul anzulegen:

Per Vorwärtsnavigation von der Ablauflogik aus

Sie können von der Ablauflogik Ihres Dynpros aus per Doppelklick auf einen Modulverweis das entsprechende Modul anlegen (siehe obige Grafik).

Über den Navigationsbereich des Object Navigator

Sie können ein Modul auch von der Objektliste Ihres Programms aus anlegen. Benutzen Sie dazu das Kontextmenü des Programms. Beachten Sie aber, dass in diesem Fall ein entsprechender Verweis auf das neu angelegte Modul noch in der Ablauflogik Ihres Dynpros aufzunehmen ist.

Ein Modul kann aus der Ablauflogik verschiedener Dynpros heraus gerufen werden (Wiederverwendbarkeit).

Beachten Sie, dass Module, die mit `MODULE ... OUTPUT` beginnen, **PBO-Module** sind und nur vom PBO eines Dynpros aus gerufen werden können. Entsprechendes gilt für **PAI-Module**, die mit `MODULE ... INPUT` beginnen.

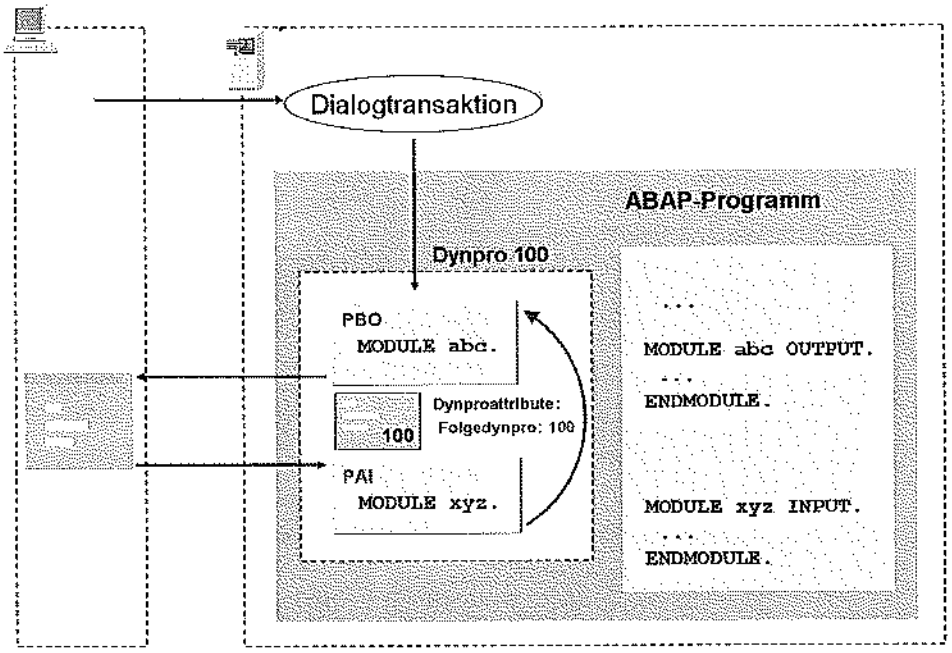


Abbildung 182: Folgedynpro „selbes Dynpro" (Wirkung)

Wird einem Dynpro seine eigene Bildnummer als Folgedynpro zugeordnet, so wird es nach seiner Beendigung erneut prozessiert.

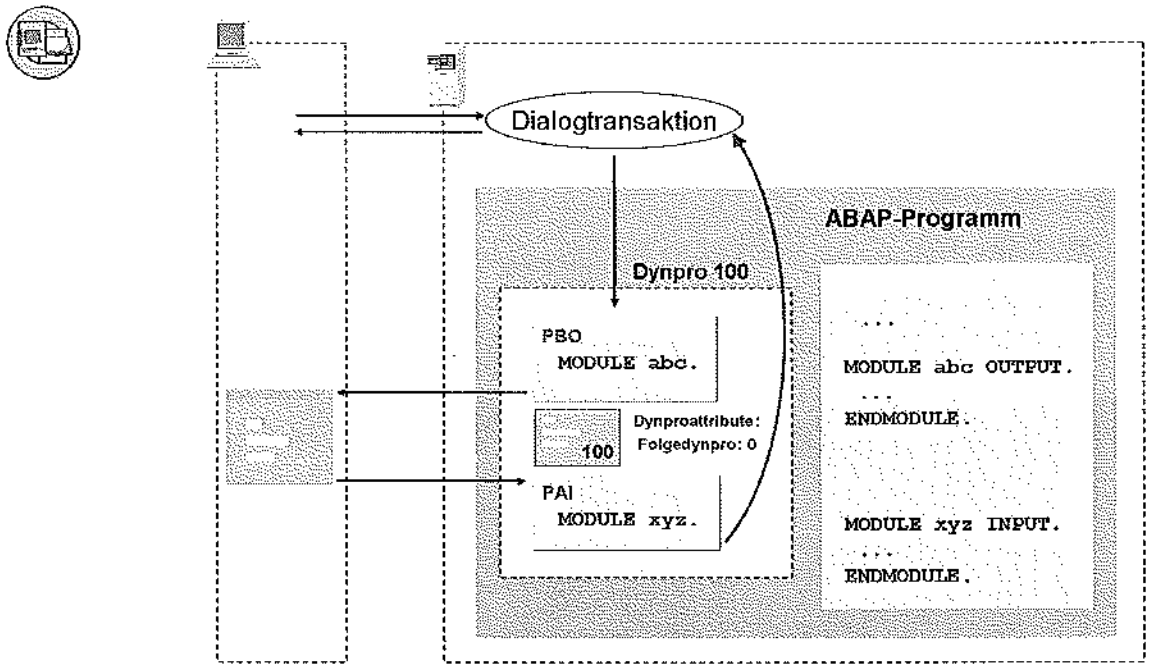


Abbildung 183: Folgedynpro 0 (Wirkung)

Das Dynproattribut „Folgedynpro =0“ bewirkt, dass nach dem kompletten Abarbeiten des Dynpros die Verarbeitung an der Dynproaufrufstelle fortgesetzt wird.

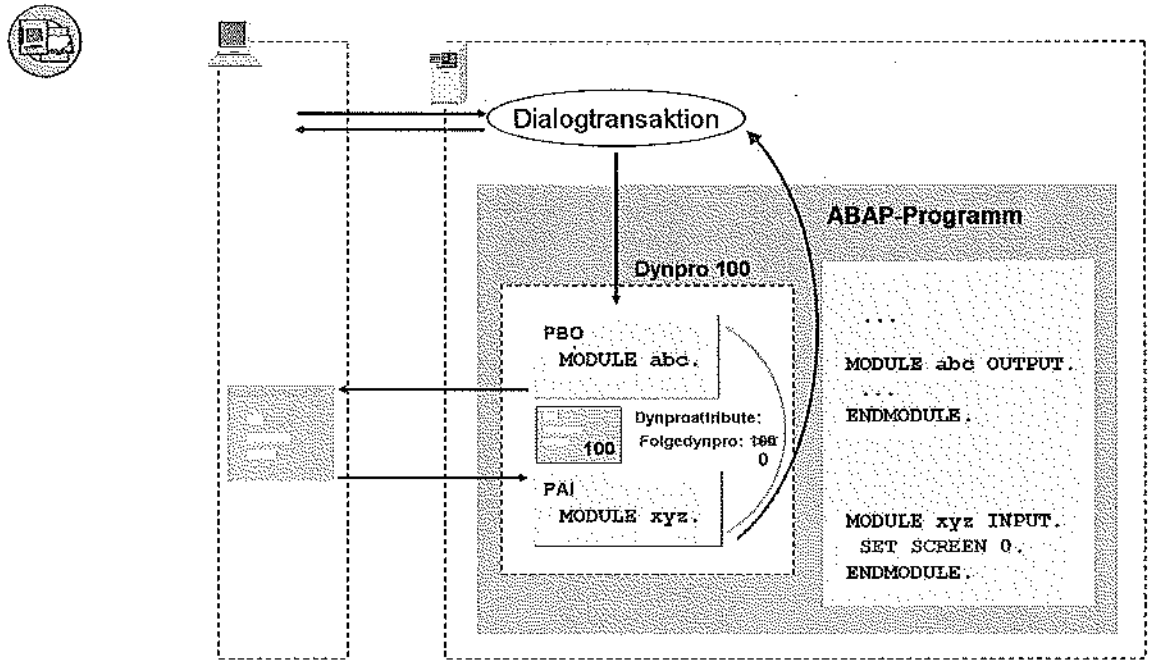


Abbildung 184: Dynamische Übersteuerung des Default-Folgedynpros

Über die ABAP-Anweisung SET SCREEN kann aus einem PAI-Modul heraus das in den Dynproattributen festgelegte Default-Folgedynpro **dynamisch übersteuert** werden (siehe obige Grafik). Diese Möglichkeit kann zur Implementierung des folgenden SAP-Standards verwendet werden: Bei Betätigung der **Enter**-Taste kommt man wieder auf das gleiche Dynpro; nur bestimmte Drucktasten verzweigen auf andere Bilder. Dazu geben Sie zu Ihrem Dynpro seine eigene Bildnummer als Folgedynpro an (Default-Folgedynpro) und verzweigen von PAI aus nur bei bestimmten Drucktasten über SET SCREEN auf andere Dynpros. (Die **Enter-Taste** stellt standardmäßig **keinen** Funtionscode in das Befehlsfeld des Dynpros. Somit wird der Initialwert des Feldes (**Space**) als Funktionscode zum Programm transportiert.)

Beachten Sie, dass bei erneutem Prozessieren desselben Dynpros auch alle PBO-Module durchlaufen werden. Falls Sie sich dafür entscheiden, die TABLES-Struktur in einem PBO-Modul zu füllen, müssen Sie sicherstellen, dass dies beim zweiten Aufruf nicht die Datenänderungen des Anwenders auf dem Dynpro überschreibt.

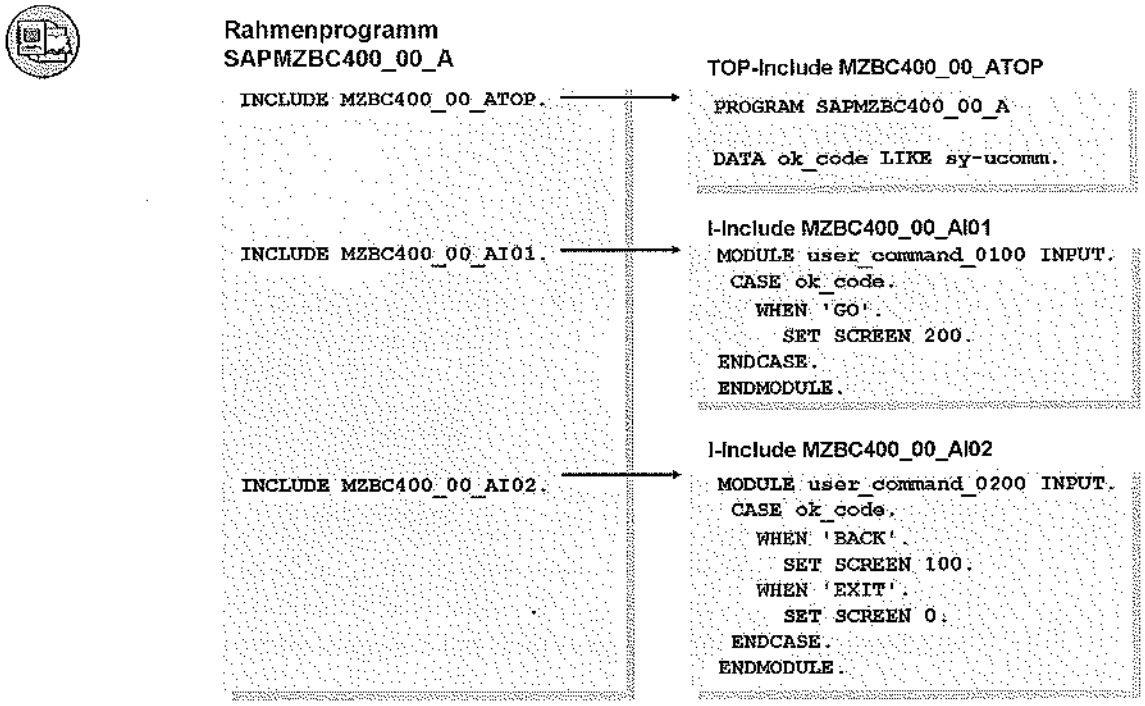


Abbildung 185: Syntaxbeispiel: Auswertung der Funktionscodes

Für unser Szenario sollen auf dem ersten Dynpro eine Drucktaste und auf dem zweiten Dynpro zwei Drucktasten abgefangen werden:

- Bei ***Weiter*** (hier Funktionscode GO) auf dem ersten Dynpro wird das Folgedynpro dynamisch auf 200 gesetzt, um zum zweiten Dynpro zu gelangen.
- Bei ***Zurück*** (hier Funktionscode BACK) auf dem zweiten Dynpro wird das Folgedynpro dynamisch auf 100 gesetzt, um zum ersten Dynpro zurückzugelangen.
- Bei **Beenden** (hier Funktionscode EXIT) wird das Folgedynpro dynamisch auf Null gesetzt, um zur Aufrufstelle des Dynpros zurückzukehren. In unserem Fall bedeutet das die Beendigung der Dialogtransaktion.

Dynpros, Ein-/Ausgabefelder anlegen

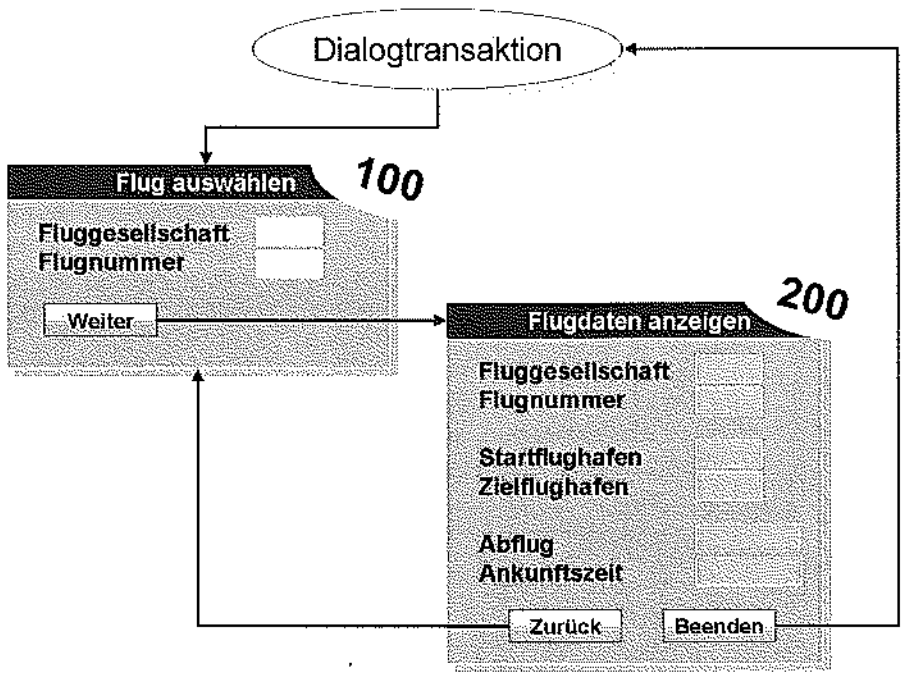


Abbildung 186: Realisierungsstufe 2: Ein-/Ausgabefelder anlegen

Im zweiten Schritt sollen auf den beiden Dynpros Ein-/Ausgabefelder angelegt werden. Das erste Dynpro soll die eingabebereiten Felder *Fluggesellschaft* und*Flugnummer* enthalten. Das zweite Dynpro soll ausschließlich Anzeigefelder enthalten und zwar zusätzlich zu *Fluggesellschaft* und*Flugnummer* noch die Felder *Startflughafen*, *Zielflughafen*, *Abflug* und *Ankunftszeit* beinhalten.

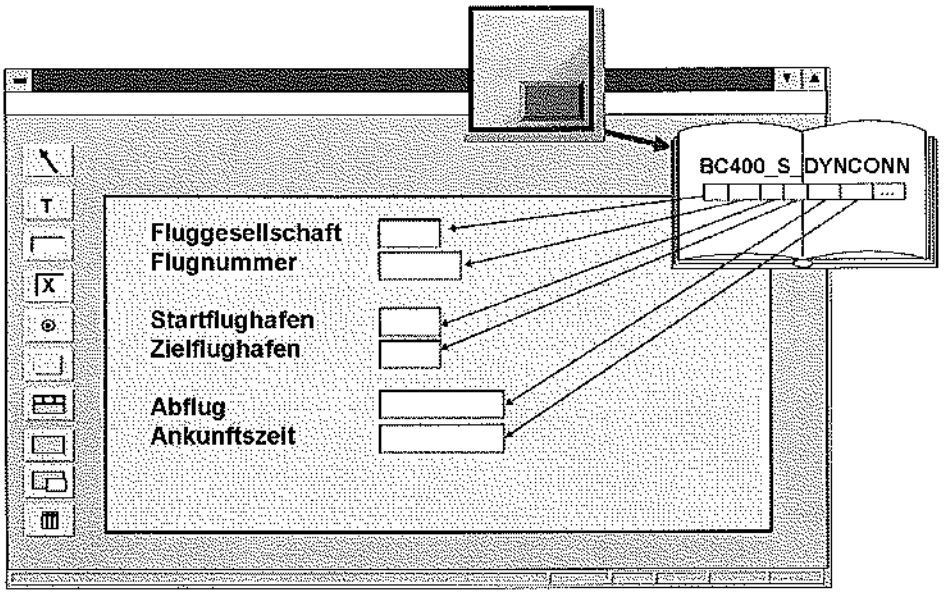


Abbildung 187: Ein-/Ausgabefelder anlegen (mit Bezug zu Dictionary-Feldern)

Sie haben zwei Möglichkeiten, beim Anlegen von Dynprofeldern die Zuordnung der Feldeigenschaften zu realisieren:

Übernehmen aus dem Dictionary

Sie können beim Anlegen eines Dynprofeldes den Typ sowie Feldeigenschaften eines Dictionary-Feldes übernehmen. Ihnen steht dann die gesamte Information zur Verfügung einschließlich der semantischen Informationen des entsprechenden Datenelementes und der Fremdschlüsselbeziehungen. Auch der Feldname wird aus dem *ABAP Dictionary* übernommen.

Übernehmen aus dem Programm

Sie können für ein Dynprofeld die Feldeigenschaften eines programminternen Datenobjektes übernehmen. Hierzu muss eine **generierte** Version des Programms vorliegen (wird beim Aktivieren automatisch erzeugt). Als Feldname wird der Name des Datenobjektes übernommen.

Der *grafische Layout-Editor* bietet Ihnen auch bequeme Möglichkeiten, weitere Bildschirmelemente, wie z. B. Texte, Rahmen, Auswahlknöpfe, Ankreuzfelder, zu definieren. Dabei wählen Sie zunächst per Mausklick in der Werkzeugleiste das gewünschte Bildschirmelement aus und positionieren es dann auf den Dynpropflegebereich.

Sie löschen Bildschirmelemente, indem Sie das Element zunächst mit der Maus markieren und die Drucktaste *Löschen* wählen.

Sie verschieben Bildschirmelemente, indem Sie sie mit gedrückter linker Maustaste neu positionieren.

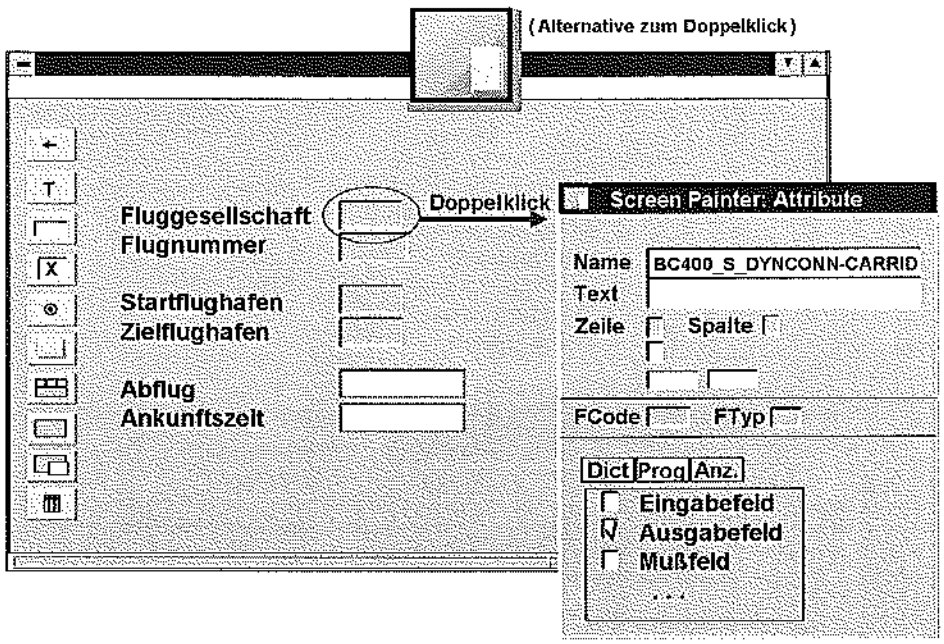


Abbildung 188: Attribute eines Dynpro-Elements pflegen

Zur Pflege der Eigenschaften eines Dynpro-Elements doppelklicken Sie es. Seine Attribute werden dann in einem Zusatzfenster zur Pflege angezeigt. (Alternativ zum Doppelklick können Sie auch das Element markieren und die Drucktaste *Attribute* betätigen.)

Sie können einem Dynprofeld das Attribut *Eingabe obligatorisch (Mussfeld)* zuordnen. Zur Laufzeit erscheint dann eine entsprechende Markierung in dem Feld, falls es initial ist. Wird das Feld vom Benutzer nicht ausgefüllt, so folgt bei beliebigen Benutzeraktionen ein Fehlerdialog, wonach die Eingabefelder wieder eingabebereit zur Verfügung stehen.

Datentransport zwischen Programm und Dynpro

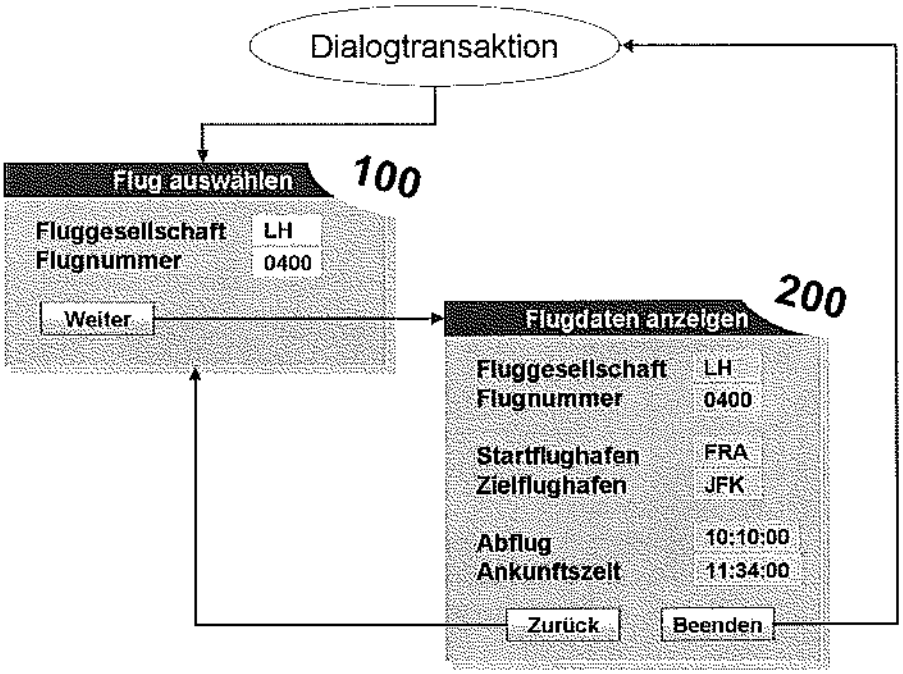


Abbildung 189: Realisierungsstufe 3: Daten auf Dynpro anzeigen

Nach der zweiten Realisierungsstufe ist es zwar möglich, auf dem ersten Dynpro eine Flugverbindung auszuwählen und auf das zweite Dynpro zu navigieren. Dort haben aber nur die Dynprofelder *Fluggesellschaft* und *Flugnummer* einen Wert. Das liegt daran, dass sie exakt die gleichen Namen tragen wie die Eingabefelder auf dem ersten Dynpro. In der zweiten Stufe soll nun der Datentransport programmiert werden und zwar sowohl vom (ersten) Dynpro in das Programm als auch vom Programm auf das zweite Dynpro. Dadurch können die Benutzereingaben im Programm verarbeitet werden und zugehörige Detailinformationen auf dem zweiten Dynpro angezeigt werden.

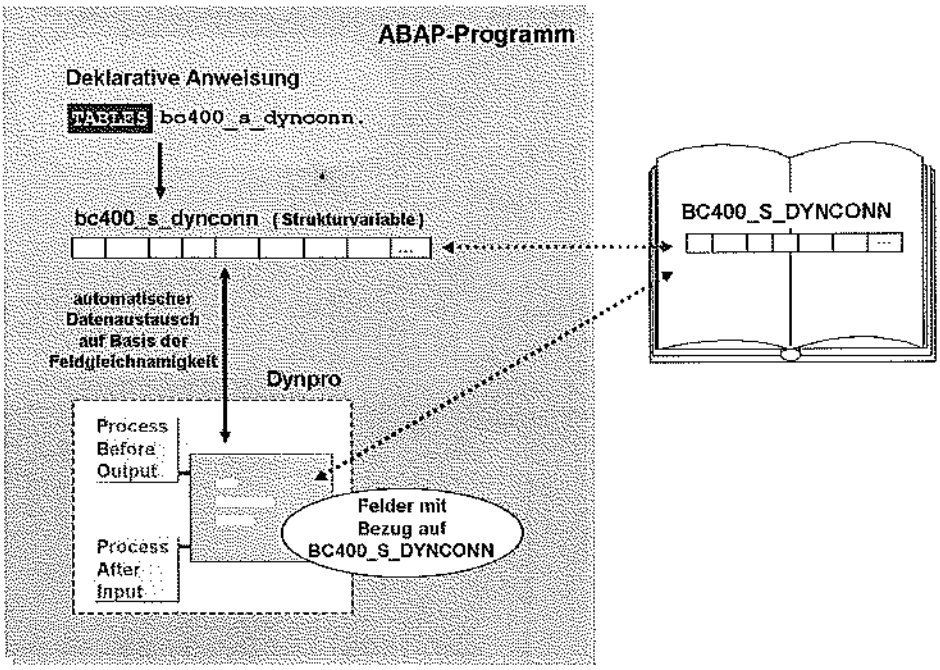


Abbildung 190: TABLES-Struktur als Schnittstelle zum Dynpro

Mit der TABLES-Anweisung wird mit Bezug auf die angegebene Dictionary-Struktur (bzw. transparente Tabelle) programmintern eine gleichnamige, typgleiche Strukturvariable definiert, die als Schnittstelle zwischen Programm und Dynpro dient:

Beziehen sich die Dynprofelder und die TABLES-Anweisung auf denselben strukturierten Typ, werden die Daten ihrer Felder auf Basis der Feldgleichnamigkeit **automatisch** ausgetauscht: Nach PBO von der TABLES-Struktur zu den Dynprofeldern und vor PAI in die umgekehrte Richtung.

Es ist üblich, dass im *ABAP Dictionary* eine spezielle Struktur angelegt wird, die die Felder enthält, die auf den Dynpros einer Anwendung angezeigt werden sollen. Diese Struktur kann auch Felder mehrerer Datenbanktabellen enthalten. Auf dem Dynpro werden die Felder mit Bezug auf diese Struktur definiert und programmintern über die entsprechende TABLES-Anweisung eine Schnittstellenstruktur realisiert. Auf diese Weise hat man trotz Felder verschiedener Tabellen auf dem Dynpro eine **eindeutige** Schnittstellenstruktur für den Datenaustausch zwischen Programm und Dynpro.

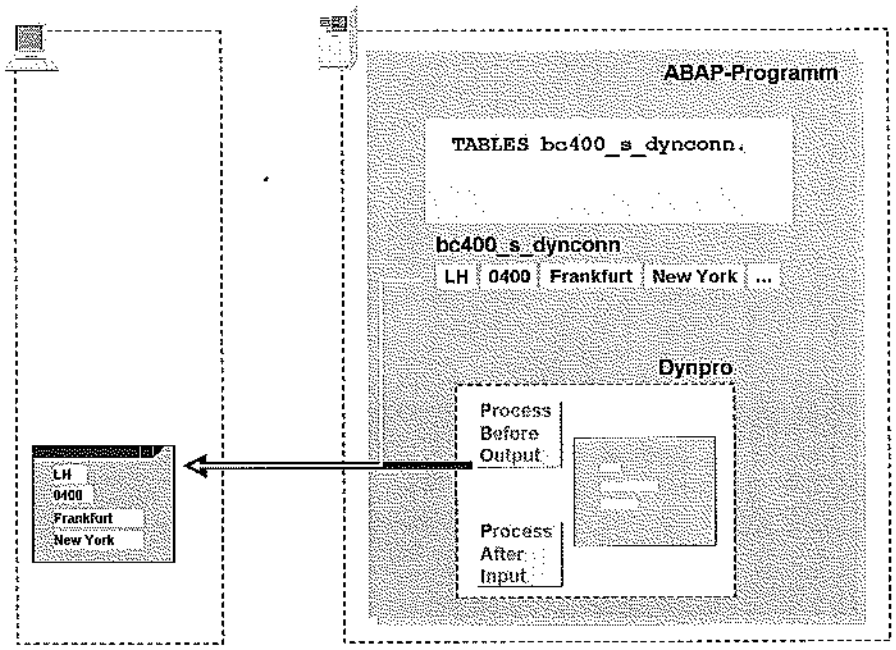


Abbildung 191: Datentransport vom Programm zum Dynpro

Nach Abarbeitung des Ereignisses **PBO** und unmittelbar vor Senden eines Dynpros zum Präsentationsserver werden die Feldinhalte der programminternen TABLES-Struktur automatisch in die gleichnamigen Dynprofelder kopiert. Der Zeitpunkt dieses automatischen Datentransports ist sinnvoll, da im PBO oft noch Daten für die Dynproanzeige in der TABLES-Struktur vorbereitet werden.

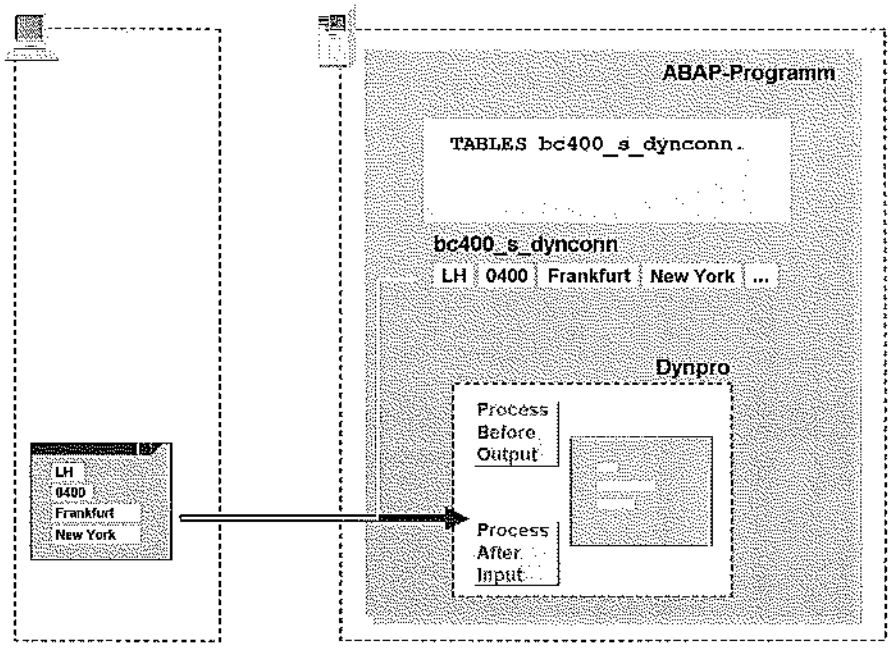


Abbildung 192: Datentransport vom Dynpro zum Programm

Bei Benutzeraktionen auf dem Dynpro werden noch **vor** Abarbeitung des Ereignisses **PAI** die Inhalte der Dynprofelder in die namensgleichen TABLES-Strukturfelder transportiert. Der Zeitpunkt dieses automatischen Datentransports ist sinnvoll, da im PAI die Weiterverarbeitung der Benutzereingaben stattfinden soll und diese daher zu diesem Zeitpunkt im Programm verfügbar sein müssen.

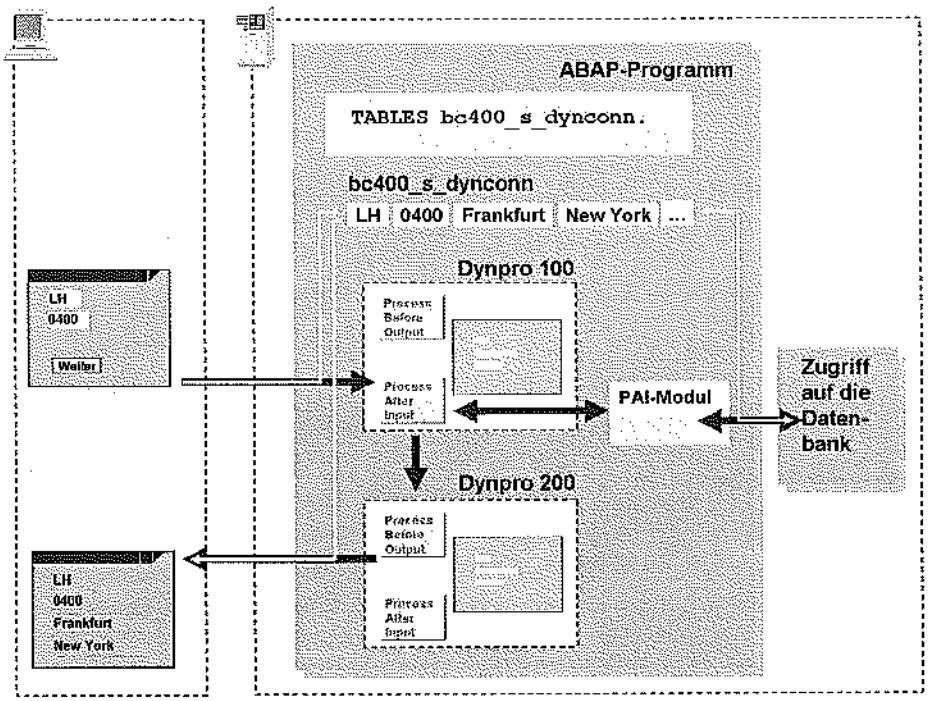


Abbildung 193: Datentransport im Beispielprogramm

Der Ablauf und Datentransport sieht in unserem Beispielprogramm wie folgt aus:

1. Bei einer Benutzeraktion auf Dynpro 100 werden die Benutzereingaben automatisch in die programminterne TABLES-Struktur BC400_S_DYNCONN transportiert.
2. Anschließend wird das PAI-Modul USER_COMMAND_0100 prozessiert.
3. Abhängig vom Funktionscode wird im PAI-Modul ein Funktionsbaustein zur Datenbeschaffung aufgerufen. Dabei werden dem Funktionsbaustein die Benutzereingaben aus der TABLES-Struktur übergeben. Die Daten, die auf Dynpro 200 angezeigt werden sollen, werden anschließend in die gleiche TABLES-Struktur gestellt
4. Nach PAI des Dynpro 100 wird zum Dynpro 200 navigiert.
5. Nach PBO des zweiten Dynpros werden die Daten automatisch von der TABLES-Struktur in die Dynprofelder transportiert.

```
MODULE user_command_0100 INPUT.

  CASE ok_code.
    WHEN 'GO'.
      MOVE-CORRESPONDING bc400_s_dynconn TO gs_connection.

      CALL FUNCTION 'BC400_DDS_CONNECTION_GET'
        EXPORTING
          iv_carrid     = gs_connection-carrid
          iv_connid     = gs_connection-connid
        IMPORTING
          es_connection = gs_connection
        EXCEPTIONS
          no_data       = 1
          OTHERS        = 2.

      IF sy-subrc <> 0.
        MESSAGE e042(bc400) WITH gs_connection-carrid
                                 gs_connection-connid.
      ELSE.
        MOVE-CORRESPONDING gs_connection TO bc400_s_dynconn.
        SET SCREEN 200.
      ENDIF.
  ENDCASE.

ENDMODULE.                    " USER_COMMAND_0100  INPUT
```

Abbildung 194: Syntax zum Beispielprogramm

Im PAI-Modul USER_COMMAND_0100 wird der Funktionscode ausgewertet. Falls der Benutzer die Drucktaste *Weiter* (Funktionscode GO) betätigt hat, wird ein Funktionsbaustein zur Datenbeschaffung aufgerufen.

Dem Funktionsbaustein werden die Benutzereingaben aus der TABLES-Struktur übergeben. Um die Schnittstelle zum Dynpro und die Schnittstelle zum Funktionsbaustein klar voneinander zu trennen, werden die Daten nicht direkt aus der TABLES-Struktur an den Funktionsbaustein übergeben. Sie werden zunächst in das Datenobjekt kopiert, das als Schnittstelle zum Funktionsbaustein dient (globales strukturiertes Datenobjekt gs_connection vom Typ BC400_S_CONNECTION.).

War die Datenbeschaffung erfolgreich, wird das Ergebnis zurück in die TABLES-Struktur kopiert und die Navigation auf das Anzeigedynpro angestoßen. Falls der Funktionsbaustein mit einer Ausnahme abbricht, wird eine entsprechende Fehlermeldung abgesetzt. Diese Fehlermeldung wird direkt auf Dynpro 100 angezeigt.

Hinweis: Fehlermeldungen (Nachrichtentyp „E“) während der Verarbeitung eines PAI-Modules führen immer dazu, dass das entsprechende Dynpro sofort wieder angezeigt wird – ohne dass PBO erneut durchlaufen wird. Durch entsprechende Techniken lässt sich erreichen, dass die Dynprofelder direkt wieder eingabebereit sind. Näheres dazu können Sie in der Schulung BC410 „Entwicklung dynprobasierter Benutzerdialoge“ erfahren.

Hinweis: Das Lesen der Daten erfolgt in unserem Beispiel zu PAI des Eingabedynpros. Alternativ könnte die Datenbeschaffung auch erst in einem PBO-Modul des Anzeigedynpros erfolgen, also nach der Navigation. Die zweite Möglichkeit wäre sogar zu bevorzugen, um die Wiederverwendung des Anzeigedynpros zu erleichtern.

Übung 20: Dynpros anlegen und dynamische Folgebildsteuerung

Lernziele der Übung

Am Ende dieser Übung können Sie

- klassische Dynpros anlegen
- Drucktasten auf Dynpros anlegen und mit Funktionscodes verknüpfen
- Dialogmodule anlegen und Funktionscodes verarbeiten
- das Folgebild dynamisch festlegen
- eine Dialogtransaktion anlegen

Unternehmensszenario

Sie wollen eine Dialoganwendung mit zwei klassischen Dynpros entwickeln. Auf den Dynpros wollen Sie Drucktasten zur Verfügung stellen, damit der Benutzer die Navigation zum jeweils anderen Dynpro initiieren kann.

Vorlag:

keine

Lösung:

SAPMBC400_UDS_A

Aufgabe 1:

Legen Sie einen Module-Pool (Typ M Programm) mit TOP Include an.

1. Legen Sie ein Programm mit Namen **SAPMZBC400_##_A** an. Lassen Sie das Häkchen *Mit TOP-Include* gesetzt. Vergewissern Sie sich, dass dadurch automatisch der Programmtyp *M Modulpool* vorgeschlagen wird.

Aufgabe 2:

Legen Sie in Ihrem Module-Pool zwei Dynpros mit Dynprotyp *Normal* an.

1. Legen Sie ein erstes Dynpro an (vorgeschlagene Nummer: **100**). Vergeben Sie einen beschreibenden Text (vorgeschlagene Kurzbeschreibung: **Eingabedynpro**) und vergewissern Sie Sich, dass der Dynprotyp richtig gewählt sowie als Folgedynpro das aktuelle Dynpro eingetragen ist.
2. Legen Sie in gleicher Weise ein zweites Dynpro an (vorgeschlagene Nummer: **200**, vorgeschlagene Kurzbeschreibung: **Ausgabedynpro**).

Fortsetzung auf der nächsten Seite

Aufgabe 3:

Legen Sie auf den Dynpros Drucktasten an für die Navigation: Auf dem ersten Dynpro eine Drucktaste, um auf das zweite Dynpro zu gelangen, auf dem zweiten eine Drucktaste, um auf das erste Dynpro zurückzukommen. Legen Sie dann auf dem zweiten Dynpro eine weitere Drucktaste an, um das Verlassen der Anwendung zu ermöglichen.

1. Bearbeiten Sie das erste Dynpro mit dem Werkzeug *Grafischer Screen Painter* und legen Sie eine Drucktaste an.
2. Vergeben Sie für die Drucktaste einen Namen (Vorschlag: **BUTTON_GO**) und erfassen Sie einen Text (Vorschlag: **„Weiter“**).
3. Ordnen Sie der Drucktaste einen Funktionscode zu (Vorschlag: **GO**) und sichern Sie das Dynpro.
4. Legen Sie auf dem anderen Dynpro in gleicher Weise zwei Drucktasten an. Vorgeschlagene Werte:

Name	*Text*	*FktCode*
BUTTON_BACK	„Zurück“	BACK
BUTTON_EXIT	„Beenden“	EXIT

Aufgabe 4:

Legen Sie im TOP-Include des Programms ein globales Datenobjekt an, das den Funktionscode aufnehmen kann. Sorgen Sie dafür, dass für beide Dynpros beim Verlassen der Funktionscode in dieses Datenobjekt geschrieben wird.

1. Editieren Sie das TOP-Include des Programms und deklarieren Sie dort ein Datenobjekt vom Typ *SYUCOMM* (Vorgeschlagener Name: OK_CODE).
2. Editieren Sie die Elementliste des ersten Dynpros und tragen Sie den Namen Ihres Datenobjekts beim Element vom Element-Typ *OK* ein.
3. Editieren Sie die Elementliste des zweiten Dynpros entsprechend.

Aufgabe 5:

Legen Sie für jedes Dynpro ein PAI-Modul an, in dem abhängig vom Funktionscode das Folgebild festgelegt wird.

1. Fügen Sie in die Ablauflogik des ersten Dynpros den Aufruf eines PAI-Moduls ein (Namensvorschlag: **USER_COMMAND_0100**)
2. Legen Sie das Modul über Vorwärtsnavigation an. Erzeugen Sie dabei, wie vom System vorgeschlagen, ein neues Include für das Modul und vergeben Sie einen passenden Namen (Namensvorschlag: **MZBC400_##_AI01**).

Fortsetzung auf der nächsten Seite

3. Implementieren Sie das PAI-Modul. Werten Sie dabei den Inhalt des OK-Feldes aus und sorgen Sie dafür, dass nach Betätigung der Drucktaste das Folgedynpro dynamisch auf 200 gesetzt wird.
4. Legen Sie in gleicher Weise auch für das zweite Dynpro ein PAI-Modul an und implementieren Sie es entsprechend. Zum Beenden des Programms setzen Sie das Folgedynpro dynamisch auf den Wert 0.

Aufgabe 6:

Legen Sie eine Dialogtransaktion an, um die Anwendung mit dem ersten Dynpro zu starten. Aktivieren und testen Sie Ihr Programm.

1. Legen Sie eine Transaktion an mit Startobjekt *Programm und Dynpro (Dialogtransaktion)* (Namensvorschlag: **ZBC400_##_A**).

Lösung 20: Dynpros anlegen und dynamische Folgebildsteuerung

Aufgabe 1:

Legen Sie einen Module-Pool (Typ M Programm) mit TOP Include an.

1. Legen Sie ein Programm mit Namen **SAPMZBC400_##_A** an. Lassen Sie das Häkchen *Mit TOP-Include* gesetzt. Vergewissern Sie sich, dass dadurch automatisch der Programmtyp *M Modulpool* vorgeschlagen wird.

   a) Wählen Sie im Kontextmenü zu Ihrem Paket *Anlegen → Programm*.

   b) Geben Sie den Namen des Programms ein, lassen Sie das Häkchen *Mit TOP-Include* gesetzt und wählen Sie *Weiter* ✔.

   c) Bestätigen Sie den vorgeschlagenen Namen des TOP-Include.

   d) Überprüfen Sie die Programmattribute und schließen Sie den Vorgang ab.

   e) Ordnen Sie das Programm und die Includes wie gewohnt Ihrem Paket und Transportauftrag zu.

Aufgabe 2:

Legen Sie in Ihrem Module-Pool zwei Dynpros mit Dynprotyp *Normal* an.

1. Legen Sie ein erstes Dynpro an (vorgeschlagene Nummer: **100**). Vergeben Sie einen beschreibenden Text (vorgeschlagene Kurzbeschreibung: **Eingabedynpro**) und vergewissern Sie Sich, dass der Dynprotyp richtig gewählt sowie als Folgedynpro das aktuelle Dynpro eingetragen ist.

   a) Wählen Sie im Kontextmenü zum Programm *Anlegen → Dynpro* und geben Sie die Dynpronummer ein.

   b) Pflegen Sie den beschreibenden Text, überprüfen Sie die übrigen Angaben und schließen Sie den Vorgang ab.

2. Legen Sie in gleicher Weise ein zweites Dynpro an (vorgeschlagene Nummer: **200**, vorgeschlagene Kurzbeschreibung: **Ausgabedynpro**).

   a) Führen Sie diesen Schritt wie zuvor durch.

Fortsetzung auf der nächsten Seite

Aufgabe 3:

Legen Sie auf den Dynpros Drucktasten an für die Navigation: Auf dem ersten Dynpro eine Drucktaste, um auf das zweite Dynpro zu gelangen, auf dem zweiten eine Drucktaste, um auf das erste Dynpro zurückzukommen. Legen Sie dann auf dem zweiten Dynpro eine weitere Drucktaste an, um das Verlassen der Anwendung zu ermöglichen.

1. Bearbeiten Sie das erste Dynpro mit dem Werkzeug *Grafischer Screen Painter* und legen Sie eine Drucktaste an.

   a) Öffnen Sie das Werkzeug über die Drucktaste ⇨*Layout*.

   b) Wählen Sie in der Werkzeugleiste am linken Rand *Drucktaste* und platzieren Sie die Drucktaste im Zeichenbereich.

2. Vergeben Sie für die Drucktaste einen Namen (Vorschlag: **BUTTON_GO**) und erfassen Sie einen Text (Vorschlag: **„Weiter“**).

   a) Tragen Sie die Werte in die entsprechenden Felder oberhalb des Zeichenbereichs ein.

3. Ordnen Sie der Drucktaste einen Funktionscode zu (Vorschlag: **GO**) und sichern Sie das Dynpro.

   a) Öffnen Sie das Attribute-Fenster durch einen Doppelklick auf die Drucktaste bzw. durch Betätigen der *F2*-Taste.

   b) Pflegen Sie im Attribute-Fenster den Inhalt des Feldes *FktCode*

4. Legen Sie auf dem anderen Dynpro in gleicher Weise zwei Drucktasten an. Vorgeschlagene Werte:

Name	*Text*	*FktCode*
BUTTON_BACK	„Zurück“	BACK
BUTTON_EXIT	„Beenden“	EXIT

   a) Führen Sie diesen Schritt wie zuvor durch.

Aufgabe 4:

Legen Sie im TOP-Include des Programms ein globales Datenobjekt an, das den Funktionscode aufnehmen kann. Sorgen Sie dafür, dass für beide Dynpros beim Verlassen der Funktionscode in dieses Datenobjekt geschrieben wird.

1. Editieren Sie das TOP-Include des Programms und deklarieren Sie dort ein Datenobjekt vom Typ *SYUCOMM* (Vorgeschlagener Name: OK_CODE).

   a) Siehe Quelltextauszug der Musterlösung.

Fortsetzung auf der nächsten Seite

2. Editieren Sie die Elementliste des ersten Dynpros und tragen Sie den Namen Ihres Datenobjekts beim Element vom Element-Typ *OK* ein.

   a) Bearbeiten Sie das erste Dynpro und navigieren Sie zum Reiter *Elementliste*.

   b) Suchen Sie das Element vom Element-Typ *OK*.

   c) Tragen Sie in der Spalte *Name* den Namen des Datenobjekts ein.

3. Editieren Sie die Elementliste des zweiten Dynpros entsprechend.

   a) Führen Sie diesen Schritt wie zuvor durch.

Aufgabe 5:

Legen Sie für jedes Dynpro ein PAI-Module an, in dem abhängig vom Funktionscode das Folgebild festgelegt wird.

1. Fügen Sie in die Ablauflogik des ersten Dynpros den Aufruf eines PAI-Moduls ein (Namensvorschlag: **USER_COMMAND_0100**)

   a) Siehe Quelltextauszug der Musterlösung.

2. Legen Sie das Modul über Vorwärtsnavigation an. Erzeugen Sie dabei, wie vom System vorgeschlagen, ein neues Include für das Modul und vergeben Sie einen passenden Namen (Namensvorschlag: **MZBC400_##_AI01**).

   a) Siehe Quelltextauszug der Musterlösung.

3. Implementieren Sie das PAI-Modul. Werten Sie dabei den Inhalt des OK-Feldes aus und sorgen Sie dafür, dass nach Betätigung der Drucktaste das Folgedynpro dynamisch auf 200 gesetzt wird.

   a) Siehe Quelltextauszug der Musterlösung.

4. Legen Sie in gleicher Weise auch für das zweite Dynpro ein PAI-Modul an und implementieren Sie es entsprechend. Zum Beenden des Programms setzen Sie das Folgedynpro dynamisch auf den Wert 0.

   a) Siehe Quelltextauszug der Musterlösung.

Aufgabe 6:

Legen Sie eine Dialogtransaktion an, um die Anwendung mit dem ersten Dynpro zu starten. Aktivieren und testen Sie Ihr Programm.

1. Legen Sie eine Transaktion an mit Startobjekt *Programm und Dynpro (Dialogtransaktion)* (Namensvorschlag: **ZBC400_##_A**).

   a) Wählen Sie im Kontextmenü für das Programm den Eintrag *Anlegen → Transaktion*.

Fortsetzung auf der nächsten Seite

b) Geben Sie den Namen der Transaktion und einen Kurztext ein und wählen Sie ✓.

c) Tragen Sie in den beiden Pflichtfeldern den Namen Ihres Programms und die Nummer Ihres ersten Dynpros ein. Setzen Sie die GUI-Fähigkeit auf *SAP GUI für Windows*.

d) Sichern Sie die Transaktion, aktivieren Sie alle Teile des Programms und testen Sie die Transaktion.

Ergebnis

Dynpro 100 Ablauflogik:

```
PROCESS BEFORE OUTPUT.
* MODULE STATUS_0100.
*
PROCESS AFTER INPUT.
 MODULE USER_COMMAND_0100.
```

Dynpro 200 Ablauflogik:

```
PROCESS BEFORE OUTPUT.
* MODULE STATUS_0200.
*
PROCESS AFTER INPUT.
 MODULE USER_COMMAND_0200.
```

Quelltextauszug aus der Musterlösung:

```
*&---------------------------------------------------------------------*
*& Include MBC400_UDS_ATOP          Modulpool           SAPMBC400_UDS_A
*&---------------------------------------------------------------------*
PROGRAM  sapmbc400_uds_a.

DATA ok_code TYPE syucomm.


*---------------------------------------------------------------------*
***INCLUDE MBC400_UDS_AI01 .
*---------------------------------------------------------------------*
*&---------------------------------------------------------------------*
*&      Module  USER_COMMAND_0100  INPUT
*&---------------------------------------------------------------------*
MODULE user_command_0100 INPUT.
```

Fortsetzung auf der nächsten Seite


```
  CASE ok_code.
    WHEN 'GO'.
      SET SCREEN 200.
  ENDCASE.

ENDMODULE.                 " USER_COMMAND_0100  INPUT


*---------------------------------------------------------------------*
***INCLUDE MBC400_UDS_AI02 .
*---------------------------------------------------------------------*
*&---------------------------------------------------------------------*
*&      Module  USER_COMMAND_0200  INPUT
*&---------------------------------------------------------------------*
MODULE user_command_0200 INPUT.
 CASE ok_code.
    WHEN 'BACK'.
      SET SCREEN 100.
    WHEN 'EXIT'.
      SET SCREEN 0.
  ENDCASE.

ENDMODULE.                 " USER_COMMAND_0200  INPUT
```

Übung 21: Dynpro – Anlegen von Ein-/Ausgabefeldern

Lernziele der Übung

Am Ende dieser Übung können Sie

- mit dem Werkzeug *Grafischer Screen Painter* Ein-/Ausgabefelder mit Bezug zum Dictionary anlegen.
- Eigenschaften von Ein-Ausgabefeldern ändern

Unternehmensszenario

Sie wollen eine Dialogtransaktion entwickeln und dabei auf den Dynpros sowohl Benutzereingaben als auch die Anzeige von Daten ermöglichen.

Vorlage:

SAPMBC400_UDS_A

Lösung:

SAPMBC400_UDS_B

Aufgabe 1:

Kopieren Sie Ihr Programm SAPMZBC400_##_A oder die Vorlage SAPMBC400_UDS_A auf den Namen **SAPMZBC400_##_B**. Stellen Sie unbedingt sicher, dass die Dynpros und die Includes mitkopiert werden.

1. Kopieren Sie das Programm mit allen Dynpros und Includes. Wählen Sie passende Namen für die neuen Includes (Namensvorschlag: **MZBC400_##_BI01** bzw. **MZBC400_##_BI02**).

Aufgabe 2:

Legen Sie auf dem ersten Dynpro zwei Eingabefelder an, eines für das Fluggesellschaftskürzel und eines für die Flugnummer. Beziehen Sie sich dabei auf die entsprechenden Komponenten des Dictionary-Strukturtyps BC400_S_DYNCONN. Pflegen Sie die Eigenschaften der Felder so, dass das Dynpro erst dann verlassen wird, wenn der Benutzer in beiden Feldern eine Eingabe gemacht hat.

1. Verwenden Sie im Werkzeug *Grafischer Screen Painter* die Funktion *Dict/Programmfelder-Fenster*, um für die Komponenten *CARRID* und *CONNID* des Dictionary-Strukturtyps Eingabefelder anzulegen.
2. Setzen Sie für beide Eingabefelder das Attribut *Eingabe* auf den Wert *obligatorisch (Mussfeld)*.

Fortsetzung auf der nächsten Seite

3. (OPTIONAL) Legen Sie ein Element vom Typ Rahmen an, pflegen Sie dafür eine Überschrift und verschieben Sie die Eingabefelder in diesen Rahmen.

Aufgabe 3:

Legen Sie auch auf dem zweiten Dynpro Ein-Ausgabefelder an für Fluggesellschaftskürzel, Flugnummer, Abflugstadt, Ankunftsstadt, Abflugzeit und Ankunftszeit. Beziehen Sie sich dabei ebenfalls auf den Dictionary-Strukturtyp **BC400_S_DYNCONN**. Pflegen Sie die Eigenschaften der Felder so, dass der Benutzer hier keine Eingaben machen kann.

1. Legen Sie wie auf dem ersten Dynpro Ein-Ausgabefelder an. Wählen Sie die Komponenten *CARRID*, *CONNID*, *CITYFROM*, *CITYTO*, *DEPTIME* und *ARRTIME* des Dictionary-Strukturtyps.
2. Setzen Sie bei allen Eingabefeldern das Attribut *Eingabe* auf den Wert *nicht möglich*.
3. (OPTIONAL) Legen Sie jeweils einen Rahmen um die Schlüsselfelder der Flugverbindung und um die zusätzlichen Information.

Aufgabe 4:

Legen Sie eine Dialogtransaktion an, um die Anwendung mit dem ersten Dynpro zu starten. Aktivieren und testen Sie Ihr Programm.

1. Legen Sie eine Transaktion an mit Startobjekt *Programm und Dynpro (Dialogtransaktion)* (Namensvorschlag: **ZBC400_##_B**).

Lösung 21: Dynpro – Anlegen von Ein-/Ausgabefeldern

Aufgabe 1:

Kopieren Sie Ihr Programm SAPMZBC400_##_A oder die Vorlage SAPMBC400_UDS_A auf den Namen **SAPMZBC400_##_B**. Stellen Sie unbedingt sicher, dass die Dynpros und die Includes mitkopiert werden.

1. Kopieren Sie das Programm mit allen Dynpros und Includes. Wählen Sie passende Namen für die neuen Includes (Namensvorschlag: **MZBC400_##_BI01** bzw. **MZBC400_##_BI02**).

   a) Lassen Sie sich das Programm im Navigationsbereich der Workbench anzeigen. Öffnen Sie im Navigationsbereich das Kontextmenü zum Programmnamen und wählen Sie *Kopieren*.

   b) Geben Sie auf dem nächsten Dialogfenster den Namen des Zielprogramms ein und wählen Sie ✔*Kopieren*.

   c) Kreuzen Sie auf dem folgenden Dialogfenster alle Felder an und wählen Sie ✔*Kopieren*.

   d) Überprüfen Sie die vorgeschlagenen Namen für die kopierten Includes. Vergewissern Sie sich, dass **alle Includes markiert sind**, bevor Sie weitermachen.

   e) Ordnen Sie das kopierte Programm und die Includes wie gewohnt Ihrem Paket und Transportauftrag zu.

Fortsetzung auf der nächsten Seite

Aufgabe 2:

Legen Sie auf dem ersten Dynpro zwei Eingabefelder an, eines für das Fluggesellschaftskürzel und eines für die Flugnummer. Beziehen Sie sich dabei auf die entsprechenden Komponenten des Dictionary-Strukturtyps BC400_S_DYNCONN. Pflegen Sie die Eigenschaften der Felder so, dass das Dynpro erst dann verlassen wird, wenn der Benutzer in beiden Feldern eine Eingabe gemacht hat.

1. Verwenden Sie im Werkzeug *Grafischer Screen Painter* die Funktion *Dict/Programmfelder-Fenster*, um für die Komponenten *CARRID* und *CONNID* des Dictionary-Strukturtyps Eingabefelder anzulegen.
   a) Bearbeiten Sie das erste Dynpro und starten Sie das Werkzeug *Grafischer Screen Painter* wie gewohnt.
   b) Wählen Sie die Drucktaste *Dict/Programmfelder-Fenster* um das entsprechende Fenster zu öffnen.
   c) Geben Sie den Namen des Strukturtyps ein und drücken Sie die Taste *Holen aus Dict.*
   d) Wählen Sie die gewünschten Strukturkomponenten aus und wählen Sie ✔.
   e) Positionieren Sie die Eingabefelder und zugehörigen Bezeichner auf dem Dynpro.
2. Setzen Sie für beide Eingabefelder das Attribut *Eingabe* auf den Wert *obligatorisch (Mussfeld).*
   a) Doppelklicken Sie auf das erste Eingabefeld, um das Attributefenster zu öffnen.
   b) Wählen Sie im unteren Teil die Registerkarte *Programm.*
   c) Pflegen Sie das Attribut *Eingabe* über die Wertehilfe.
   d) Verfahren Sie für das zweite Feld entsprechend.
3. (OPTIONAL) Legen Sie ein Element vom Typ Rahmen an, pflegen Sie dafür eine Überschrift und verschieben Sie die Eingabefelder in diesen Rahmen.
   a) Wählen Sie in der Werzeugleiste am linken Rand das Werkzeug *Rahmen* und legen Sie den Rahmen auf dem Dynpro-Layout an.
   b) Pflegen Sie für den Rahmen einen Namen und Text.
   c) Markieren Sie alle Eingabefelder und ihre Bezeichner (Anklicken bei gedrückter `Strg`-Taste) und verschieben Sie sie in den Rahmen.

Fortsetzung auf der nächsten Seite

Aufgabe 3:

Legen Sie auch auf dem zweiten Dynpro Ein-Ausgabefelder an für Fluggesellschaftskürzel, Flugnummer, Abflugstadt, Ankunftsstadt, Abflugzeit und Ankunftszeit. Beziehen Sie sich dabei ebenfalls auf den Dictionary-Strukturtyp **BC400_S_DYNCONN**. Pflegen Sie die Eigenschaften der Felder so, dass der Benutzer hier keine Eingaben machen kann.

1. Legen Sie wie auf dem ersten Dynpro Ein-Ausgabefelder an. Wählen Sie die Komponenten *CARRID*, *CONNID*, *CITYFROM*, *CITYTO*, *DEPTIME* und *ARRTIME* des Dictionary-Strukturtyps.

   a) Verfahren Sie wie beim ersten Dynpro.

2. Setzen Sie bei allen Eingabefeldern das Attribut *Eingabe* auf den Wert *nicht möglich*.

   a) Verfahren Sie wie beim ersten Dynpro.

3. (OPTIONAL) Legen Sie jeweils einen Rahmen um die Schlüsselfelder der Flugverbindung und um die zusätzlichen Information.

   a) Verfahren Sie wie beim ersten Dynpro.

Aufgabe 4:

Legen Sie eine Dialogtransaktion an, um die Anwendung mit dem ersten Dynpro zu starten. Aktivieren und testen Sie Ihr Programm.

1. Legen Sie eine Transaktion an mit Startobjekt *Programm und Dynpro (Dialogtransaktion)* (Namensvorschlag: **ZBC400_##_B**).

   a) Wählen Sie im Kontextmenü für das Programm den Eintrag *Anlegen → Transaktion*.

   b) Geben Sie den Namen der Transaktion und einen Kurztext ein und wählen Sie ✔.

   c) Tragen Sie in den beiden Pflichtfeldern den Namen Ihres Programms und die Nummer Ihres ersten Dynpros ein und sichern Sie die Transaktion.

   d) Sichern Sie die Transaktion, aktivieren Sie das Programm und Testen Sie die Transaktion.

Übung 22: Dynpro – Datentransport

Lernziele der Übung

Am Ende dieser Übung können Sie

- Dynprofelder mit Daten aus dem Programm versorgen
- Benutzereingaben im Programm weiterverarbeiten

Unternehmensszenario

Sie wollen eine Dialogtransaktion entwickeln, bei der die Eingaben auf dem ersten Dynpro zur Abgrenzung bei der Datenbeschaffung verwendet werden. Die ermittelten Daten sollen nach der Navigation auf dem zweiten Dynpro angezeigt werden.

Vorlage:

SAPMBC400_UDS_B

Lösung:

SAPMBC400_UDS_C

Aufgabe 1:

Kopieren Sie Ihr Programm SAPMZBC400_##_B oder die Vorlage SAPMBC400_UDS_B auf den Namen **SAPMZBC400_##_C**. Stellen Sie unbedingt sicher, dass die Dynpros und die Includes mitkopiert werden.

1. Kopieren Sie das Programm mit allen Dynpros und Includes. Wählen Sie passende Namen für die neuen Includes (Namensvorschlag: **MZBC400_##_CI01** bzw. **MZBC400_##_CI02**).

Aufgabe 2:

Legen Sie in Ihrem Programm ein globales, strukturiertes Datenobjekt an, das als Schnittstelle zu den Dynpros dienen kann, dessen Komponenten also die gleichen Namen haben wie die Dynprofelder.

1. Editieren Sie das TOP-Include Ihres Programms. Verwenden Sie dort die TABLES-Anweisung, um ein strukturiertes Datenobjekt anzulegen, das auf dem Dictionary-Strukturtyp BC400_S_DYNCONN basiert.

Fortsetzung auf der nächsten Seite

Aufgabe 3:

Stellen Sie sicher, dass vor der Navigation auf das zweite Dynpro die Details zur Flugverbindung von der Datenbank gelesen und die Felder der TABLES-Struktur mit diesen gefüllt werden. Verwenden Sie zur Datenbeschaffung den Funktionsbaustein BC400_DDS_CONNECTION_GET oder den von Ihnen selbst entwickelten Funktionsbaustein.

1. Editieren Sie das PAI-Modul, in dem der Funktionscode des ersten Dynpros ausgewertet wird. Implementieren Sie den Aufruf des Funktionsbausteins vor der Navigation. Legen Sie dazu im TOP-Include ein globales strukturiertes Datenobjekt an (Namensvorschlag: **gs_connection**), das Sie als Aktualparameter für den EXPORT-Parameter des Funktionsbausteins verwenden. Übergeben Sie die Benutzereingaben aus der TABLES-Struktur an die IMPORT-Parameter des Funktionsbausteins.

   Hinweis: Obwohl es möglich ist, direkt die Komponenten der TABLES-Struktur als Aktualparameter zu verwenden, sollten die Benutzereingaben zunächst in das globale Datenobjekt übertragen und von dort an den Funktionsbaustein übergeben werden. Man erhält so eine saubere Trennung zwischen der Schnittstelle zur Benutzeroberfläche (TABLES-Struktur) und der Schnittstelle zum Funktionsbaustein (globales Datenobjekt).

2. Weshalb kann das strukturierte Datenobjekt nicht als lokales Datenobjekt im PBO-Modul angelegt werden?

   _____

   _____

   _____

   _____

3. Behandeln Sie Ausnahmen des Funktionsbausteins. Reagieren Sie gegebenenfalls mit Informationsnachricht 042 aus Nachrichtenklasse BC400. Verlassen Sie das Eingabedynpro nur, falls keine Ausnahme aufgetreten ist.
4. Falls der Funktionsbaustein ohne Fehler durchlaufen wurde, füllen Sie die Felder der TABLES-Struktur mit dem Ergebnis und initiieren die Navigation auf das zweite Dynpro an.
5. Aktivieren Sie Ihr Programm, legen Sie eine Dialogtransaktion an und testen Sie die Anwendung.

Lösung 22: Dynpro – Datentransport

Aufgabe 1:

Kopieren Sie Ihr Programm SAPMZBC400_##_B oder die Vorlage SAPMBC400_UDS_B auf den Namen **SAPMZBC400_##_C**. Stellen Sie unbedingt sicher, dass die Dynpros und die Includes mitkopiert werden.

1. Kopieren Sie das Programm mit allen Dynpros und Includes. Wählen Sie passende Namen für die neuen Includes (Namensvorschlag: **MZBC400_##_CI01** bzw. **MZBC400_##_CI02**).

   a) Führen Sie diesen Schritt wie gewohnt durch.

Aufgabe 2:

Legen Sie in Ihrem Programm ein globales, strukturiertes Datenobjekt an, das als Schnittstelle zu den Dynpros dienen kann, dessen Komponenten also die gleichen Namen haben wie die Dynprofelder.

1. Editieren Sie das TOP-Include Ihres Programms. Verwenden Sie dort die TABLES-Anweisung, um ein strukturiertes Datenobjekt anzulegen, das auf dem Dictionary-Strukturtyp BC400_S_DYNCONN basiert.

   a) Siehe Quelltextauszug der Musterlösung.

Aufgabe 3:

Stellen Sie sicher, dass vor der Navigation auf das zweite Dynpro die Details zur Flugverbindung von der Datenbank gelesen und die Felder der TABLES-Struktur mit diesen gefüllt werden. Verwenden Sie zur Datenbeschaffung den Funktionsbaustein BC400_DDS_CONNECTION_GET oder den von Ihnen selbst entwickelten Funktionsbaustein.

1. Editieren Sie das PAI-Modul, in dem der Funktionscode des ersten Dynpros ausgewertet wird. Implementieren Sie den Aufruf des Funktionsbausteins vor der Navigation. Legen Sie dazu im TOP-Include ein globales strukturiertes Datenobjekt an (Namensvorschlag: **gs_connection**), das Sie

Fortsetzung auf der nächsten Seite

als Aktualparameter für den EXPORT-Parameter des Funktionsbausteins verwenden. Übergeben Sie die Benutzereingaben aus der TABLES-Struktur an die IMPORT-Parameter des Funktionsbausteins.

Hinweis: Obwohl es möglich ist, direkt die Komponenten der TABLES-Struktur als Aktualparameter zu verwenden, sollten die Benutzereingaben zunächst in das globale Datenobjekt übertragen und von dort an den Funktionsbaustein übergeben werden. Man erhält so eine saubere Trennung zwischen der Schnittstelle zur Benutzeroberfläche (TABLES-Struktur) und der Schnittstelle zum Funktionsbaustein (globales Datenobjekt).

a) Siehe Quelltextauszug aus der Musterlösung.

2. Weshalb kann das strukturierte Datenobjekt nicht als lokales Datenobjekt im PBO-Modul angelegt werden?

   Antwort: Lokale Datenobjekte gibt es nur in Unterprogrammen, Funktionsbausteinen und Methoden. Mit DATA-Anweisungen in Dialogmodulen werden immer programmglobale Datenobjekte angelegt. Um Verwirrungen zu vermeiden, sollen deshalb in Dialogmodulen grundsätzlich keine Datenobjekte deklariert werden. Globale Datenobjekte gehören ins TOP-Include!

3. Behandeln Sie Ausnahmen des Funktionsbausteins. Reagieren Sie gegebenenfalls mit Informationsnachricht 042 aus Nachrichtenklasse BC400. Verlassen Sie das Eingabedynpro nur, falls keine Ausnahme aufgetreten ist.

   a) Siehe Quelltextauszug aus der Musterlösung.

4. Falls der Funktionsbaustein ohne Fehler durchlaufen wurde, füllen Sie die Felder der TABLES-Struktur mit dem Ergebnis und initiieren die Navigation auf das zweite Dynpro an.

   a) Siehe Quelltextauszug aus der Musterlösung.

5. Aktivieren Sie Ihr Programm, legen Sie eine Dialogtransaktion an und testen Sie die Anwendung.

   a) Führen Sie diesen Schritt wie in vorherigen Übungen durch.

Ergebnis

Quelltextauszug aus der Musterlösung:

```
*&---------------------------------------------------------------------*
*& Include MBC400_UDS_CTOP         Modulpool            SAPMBC400_UDS_C
*&---------------------------------------------------------------------*
```

Fortsetzung auf der nächsten Seite


```
PROGRAM  sapmbc400_uds_c.

DATA ok_code TYPE syucomm.

TABLES bc400_s_dynconn.

DATA gs_connection TYPE bc400_s_connection.

*---------------------------------------------------------------------*
***INCLUDE MBC400_UDS_CI01 .
*---------------------------------------------------------------------*
*&---------------------------------------------------------------------*
*&      Module  USER_COMMAND_0100  INPUT
*&---------------------------------------------------------------------*
MODULE user_command_0100 INPUT.

  CASE ok_code.

    WHEN 'GO'.

      MOVE-CORRESPONDING bc400_s_dynconn TO gs_connection.

      CALL FUNCTION 'BC400_DDS_CONNECTION_GET'
        EXPORTING
          iv_carrid     = gs_connection-carrid
          iv_connid     = gs_connection-connid
        IMPORTING
          es_connection = gs_connection
        EXCEPTIONS
          no_data       = 1
          OTHERS        = 2.

      IF sy-subrc <> 0.
        MESSAGE i042(bc400) WITH gs_connection-carrid
                                 gs_connection-connid. "flight not found
      ELSE.

        MOVE-CORRESPONDING gs_connection TO bc400_s_dynconn.

        SET SCREEN 200.

      ENDIF.

  ENDCASE.
ENDMODULE.                 " USER_COMMAND_0100  INPUT
```

Zusammenfassung der Lektion

Nun können Sie

- Eigenschaften und Stärken des Dynpros aufzählen
- einfache Dynpros mit Ein-/Ausgabefeldern sowie Drucktasten realisieren und zu einer Dialoganwendung verknüpfen
- die programminterne Verarbeitung bei Dynpro-Aufrufen erläutern und implementieren

Lektion: ABAP Web Dynpro

Überblick über die Lektion

In dieser Lektion erhalten Sie einen Überblick über die Eigenschaften und Nutzungsszenarien sowie die Programmier- und Laufzeitarchitektur des ABAP Web Dynpros. Außerdem lernen Sie die Gestaltung und Programmierung einer einfachen ABAP-Web-Dynpro-Anwendung mit Ein-/Ausgabefeldern und Drucktasten.

Lernziele der Lektion

Am Ende dieser Lektion können Sie

- die Eigenschaften und Nutzungsszenarien des ABAP Web Dynpros aufzählen
- die Programmier- und Laufzeitarchitektur des ABAP Web Dynpros grob erläutern
- einfache Web-Dynpro-Anwendungen mit Ein-/Ausgabefeldern und Drucktasten realisieren

Unternehmensszenario

Sie sollen die Eigenschaften und Nutzungsszenarien sowie das Programmierkonzept und die Laufzeitarchitektur von ABAP Web Dynpro erläutern.

ABAP Web Dynpro: Eigenschaften und Programmiermodell

ABAP-Web-Dynpro-Eigenschaften

- client- und protokollunabhängiges Programmiermodell
- Trennung von Layout, Programmfluss und Business-Logik
- Grafische Unterstützung bei Layout-Design und Programmflussgestaltung
- Entwicklungstools eingebettet in Object Navigator
- wiederverwendbare Komponenten
- beliebige Backend-Systeme durch Nutzung von Web Services
- Unterstützung des Accessibility-Standards 508

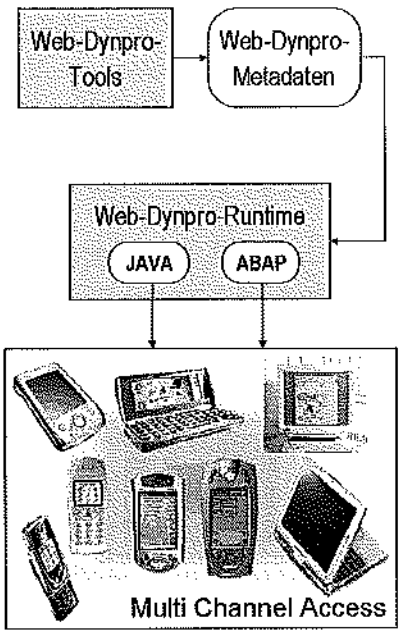


Abbildung 195: ABAP-Web-Dynpro-Eigenschaften

Zu *SAP NetWeaver 7.0* hat SAP mit ABAP Web Dynpro ein neues Programmiermodell realisiert, das unabhängig ist von den eingesetzten Clients (zur Bedienung von Web-Dynpro-Anwendungen) und dem zugehörigen Protokoll. Dies wird durch den Einsatz von Metadaten erreicht.

ABAP Web Dynpro wird die Standard-UI-Technologie von SAP für die Entwicklung von Web-Anwendungen im ABAP-Umfeld.

Neben einer eigenen Laufzeitumgebung gehört zum ABAP Web Dynpro vor allem eine grafische Entwicklungsumgebung (*Web Dynpro Explorer*), deren Werkzeuge vollständig in die ABAP-Entwicklungsumgebung (*Object Navigator*) integriert sind. Die Entwicklung einer Web-Dynpro-Anwendung wird über den *SAPGUI* im SAP-System durchgeführt. Ihr Aufruf geschieht über die entsprechende URL beispielsweise vom Internet-Browser oder vom *SAP Portal* aus.

Durch die Wiederverwendbarkeit von Komponenten sowie die strikte Trennung von Layout, Programmfluss und Business-Logik wird die Wartung und Weiterentwicklung von ABAP-Web-Dynpro-Anwendungen erleichtert. Ferner können Web-Dynpro-Anwendungen für Sehbehinderte bedienbar gemacht werden, da Web Dynpro durch Tastatureingabe und Screenreading den Accessibility-Standard 508 (Barrierefreiheit) unterstützt.

ABAP Web Dynpro unterstützt die Nutzung von Web-Services, wodurch aus einer Anwendung heraus in standardisierter und komfortabler Weise auf Business-Funktionalitäten unterschiedlichster Systeme zugegriffen werden kann.

Entkopplung von Business-Logik, Layout und Programmfluss

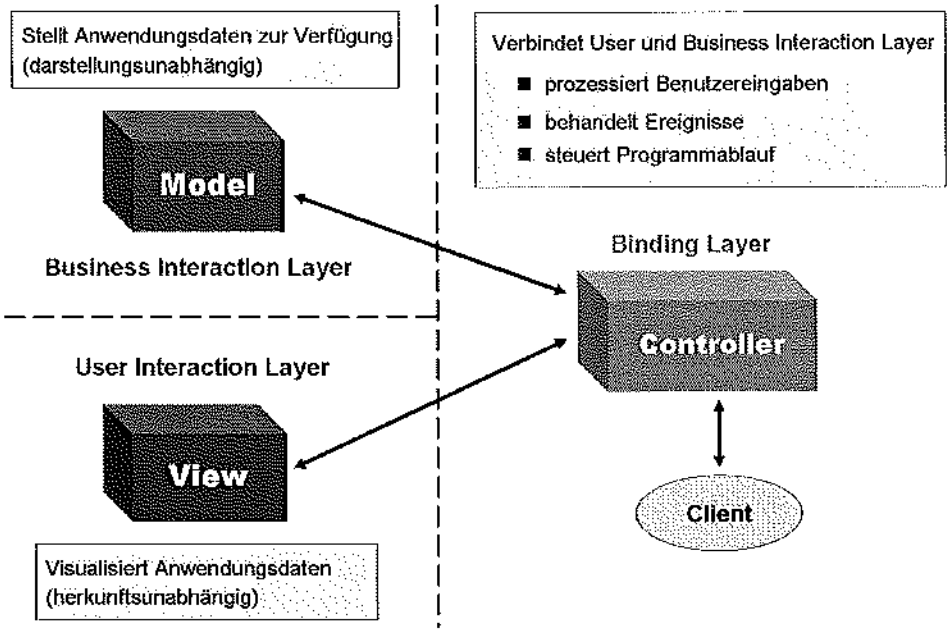


Abbildung 196: Das Model-View-Controller-Programmiermodell

ABAP-Web-Dynpro-Anwendungen sind nach dem Model-View-Controller-Programmiermodell (MVC-Programmiermodell) strukturiert.

Während das **Model** in der *Business Interaction Layer* für den darstellungsunabhängigen Datenzugriff auf das Backend-System zuständig ist, übernimmt die **View** in der *User Interaction Layer* die herkunftsunabhängige Darstellung der Daten (Layout). Der **Controller** in der *Binding Layer* liegt zwischen View und Model. Er bereitet die Daten des Models auf, die in der View angezeigt werden sollen, verarbeitet die Benutzereingaben und gibt sie an das Model zurück. Außerdem steuert er den Ablauf des Programms.

In ABAP Web Dynpro kann das Model aus Anwendungsklassen bestehen, die die Datenzugriffe kapseln. Es können aber auch vorhandene BAPIs, Funktionsbausteine oder Web-Services verwendet werden.

View und Controller werden mittels der grafischen Werkzeuge realisiert.

Elemente des ABAP Web Dynpros

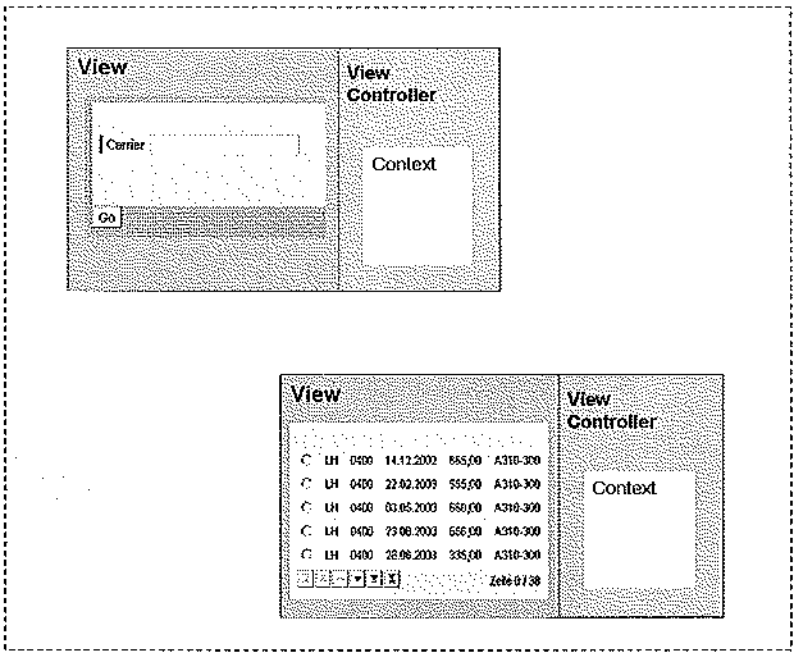


Abbildung 197: ABAP Web Dynpro Views

Das Layout einer **View** besteht aus einer Menge von **UI-Elementen** (Bildschirmelemente wie Eingabefeld, Tabellen und Drucktasten), die über ein grafisches Werkzeug zusammengestellt werden. Zur Laufzeit werden Views nacheinander und nicht parallel auf dem Bildschirm angezeigt. Sie entsprechen also den klassischen Dynpros ohne Ablauflogik.

Wie beim klassischen ABAP-Dynpro müssen Eingabeprüfungen und Wertehilfen (F4) nicht händisch implementiert werden, sondern können durch Bezug auf das ABAP Dictionary als UI-Services genutzt werden.

Durch die Container-Technik lassen sich Views in andere Views integrieren, wodurch eine Modularisierung ermöglicht wird.

Jede View besitzt einen ihr zugeordneten View-Controller. Technisch handelt es sich beim View-Controller um eine ABAP-Klasse. Das Coding des View-Controllers wird vom Entwickler aber nur zum Teil direkt implementiert, ein großer Teil wird gemäß Design-Angaben des Entwicklers automatisch generiert. Der **Context** des View-Controllers dient als Daten-Container und enthält die in der View anzuzeigenden Daten (entspricht der TABLES-Struktur beim klassischen Dynpro).

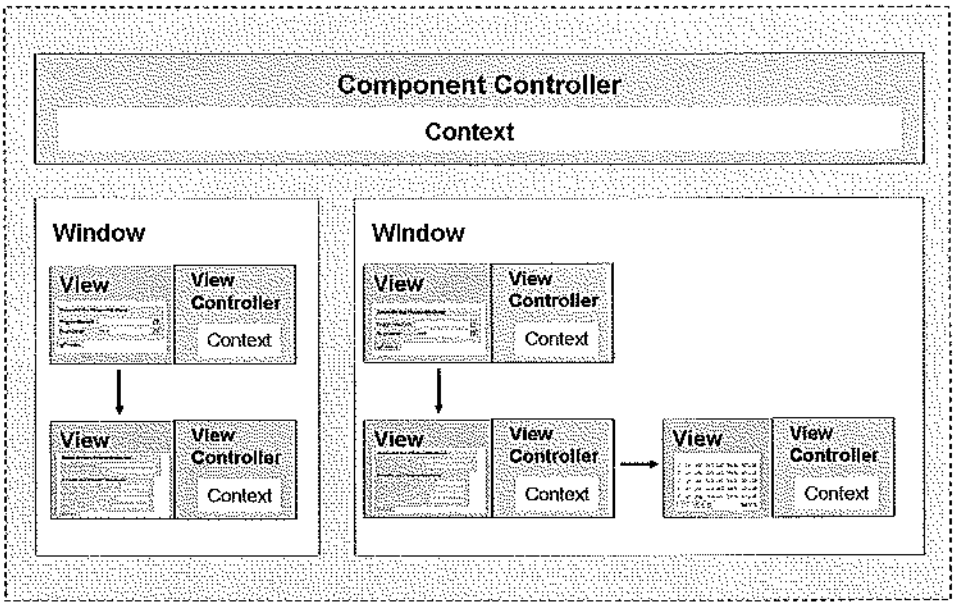


Abbildung 198: ABAP Web Dynpro Component

Windows dienen dazu, mehrere Views zu bündeln und die Navigationsmöglichkeiten zwischen ihnen festzulegen (View-Abfolgen). Eine **Component** enthält ein oder mehrere Windows und besitzt einen eigenen Controller mit Context, in dem Daten abgelegt werden, die auf mehreren Views der Component angezeigt werden sollen.

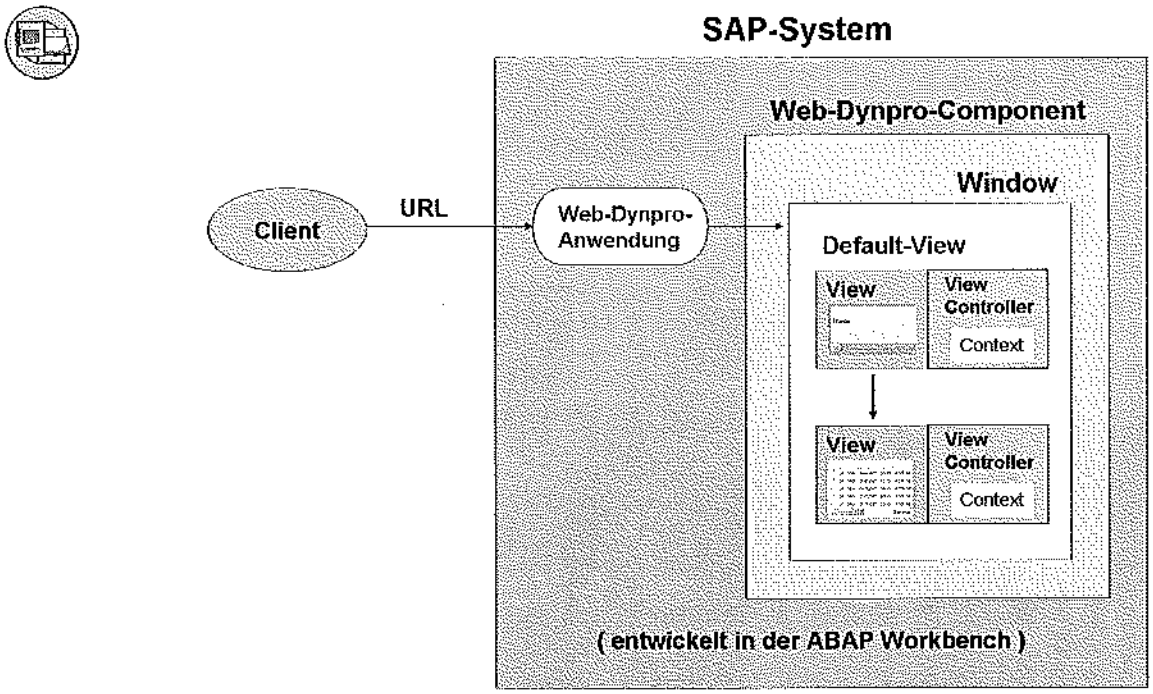


Abbildung 199: Entwicklungsumgebung und Laufzeitarchitektur

Eine **Web-Dynpro-Anwendung** verweist auf ein Window mit Default-View (Start-View), die bei Aufruf der Anwendung angezeigt wird.

Eine Component mit zugehörigen Windows ist als abgeschlossene Einheit verwendbar in verschiedenen Web-Dynpro-Anwendungen. Dadurch kann der Implementierungs- und Wartungsaufwand verringert werden.

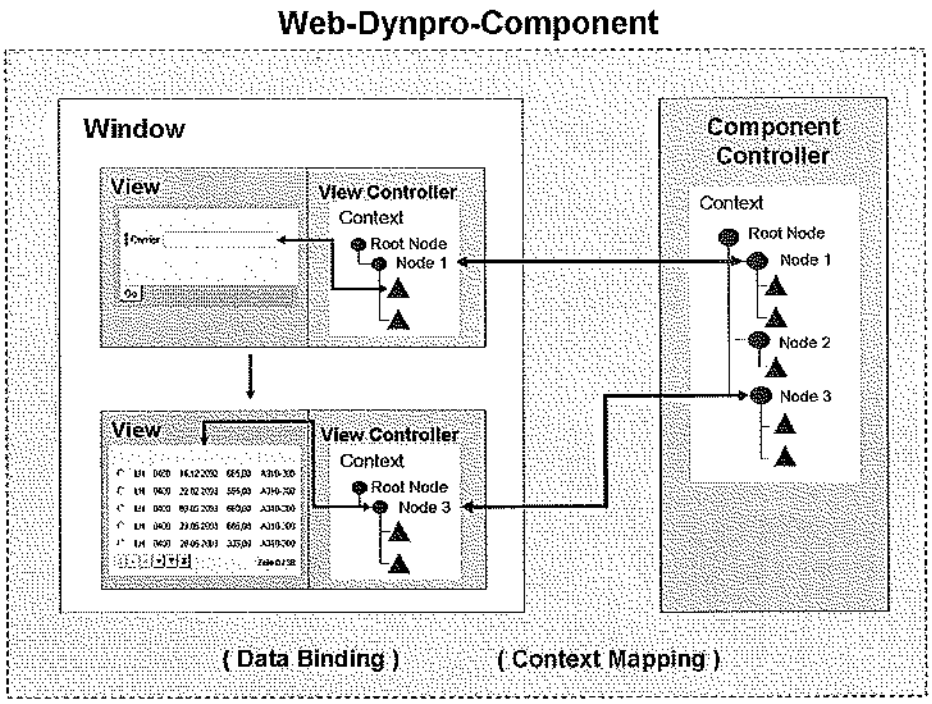


Abbildung 200: ABAP Web Dynpro Context

Der **Context** eines jeden View-Controllers enthält Daten, die auf der View angezeigt werden sollen (Daten-Container). Der Datentransport zwischen View Context und den UI-Elementen der View erfolgt durch entsprechende Zuordnung (Data Binding).

Auch eine Component hat einen **Context**. Darin werden Daten abgelegt, die in verschiedenen Views der Component angezeigt werden sollen. In solchen Fällen werden meist in den entsprechenden View-Contexts Referenzen auf die Component-Daten realisiert (Context Mapping).

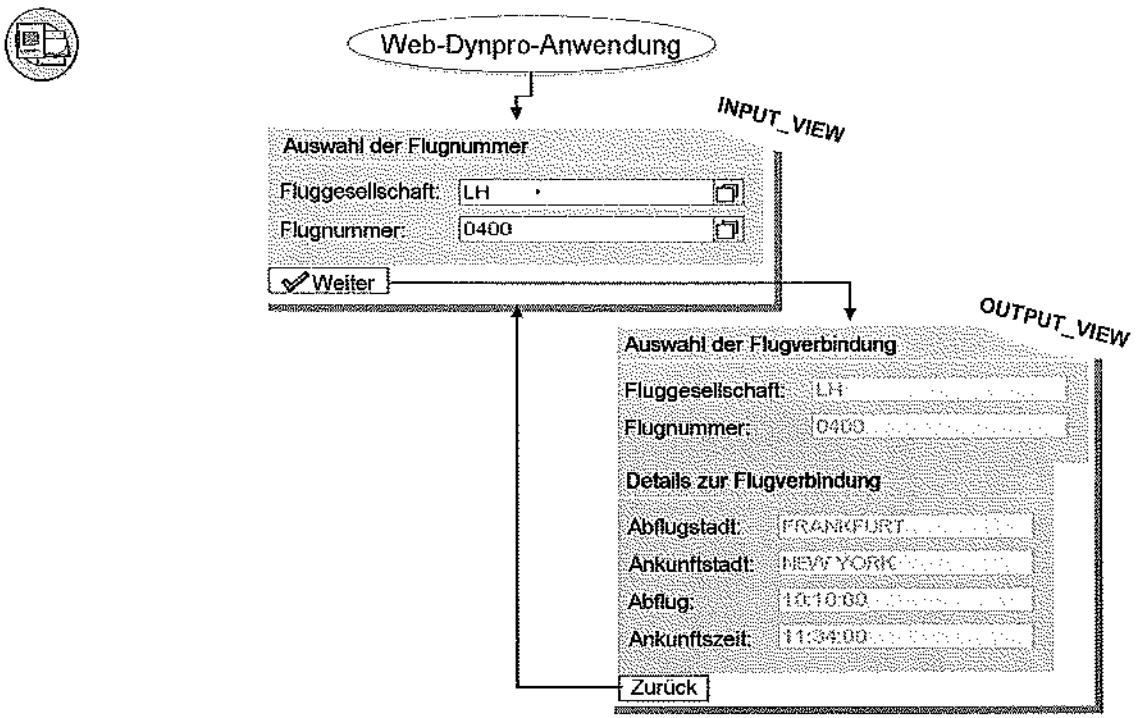


Abbildung 201: Anwendungsbeispiel

In den folgenden Abschnitten wird in mehreren Stufen eine ABAP-Web-Dynpro-Anwendung entwickelt, mit der die Daten zu einer einzelnen Flugverbindung angezeigt werden können.

Das Programm besteht aus einer Web Dynpro Component mit zwei Views. Auf der ersten View wählt der Benutzer die Flugverbindung aus, auf der zweiten View werden die Details zur Flugverbindung angezeigt.

Die beiden Views werden in ein Web Dynpro Window der Component eingebettet und dort so über Navigationslinks miteinander verbunden, dass der Anwender durch Betätigen der Drucktasten zwischen den Views hin- und hernavigieren kann.

Die im Window definierte View-Folge wird über eine Web-Dynpro-Anwendung gestartet.

Web Dynpro, Drucktasten und Navigation

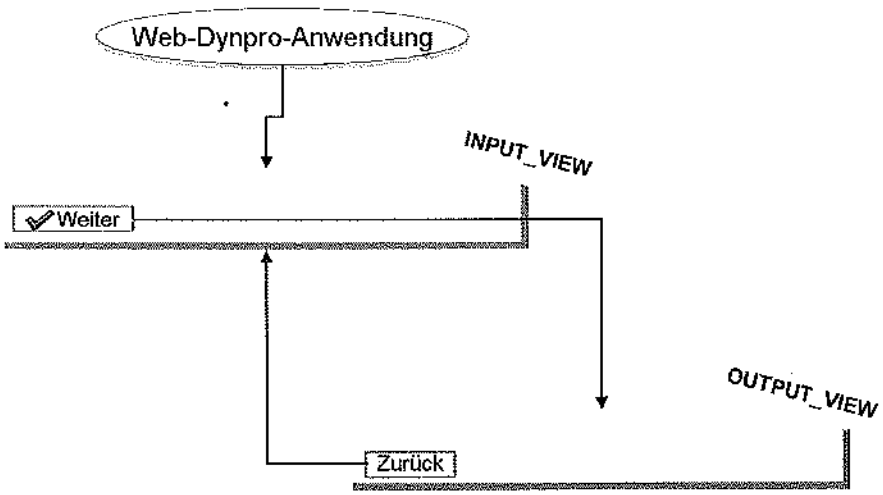


Abbildung 202: Realisierungsstufe 1: Web Dynpro Views mit Drucktasten und Navigation

In der ersten Stufe unseres Beispiels soll eine Web Dynpro Component mit zwei Views angelegt werden. Auf den beiden Views soll sich jeweils eine Drucktaste befinden. Das Programm soll so implementiert werden, dass bei Betätigen der Drucktasten zur jeweils anderen View navigiert wird. Zum Start der View-Folge wird eine Web-Dynpro-Anwendung angelegt.

Dieser Abschnitt befasst sich mit Folgendem:

- Anlegen einer Web-Dynpro-Component
- Erstellen von Web Dynpro Views
- Einbetten von Views in Web Dynpro Windows
- Einrichten der Navigation
- Anlegen von Drucktasten und Auswerten der Benutzeraktion
- Anlegen von Web-Dynpro-Anwendungen

Anlegen einer Web Dynpro Component

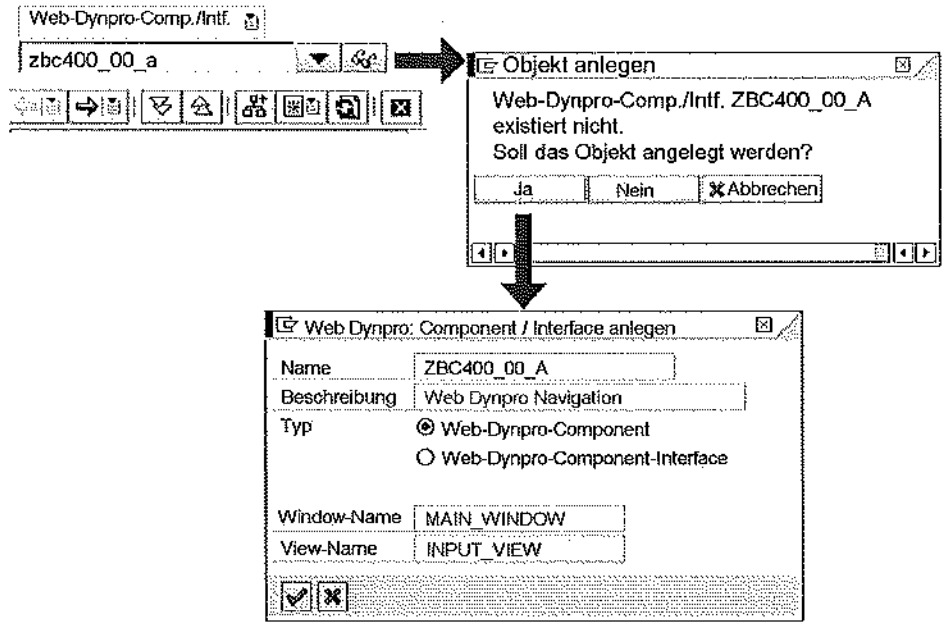


Abbildung 203: Web Dynpro Component anlegen

Das Anlegen einer Web-Dynpro-Component erfolgt ganz analog zum Anlegen eines neuen ABAP Programs über das Kontextmenü im Navigationsbereich des *Object Navigator* (siehe obige Grafik)

Beim Anlegen einer neuen Web-Dynpro-Component lassen sich automatisch ein Web Dynpro Window und eine Web Dynpro View anlegen. Später können in der Component auch noch weitere Windows und Views angelegt werden.

Erstellen von Web Dynpro Views und Einbetten ins Window

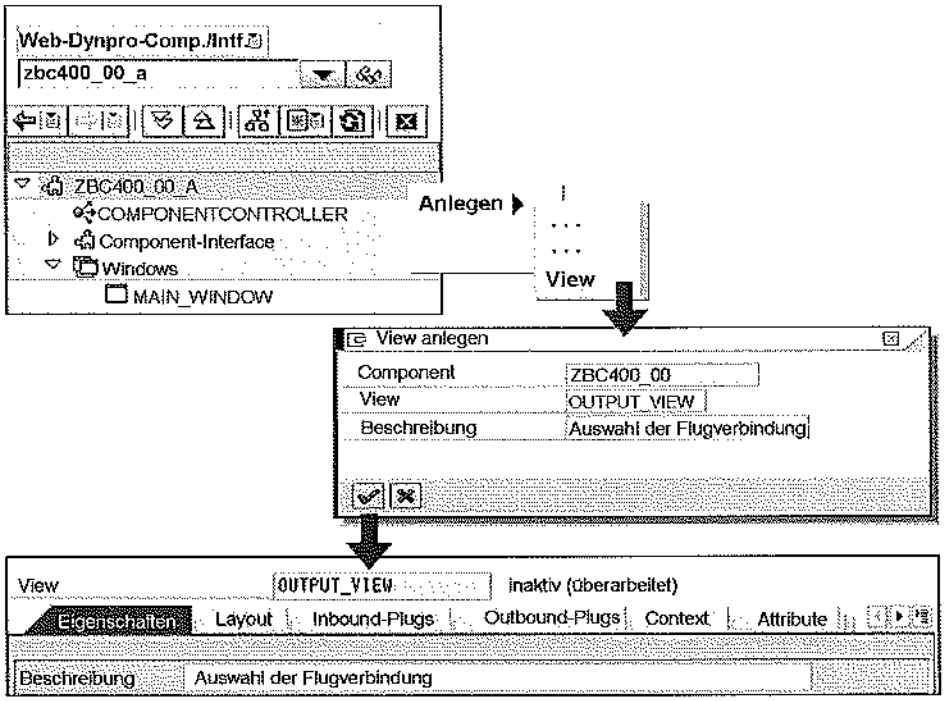


Abbildung 204: Web Dynpro View anlegen

Zum Anlegen einer Web Dynpro View verwenden Sie ebenfalls das Kontextmenü im Navigationsbereich der *Object Navigator*. Vergeben Sie einen Namen für die View und einen beschreibenden Text.

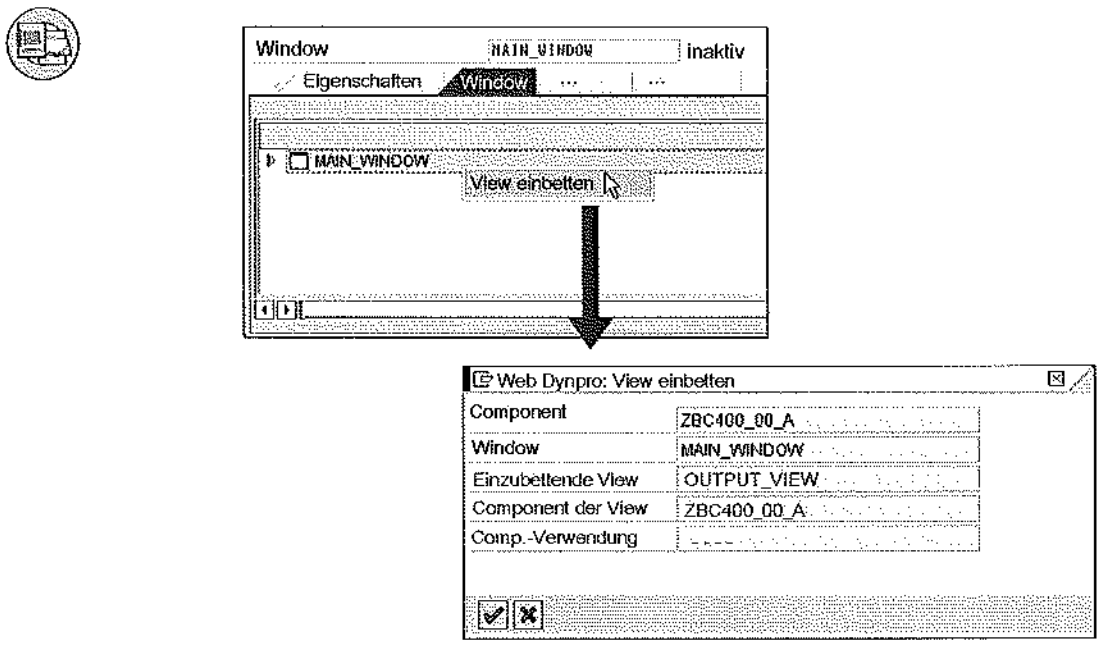


Abbildung 205: Web Dynpro View einbetten

Um eine Web Dynpro View in ein Window einzubetten, öffnen Sie im *Object Navigator* das Window und lassen sich die Window-Struktur anzeigen. Für das Einbetten selbst haben Sie zwei Möglichkeiten:

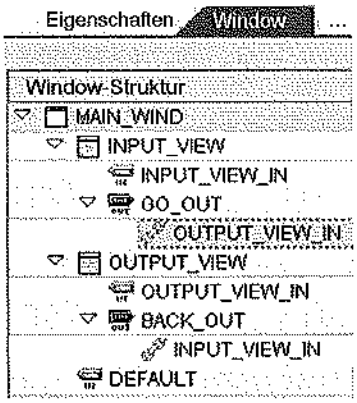
Über das Kontextmenü

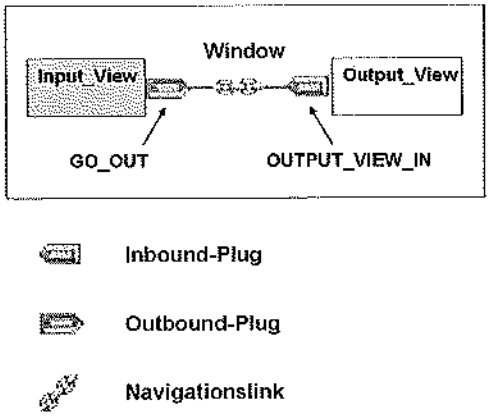
In der Window-Struktur finden Sie im Kontextmenü zum Namen des Window den Menüeintrag *View einbetten* (siehe obige Grafik). Den einzubettenden View wählen Sie auf dem folgenden Eingabebild über die Wertehilfe (*F4*-Hilfe) aus.

Über Drag & Drop aus dem Navigationsbereich

Wenn Sie im Navigationsbereich den Knoten *Views* öffnen, können Sie die einzubettenden View direkt per Drag & Drop in die Window-Struktur ziehen und auf dem Window fallen lassen.

Einrichten der Navigation zwischen Web Dynpro Views





- Um die Navigation zwischen zwei Views zu definieren, müssen Sie für jede View Aus- und Einstiegspunkte in Form von Outbound- und Inbound-Plugs anlegen.
- Erst dann können Sie mit Navigationslinks den Navigationsablauf festlegen.

Abbildung 206: Navigation zwischen Web Dynpro Views

Um innerhalb eines Windows von einer View auf eine andere navigieren zu können, muss zwischen den Views ein **Navigationslink** angelegt werden. Ein Navigationslink führt immer von einem **Outbound-Plug** einer View zu einem **Inbound-Plug** einer anderen View.

Outbound-Plugs und Inbound-Plugs einer View legen die möglichen „Eingänge" und „Ausgänge" fest. Eine View kann prinzipiell mehrere Inbound-Plugs und Outbound-Plugs besitzen. Der Navigationslink in obiger Grafik bewirkt, dass nach dem Verlassen des INPUT_VIEW über seinen Outbound-Plug *GO_OUT* zum OUTPUT_VIEW navigiert wird und zwar wird dieser über seinen Inbound-Plug *OUTPUT_VIEW_IN* gestartet.

Durch das Anlegen der Navigationslinks wird die Abfolge der Views innerhalb des jeweiligen Windows festgelegt. Das geschieht rein anhand der vorhandenen Plugs und unabhängig von der Implementierung der Views.

Die Plugs legen Sie ganz einfach an, indem Sie die entsprechende View bearbeiten und auf dem Reiter *Inbound-Plugs* oder *Outbound-Plugs* einen Namen und einen beschreibenden Text für den Plug eintragen. Für fortgeschrittene Anwendungen ist es außerdem möglich, die Outbound-Plugs mit Parametern zu versehen.

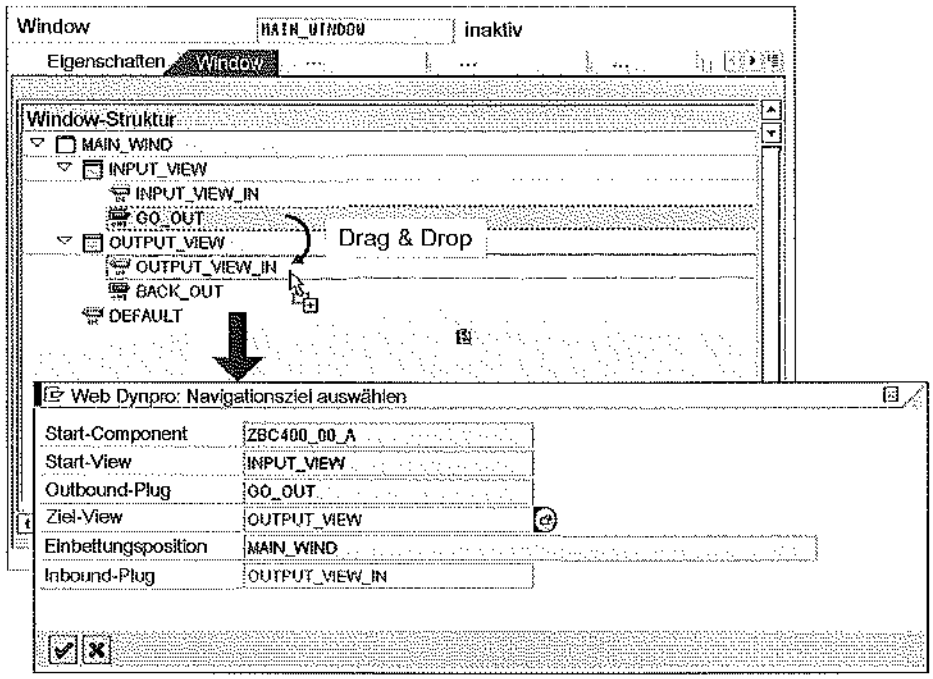


Abbildung 207: Anlegen eines Navigationslinks

Nach dem Anlegen der Plugs sind In der Struktur eines Windows neben den eingebetteten Views auch deren Plugs sichtbar. Durch einfaches Drag & Drop lassen sich zwischen diesen Plugs Navigationslinks anlegen (siehe obige Grafik).

Anlegen von Drucktasten und Anstoßen der Navigation

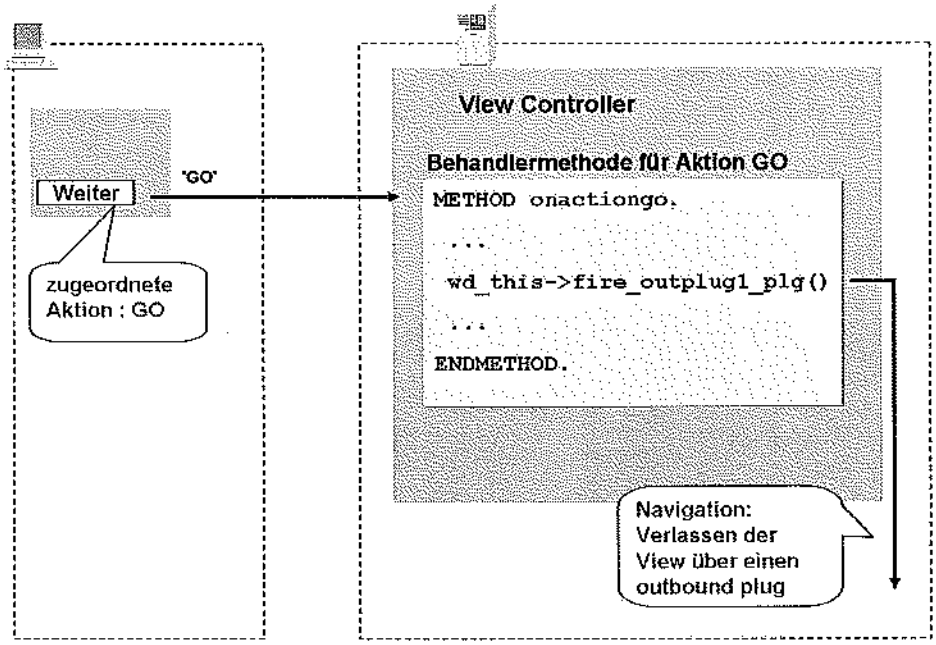


Abbildung 208: Ablauf bei Betätigung einer Drucktaste

In unserem Beispielprogramm soll die Navigation jeweils durch das Betätigen einer Drucktaste angestoßen werden. Der Ablauf ist dabei wie folgt:

- Betätigt der Anwender die Drucktaste, wird ein Aktionscode vom Browser an den Applikationsserver übertragen. Dieser Aktionscode wurde vom Entwickler für die jeweilige View definiert und der Drucktaste zugeordnet.
- Zu jedem Aktionscode gehört eine spezielle Methode des View-Controllers (Aktionsbehandlermethode), die jetzt als Reaktion auf die Aktion ausgeführt wird.
- In unserem Fall soll nur die Navigation zur nächsten View angestoßen werden. Dazu wird die View über den entsprechenden Outbound-Plug verlassen. Man sagt, der Outbound-Plug wird „gefeuert".

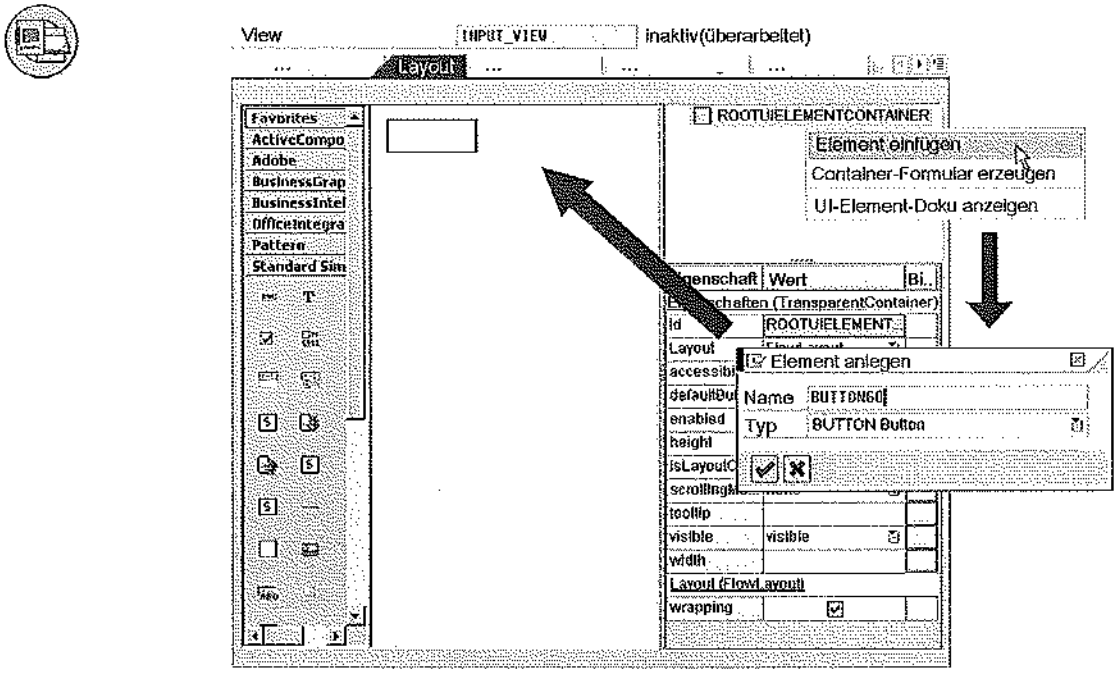


Abbildung 209: Drucktaste anlegen

Eine Drucktaste wird auf der Layout-Sicht der View angelegt. Es gibt dazu mehrere Möglichkeiten:

Über das Kontextmenü in der Elementhierarchie

Öffnen Sie in der Elementhierarchie rechts oben das Kontextmenü für den Knoten *ROOTUIELEMENTCONTAINER* und wählen Sie *Element einfügen*. Geben Sie Namen und Typ des UI-Elements an. (Siehe obige Grafik).

Über die Werkzeugleiste

Wählen Sie in der Werkzeugleiste am linken Rand das Werkzeug *Button* und platzieren Sie die Drucktaste in der Layout-Vorschau in der Mitte.

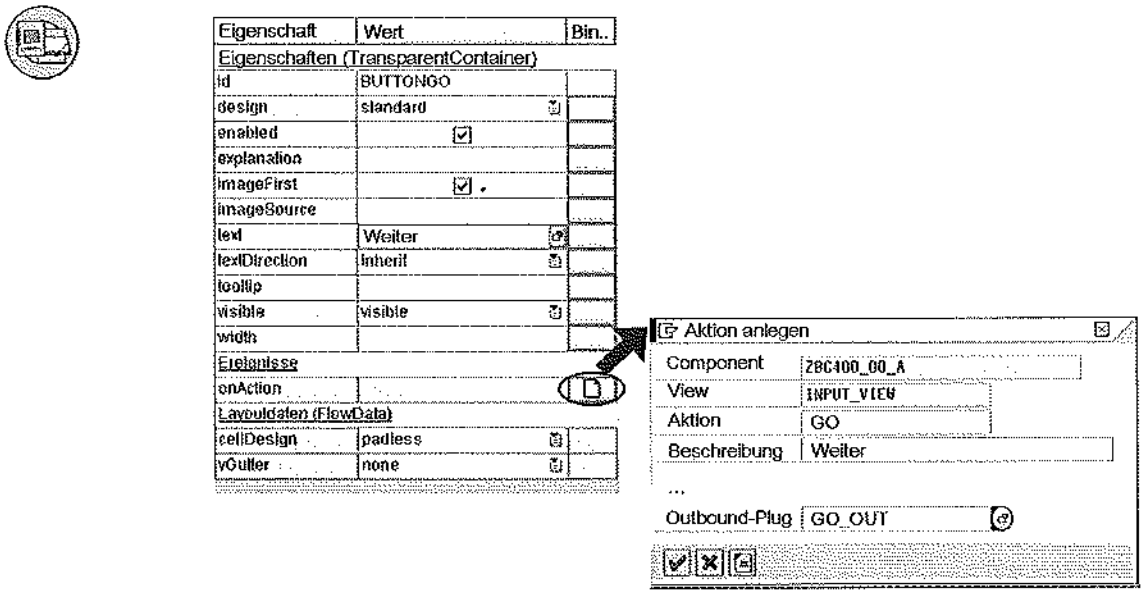


Abbildung 210: Attribute von Drucktasten pflegen

Im Attribute-Fenster am rechten unteren Rand werden jeweils für ein Element des Layouts die Attribute angezeigt. Für eine Drucktaste kann hier unter anderem ein Text eingetragen oder auch eine Ikone zugeordnet werden.

Um der Drucktaste eine Aktion zuzuordnen, gehen Sie zum Attribut *onAction* und klicken Sie auf das Anlegen-Symbol in der dritten Spalte (siehe obige Grafik). Im folgenden Dialog ordnen Sie der Drucktaste eine Aktion zu. Wenn die Aktion nicht schon vorher für die View definiert wurde (Reiter *Aktionen*) kann sie hier auch direkt neu angelegt werden.

Geben Sie zusätzlich im Feld *Outbound-Plug* den Outbound-Plug an, der als Reaktion auf diese Drucktaste gefeuert werden soll. Dann wird die Aktionsbehandlermethode nicht leer angelegt, sondern der Quelltext zum Feuern des Outbound-Plugs automatisch mitgeneriert.

Anlegen einer Web -Dynpro-Anwendung

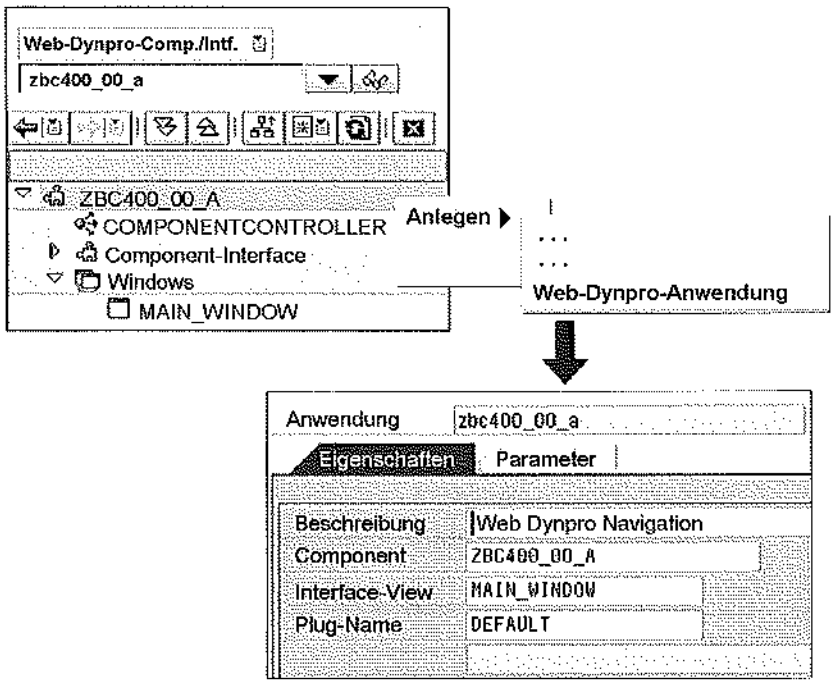


Abbildung 211: Web-Dynpro-Anwendung anlegen

Der Einstiegspunkt in eine Web Dynpro View-Folge, also ein Web Dynpro Window, ist immer eine **Web-Dynpro-Anwendung**. Sie zeigt auf das Window bzw. die gleichnamige **Interface-View** und den dafür automatisch angelegten Inbound-Plug *DEFAULT*.

Eine Web-Dynpro-Anwendung wird über das Kontextmenü des *Object Navigator* angelegt (siehe obige Grafik). Im nachfolgenden Dialog wird sie mit der Component, dem Interface-View und dem Standard-Plug verknüpft.

Eine Web-Dynpro-Anwendung lässt sich über eine entsprechende Drucktaste direkt aus der Entwicklungsumgebung testen. Alternativ kann die zugeordnete URL (Internetadresse) direkt in die Adresszeile eines Browsers eingegeben werden.

Web Dynpro, Layout und Datentransport

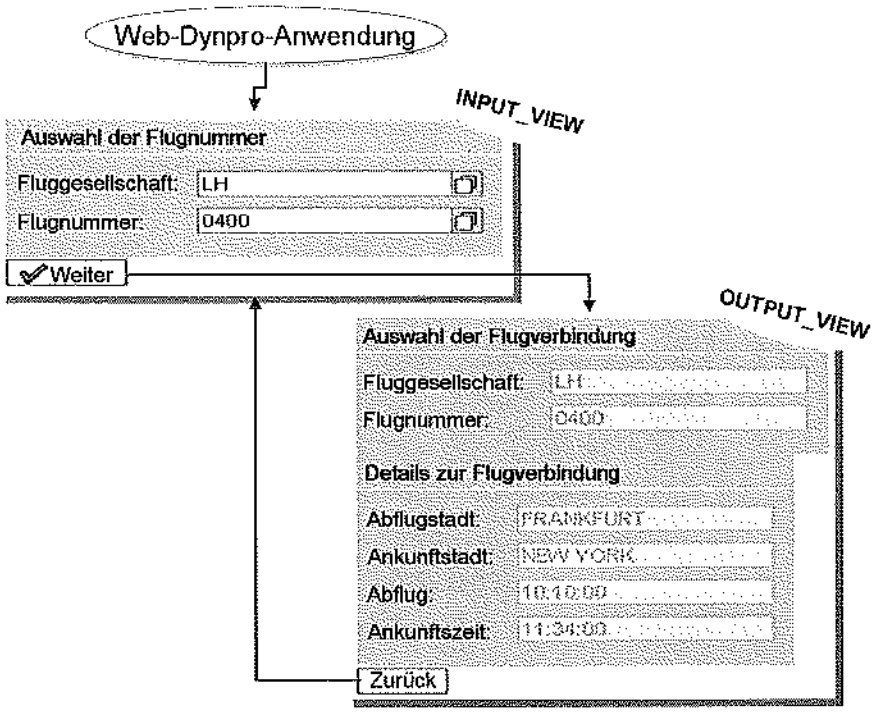


Abbildung 212: Realisierungsstufe 2: Ein-/Ausgabefelder anlegen und Daten anzeigen

In der zweiten Stufe unseres Beispiels sollen die Web Dynpro Views mit Ein-/Ausgabefeldern erweitert werden und die Datenbeschaffung und der Datentransport implementiert werden. Auf dem ersten View soll der Benutzer über Fluggesellschaftskürzel und Flugnummer eine Flugverbindung auswählen. Nach dem Betätigen der Drucktaste sollen diese Informationen im Programm weiterverarbeitet und die Detailinformationen zur Fluggesellschaft von der Datenbank gelesen werden. Nach der Navigation auf den zweiten View, sollen die Detailinformationen dort angezeigt werden.

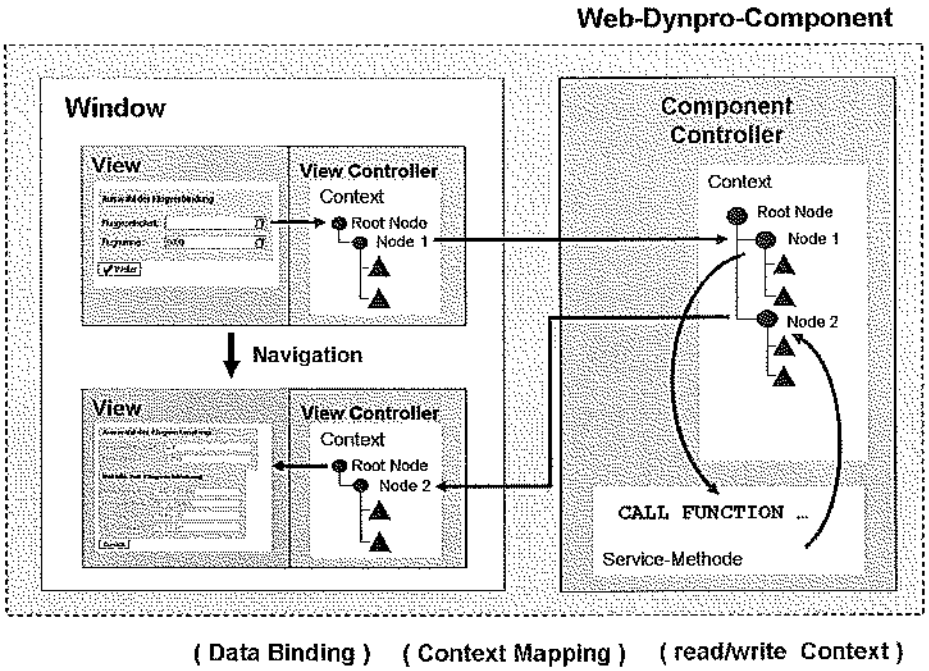


Abbildung 213: Datenbeschaffung und Datentransport

Um dem MVC-Programmiermodell zu entsprechen, soll jede Datenbeschaffung einer Web-Dynpro-Component zentral in einer Methode des Component-Controllers gekapselt werden. In unserem Fall wird dieser **Service-Aufruf** den Aufruf eines Funktionsbausteins enthalten. Die vom Funktionsbaustein benötigten Daten werden dabei dem Context des Component-Controllers entnommen. Entsprechend wird nach dem Funktionsbausteinaufruf das Ergebnis wieder im Context abgelegt.

Der Datentransport zwischen dem zentralen Component-Controller und den einzelnen View-Controllern erfolgt über das **Context Mapping**. Es handelt sich dabei um ein Referenzieren von einem Context auf einen anderen, der Datentransport funktioniert also automatisch in beide Richtungen.

Der Datentransport zwischen den Ein-/Ausgabefeldern einer View und dem Context dieser View erfolgt durch das **Data Binding**.

Dieser Abschnitt befasst sich mit Folgendem:

- Anlegen eines Web Dynpro Service-Aufruf
- Anlegen des Controller Context
- Einrichten des Context Mappings
- Anlegen von Ein-Ausgabefeldern
- Einrichten des Data Bindings
- Aufruf der Service-Methode zur Datenbeschaffung

Anlegen eines Web Dynpro Service-Aufrufs und Generieren des Controller Context

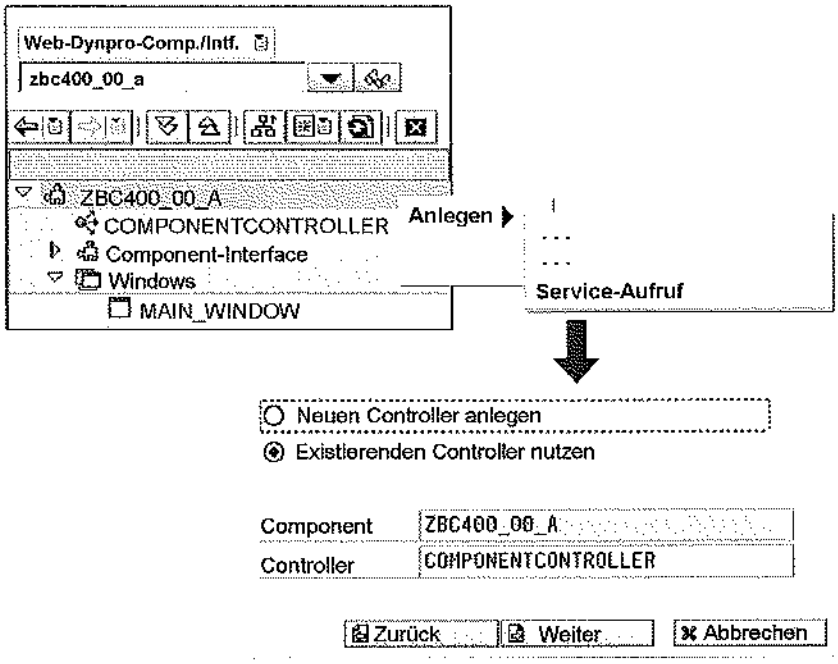


Abbildung 214: Service-Aufruf anlegen

Zum Anlegen eines Service-Aufrufs stellt die *ABAP Workbench* einen *Web Dynpro Wizard* zur Verfügung, der im Navigationsbereich des *Object Navigator* wie gewohnt über das Kontextmenü angesprochen werden kann. Nach dem Start des Wizards muss zunächst der Controller ausgewählt werden, in dem der Service-Aufruf angelegt werden soll. In unserem Fall ist das der existierende Component-Controller. Es besteht aber prinzipiell auch die Möglichkeit, hier einen zusätzlichen Controller anlegen zu lassen.

Anschließend benötigt der Wizard noch die Information, welcher Funktionsbaustein aufgerufen werden soll, wie die Service-Methode heißen soll und welche Parameter des Funktionsbausteins er beim Aufruf mit Aktualparametern versorgen soll.

Der Wizard generiert im ausgewählten Controller folgende Bestandteile:

- eine Service-Methode (Name beginnt mit „EXECUTE_)“
- Kontextknoten für alle Funktionsbaustein-Parameter, die beim Aufruf versorgt werden sollen
- den Aufruf des Funktionsbausteins in der Service-Methode
- den Quelltext zum Lesen der Daten aus dem Context
- den Quelltext zum Füllen des Context mit dem Ergebnis des Funktionsbaustein-Aufrufs

Einrichten des Context Mapping

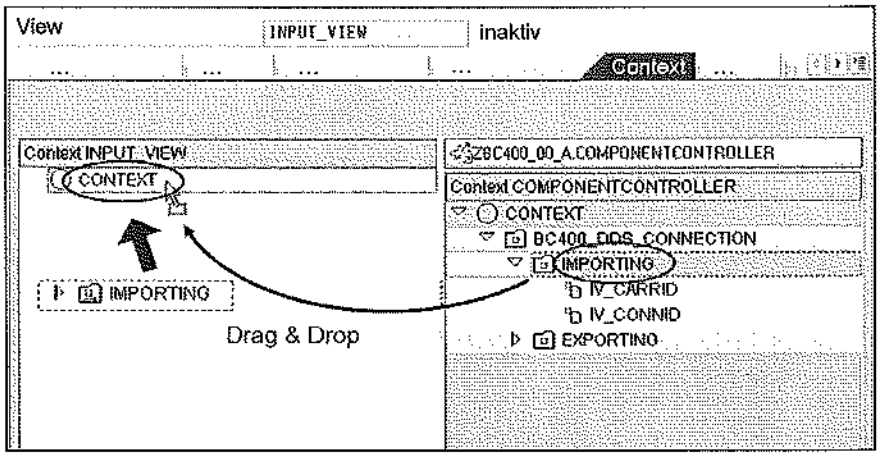


Abbildung 215: Context Mapping einrichten

Damit die Daten aus dem Context des Component-Controllers auch in den Views zur Verfügung stehen, müssen die entsprechenden Knoten in den Context des View-Controllers kopiert und auf die Knoten des Component-Controllers abgebildet (gemappt) werden. Das Context-Mapping ist auch erforderlich, damit Benutzereingaben auf einer View im Component-Controller zur Verfügung stehen.

Das Kopieren der Knoten kann mit dem Einrichten des Context Mapping gemeinsam vorgenommen werden. Öffnen Sie dazu die entsprechende View im Änderungsmodus und gehen Sie auf den Reiter *Context*. Links sehen Sie den Context des View Controllers (noch leer bis auf einen Wurzel-Knoten mit Namen *CONTEXT*) und rechts den generierten Context des Component-Controllers. Sie kopieren einen Knoten indem Sie ihn von rechts nach links ziehen (Drag & Drop) und auf dem Wurzelknoten des View Controller Context fallen lassen. Im folgenden Dialog bestätigen Sie, dass der Knoten sowohl kopiert als auch gemappt werden soll.

Anlegen von Ein-/Ausgabefeldern und einrichten des Data Binding

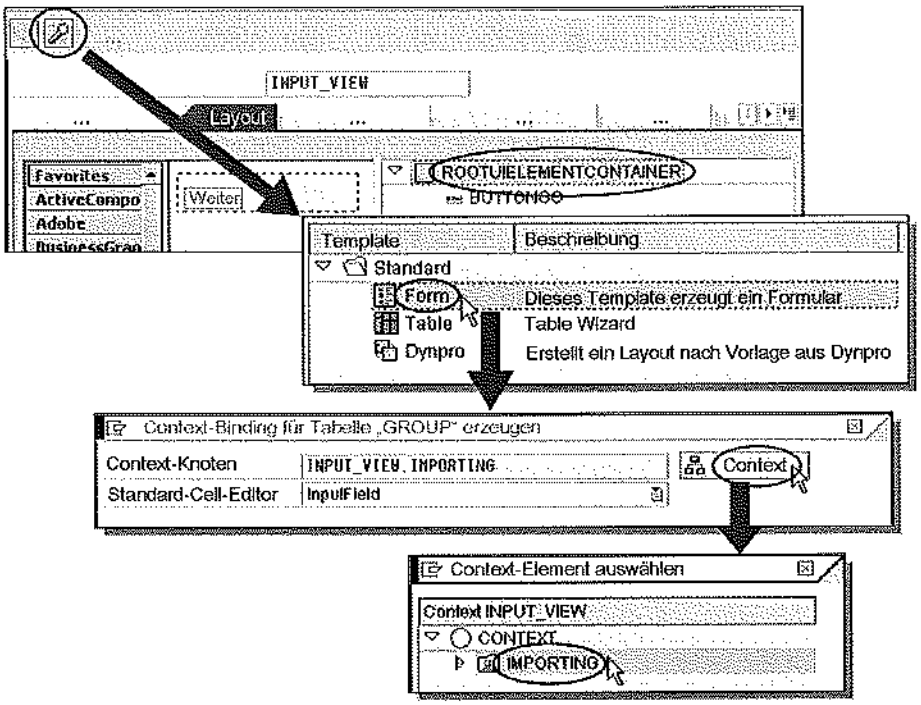


Abbildung 216: Layout-Formular anlegen

Auch für das Anlegen von Ein-/Ausgabefeldern und Einrichten des Data Binding stellt die *ABAP Workbench* einen Wizard zur Verfügung. Dieser setzt voraus, dass der Knoten, an den gebunden werden soll, bereits im Context des View-Controllers vorhanden ist. Dann kann für diesen Knoten sehr einfach ein Layout-Formular angelegt werden.

Öffnen Sie die View im Änderungsmodus und gehen Sie zum Reiter *Layout*. Markieren Sie in der Element-Hierarchie (rechts oben) den Eintrag *ROOTUIELEMENTCONTAINER*. Starten Sie den *Web Dynpro Wizard* über die entsprechende Drucktaste (siehe obige Grafik) und wählen Sie die Vorlage *Form*. Durch Betätigen der Drucktaste *Context* gelangen Sie in einen Dialog, in dem Sie direkt den Knoten auswählen können, für den das Formular erstellt werden soll. Enthält der Knoten mehrere Felder, können diese anschließend noch gezielt ausgewählt bzw. ausgeschlossen werden.

Aufruf der Service-Methode zur Datenbeschaffung

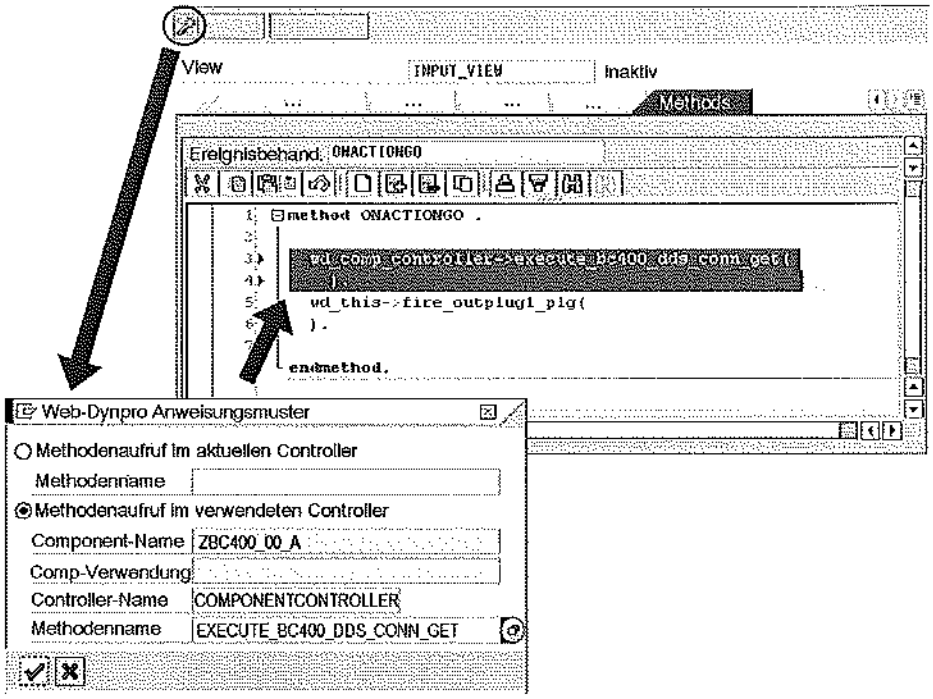


Abbildung 217: Methodenaufruf implementieren

Damit auf der zweiten View die Details zur Flugverbindung angezeigt werden können, muss vor der Navigation die Service-Methode aufgerufen werden. In unserem Beispiel erfolgt dieser Aufruf in der Aktionsbehandlermethode (Methode *ONACTION...*) der Eingabeview.

Auch der Aufruf von Methoden kann mit dem *Web Dynpro Wizard* generiert werden. Öffnen Sie dazu den Quelltext der Aktionsbehandlermethode, positionieren Sie den Cursor vor dem Quelltext zum Feuern des Outbound-Plugs und betätigen Sie die Drucktaste zum Starten des Wizard (siehe obige Grafik).

Wählen Sie auf dem nachfolgenden Bild *Methodenaufruf im verwendeten Controller* und tragen Sie mittels Wertehilfe den Component-Controller als verwendeten Controller ein. Wählen Sie anschließend, ebenfalls über die Wertehilfe, die Methode ein, die aufgerufen werden soll.

Zum Abschluss des Vorgangs erzeugt der Wizard den Quelltext zum Aufruf der Methode.