Enhancing Output Determination

Technical Solution Guide
Enhancing Output Determination

Enhancing Output Determination ................................................................. 3
Condition Usage Code ................................................................................. 3
Application Area ............................................................................................ 4
  Application Areas Relevant for Output Determination .................................. 4
Communication Structures ............................................................................ 6
  Communication Structures by Application Area ............................................ 8
The Field Catalog ............................................................................................ 10
  Field Catalog Structure ............................................................................... 10
  Maintaining the Field Catalog ................................................................... 12
  Building Condition Tables ......................................................................... 14
Processing Function Modules ......................................................................... 15
Relevant OSS Notes ....................................................................................... 15

Process Flow .................................................................................................. 15
  Step 1  Filling the Header Structure ............................................................... 17
  Step 2  Call Function COMMUNICATION_AREA_KOMKBV1 ......................... 18
  Step 3  Call Function KOMKBV1_FILL .......................................................... 19
  Step 4  Filling the Item Structure ................................................................ 20
  Step 5  Call Function COMMUNICATION_AREA_KOMPBV1 ......................... 21
  Step 6  Call Function KOMPBV1_FILL .......................................................... 22
  Step 7  Call Function MESSAGING .............................................................. 23
Sales Order Confirmation Output Enhancement Example ........................... 24
  Business Requirement ................................................................................. 24
  Solution ....................................................................................................... 24
  Step 1  Create a new field for use in Output Determination ......................... 25
  Step 2  Make ZZ_RTI_ORDER available to the field catalog ....................... 27
  Step 3  Populating the ZZ_RTI_ORDER indicator ........................................ 29
  Step 4  Adding ZZ_RTI_ORDER to the field catalog .................................... 31
  Step 5  Building the new condition table ..................................................... 32
  Step 6  Adding the new condition table to the access sequence .................. 33
  Step 7  Creating a Sales Rep customer master ............................................ 34
  Step 8  Maintaining the condition record ................................................... 35
Billing Output Enhancement Example ......................................................... 37
  Business Requirement ................................................................................. 37
  Solution ....................................................................................................... 37
  Step 1  Create a new field for use in Output Determination ......................... 38
  Step 2  Populating the AUART field ............................................................ 40
  Step 3  Build the Requirement Routine ...................................................... 42
  Step 4  Assigning the routine to an output type ......................................... 44
  Step 5  Testing the process ......................................................................... 45
  Step 6  Running the VOFM regeneration ..................................................... 46
Enhancing Output Determination

Output Determination is one of several functions that use the R/3 condition technique. The purpose of this document is to demonstrate how to enhance output determination using the standard user exits built into R/3. In many cases, designing a successful enhancement requires both functional configuration and program logic. This document explains the user exit functionality, and then details two sample enhancements taken from actual projects. The examples were built on a 4.6C system.

This document assumes that the reader understands the condition technique and its configuration for Output Determination.

Condition Usage Code

The condition technique is used in several areas of SD (Pricing, Output Determination, etc.). All of these areas share the same functions and programs for maintaining condition records, etc. To differentiate between these areas, the Condition Usage Code is used. The condition technique for output determination is identified via usage code ‘B’. Many of the objects related to output determination contain this code in either the object name or as a high order key in configuration tables. For example, building condition table 900 in output determination creates a physical table name B900. In pricing, where the condition usage code is ‘A’, the physical table name is A900.
Enhancing Output Determination

Application Area

The Application Area subdivides the condition usage of the condition technique (for example, sales order, billing, purchase order). The communication structures and processing functions for output determination are built separately for each application area. Table T681A contains all application areas defined in R/3. Table T681Z contains the subset of those application areas that are relevant for output determination. The following list is from T681Z.

Application Areas Relevant for Output Determination

<table>
<thead>
<tr>
<th>Area</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CF</td>
<td>Error control</td>
</tr>
<tr>
<td>CV</td>
<td>Document administration</td>
</tr>
<tr>
<td>DB</td>
<td>Customers</td>
</tr>
<tr>
<td>E1</td>
<td>Inbound delivery</td>
</tr>
<tr>
<td>EA</td>
<td>Purchasing RFQ</td>
</tr>
<tr>
<td>EF</td>
<td>Purchase Order</td>
</tr>
<tr>
<td>EL</td>
<td>Purchasing Scheduling Agreement Release</td>
</tr>
<tr>
<td>ES</td>
<td>Data entry sheet</td>
</tr>
<tr>
<td>EV</td>
<td>Purchasing Outline Agreement</td>
</tr>
<tr>
<td>K1</td>
<td>Sales activities</td>
</tr>
<tr>
<td>M1</td>
<td>Rough Goods Receipt</td>
</tr>
<tr>
<td>MD</td>
<td>Direct buying</td>
</tr>
<tr>
<td>ME</td>
<td>Inventory management</td>
</tr>
<tr>
<td>MR</td>
<td>Invoice verification</td>
</tr>
<tr>
<td>NF</td>
<td>Nota fiscal</td>
</tr>
<tr>
<td>P1</td>
<td>Direct Manufacturing</td>
</tr>
<tr>
<td>PA</td>
<td>JIT call</td>
</tr>
<tr>
<td>RS</td>
<td>Automotive</td>
</tr>
<tr>
<td>V1</td>
<td>Sales</td>
</tr>
<tr>
<td>V2</td>
<td>Shipping</td>
</tr>
<tr>
<td>V3</td>
<td>Billing</td>
</tr>
<tr>
<td>V4</td>
<td>Shipping (Special case)</td>
</tr>
</tbody>
</table>
## Enhancing Output Determination

<table>
<thead>
<tr>
<th>Area</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>V5</td>
<td>Groups</td>
</tr>
<tr>
<td>V6</td>
<td>Handling units</td>
</tr>
<tr>
<td>V7</td>
<td>Transport</td>
</tr>
<tr>
<td>W0</td>
<td>All. table Group</td>
</tr>
<tr>
<td>W1</td>
<td>Sales promotion group</td>
</tr>
<tr>
<td>WA</td>
<td>Allocation table</td>
</tr>
<tr>
<td>WB</td>
<td>Retail promotion</td>
</tr>
<tr>
<td>WE</td>
<td>Labeling</td>
</tr>
<tr>
<td>WR</td>
<td>Agency Business</td>
</tr>
<tr>
<td>WS</td>
<td>Free Sett. Agreement</td>
</tr>
<tr>
<td>WT</td>
<td>Trading contract</td>
</tr>
</tbody>
</table>
Communication Structures

R/3 uses work areas called communication structures to pass data from the calling application to the output determination functions. There are separate structures for header and item data (when relevant). In many cases, the communication structures contain special includes for customer specific enhancements. Only these special includes should be modified by the customer. The following example is the delivery header communication structure. The first screen displays the top portion of the structure.
Towards the bottom of the structure, the *include* for customer enhancements is found (KOMKBZ4). These includes always contain the field DUMMY. This field exists because in earlier versions of R/3, the data dictionary did not allow structures that contained no fields. SAP put the DUMMY field into these structures just to get around this requirement. When customer specific fields are added to these structures, the DUMMY field should be deleted.

On several occasions, we have seen programmers actually leave the DUMMY field as is, but use it as an indicator for some purpose. We strongly recommend that this practice not be used. If a field is required, it should be created with a relevant name and data element.
## Communication Structures by Application Area

Table T681Z contains the communication structure names for each application area. The name usually corresponds to the application area, but there are exceptions. Notice in the following list that application areas EA and EF use the same header structure.

<table>
<thead>
<tr>
<th>Area</th>
<th>Header Structure</th>
<th>Item Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>CF</td>
<td>KOMKBCF</td>
<td></td>
</tr>
<tr>
<td>CV</td>
<td>KOMKBCV</td>
<td></td>
</tr>
<tr>
<td>DB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>E1</td>
<td>KOMKBE1</td>
<td></td>
</tr>
<tr>
<td>EA</td>
<td>KOMKBEA</td>
<td></td>
</tr>
<tr>
<td>EF</td>
<td>KOMKBEA</td>
<td>KOMPBEA</td>
</tr>
<tr>
<td>EL</td>
<td>KOMKBEA</td>
<td></td>
</tr>
<tr>
<td>ES</td>
<td>KOMKBES</td>
<td></td>
</tr>
<tr>
<td>EV</td>
<td>KOMKBEA</td>
<td></td>
</tr>
<tr>
<td>K1</td>
<td>KOMKBEA</td>
<td></td>
</tr>
<tr>
<td>M1</td>
<td>KOMKBM1</td>
<td></td>
</tr>
<tr>
<td>MD</td>
<td>KOMKBMD</td>
<td></td>
</tr>
<tr>
<td>ME</td>
<td>KOMKBEA</td>
<td>KOMPBE</td>
</tr>
<tr>
<td>MR</td>
<td>KOMKBME</td>
<td>KOMPBM</td>
</tr>
<tr>
<td>NF</td>
<td>KOMKBNF</td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td>KOMKBP1</td>
<td></td>
</tr>
<tr>
<td>PA</td>
<td>KOMKBPA</td>
<td></td>
</tr>
<tr>
<td>RS</td>
<td>KOMKBRS</td>
<td></td>
</tr>
<tr>
<td>V1</td>
<td>KOMKBEA</td>
<td>KOMPBV1</td>
</tr>
<tr>
<td>V2</td>
<td>KOMKBEA</td>
<td>KOMPBV2</td>
</tr>
<tr>
<td>V3</td>
<td>KOMKBEA</td>
<td>KOMPBV3</td>
</tr>
<tr>
<td>V4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>V5</td>
<td>KOMKBEA</td>
<td></td>
</tr>
<tr>
<td>V6</td>
<td>KOMKBEA</td>
<td></td>
</tr>
<tr>
<td>V7</td>
<td>KOMKBEA</td>
<td></td>
</tr>
<tr>
<td>W0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Enhancing Output Determination

<table>
<thead>
<tr>
<th>Area</th>
<th>Header Structure</th>
<th>Item Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>W1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WA</td>
<td>KOMKBWA</td>
<td>KOMPBWA</td>
</tr>
<tr>
<td>WB</td>
<td>KOMKBWB</td>
<td>KOMPBWB</td>
</tr>
<tr>
<td>WE</td>
<td>KOMKBWE</td>
<td></td>
</tr>
<tr>
<td>WN</td>
<td>KOMKBWN</td>
<td></td>
</tr>
<tr>
<td>WR</td>
<td>KOMKBWR</td>
<td>KOMPBWR</td>
</tr>
<tr>
<td>WS</td>
<td>KOMKBWS</td>
<td></td>
</tr>
<tr>
<td>WT</td>
<td>KOMKBWT</td>
<td>KOMPBWT</td>
</tr>
</tbody>
</table>
The Field Catalog

The field catalog is a configuration table (T681F) that contains all fields that can be used as key fields when building condition tables. When the field catalog is maintained, the available field list is based on the fields contained in the field catalog structure.

Field Catalog Structure

There are separate field catalogs for each condition usage code. For output determination, the field catalog structure is KOMB. Like the communication structures, the field catalog structure contains an include for customer specific fields (KOMBZ).
Enhancing Output Determination

In this example, the KOMBZ include has been modified with several new fields. The DUMMY field was removed.
Enhancing Output Determination

Maintaining the Field Catalog

The field catalog maintenance transaction is found next to the corresponding condition table maintenance transaction in the IMG. Although the field catalog transaction is listed second, it is processed first.
Enhancing Output Determination

The field catalog maintenance program is a standard SM30 screen that maintains table T681F. All field catalogs used in the condition technique are contained in T681F. The high order keys of this table are the condition usage code and application area. Since this transaction maintains the field catalog for sales orders, the T681F entries maintained are from condition usage code ‘B’ and application area ‘V1’. While adding fields, the valid field list consists of the fields in the field catalog structure (KOMB).
Building Condition Tables
When building condition tables, the available fields in the right pane, are the fields from T681F in the corresponding condition usage and application area. (‘B’, ‘V1’).
Enhancing Output Determination

Processing Function Modules
There are two main function groups used for output determination. V61B contains the main functions that enable output determination. VCOM contains functions and user exits for filling the communication structures with document data from the calling programs.

Relevant OSS Notes
The following OSS notes may be helpful in understanding output determination enhancements.

<table>
<thead>
<tr>
<th>Note</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>32662</td>
<td>User name added to new condition table</td>
</tr>
<tr>
<td>39462</td>
<td>Expand field catalog in message determination</td>
</tr>
<tr>
<td>159684</td>
<td>Not all fields are displayed in condition table</td>
</tr>
</tbody>
</table>

Process Flow
The following diagram and screen shots illustrate how the processing functions are called using sales order output as an example.
Enhancing Output Determination

---

### Function Group V61B

**Global Data Area:**

- Calls Function `COMMUNICATION_AREA_KOMKBV1` passing sales order data.

<table>
<thead>
<tr>
<th>Function Modules</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>KOMKBV1</code></td>
<td>Receives sales order data. Calls Function <code>KOMKBV1_FILL</code> passing sales order data. Updates KOMKBV1 work area.</td>
</tr>
<tr>
<td><code>KOMPBV1</code></td>
<td>Receives sales order data. Calls Function <code>KOMPBV1_FILL</code> passing sales order data. Updates KOMPBV1 work area.</td>
</tr>
<tr>
<td><code>FUNCTION MESSAGING</code></td>
<td>Executes the condition technique routines for output determination using the filled data areas.</td>
</tr>
</tbody>
</table>

### Function Group VCOM

**Global Data Area:**

- Calls Function `KOMKBV1_FILL` passing sales order data.
- Calls Function `KOMPBV1_FILL` passing sales order data.
- Calls Function `MESSAGING`.

<table>
<thead>
<tr>
<th>Function Modules</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>KOMKBV1</code></td>
<td>Receives sales order data. Populates KOMKBV1 with sales order header data. Performs USEREXIT_KOMKBV1_FILL. Updates KOMKBV1 work area.</td>
</tr>
<tr>
<td><code>KOMPBV1</code></td>
<td>Receives sales order data. Populates KOMPBV1 with sales order item data. Performs USEREXIT_KOMPBV1_FILL. Updates KOMPBV1 work area.</td>
</tr>
</tbody>
</table>

**Form Routines**

- `FORM USEREXIT_KOMKBV1_FILL`
  ```
  * .......
  END FORM.
  ```
- `FORM USEREXIT_KOMPBV1_FILL`
  ```
  * .......
  END FORM.
  ```
Step 1  Filling the Header Structure

Function COMMUNICATION_AREA_KOMKBV1 is called. All of the relevant order data is passed to the function module (VBAK, VBUK, VBPA, VBAP). The function module is contained in function group V61B. This function group’s global data area is where the communication structure data is held during the output process.
Step 2  Call Function COMMUNICATION_AREA_KOMKBV1

This function calls the communication structure fill function KOMKBV1_FILL passing the order data structures. The KOMKBV1 communication structure is returned from the call and is now completely filled. The structure is defined in the global data area of V61B and will remain in memory until the output is processed by subsequent function calls.
Step 3  Call Function KOMKBV1_FILL

This function fills the header communication structure. It also calls the user exit routines associated with the structure type. All of the ‘fill’ functions and user exits are contained in function group VCOM.

```plaintext
FUNCTION KOMKBV1_FILL
  **Globale Schnittstelle:**
  **IMPORTING**
  * VALUE (COM_TRYP) LIKE TABLE(1) TYPE COMTRYP DEFAULT 0.
  * VALUE (COM_VBAK) LIKE TABLE 1 TYPE YBAK STRUCTURE YBAK.
  * VALUE (COM_YBAK) LIKE TABLE 1 TYPE YDK STRUCTURE YDK.
  * VALUE (COM_YBAD) LIKE TABLE 1 TYPE YDK STRUCTURE YDK.
  * EXPORTING
    * VALUE (COM_KEY) LIKE COMKEY STRUCTURE COMKBV1.
    * TABLES
      * COM_PART STRUCTURE MAPA.
      * COM_YBAK STRUCTURE YBAKY.
      * COM_YBAD_TAB STRUCTURE YBAKY OPTIONAL.
  LOCAL.
  FIELD-VALUE.
  CLEAR COM_KEY.
  MOVE-CORRESPONDING COM_VBAK TO COM_KEY.
  MOVE-CORRESPONDING COM_YBAK TO COM_KEY.
  MOVE-CORRESPONDING COM_YBAD TO COM_KEY.
  MOVE COM_TRYP TO COM_KEY-TRYP.
  PERFORM USEREXIT_KOMKBV1_FILL.
  LOOP AT COM_YBAK.
    CLEAR COM_PART.
    IF COM_YBAK-PGNR EQ 0.
      CASE COM_YBAK-PAR.
        WHEN 'AG'.
          COM_KEY-KUNNR = COM_YBAK-KUNNR.
        WHEN 'CA'.
          COM_KEY-KUNNR = COM_YBAK-KUNNR.
        WHEN 'GR'.
          COM_KEY-KUNRE = COM_YBAK-KUNRE.
        WHEN 'RE'.
          COM_KEY-KUNRE = COM_YBAK-KUNRE.
        WHEN 'EX'.
          COM_KEY-EXKRF = COM_YBAK-EXKRF.
        END-CASE.
  ENDLOOP.
  **ENDFUNCTION.**
```
Step 4  **Filling the Item Structure**

After the previous function calls, control has been returned to the sales order program. The next step is to fill the item structure. Function COMMUNICATION_AREA_KOMPBV1 is called. The same order data is passed to the function module (VBAK, VBUK, VBPA, VBAP).

```
LOOP AT XVAB WHERE UPKZ NE UPKZ_DELETE AND XVPOS IS INITIAL.
  * Tuning
  PERFORM XVAP_SELECT(SAPMM45A) USING XVAB-PSY V.
  SPACE SY-SUBRC.
  CHECK XVAP-KALSME SPACE.
  PERFORM XVAP_BEARBEITEN_VORBEREITEN(SAPIV45P).
  * Objektiv aufbauen
  MOVE XVBAK-VABLN TO XVPOS-POS.
  WRITE XVBAK-POSNR TO XVPOS-POS+10.
  * Kommunikationsblock aufbauen
  XVUP = XVAB
  CALL FUNCTION 'COMMUNICATION_AREA_KOMPBV' EXEPTING.
  MSG_XVUP = XVUP
  MSG_XVBK = XVBAK
  MSG_XVBAK = XVBAK.
  * EHS environment
  XVBAK = XVBAK.
  TABLES XVBAK = XVBAK.
  XVPART = XVBAK.
  IF XVPOS-TRIPH = CHARG.
  IF XVPOS-TRIH = CHARG.
  * Kommunikationsblock ait aufbauen
  PERFORM XVBAK_LESEN(SAPIV45P) USING XVBAK-POSNR SY-TABIX.
  XVBAK = XVBAK.
  PERFORM XVBAK_LESEN(SAPIV45P) USING XVBAK-POSNR SY-TABIX 'IF SY-SUBRC = 0.
  XVBAK = XVBAK.
  ELSE
  XVBAK-POSNR = 0.
  PERFORM XVBAK_LESEN(SAPIV45P) USING XVBAK-POSNR SY-TABIX 'IF SY-SUBRC = 0.
  XVBAK = XVBAK.
  EXIT.
```
Step 5 Call Function COMMUNICATION_AREA_KOMPBV1

This function calls the communication structure fill function KOMPBV1_FILL passing the order data structures. The KOMPBV1 communication structure is returned from the call and is now completely filled. The structure is defined in the global data area of V61B and like the header structure, will remain in memory until the output is processed by subsequent function calls.
Step 6  Call Function KOMPBV1_FILL

This function fills the item communication structure. It also calls the user exit routines associated with the structure type.
Step 7  Call Function MESSAGING

This function triggers the output. It is contained in function group V61B and has access to the communication structures that were filled by the previous function calls.
Sales Order Confirmation Output Enhancement Example

The following section demonstrates how to enhance output determination using the user exits provided. Although this is a sales order example, the basic process is the same for all other output.

Business Requirement

During order entry, the system has been configured to generate an order confirmation using the standard configuration. The order confirmation is sent to the customer. The company has introduced a new product line and wants to re-direct the order confirmations to an internal sales person for all orders that contain at least one of the new products. All of these confirmations are to be emailed to the same sales person.

Solution

To solve this problem, a new condition table will be added to the access sequence for the order confirmation. A condition record will be found only if one of the new products is contained in the order. The table will be positioned first in the access sequence and when accessed successfully, the remaining tables will not be processed. A field will be created to denote that one of the new products is present on the order. A user exit will be written to examine the materials on the order, and set the indicator accordingly.
Step 1  Create a new field for use in Output Determination

For header level sales order output determination, the system uses communication structure KOMKBV1 to pass order data to the output determination processing routines. In addition to the standard fields, this structure contains an include for customer defined fields (KOMKBZ3).
Using the customer modification structure KOMKBZ3, new field ZZ_RTI_ORDER was added. This field is an indicator that will be set if the sales order contains an item for the special subset of materials.
Step 2  Make ZZ_RTI_ORDER available to the field catalog

Since we plan to use this field as a key to a condition table, it must be available in the field catalog for output determination. Structure KOMB contains all fields that are available in the field catalog. This structure contains an include for customer defined fields (KOMBZ).
Using the customer modification structure KOMBZ, new field ZZ_RTI_ORDER was added. This field is an indicator that will be set if the sales order contains an item for the special subset of materials.
Step 3  Populating the ZZ_RTI_ORDER indicator

During order output, the system calls function KOMKBV1_FILL to populate the communication structures with sales order data. All of the relevant order data is passed into this function having been read previously. The function calls form USEREXIT_KOMKBV1_FILL, which is the header level user exit routine.
Enhancing Output Determination

In this user exit, the new logic loops thru the items (COM_VBAP_TAB) and using the product hierarchy, determines if any of the new products exist on the order. If any are found, the ZZ_RTI_ORDER indicator is set to ‘X’.

Note that no database selects are required for this. The most common error made with these enhancements is the programmer redundantly selecting data that already resides in memory.
Step 4  ADDING ZZ_RTI_ORDER TO THE FIELD CATALOG

Using transaction V/86, the ZZ_RTI_ORDER field is added to the field catalog for sales order related output determination.
Step 5  Building the new condition table

Using transaction V/57, condition table 901 is created. The only field in this table will be the ZZ_RTI_ORDER indicator.
Step 6 Adding the new condition table to the access sequence

Using transaction V/48, condition table 901 is added to access sequence Z002. The table is inserted as the first table in the sequence. Access Z002 is assigned to the order confirmation output type.
Step 7  Creating a Sales Rep customer master

At this client, sales reps are defined using customer master records. Order confirmations are sent via email to a contact person assigned to the customer. Here we defined the contact person for our special email.
**Step 8**  
*Maintaining the condition record.*

Using transaction VV11, a single condition record is added to table 901.
## Enhancing Output Determination

![Screenshot of SAP interface](image)

### Create Condition Records (Order Conf. non EDI): Fast Entry

<table>
<thead>
<tr>
<th>Name</th>
<th>PartID</th>
<th>Partbar</th>
<th>ML</th>
<th>Time</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>All BII Items</strong></td>
<td>CP</td>
<td>00000004077</td>
<td>5</td>
<td>4</td>
<td>EN</td>
</tr>
</tbody>
</table>
Billing Output Enhancement Example

The following section demonstrates how to implement a requirement routine to prevent condition access in output determination. Although this is a billing document example, the basic process is the same for all other output.

Business Requirement
The customer has defined a special ‘No charge’ order type that is used to ship sample products. For these orders, billing output is to be suppressed. The billing type associated with these orders is the standard F2.

Solution
To solve this problem, an output requirement routine will be coded and assigned to the output type within the relevant output determination procedure. Within this routine, we need to examine the sales order type associated with the invoice. Unfortunately, the order type is not included in the communication structures used for billing output. Therefore, a user exit will be coded to provide it.
Step 1  Create a new field for use in Output Determination

For header level billing output determination, the system uses communication structure KOMKBV3 to pass order data to the output determination processing routines. In addition to the standard fields, this structure contains an include for customer defined fields (KOMKBZ5).
Using the customer modification structure KOMKBZ5, the order type field AUART was added.

There are many who debate whether or not a standard field name (i.e. AUART) should be specified in a user exit structure. Even SAP would probably recommend using a ‘Z’ field (i.e. ZZAUART). They will argue that if SAP were to add the standard field in a future release, a syntax error would result. While true, we feel that ending up with redundant fields is worse than fixing a syntax error. To correct the error would simply require that the field be deleted from the user exit structure.
Step 2  Populating the AUART field
During billing output, the system calls function KOMKBV3_FILL to populate the communication structures with billing document data. All of the relevant billing data is passed into this function having been read previously. The function calls form USEREXIT_KOMKBV3_FILL which is the header level user exit routine. For this project, we need to add the sales order type. Since order information is not passed into this function module, we will need to select it in our user exit.
Enhancing Output Determination

In this user exit, the new logic determines the sales order type and moves it to the KOMKBV3 communication structure.
Step 3  **Build the Requirement Routine**

Using transaction VOFM, create the requirement routine. The standard SAP routine is number 062. We cloned it to 962.
Enhancing Output Determination

One line of code was added to each form routine to check the sales order type. When it is ZNC, the return code will be set to 4 and the output will not be sent.
Step 4  Assigning the routine to an output type

Using transaction V/42, assign the new requirement routine to the relevant output types in the output determination procedure.
**Step 5  Testing the process**

Below is the output determination process log of an invoice that originated from a ZNC order. The output was not sent due to requirement 962.
Step 6  Running the VOFM regeneration

Whenever a VOFM routine is created, a special generation program should be run in each system that the routine is moved to. The program is usually executed by the Basis Team.

Note: This is a critical step that should not be overlooked. If this step is not executed, short dumps can be the result.
Enhancing Output Determination

The RV80HGEN completed.