ABAP® Development for Sales and Distribution in SAP®

Exits, BAdIs, and Enhancements
Contents at a Glance

1 Introduction to Enhancements in Sales and Distribution .......... 19

PART I Enhancements in Sales Order Processing
2 Validating Sales Order Data ...................................................... 47
3 Capturing and Saving Additional Data Fields in Sales Order Processing ................................................................. 61
4 Creating an SAP CRM Activity after SD Order Billing ............. 83
5 Filtering Pricing Data within a Web Service ............................. 101
6 Using a Custom Field in SD Pricing ......................................... 121

PART II Enhancements in Delivery Processing
7 Setting a Delivery Block on the Header Level ....................... 135
8 Using a BAdI to Keep Track of Delivery KPIs ........................ 145
9 Using a Customer Exit to Enhance the Outbound Delivery Monitor .............................................................................. 163

PART III Enhancements in Billing
10 Invoice Splitting Using a VOFM Routine ............................... 179
11 Changing Invoice Reference and Numbering Range during Document Creation ............................................................ 195

PART IV Conclusion
12 Finding the Right Enhancement Technique .......................... 211
13 Future Outlook ........................................................................ 217

Appendices
A Exits and BAdIs in Sales and Distribution ............................. 223
B Code Listing for Class ZCL_SURVEY ........................................ 249
C The Author ............................................................................. 253
## 3 Capturing and Saving Additional Data Fields in Sales Order Processing

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Business Scenario</td>
<td>61</td>
</tr>
<tr>
<td>3.2</td>
<td>Finding the Right Enhancement</td>
<td>62</td>
</tr>
<tr>
<td>3.2.1</td>
<td>Using the Additional Data B Tab</td>
<td>63</td>
</tr>
<tr>
<td>3.2.2</td>
<td>Using Implicit Enhancements</td>
<td>63</td>
</tr>
<tr>
<td>3.2.3</td>
<td>Using Persistent Objects</td>
<td>65</td>
</tr>
<tr>
<td>3.2.4</td>
<td>USEREXIT_SAVE_DOCUMENT</td>
<td>65</td>
</tr>
<tr>
<td>3.3</td>
<td>Implementing the Solution</td>
<td>66</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Declarations of Data Elements, Database Tables, and Table Types</td>
<td>66</td>
</tr>
<tr>
<td>3.3.2</td>
<td>Create Persistency Classes for Survey Data</td>
<td>68</td>
</tr>
<tr>
<td>3.3.3</td>
<td>Create Database Table Lock Entries</td>
<td>71</td>
</tr>
<tr>
<td>3.3.4</td>
<td>Creating Modification and Implicit Enhancements for a Custom Screen</td>
<td>71</td>
</tr>
<tr>
<td>3.3.5</td>
<td>Create the Implicit Enhancement for PBO Modules</td>
<td>73</td>
</tr>
<tr>
<td>3.3.6</td>
<td>Create the Implicit Enhancement for Module PAI</td>
<td>76</td>
</tr>
<tr>
<td>3.3.7</td>
<td>Adding Layout Changes to Screen 8309 of SAPMV45A</td>
<td>81</td>
</tr>
<tr>
<td>3.4</td>
<td>Summary</td>
<td>82</td>
</tr>
</tbody>
</table>

## 4 Creating an SAP CRM Activity after SD Order Billing

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Business Scenario</td>
<td>83</td>
</tr>
<tr>
<td>4.2</td>
<td>Finding the Right Enhancement</td>
<td>84</td>
</tr>
<tr>
<td>4.3</td>
<td>Communication between SAP Systems via RFC</td>
<td>90</td>
</tr>
<tr>
<td>4.3.1</td>
<td>ABAP-to-ABAP Communications</td>
<td>90</td>
</tr>
<tr>
<td>4.3.2</td>
<td>Setting Up RFC Connections</td>
<td>91</td>
</tr>
<tr>
<td>4.4</td>
<td>Implementing the Solution</td>
<td>93</td>
</tr>
<tr>
<td>4.4.1</td>
<td>Creating a New Function Group in the SAP CRM System</td>
<td>94</td>
</tr>
<tr>
<td>4.4.2</td>
<td>Creating an RFC Function Module in the SAP CRM System</td>
<td>94</td>
</tr>
<tr>
<td>4.4.3</td>
<td>Specify Function Module Parameters</td>
<td>95</td>
</tr>
<tr>
<td>4.4.4</td>
<td>Implement ABAP Logic to Create an Activity in SAP CRM</td>
<td>95</td>
</tr>
</tbody>
</table>
# PART II  Enhancements in Delivery Processing

## 7  Setting a Delivery Block on the Header Level ................................. 135

7.1 Business Scenario ........................................................................ 135  
7.2 Finding the Right Enhancement .................................................. 136  
7.3 Implementing the Solution ......................................................... 137  
   7.3.1 Locating the BAdI .......................................................... 137  
   7.3.2 Creating a BAdI Implementation .................................... 140  
   7.3.3 Adding Code to the Implementation .............................. 143  
7.4 Summary .................................................................................... 144  

## 8  Using a BAdI to Keep Track of Delivery KPIs ................................. 145

8.1 Business Scenario ........................................................................ 145  
   8.1.1 Current KPI Solution ...................................................... 145  
   8.1.2 To-Be Process for KPI Reporting ..................................... 146  
8.2 Finding the Right Enhancement .................................................. 148  
8.3 Implementing the Solution ......................................................... 150  
   8.3.1 Custom KPI Field Table Layouts ...................................... 150  
   8.3.2 Define Custom Table ZKPI_FIELDS .................................. 151  
   8.3.3 Generating a Table Maintenance for ZKPI_FIELDS ....... 154  
   8.3.4 Define Custom Table ZKPI_DATA ................................... 155  
   8.3.5 Implement ABAP Coding in Method SAVE_AND_  
      PUBLISH_DOCUMENT ................................................... 156  
8.4 Summary .................................................................................... 161  

## 9  Using a Customer Exit to Enhance the Outbound Delivery  
    Monitor ..................................................................................... 163

9.1 Business Scenario ........................................................................ 163  
9.2 Finding the Right Enhancement .................................................. 165  
9.3 Implementing the Solution ......................................................... 166  
   9.3.1 Implement the Customer Exit Routine ............................ 166  
   9.3.2 Add New Field Entries to Append Structure LIPOVZ ..... 168  
   9.3.3 Add Coding to Derive KPI Data and Calculate  
      Delivery Age .................................................................. 172  
   9.3.4 Testing the Enhanced Delivery Monitor .......................... 175  
9.4 Summary .................................................................................... 175
**Contents**

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>13 Future Outlook</td>
<td>217</td>
</tr>
<tr>
<td><strong>Appendices</strong></td>
<td>221</td>
</tr>
<tr>
<td>A Exits and BAdIs in Sales and Distribution</td>
<td>223</td>
</tr>
<tr>
<td>A.1 Sales Orders</td>
<td>223</td>
</tr>
<tr>
<td>A.2 Deliveries</td>
<td>232</td>
</tr>
<tr>
<td>A.3 Invoicing</td>
<td>238</td>
</tr>
<tr>
<td>A.4 Transport Documents</td>
<td>242</td>
</tr>
<tr>
<td>A.5 Shipment Costs</td>
<td>246</td>
</tr>
<tr>
<td>B Code Listing for Class ZCL_SURVEY</td>
<td>249</td>
</tr>
<tr>
<td>C The Author</td>
<td>253</td>
</tr>
<tr>
<td><strong>Index</strong></td>
<td>255</td>
</tr>
</tbody>
</table>
Today, SAP ERP counts for more than 65,000 installations worldwide. Experts estimate that 90-95% of these installations use Sales and Distribution (SD), so we can assume that roughly 60,000 businesses around the globe use it. In other words, SAP SD is a true ERP success story.

There are many reasons for this success, but one that repeatedly ranks the highest is the level of configuration and customization that can be achieved within SD. Both sales and distribution form integral parts of most modern businesses. Therefore, the ability to mold the system to key business processes—where deemed necessary—can be paramount. While configuration enables a business to add system settings within the boundaries of the programming code defined by SAP, customization goes deeper and requires additional code in ABAP, SAP’s programming language.

However, customization of standard software is often frowned upon. Experts perceive any deviation from the standard as a potential headache when it comes to future upgrades. This leaves customers in a dilemma, as shown in the following light-hearted illustration.

A Customer’s Enhancement Dilemma
Over the past 20 years, however, SAP has worked hard to allow its customers to slightly adjust key processes if necessary, yet still remain on the upgrade path. This is enabled by different kinds of technologies such as exits, business add-ins (BAdIs), and the more recent introduction of the Enhancement Framework. These technologies support ABAP code customizations and play a major role when modifying SAP’s standard ABAP delivery.

About This Book

This book is predominantly based on my own consulting experience and tasks I have worked on over the years. I have tried to include helpful, real-world examples of varying difficulty and complexity.

Chapter 1 provides background information and comparisons of the different enhancement techniques used in this book. After this, Chapter 2 through Chapter 11 will show you how to do the following:

- Add new field validations for a sales order.
- Capture additional data in custom fields within Transactions VA01/02.
- Automatically create a SAP CRM activity after billing cancellation.
- Filter pricing data within a SAP standard web service for order confirmation.
- Add custom fields within SD pricing.
- Set a delivery block from new validation within a code enhancement.
- Use a BAdI to keep track of delivery KPIs within a little custom reporting system.
- Enhance a standard delivery monitor report to include custom KPI data.
- Split invoice items into separate documents by using a VOFM routine.
- Change the reference field and numbering range while a new invoice is created.

Chapter 12 then summarizes the techniques and enhancement methods. Lastly, Chapter 13 concludes with an outlook on ERP enhancements and their role in the future.

Within the appendices you will find an overview of all SD-related exits and BAdIs, as well as one of the longer coding examples (which can also be found online).
Who Benefits from This Book

This book will be beneficial to the following groups of people:

- ABAP developers (beginner, intermediate)
- SD consultants (all levels of experience)
- Support consultants (all levels of experience)
- SAP architects (all levels of experience)
- Project team members who require a deeper understanding of how to enhance standard SAP ERP

System Prerequisites

All examples in this book have been created and tested in a SAP ECC 6.06 system on SAP NetWeaver AS ABAP 7.31. However, none of the shown examples require such a recent release level (as of July 2012) and any SAP ECC 6.0 and SAP NetWeaver AS ABAP 7.x and above will suffice. If you work on earlier releases, some of the newer enhancement techniques such as BAdIs or the Enhancement Framework might not be available.

If in doubt, refer to the overviews in Chapter 1 to check whether an enhancement point or BAdI is available within the application release you are using.

Fictional Project

Software development doesn't take place in a vacuum, and most of the time there are underlying pains and purposes for it. The same goes for any enhancements to SAP systems, which invariably often stem from business requirements. Therefore, simply showcasing the most common exits and enhancements with instructions on how to implement them doesn’t reach far enough. It's important to tell a business story. It's equally important for IT professionals to realize this concept and have a grasp of the business story behind a proposed change.

You’ll find that this book mirrors the “real world” as much as possible. Throughout the course of this book, I have used stories and anecdotes from a fictional project to better illustrate the discussed topics.
You’ll come across a junior ABAP developer and a senior SD consultant, who will accompany you on your learning journey. So let’s meet the business, Sean, and Christine...

The Byrell Corporation

Byrell Corporation has been trading for more than 50 years as a logistics business with depots and warehouses in various locations on the British Isles. The company has been growing steadily over the past 40 years, in particular in the past 20, which among other things led to the board’s decision to implement SAP R/3 and replace the various legacy systems previously in place. Apart from the traditional haulage business, Byrell also offers storage and warehousing solutions, based on its SAP systems and custom developments.

In 2004, Byrell decided to add SAP CRM alongside its SAP ERP system to offer its customers a better integration of opportunity management and quotations.

Byrell envisages an upgrade to the latest SAP ECC Enhancement Pack in the near future and has already instructed its SAP team to prepare for this.

Sean, Senior SD Consultant

Sean has been with the Byrell Corporation for more than 30 years now. He started with Byrell when he was in his early twenties, working as a packer in the warehouse. Over the years, his bosses recognized his technical aptitude and gave him the option to cross over into IT, where he played a pivotal role in the implementation of SAP R/3 3.1i in 1995. Since then, he has worked on other major upgrades, all the way to the current SAP ECC 6.0 release. Apart from his upgrade project work, Sean deals with smaller SD support and improvements, liaising between business and IT.

Sean acknowledges that SAP ERP has changed a lot since its first implementation at Byrell back in 1995. He is well versed with how to enhance the SD module using user exits, but newer enhancement methods such as business add-ins often raise a few questions. He also realizes that over the same period of time, the ABAP language has become more sophisticated. A lot of the code that he looks at during his support work is not always understandable for him. For some time now, he has vowed to refresh his ABAP knowledge, but work pressures and deadlines have prevented this so far.
Sean’s role is to understand the functional side of new development requirements, write specifications for Christine, and apply any necessary SD configuration, if needed.

Christine, Junior ABAP Developer

Christine started at Byrell last year as part of the company's graduate program. She came straight from the university, where she studied computing. Christine's first real programming language was Java at college, which introduced her to object-oriented (OO) programming right from the start. During some of her college and university assignments, she also picked up XHTML, Cascading Style Sheets (CSS), and a little bit of JavaScript. Christine loves coding and spends a lot of her spare time in front of computers.

Last year after Christine started at Byrell, she was sent to a couple of introductory ABAP training courses, which taught her all about report programming, screen (Dynpro) development, and ABAP OO. She was delighted that there was such a thing as object orientation in ABAP and has been using her new OO skills on many occasions. Recently Christine has also attended a course on the SAP Enhancement Framework, because her skills will be required to help Sean with some of the SD development requests coming out of support issues. In addition, there is also an SAP ECC upgrade project on the horizon, which might require some of Christine’s time.

Christine’s role is to provide technical consultancy input to new development requirements and to create ABAP code per Sean’s functional specifications.

Acknowledgments

Working on this book has been rewarding, insightful and taxing, but most of all it has been an extreme honor.

First of all, a big thank you goes out to the SAP Mentor initiative, which allowed me to reach out and receive assistance from such great individuals as Mark “wolf pack leader” Finnern, Matthias Steiner, and Thomas Jung, who helped me to get in touch with the right people within SAP for this project. Most of all, I would like to thank Martin Lang of SAP, who was instrumental in providing access to systems where I could code and test the solutions for this book.
Moreover, this book would have never come to fruition without the initiative, hard work, determination, trust, and patience of Kelly Harris and Laura Korslund of SAP PRESS. Writing a book such as this is bound to come with a few ups and downs. Kelly and Laura’s experience and assistance certainly helped in coping with some of them.

I would also like to thank friends and colleagues, namely Peter Richardson, Zoe Gill, Arpit Oberoi, Ben McGrail, Stefan Karaivanov, Thomas Otter, and Nick Watkins, whose feedback, professional advice, criticism, and ideas were essential during the entire duration of writing this book. In particular I would like to thank Cath Laursen and Arran McMillan for their assistance with questions around SAP CRM.

Special thanks also go out to Stuart Trotter of Rockpool Children’s Books (http://rockpoolchildrensbooks.com), who helped me to transfer my idea for the “SAP SD” illustration from my brain and onto paper and screen.

Finally, I want to express my love and gratefulness to my wife Sally and our beautiful daughters Milly and Hannah, who have all been helpful and supportive throughout the realization of this project. Sally, Milly, and Hannah—a lot of this book is also due to you! Thank you for giving me your understanding and patience to fulfill this dream.
Enhancements can be an excellent way to enrich reports that are delivered by standard functionality with custom customer data.

9 Using a Customer Exit to Enhance the Outbound Delivery Monitor

After the new KPI data collection was introduced (refer back to Chapter 8), Byrell’s distribution department started looking at different ways to use this data within their existing planning processes.

In Chapter 8, Christine facilitated a data extraction for delivery KPI reporting. Byrell’s distribution department now wants to get more use out of this KPI data by incorporating it directly into their reports. In this chapter, you’ll learn how Christine and Sean use the extracted KPIs to enhance a standard delivery monitor report, thereby combining two important sources of delivery data.

9.1 Business Scenario

One of the central tools used by Byrell’s distribution planners is the standard outbound delivery monitor report (Transaction VL06O). This report has many uses; for example, it provides an overview of all deliveries according to picking dates, planned transport dates, or goods issue date.

Currently, distribution planners merge the standard delivery monitor report with the newly introduced KPI data using spreadsheets, but now the distribution department wants to create flexible and integrated reports. This improvement will eliminate the need to merge data in a spreadsheet. By doing this, Byrell’s planners could benefit from the combined real-time information of the monitor report and the performance and history data of the KPI extract.

Sean attends a meeting with distribution planners at which the decision is made to change the delivery monitor to contain additional fields for the following:
- Number of KPI-related changes applied (for statistical purposes)
- Delivery age calculation to help with prioritization of backlog deliveries
- Flag to indicate whether the picking date or planned transport date has been changed within the past 24 hours (also to help with late changes to delivery priorities)

On closer inspection, Sean notices that only Byrell’s first and last request actually requires the use of KPI data, while the second request (delivery age calculation) can actually be achieved with data already available in the report. It simply requires the addition of a new field to display the difference in days between two dates.

During the meeting with the delivery planners, Sean displays the delivery report, and the group debates options for enhancing it. Currently the report looks like the screen shown in Figure 9.1.

![Figure 9.1 Outbound Delivery Monitor Report and Selector for Additional Fields](image)

When it comes to enhancing this standard report, Sean knows that there are a couple of easy options:

1. Simply copy the report and amend a customer (Z*) version of it.
2. Use implicit enhancements to change the standard report.
Neither option really seems right to him, though. While he considers these solutions, Sean remembers that there is an SAP Note that deals with the enhancement of delivery monitor reports.

### 9.2 Finding the Right Enhancement

Sean logs on to the SAP Service Marketplace (www.service.sap.com) and uses the search string “enhance VL06O delivery monitor” in the SAP Notes section (see Figure 9.2).

![Figure 9.2 Search Screen for SAP Notes](image)

Among the search results, Sean finds the one he was looking for—SAP Note 128150.

**SAP Note 128150 – VL06: Designing Your Own Display Variants**

At first, this note appears to be about the creation of report variants. However, upon closer inspection, the text also describes how additional fields (those not provided via standard delivery Tables LIKP, LIPS, and VBUP) can be included in the report and populated via customer exit V5000001. Additional fields are defined in append structure LIPOVZ, which is included in delivery monitor structure LIPOV. This structure acts as the ABAP List Viewer (ALV) line item structure for the delivery monitor.

For more information please refer to SAP Note 128150, which can be found on the SAP Service Marketplace (www.service.sap.com).
SAP Notes are usually regarded as a resource for support solutions, but as you can see, sometimes a search on the SAP Service Marketplace can also reveal useful information regarding exits and enhancements.

9.3 Implementing the Solution

After reading through the note and refreshing his memory about how to enhance the delivery monitor using customer exit V50Q0001, Sean realizes that structure LIPOVZ plays a central role in this development. All new fields to be included in the report need to be added to this structure. In addition, coding has to be added to an exit routine where the additional KPI data is selected from the extract table and fed it into the new LIPOVZ fields.

Based on this, Sean anticipates the following steps to be included in the functional specification that he will create for Christine:

1. Implement the customer exit routine.
2. Add new field entries (number of changes applied, delivery age, 24h change flag) to append structure LIPOVZ.
3. Add coding to derive the KPI data.
4. Add coding to calculate the delivery age.

Let's go through each of these steps in the following sections.

9.3.1 Implement the Customer Exit Routine

Christine’s first task is to create the actual customer exit routine. To do this she uses Transaction CMOD, where she creates a new project for this particular exercise, which serves as a project envelope for the customer exit. She names the new project “ZDLV_MON” and clicks the CREATE button.

On the next screen, Christine has to specify a description for this new project (she enters “Customer Exit for Delivery Monitor Enhancement” in the SHORT TEXT field). She clicks the SAVE button and then clicks the ENHANCEMENT ASSIGNMENTS button to navigate to the overview screen (see Figure 9.3).
According to the note Sean found on the SAP Marketplace (128150), Enhancement V50Q0001 should be used to include new fields in the delivery monitor. Christine enters the enhancement ID into the first column field and presses Enter to validate her entry. She then saves her work.

Now Christine clicks on the Components button. This is where she actually has to specify which implementation to use. In the case of this particular user exit, there is a choice between two different function module implementations. Per SAP Note 128150, Christine chooses the first function module (EXIT_SAPLV50Q_001) by double-clicking on it. This takes her straight to the function module editor (Transaction SE37) and displays the INCLUDE statement the user exit that was created for it by SAP developers (INCLUDE ZXV50QU01). She double-clicks on the INCLUDE ZXV50QU01 statement and receives a popup, asking whether she wants to create the object (see Figure 9.4).

By clicking the Yes button (and specifying the development class and transport request details), she is taken to the editor, which displays a new and empty function group Include (see Figure 9.5). This is where she will add her coding later on in Sections 9.3.3 and 9.3.4. The coding can't be completed now because Christine hasn't added the new fields to the append structure LIPOVZ yet. She therefore saves her work by clicking the Save button.
9.3.2 Add New Field Entries to Append Structure LIPOVZ

Next, Christine adds the three new fields into the report line item structure (as mentioned in SAP Note 128150), which are described in the functional specification she received from Sean. She now has to define these fields as shown in Table 9.1.
Creating New Data Elements

First, Christine defines new data elements for all three report fields, which will be used for the new fields in append LIPOVZ. She accesses Transaction SE11, enters “ZDEL_MON_CHNGES” on the initial selection screen, and clicks on the Create button to create a new data element.

She then assigns a short description for the new field (“Delivery Monitor – number of changes applied”). When it comes to defining the field type, Christine isn’t sure how many changes the new report field will have to handle. To be on the safe side, she specifies it as INT4 with a length of 10.

Finally, she assigns field labels by clicking on the Field Label tab for the new data element (see Figure 9.6) and then activates her new data element by clicking the Activate button.

Assigning New Fields and Data Elements to Append Structure LIPOVZ

Christine now adds new fields into append structure LIPOVZ by using the new data elements she has just created. In Transaction SE11, she selects structure LIPOVZ and clicks on the Change button. Figure 9.7 shows the field names and data elements she adds to the append structure by typing the field names into the Component column.
All three fields are typed with the data elements that she just created in the previous step (ZDEL_MON_CHNGES, ZDEL_MON_DELIVERY_AGE, and ZDEL_MON_DATE_CHANGE_FLAG).
After she has finished adding the three news fields, she clicks on the Activate button.

**Testing the New Fields in the Delivery Monitor (VL06O)**

Although Christine has not added any logic to populate her three new fields yet, she wants to test whether they appear in the report. She therefore starts Transaction VL06O and runs the report. In the ALV report screen, she selects the new fields from the ALV field selector popup (see Figure 9.8), which is displayed by clicking on the Change Layout button ().

![Figure 9.8 ALV Field Selector Displays Added New Fields](image)

Once selected, the values are also displayed in the report (albeit without values, because no coding has been added to populate them yet). Nevertheless, it's a pass for Christine's first test of the enhanced delivery monitor (see Figure 9.9).
9.3.3 Add Coding to Derive KPI Data and Calculate Delivery Age

Christine then goes back to Transaction SE37 where she displays function module EXIT_SAPLV500_001 and double-clicks on INCLUDE ZXV50QU01. She then goes into change mode by clicking on the Display/Change button. To populate the new LIPOV structure fields with the correct values, Christine adds the coding shown in Listing 9.1.

```
*------------------------------------------------------------------------*
* Include           ZXV50QU01                                          *
*------------------------------------------------------------------------*
DATA: ls_postab      TYPE lipov,
      lv_change_made TYPE abap_bool VALUE abap_false,
      lv_erdat       TYPE sy-datum,
      lv_prev_day    TYPE sy-datum,
      lt_kpi_data    TYPE STANDARD TABLE OF zkpi_data,
      ls_kpi_data    TYPE zkpi_data.

LOOP AT ct_postab INTO ls_postab.
  * Step 1 - Get KPI changes for particular delivery document
  SELECT *
    FROM zkpi_data
    INTO TABLE lt_kpi_data
    WHERE vbeln = ls_postab-vbeln.

  IF sy-subrc = 0 AND lines( lt_kpi_data ) > 0.
    ls_postab-dm_change_ctr = lines( lt_kpi_data ).
    lv_change_made = abap_true.
```
Implementing the Solution

* Step 2 - check whether change flag to be set

```plaintext
lv_prev_day = sy-datum - 1.  " date for yesterday
LOOP AT lt_kpi_data INTO ls_kpi_data
  WHERE kpi_date >= lv_prev_day.
  CASE ls_kpi_data-kpi_date.
    WHEN sy-datum.
      IF ls_kpi_data-kpi_time <= sy-uzeit.
        ls_postab-dm_change_24 = abap_true.
        lv_change_made = abap_true.
      ENDIF.
    WHEN lv_prev_day.
      IF ls_kpi_data-kpi_time >= sy-uzeit.
        ls_postab-dm_change_24 = abap_true.
        lv_change_made = abap_true.
      ENDIF.
  END_CASE.
ENDLOOP.
```

* Step 3 - calculate delivery age

* only interested in documents that have been picked,
* and goods issued

```plaintext
IF NOT ( ls_postab-koquk = 'C' AND  " conf. status
            ls_postab-kostk = 'C' AND  " picking status
            ls_postab-wbstk = 'C' ).  " GI status
  SELECT SINGLE erdat
    FROM likp
    INTO lv_erdat
    WHERE vbeln = ls_postab-vbeln.
  IF sy-subrc = 0.
    ls_postab-dm_delv_age = sy-datum - lv_erdat.
    lv_change_made = abap_true.
  ENDIF.
ENDIF.
```

* Step 4 - modify ct_postab table

```plaintext
IF lv_change_made = abap_true.
  MODIFY ct_postab FROM ls_postab.
ENDIF.
```
Christine’s coding loops around internal interface Table CT_POSTAB, which contains all report lines and has a LIPOV structure. Processing is separated into four steps as you can see in the code, with the first three steps populating each of the custom fields Christine added earlier.

**Step 1**
The process starts with a select on the KPI data table for the current delivery document. Items found are stored in internal Table LT_KPI_DATA. A `lines()` function is used to establish the number of lines in Table LT_KPI_DATA and store it in `LS_POSTAB-DM_CHANGE_CTR`, our first custom field. If the field value was set, then the `LV_CHANGE_MADE` flag is set, which will display the flag to indicate that the delivery was changed within the past 24 hours.

**Step 2**
The next step calculates yesterday’s date by subtracting 1 from today’s date. It then loops around all KPI data lines of yesterday (or today) and finds out whether the KPI-related change was made within the past 24 hours or not. If a change was made within the last 24 hours, then flag `DM_CHANGE_24` is set in structure `LS_POSTAB`. Again, flag `LV_CHANGE_MADE` is set.

**Step 3**
Christine’s code calculates the age of the delivery (i.e., establishes how many days have passed from the creation of the delivery to today). However, per Sean’s specification, the program also needs to filter out any documents that have already been completed, confirmed, and goods issued. For these documents, no delivery age needs to be calculated because they are regarded as completed by Byrell’s delivery process. Christine’s code checks three status fields (`koquk`, `kostk`, and `wbstk`) to ensure that only those documents are processed.

The code selects the creation date (field `ERDAT`) from delivery header Table LIKP, calculates the difference in days from today’s date (field `SY-DATUM`), and stores it in field `DM_DELV_AGE` of structure `LS_POSTAB`. Once again, flag `LV_CHANGE_MADE` is set.
Step 4

The customer exit checks whether LV_CHANGE_MADE is set to “true.” If this is the case, then the current CT_POSTAB is changed using structure LS_POSTAB.

9.3.4 Testing the Enhanced Delivery Monitor

Finally, Christine demonstrates the delivery monitor report to Sean, including the three new custom fields (see Figure 9.10).

As introduced in Chapter 8, the flexible setup of the new KPI solution now enables Byrell to introduce new extraction fields and criteria more easily. In return this saves time and money, because it isn’t necessary to change report or extract coding whenever a different field is to be reported on. In this chapter, Christine and Sean went one step further and introduced the KPI extract figures into a standard report by using a customer exit, thus eliminating the need for Byrell’s planners to merge data into spreadsheets or other external documents. Planning is now made easier and quicker as all data is brought into one place, delivery monitor VL06O.

9.4 Summary

This chapter showed how a user exit can be used to enhance an SAP standard report. When it comes to important standard reports such as the delivery monitor, SAP has provided various options to use standard reports and yet still add custom fields and processing. Too often, developers simply copy the existing report and create a Z or Y version of it, thereby risking the loss of new and additional functionality whenever the system is upgraded.
Another novel approach in this chapter was the way Sean actually came across the user exit. SAP Notes represent another way of finding out about exits and shouldn’t be overlooked. In your own enhancement research work, make it a habit to also check for any suitable SAP Notes—you might be surprised by what you find.

In the next chapter, you’ll learn how to introduce VOFM routines as a nifty and lightweight way to perform an invoice split.
Index

A

ABAP Debugger, 118
ABAP Editor, 55, 205
ABAP INCLUDE RV6OC901, 186
ABAP List Viewer (ALV), 146
  create report, 148
  line item structure, 165
  report screen, 171
ABAP-to-ABAP, 90
Access sequence, 123, 128
Accounting (exits), 239, 240, 241
Activate enhancement, 55
Additional Data tabs A and B, 62
Advantages of exits, 21
Append structure, 124
  LIPOVZ, 168
Application programming interface (API), 95
Availability check, 230
Billing, 83, 179
  interface (exits), 238, 239
  plan (exits), 240
  reference, 196
  reference field, 199
  type, 180
Billing header table
  VBRK, 180
Boolean variable, 58
Breakpoint, 200
BREAK statement, 116
Business add-in (see BAdI)
Business Application Program Interface (see BAPI)
Business intelligence, 145
Business Object Layer (BOL), 95

B

Backend, 101
BAdI, 20, 34, 37, 85
  breakpoint, 116
  classic, 37, 135
  create implementation, 140
  debug, 116
  filter, 149
  kernel-based, 37
  LE_SHP_BADI, 137
  LE_SHP_DELIVERY_PROC, 137, 148
  migrated, 138
  outbound, 113
  reuse implementation, 156
BADI_SD_BILLING, 88
BADI_SD_SALES, 38
BAPI, 95, 102
BASIS administrator, 91
Batch scheduling, 146
Best Practices, 217
CALL CUSTOMER-FUNCTION, 29
CALL FUNCTION...DESTINATION, 90
Class
  CL_CRM_BOL_CORE, 96
  CL_CRM_BOL_ENTITY, 96
  CL_RECA_GUID, 151, 161
  ZCL_LE_SHP_DLV, 156
Class Builder, 70
Classifications, 106
Client frontend applications, 109
CMOD, 25, 31
Code exit, 192
Combination criteria in the billing document, 180
Communication structure, 121, 123
Competitive advantage, 219
Components, 102
Component supply processing, 232
Condition record, 123
Condition type, 123, 129
Confirmations, 231
Contract data processing, 231
Copy control, 191
Copying requirements routine, 33
Create empty class, 141
Cross-reference table, 202
CRUD, 148
Custom container, 81
Custom domain element, 152
Customer-defined routines, 20, 32
Customer exit, 22, 85, 87, 163
create routine, 166
EXIT_SAPLV60A_002, 85
routine, 166
V50Q0001, 166
Customer service, 135
Custom field, 121
Customization, 217
Customizing, 20
Custom routine, 185
Custom table, 151, 155

D
Database table, 51, 64
  lock entries, 71
Data Dictionary, 201
Data Dictionary object, 65, 154
Data elements, 66
  create new, 169
  ZKPI_EL_TABLE, 152
Data model class, 68
Data transfer routine, 33, 182, 186
Debugging, 116
Declaration, 55
Define number range for billing documents, 201
Delivery block, 135
delivery_final_check(), 141
Delivery (IDoc exits), 236, 237
Delivery monitor, 163
  enhanced, 175
  new fields, 171
Delivery monitor (exits), 237
Delivery processing (exits), 233, 234
Delivery processing (user exits), 232, 233
Delivery-related user exits, 232
Delivery processing (BAdIs), 234, 235, 236
DEQUEUE_EZ_SURVEY, 80
Design by contract, 103
Destination system, 99
Disruption, 34
Duty, 190
  item, 181

E
ECC_SALESORDER010QR, 111
END ENHANCEMENT, 39
END ENHANCEMENT-SECTION, 36
Enhancement Framework, 20, 34, 140
Enhancement ID, 36
Enhancement implementation, 97
Enhancement point, 35, 55
  implementation, 54, 97
Enhancement section, 36
ENHANCEMENT-SECTION, 36
Enhancement spot, 32, 36
  LE_SHP_DELIVERY_PROC, 139
Enhancement V50Q0001, 167
Enterprise services, 101, 102, 104
Error exception, 95
Error log, 90
Establish delivery type, 206
Exception, 95
EXECUTE_SYNCHRONOUS, 108, 113
Exit, 20
Explicit enhancement, 35, 212

F
Fast Moving Consumer Goods (FMCG), 179
Field validation, 54
Filter, 38
Filter out pricing data, 111
Foreign key relationship, 67
Form exit, 22
Form subroutine, 22
Formula, 184
  routine, 34
Functional consultant, 20
Functionality, 102
Functional specification (FS), 48
Index

Function group, 94
Function module, 167
  editor, 167
  ENQUEUE_EZ_SURVEY, 80
  EXIT_SAPLV50Q_001, 172
  GUID_CREATE, 151
  L_MC_TIME_DIFFERENCE, 76
  parameters, 95

G

Generate table maintenance, 52
Generator settings, 70
Global declarations, 80
Globally Unique Identifier (GUID), 151, 161

H

Hard coding, 59, 182, 190

I

IDoc, 108
Implementation Guide (IMG), 22, 179, 191
Implicit enhancement, 38, 127, 186, 212
  create, 73, 204
  custom coding, 190
Import parameters, 95
Include
  MV45AFZZ, 50, 76, 127
  MV45AI0Z, 76
  MV45A002, 73
  MV45ATOP, 54, 81
  MV45ATZZ, 54
  MV50AFZ1, 136, 148
  RV60CNNN, 186
INSERT, 65
Interface attributes, 113
Internet customer orders, 121
Interval, 202
Introduction, 19
Invoice
  header, 182
  separate, 181
Invoice split, 180, 183
delivery based, 180
Invoicing, 238

K

Key combinations, 161
Key performance indicator (KPI), 145
Key users, 204
KPI, 163
coding, 160
data table, 150
derive data, 172
derive fields, 160
extract data, 161
extract scenarios, 160
extract system, 161
field layout, 150
field table, 150
populate data structure, 161

L

Layers, 35
lines(), 174
LIPOV, 165, 174
Lock object, 71
Logon and security, 92
Loose coupling, 103

M

Maintenance view, 204
Material determination, 227
Menu exit, 27
Menu Painter, 27
Mind map, 20, 43
Modification, 22, 34, 215
Modification-free, 25
MODIFY, 161
Multilevel enhancement landscape, 35
Multiple use, 139
### N

- Namespace, 24, 42
- Non-duty, 190
  - item, 181
- Numbering range, 197, 200
  - amend, 207
- Number range, 200

### O

- Object attributes, 97
- Off-the-shelf, 20
- Operations, 101
- Order value category, 48, 51
- Outbound delivery monitor, 163
- Outbound delivery orders and groups (exits), 237
- Outbound parameter, 116
- OUTBOUND_PROCESSING, 114

### P

- Package, 22
- PAI, 66, 71
- Parameter transaction, 52
- Partner determination, 227
- Payment cards, 231
- Payment terms, 47, 49
- PBO, 66, 71, 75
- Persistency class, 68, 82
- Persistent class, 68
- Persistent object, 61, 65
- PHP, 110
- Picking volume, 47
- Picking waves (exits), 238
- Pricing, 121
- Pricing (BAdIs), 229
- Pricing (exits), 228
- Product allocation, 230
- Product selection, 229
- Provider class, 112
- PSTYV, 190

### R

- Real-life scenario, 43
- Register object, 125
- Remote, 90
- Remote function call (RFC), 90
- Repository elements, 34
- Repository Infosystem (SE80), 31, 84, 85, 105, 148, 181, 213
  - search pattern, 149
- Requirements, 184
- Requirements routine, 33
- Responsible enhancements, 218
- Return on investment (ROI), 217
- Reusing BAdI implementations, 156
- RFC, 90, 93
  - connections, 91
  - connection setup, 93
  - create connection, 91
  - function module, 94, 97
  - insert, 90
  - module, 87
- RFC-enabled function module, 90
- Routine, 184
  - copy, 184
  - data transfer, 183
- RV60AFZC, 181, 199
- RV60AFZZ, 201
- RV_INVOICE_CREATE, 87, 88
- RV_INVOICE_REFRESH, 93

### S

- Sales activities, 27
- Sales district, 197
- Sales documents, 223, 231
- Sales order, 47
- SalesOrderByIDQueryResponse_In, 107
- Sales support, 231
- SAP Business Suite, 34
- SAP CRM, 83, 90
  - activity, 83, 87, 97
- SAP ERP, 34
- SAP internal, 88
- SAP Marketplace, 72
SAPMV45A, 63, 66, 72
SAP NetWeaver, 151
SAP Note, 165, 213, 226
SAP release 4.6, 21
SAP Service Marketplace, 165, 215
SAVE_AND_PUBLISH_DOCUMENT, 149, 156
save_document_prepare(), 141
Screen 8309, 63, 66, 75
Screen 8310, 63
Screen exit, 27
Screen flow, 71
SD
  best practices, 19
  cost efficiency, 19
  customer modifications, 198
  stability, 19
  upgradability, 19
SELECT, 65
Semantic interoperability, 103
Service consumers, 102
Service-oriented architecture (SOA), 101, 106
Service provider, 102
Shipment costs, 246
Shipment costs (BAdIs), 247, 248
Shipment costs (exits), 246, 247
SOAP messages, 110
Software Development Kit (SDK), 219
SSCR suite, 125
Standard code, 21
Standard processing (BAdIs), 226
Standard processing (exits), 223, 224, 225
Standard software, 19
Switchable, 25
Switch Framework, 34, 40, 140
Syntactic (or technical) interoperability, 103

Table (Cont.)
  XVBFS, 99
  XVBPA, 99
Table maintenance dialog program, 154
Table types, 67
Tabs, 61
Technical and functional consultants, 217
Text determination, 229
Trade-offs, 43
Transaction
  CMOD, 166
  CV01N, 27
  SE11, 66, 123, 151, 155, 159, 203
  SE37, 94, 167, 172
  SE38, 186
  SE54, 52, 154
  SE80, 68, 84, 104, 112, 137, 148, 181, 198, 204
  SE93, 52
  SM30, 54, 155
  SM37, 146
  SM59, 91
  SMOD, 30
  SPAU, 215
  SPAU_ENH, 215
  SPRO, 128
  V/03, 123, 126
  VA01/02/03, 49, 61
  VF01/02/03, 196
  VK11, 129
  VLO1N, 136
  VLO6O, 163, 171
  VMOD, 22
  VOFM, 32, 182
  VTFL, 191
  ZXVVFU08, 26
Transport document (exits), 242, 243, 244, 245
Transport documents, 242
Transport documents and handling units (BAdIs), 245
Transport request, 185

U

Universal Description, Discover and Integration (UDDI), 103
Index

User exit, 22, 49
USEREXIT_FILL_VBRK_VBRP, 39, 182, 199
USEREXIT_MOVE_FIELD_TO_VBKD, 50, 56
USEREXIT_NUMBER_RANGE, 200
USEREXIT_PRICING_PREPARE_TKOMK, 123
USEREXIT_SAVE_DOCUMENT, 148
USEREXIT_SAVE_DOCUMENT., 76
USERXIT_SAVE_DOCUMENT, 65
US_RANGE_INTERN, 208

W

Warehouse, 47
Web services, 102, 104
Web Services Description Language (WSDL), 103
Where-used search, 87
Wrapper, 140
WSDL, 106

V

Validation, 145
Variable declarations, 54
Variant configuration, 229
VBAK, 62, 123
VBKD-ZTERM, 49
VBRK-BZIRK, 203
VBRK-XBLNR, 196
VK11, 123
VMOD, 198
VOFM, 184
routine, 179, 212

X

XML Editor, 117
XML response message, 110

Z

ZTERM, 51