

BC420

Data Migration

SAP NetWeaver

Date _____
Training Center _____
Instructors _____
Education Website _____

Participant Handbook

Course Version: 62
Course Duration: 5 Day(s)
Material Number: 50085260



An SAP course - use it to learn, reference it for work

Copyright

Copyright © 2007 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Trademarks

- Microsoft®, WINDOWS®, NT®, EXCEL®, Word®, PowerPoint® and SQL Server® are registered trademarks of Microsoft Corporation.
- IBM®, DB2®, OS/2®, DB2/6000®, Parallel Sysplex®, MVS/ESA®, RS/6000®, AIX®, S/390®, AS/400®, OS/390®, and OS/400® are registered trademarks of IBM Corporation.
- ORACLE® is a registered trademark of ORACLE Corporation.
- INFORMIX®-OnLine for SAP and INFORMIX® Dynamic ServerTM are registered trademarks of Informix Software Incorporated.
- UNIX®, X/Open®, OSF/1®, and Motif® are registered trademarks of the Open Group.
- Citrix®, the Citrix logo, ICA®, Program Neighborhood®, MetaFrame®, WinFrame®, VideoFrame®, MultiWin® and other Citrix product names referenced herein are trademarks of Citrix Systems, Inc.
- HTML, DHTML, XML, XHTML are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.
- JAVA® is a registered trademark of Sun Microsystems, Inc.
- JAVASCRIPT® is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.
- SAP, SAP Logo, R/2, RIVA, R/3, SAP ArchiveLink, SAP Business Workflow, WebFlow, SAP EarlyWatch, BAPI, SAPPHIRE, Management Cockpit, mySAP.com Logo and mySAP.com are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other products mentioned are trademarks or registered trademarks of their respective companies.

Disclaimer

THESE MATERIALS ARE PROVIDED BY SAP ON AN "AS IS" BASIS, AND SAP EXPRESSLY DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR APPLIED, INCLUDING WITHOUT LIMITATION WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WITH RESPECT TO THESE MATERIALS AND THE SERVICE, INFORMATION, TEXT, GRAPHICS, LINKS, OR ANY OTHER MATERIALS AND PRODUCTS CONTAINED HEREIN. IN NO EVENT SHALL SAP BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, CONSEQUENTIAL, OR PUNITIVE DAMAGES OF ANY KIND WHATSOEVER, INCLUDING WITHOUT LIMITATION LOST REVENUES OR LOST PROFITS, WHICH MAY RESULT FROM THE USE OF THESE MATERIALS OR INCLUDED SOFTWARE COMPONENTS.

About This Handbook

This handbook is intended to complement the instructor-led presentation of this course, and serve as a source of reference. It is not suitable for self-study.

Typographic Conventions

American English is the standard used in this handbook. The following typographic conventions are also used.

Type Style	Description
<i>Example text</i>	Words or characters that appear on the screen. These include field names, screen titles, pushbuttons as well as menu names, paths, and options. Also used for cross-references to other documentation both internal (in this documentation) and external (in other locations, such as SAPNet).
Example text	Emphasized words or phrases in body text, titles of graphics, and tables
EXAMPLE TEXT	Names of elements in the system. These include report names, program names, transaction codes, table names, and individual key words of a programming language, when surrounded by body text, for example SELECT and INCLUDE.
Example text	Screen output. This includes file and directory names and their paths, messages, names of variables and parameters, and passages of the source text of a program.
Example text	Exact user entry. These are words and characters that you enter in the system exactly as they appear in the documentation.
< Example text >	Variable user entry. Pointed brackets indicate that you replace these words and characters with appropriate entries.

Icons in Body Text

The following icons are used in this handbook.

Icon	Meaning
	For more information, tips, or background
	Note or further explanation of previous point
	Exception or caution
	Procedures
	Indicates that the item is displayed in the instructor's presentation.

Contents

Course Overview	vii
Course Goals	vii
Course Objectives	vii
Unit 1: Data Migration.....	1
Data Migration.....	2
Unit 2: Batch input.....	23
Batch input.....	24
Unit 3: Basic information about DX-WB	63
DX-WB Basics	64
Unit 4: Other DX-WB Functions	99
Other DX-WB Functions	100
Unit 5: Sequential Files.....	123
Sequential Files	124
Unit 6: Legacy System Migration Workbench.....	163
LSMW - Legacy System Migration Workbench	164
Unit 7: Direct Input	225
Direct Input.....	226
Unit 8: Data Transfer Using IDocs	239
Data Transfer Using IDocs.....	240
Unit 9: BAPI	283
BAPI	284
Unit 10: TA Recorder	321
TA Recorder	322
Unit 11: Background Processing and Special Techniques....	375
Background Processing.....	376

Unit 12: Appendix I	401
Appendix I.....	402
Unit 13: Appendix II	411
Appendix II.....	412
Appendix 1: Tasks	437

Course Overview

Main Business Scenario



- When you implement an SAP system, you should transfer the data from the legacy system/external system.
- You should transfer the following objects, for example:
 - Customer data
 - FI Documents
 - Fixed assets

Target Audience

This course is intended for the following audiences:

- Developers in the data transfer area
- SAP consultants and partners who work in the data transfer area

Course Prerequisites

Required Knowledge

- Basic technological knowledge of the SAP NetWeaver Application Server (SAPTEC/SAP50)
- ABAP programming knowledge (BC400)
- You do not need to have experience in the data transfer area.



Course Goals

This course will prepare you to:

- Use the most important techniques and tools in the data transfer area.



Course Objectives

After completing this course, you will be able to:

- Use the following techniques to transfer data:
- Use batch Input
- Use call transaction
- Use direct input
- Use IDocs

- Use BAPIs
- Use the Legacy System Migration Workbench (LSMW) as a tool
- Use the data transfer workbench (DX-WB) as a tool

SAP Software Component Information

The information in this course pertains to the following SAP Software Components and releases:

- SAP NetWeaver 7.0

Unit 1

Data Migration

Unit Overview

This chapter describes problems and possible solutions during the data transfer to the SAP system.



Unit Objectives

After completing this unit, you will be able to:

- Describe how data is transferred to an SAP system
- Name the methods and tools available for data transfer

Unit Contents

Lesson: Data Migration	2
Exercise 1: Overview of Data Transfer	17

Lesson: Data Migration

Lesson Overview

- Task
- Methods
- Data Transfer Technology



Lesson Objectives

After completing this lesson, you will be able to:

- Describe how data is transferred to an SAP system
- Name the methods and tools available for data transfer

Business Example

A company wants to transfer external data to an SAP system as part of a migration. In addition, this company is planning a permanent data transfer to this SAP system. It wants to evaluate all the options for this permanent data transfer and to implement the most suitable solution option.

Overview of the Topics/Unit and the Course Structure



Figure 1: Overview of Data Transfer (1)

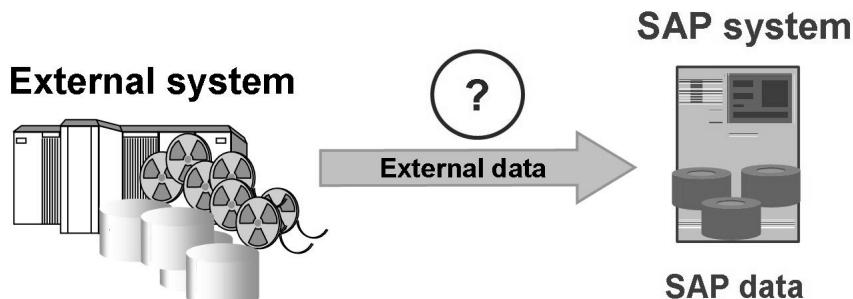


Figure 2: Transferring Data into the SAP System

Large datasets cannot be transferred manually from an external system into the SAP System. This would be unacceptable to a company for economical reasons. Therefore you need a method of transferring the data automatically in the background.

Data transfer is necessary if:

- You want to transfer data from an external system into the SAP system when it is installed.
- Data is transferred on a regular basis from external systems into the SAP system.

Example: If you have input data using your external system which needs to be integrated into the SAP System.

Therefore, you have to find a way of transporting the data into the SAP system or the database.

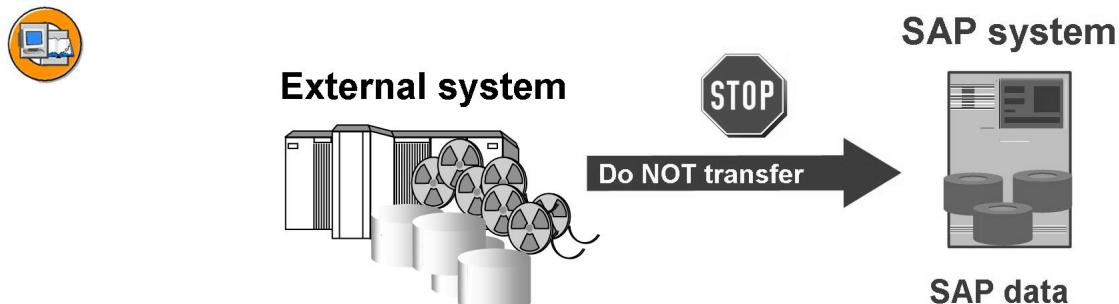


Figure 3: Transfer Data Directly to the DB?

To ensure data integrity, you cannot import data from your external system directly into the SAP System database.

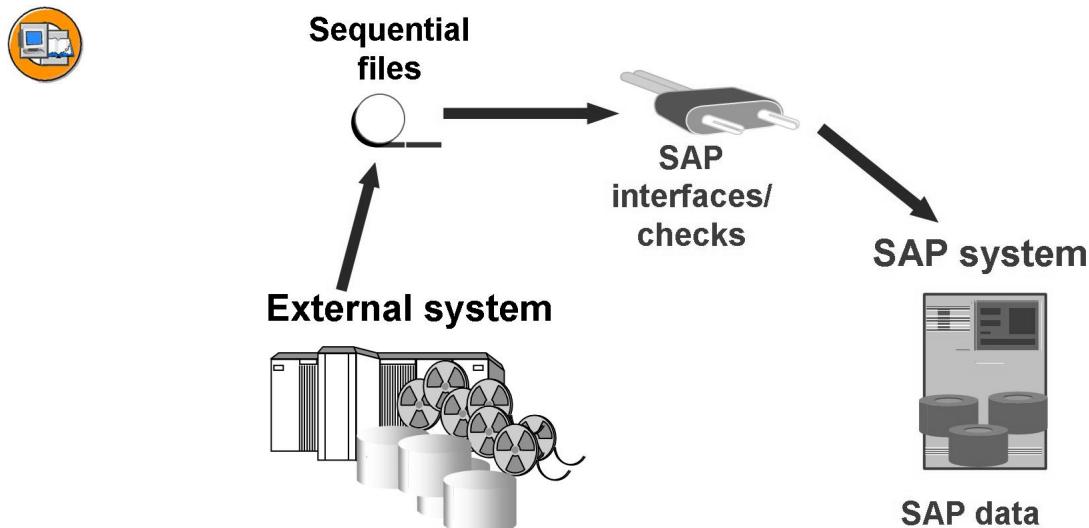


Figure 4: Data Transfer Using Interfaces

When transferring data into one SAP System from another SAP System or from an external system, you must ensure the integrity of the data. Thus, it is necessary to subject this type of data transfer to the same checks as data transfer in dialog mode.

It is difficult to execute the online checks in the transactions yourself because they are extensive and partly apply to more than one application.

SAP provides appropriate interfaces for data transfer. To some extent these use transactions and their checks to transfer data into the SAP System.

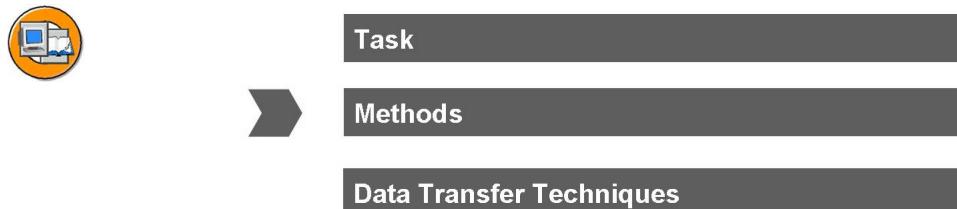


Figure 5: Overview of Data Transfer (2)

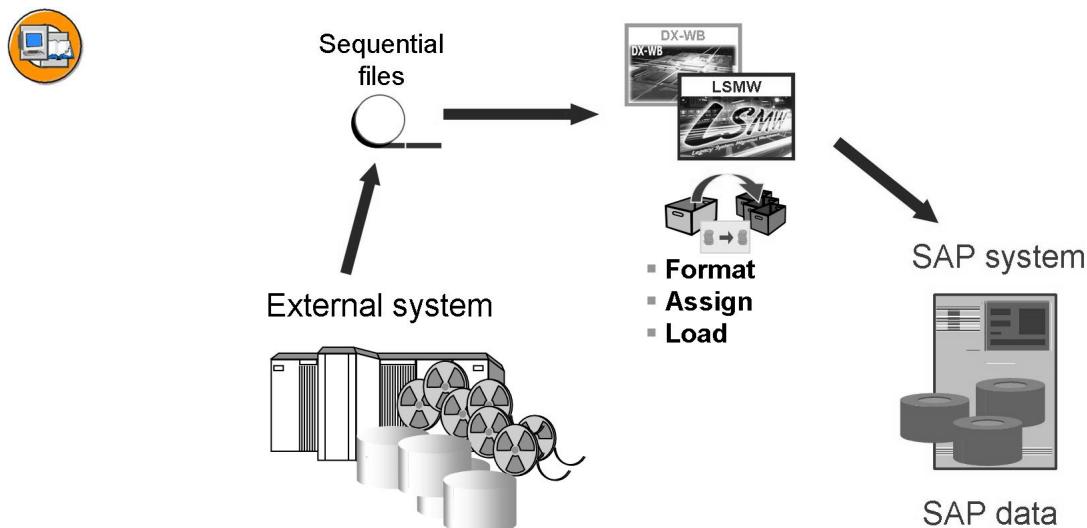


Figure 6: Data Transfer Tools and Tasks (I)

The following tools are available for transferring data into one SAP System from another SAP System or from an external system:

- The Data Transfer Workbench, also called DX-WB
- The Legacy System Migration Workbench, also called LSMW

These tools do not actually transfer data themselves. They act as the central point of access and make the transfer tasks easier for the actual transfer programs of the applications.

Before data can be transferred to the SAP system, the external data is must be prepared. The data has to be formatted, assigned, and loaded or imported. These steps are now described in detail.

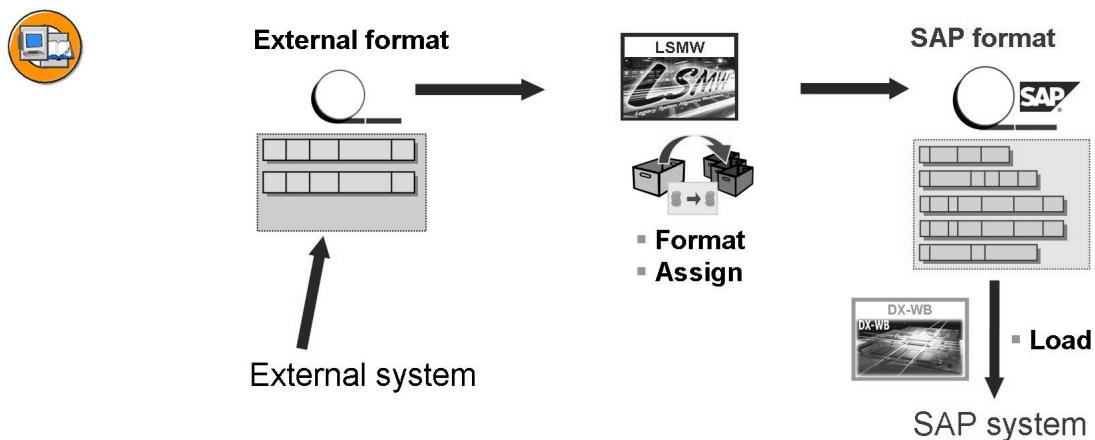


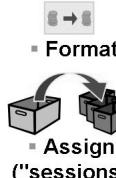
Figure 7: Data Transfer Tools and Tasks (II)



External format

\$	10,0
E	242,5

External system



10.00	USD
242.50	EUR

SAP system

Formatting and assignment = conversion

Figure 8: Conversion Process

Mostly, you have to determine the valid value sets of the SAP data fields and convert the external data to the new valid value set. This is known as **“formatting”** the data. For example, the currency symbol “\$” in the external system is formatted to “USD” in the SAP system.

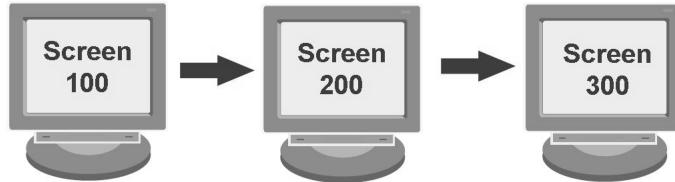
With data transfer, one of the most important tasks is to establish which data fields in the external system correspond to which data fields in the SAP System. This is known as **“assigning”** (or “mapping”) the data.

Method: Establish how the existing data should be mapped to the SAP data structure. Establish how the data has to be mapped to the SAP data structure.

- Establish the fields that can be transferred directly from the data. These fields correspond directly to the existing data field and its SAP data field.
- Establish which fields in the external system must be adapted (data type and data length converted) before they can be transferred to the SAP System.
- Check what data types and data lengths have to be converted.



SAP transaction



- Execute
- Analyze
- Determine mandatory fields
- Is Customizing completed?

Figure 9: Conversion Help: Analyzing the SAP Transaction

The SAP transactions support the conversion process by helping to identify the relevant fields.

The following information is required:

- Transaction code, if this is not known already
- Fields required for the input
- Fields for which default values can be used
- Name, type and length of the fields that are used in the transaction.



Caution: For some data transfer techniques the transfer logic is not important; so always find out beforehand about the online transaction you are using.



Task

Methods



Data Transfer Techniques

Figure 10: Overview of Data Transfer (3)

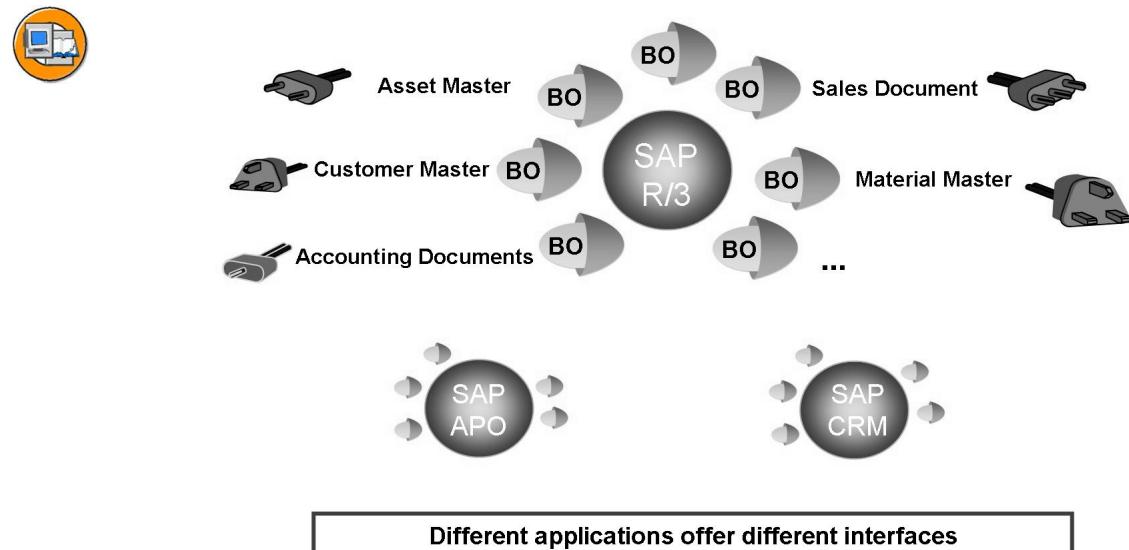


Figure 11: Data Transfer Objects

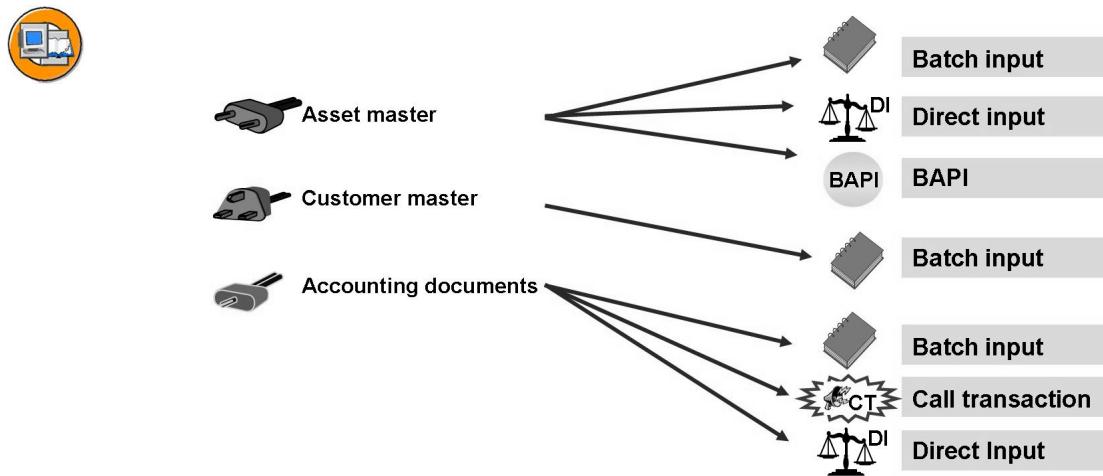
The Business Object Repository (BOR) is very important for using data transfer tools.

The BOR is the central repository in the SAP System, and contains all SAP business object types, SAP interface types, their definitions and methods, and so on. The BOR identifies and describes the available SAP business object types and interface types, and their BAPIs.

For example, if you are developing an application program, in the BOR you will find all the information about SAP business object types, SAP interface types, their key fields and their BAPI methods. The BOR contains all the information you need to use the correct object type definitions and BAPI calls in your application program.

As of 4.6 all transfer programs are organized using the BOR objects relevant to the application.

The data transfer objects in the DX-WB orient themselves by the business object numbers and names in the BOR.



Different data transfer techniques can be used in each application.

Figure 12: Overview: Data Transfer Technology

In the applications there are different data transfer interfaces. You can use different techniques for each application and interface. For example, batch input is a widely used technique for transferring external data and is used in a lot of applications.

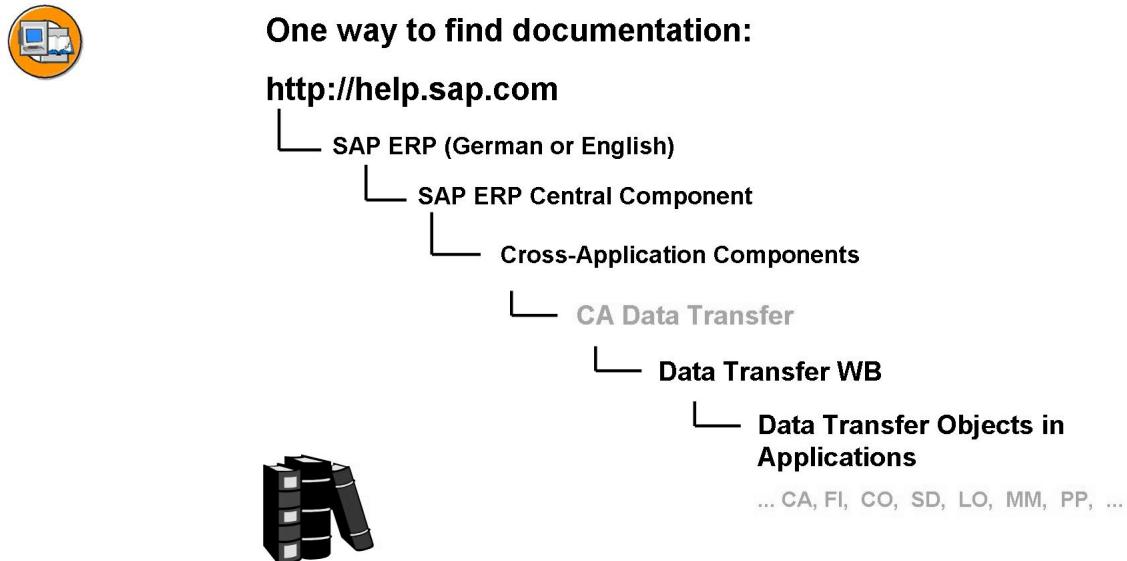


Figure 13: Finding Documentation for Data Transfer

In addition to the path described in the graphic, you can also find documentation for the data transfer for each SAP release by following the menu path *Help → Application Help* from the DX-WB.

You can also find documentation for the data transfer in the applications on the Internet under the following link:

help.sap.com →

[SAP ERP \(German or English\)](http://help.sap.com/SAP_ERP) →

[SAP ERP Central Component](http://help.sap.com/SAP_ERP_Central_Component) →

[Cross-Application Components](http://help.sap.com/Cross_Application_Components) →

[CA Data Transfer](http://help.sap.com/CA_Data_Transfer) →

[Data Transfer Workbench](http://help.sap.com/Data_Transfer_Workbench) →

[Data Transfer Objects in Applications](http://help.sap.com/Data_Transfer_Objects_in_Applications)

Batch input

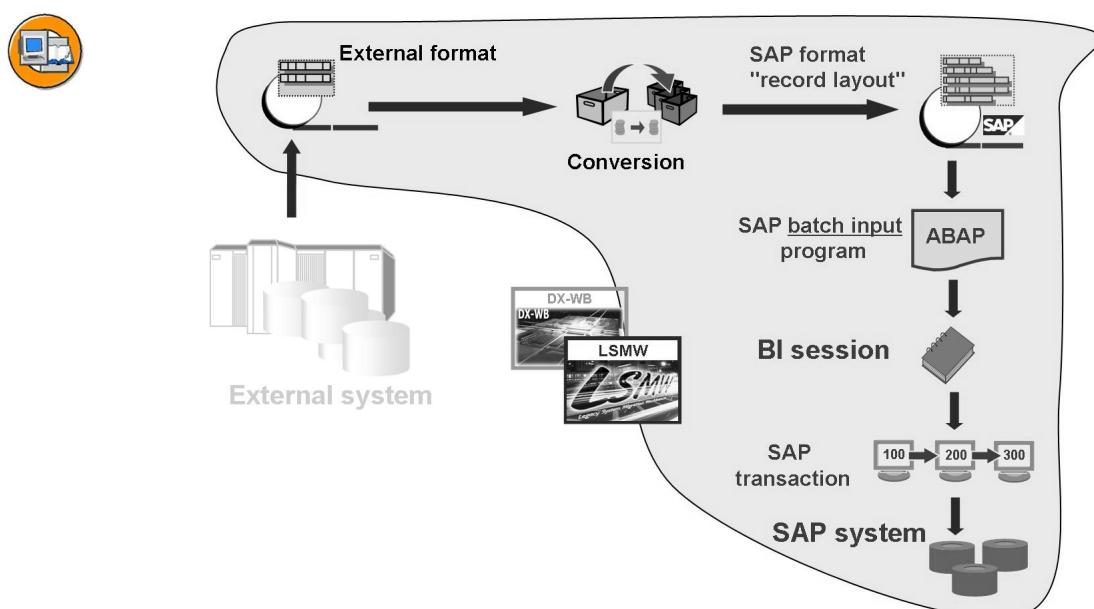


Figure 14: Batch-Input Technique

Batch input is a standard technique for transferring data into the SAP system. Transactions are triggered automatically to update data. The advantage of this technique is that all transaction checks are carried out. This ensures data remains consistent.

Batch input runs in two phases:

1. A batch input session containing all the relevant data is created.
2. This batch input session is processed and the data contained in it is imported into the SAP system using transactions.

A large number of the SAP standard data transfer programs use the batch input technique.

Call Transaction

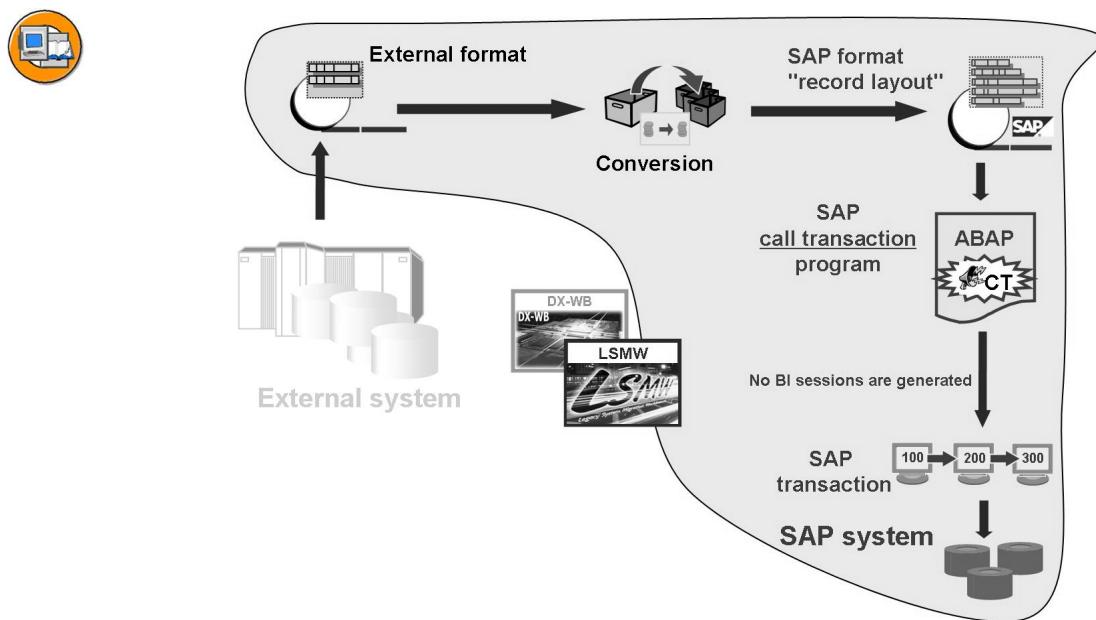


Figure 15: The Call Transaction Technique

Batch input is a standard technique for transferring data into the SAP system. As with batch input transactions are also used here to update the data. But with call transaction no batch input sessions are generated. Here transactions are triggered directly from the transfer program.

Call transaction updates the data directly using the transactions.

This technique offers the same functions as with batch input. No BI sessions are generated. Error handling is discussed later in the course.

Direct Input

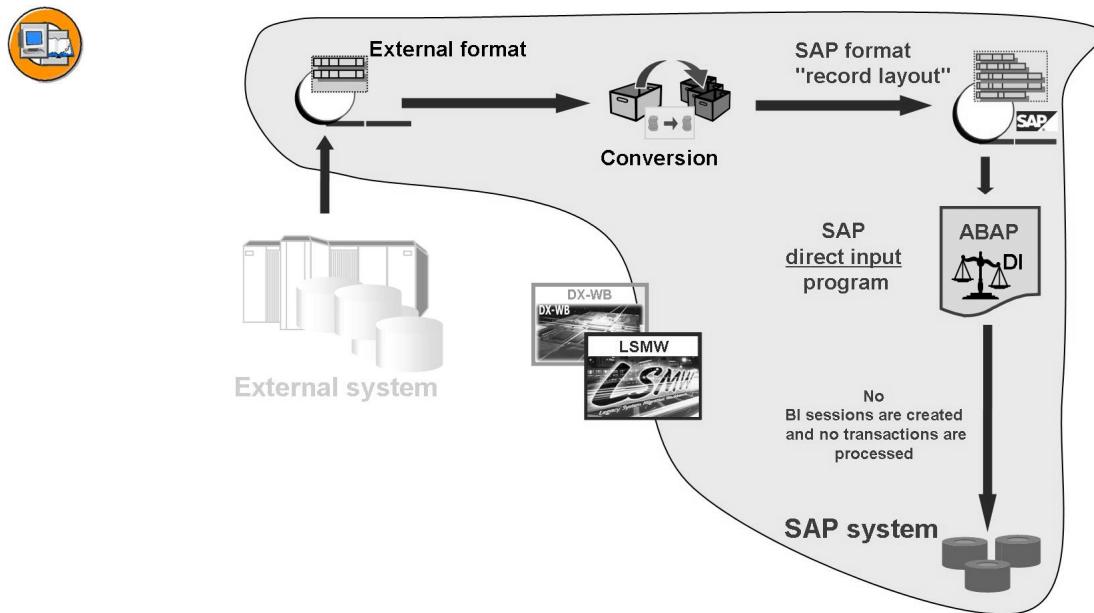


Figure 16: Direct Input Technique

With direct input all necessary checks are carried out on the data in the data transfer file, and then this data is transferred directly into the SAP system. The SAP database is immediately updated with the new data.



Hint: The data is imported and checked by a program that performs the same checks as the actual online transaction. This ensures the integrity of the data.

Transactions are not used for direct input.

IDoc

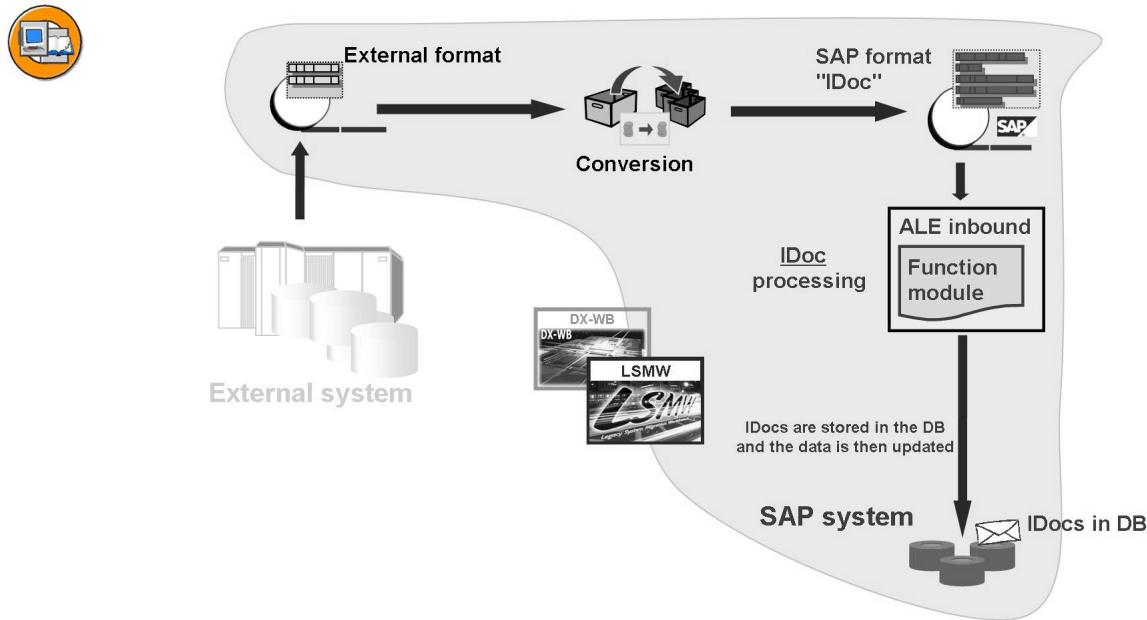


Figure 17: The IDoc Technique

IDoc is an SAP standard format for data transfer between systems. “IDoc” stands for *intermediate document*. One of the uses of IDocs is within ALE (an SAP concept for distributing business processes among autonomous SAP Systems). ALE is used to exchange data between different application systems.

Procedure:

- Data already converted to the IDoc format is passed to ALE inbound processing. An IDoc is created in the database for each data record (for example, an FI document).
- The created IDocs are read one by one and the inbound function module associated with each one is called. This checks the data and if all OK writes it to the application tables. The data (for example, an FI document) is created.

Business Application Programming Interfaces

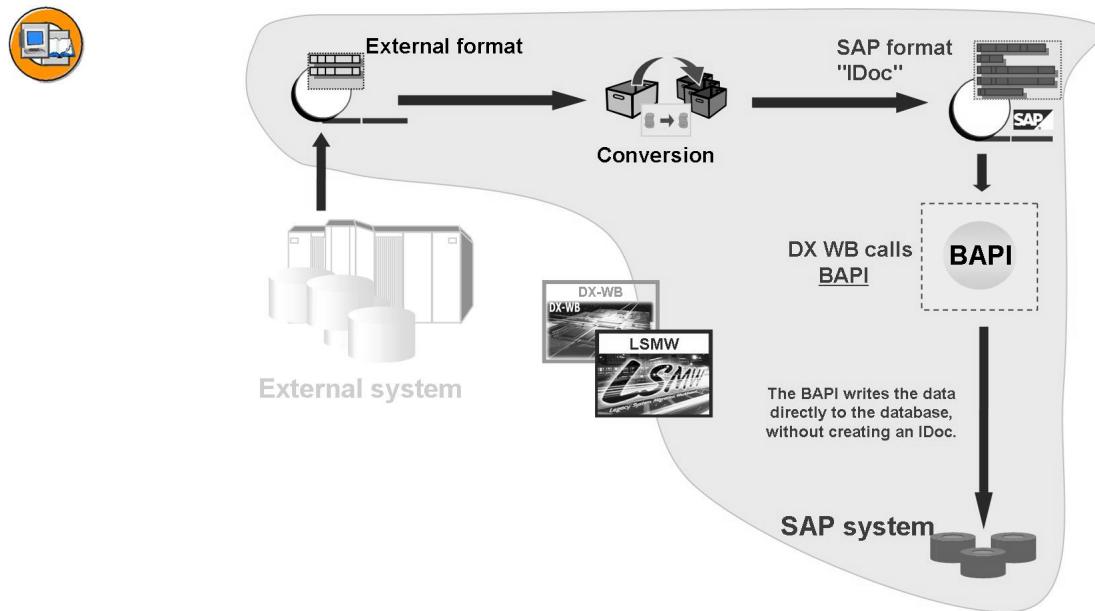


Figure 18: The BAPI Technique

Business Application Programming Interfaces (BAPIs) are standardized programming interfaces that provide external applications with access to business processes and data within the SAP System.

They are defined in the BOR as methods of SAP Business Objects or SAP interface types and give you an object-orientated view over the business components (application components) of the SAP system.

When an application BAPI is called, the data is transferred.

With this technique integrity checks are also performed online.

Transaction Recorder

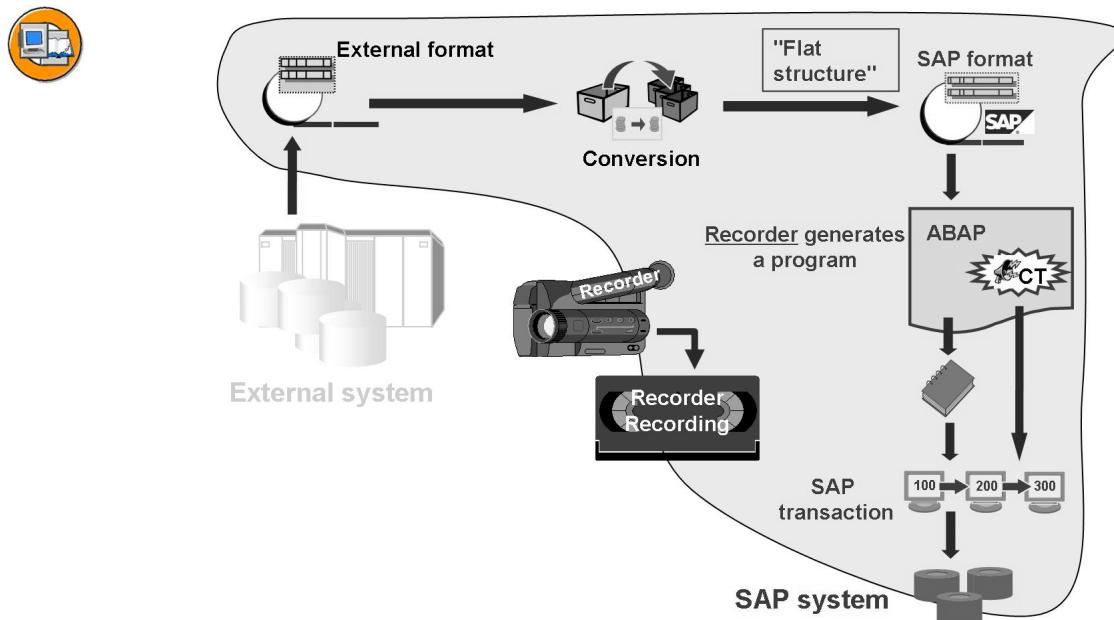


Figure 19: Customer Developments Using the Transaction Recorder

With the transaction recorder you can record a sequence of transactions including all the screens that are run through.

You can then use the recording to generate a data transfer program that imports the data into the SAP system.

The program can execute the data transfer using batch input or call transaction.

This procedure is particularly suitable for customers' own developments.

Exercise 1: Overview of Data Transfer

Exercise Objectives

After completing this exercise, you will be able to:

- Outline the SAP transactions that can be used to transfer customer data online.

Business Example

In the online transaction FD01, create customers with external number assignment in the SAP system.



Hint: (## is your group number)

Customers: Z-##-00001 and Z-##-00002

Pay particular attention to the sixth place. This should be 0 for this exercise. For each group, there are many additional customers that are required for subsequent exercises. These are all created with external number assignment.

Task 1:

To prepare the data transfer for customers with an external number assignment, take a look at the SAP transaction which is used to create customers in financial accounting. You should fill all required fields with values.

- Open SAP transaction FD01.
- On the initial screen, choose **General customers** or **KUNA** for the *Account Group* field. In the *Customer* field enter the customer number **Z-##-00001**. In the *Company Code* field enter **0001**. The fields below Reference must not be filled. Choose *Enter* to display the next screen.
- On this screen (Create Customer: General Data), enter all the required address data on the tab page *Address*. Enter a value in the field *Postal Code*. You can choose *DE* or *US* for the country. In the Communication field, enter the appropriate language for the country.
- Go to the tab page *Control Data*. Under control information, enter the sales tax ID if this customer requires one (this depends on the country of the customer). For customers from Germany, this is **DE123456789**.
- Go to the company code data (pushbutton: Company Code Data). Under Account Management, enter the value **120000** for the reconciliation account.

Continued on next page

6. To save the customer data, choose Save.
7. Enter a second customer with the number **Z-##-00002**.

Task 2:

1. You can use transaction FD03 (Display Customers: Accounting Initial Screen) to check the customers you have created.

Solution 1: Overview of Data Transfer

Task 1:

To prepare the data transfer for customers with an external number assignment, take a look at the SAP transaction which is used to create customers in financial accounting. You should fill all required fields with values.

1. Open SAP transaction FD01.
 - a)
2. On the initial screen, choose **General customers** or **KUNA** for the *Account Group* field. In the *Customer* field enter the customer number **Z-##-00001**. In the *Company Code* field enter **0001**. The fields below Reference must not be filled. Choose *Enter* to display the next screen.
 - a)
3. On this screen (Create Customer: General Data), enter all the required address data on the tab page *Address*. Enter a value in the field *Postal Code*. You can choose *DE* or *US* for the country. In the Communication field, enter the appropriate language for the country.
 - a)
4. Go to the tab page *Control Data*. Under control information, enter the sales tax ID if this customer requires one (this depends on the country of the customer). For customers from Germany, this is **DE123456789**.
 - a)
5. Go to the company code data (pushbutton: Company Code Data). Under Account Management, enter the value **120000** for the reconciliation account.
 - a)
6. To save the customer data, choose Save.
 - a)
7. Enter a second customer with the number **Z-##-00002**.
 - a)

Task 2:

1. You can use transaction FD03 (Display Customers: Accounting Initial Screen) to check the customers you have created.
 - a)



Lesson Summary

You should now be able to:

- Describe how data is transferred to an SAP system
- Name the methods and tools available for data transfer



Unit Summary

You should now be able to:

- Describe how data is transferred to an SAP system
- Name the methods and tools available for data transfer

Unit 2

Batch input

Unit Overview

The first transfer technique demonstrated is the batch input technique for the customer master data procedure. The batch input monitor is used as the tool. SAP standard transfer procedures mostly work with batch input sessions. The input format of these procedures is record layout structures



Unit Objectives

After completing this unit, you will be able to:

- Describe the structure of batch input sessions
- Process and analyze batch input sessions using the batch input monitor
- Explain what the SAP standard method is and how record layout files are used

Unit Contents

Lesson: Batch input	24
Exercise 2: Working with Batch Input Sessions	55
Exercise 3: Batch Input Sessions and Error Handling.....	57

Lesson: Batch input

Lesson Overview

- Batch Input Sessions
- Batch Input Monitor



Lesson Objectives

After completing this lesson, you will be able to:

- Describe the structure of batch input sessions
- Process and analyze batch input sessions using the batch input monitor
- Explain what the SAP standard method is and how record layout files are used

Business Example

A company wants to use batch input for a data transfer.

Batch Input Basics



Batch Input Basics

Create batch input session

Batch Input Monitor

SAP Standard Procedure "Record Layout"

Figure 20: Basics of Batch Input Processing (1)



- External data is converted to the SAP format and stored in a file in SAP format.
- A data transfer program reads the converted data and creates batch input sessions from it.
- A batch input session contains data prepared for SAP transactions.
- The data from the batch input sessions is passed to transactions.
- Once the transactions have been successfully processed, the data in the SAP database is consistent.

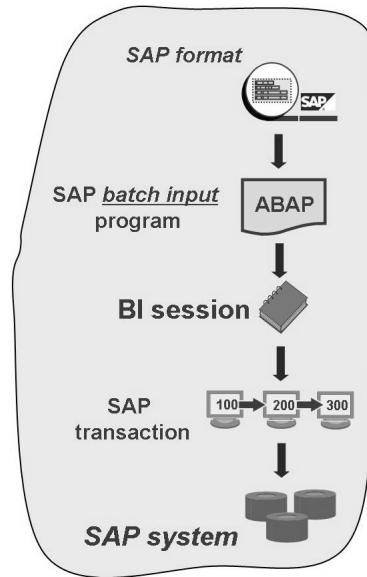


Figure 21: Batch Input Technique

Batch input is a standard technique for transferring data into the SAP system. The transaction flow is simulated and the data transferred as though it had been entered online. The advantage of this technique is that all transaction checks are carried out. This ensures data remains consistent.

Batch input runs in two phases:

1. A batch input session containing all the relevant data is created.
2. This batch input session is processed and the data contained in it is imported into the SAP system.

The majority of the SAP standard data transfer programs use the batch input technique.

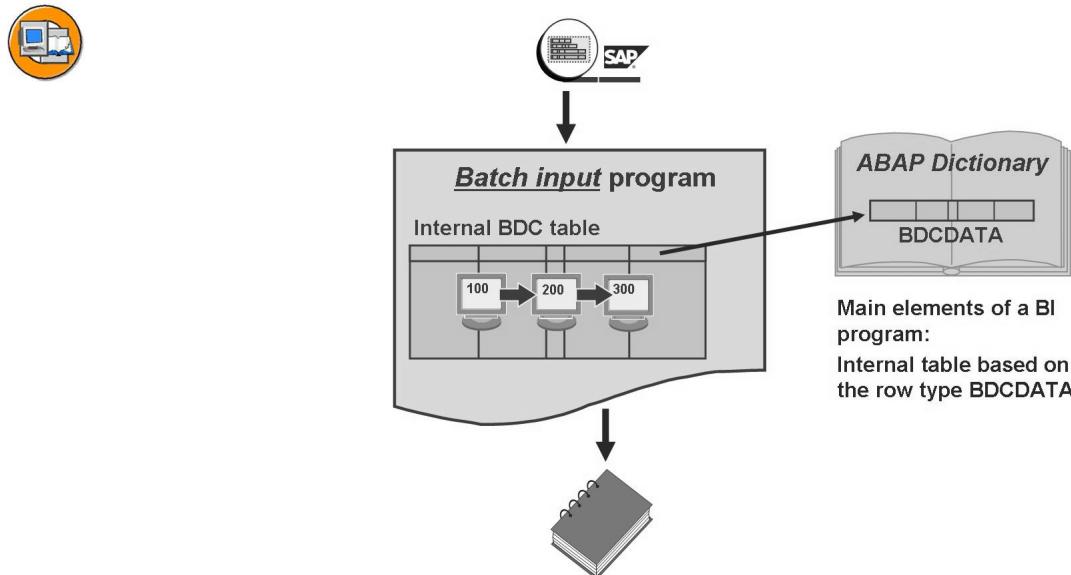


Figure 22: Batch Input Program

Data is transferred to the batch input session by batch input programs. These programs are either made available as standard transfer programs in the DX-WB, or they have to be created for an individual transfer. (For example, using the transaction recorder).

Batch input programs perform the following functions:

- They provide structured work areas for the data to be transferred in the form of an internal table (BDC table).
- They import the data.
- They enter the imported data into the internal table of row type BDCDATA.
- They transfer the filled BDC table to the body of the batch input session.

The BDC table always has the same structure (same row type) and contains the relevant transaction data for the data transfer.

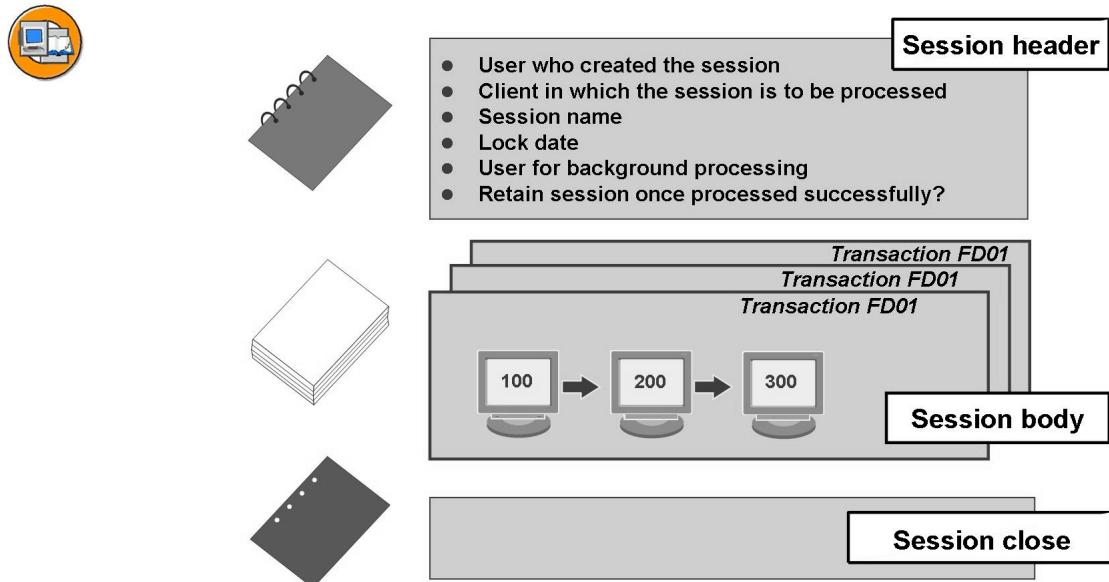


Figure 23: Format of a Batch Input Session

A batch input session consists of a header part (session header, described by the ABAP Dictionary structure APQI) and a data part (transactions, described by the ABAP Dictionary structure APQD).

The session header contains general data on the whole batch input session.

The session body contains all application-specific transaction data.

The session close closes the batch input session. You can then process it using the batch input monitor. The data is then imported into the SAP database.

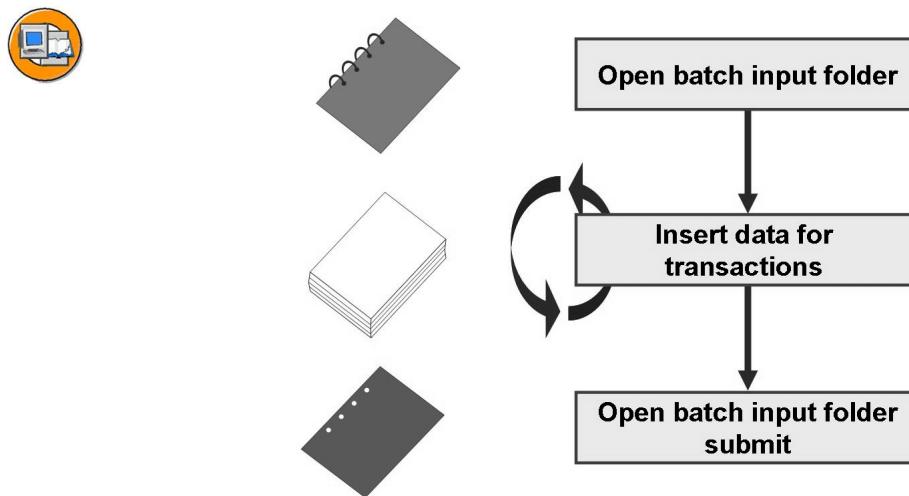


Figure 24: Creating Batch Input Sessions

Each batch input session must be opened and closed. The data for the application functions (transactions) is inserted in the batch input session.

A batch input session contains screen data prepared for individual transactions.

Data is inserted into batch input sessions per transaction. That is, for each insertion step, **one** transaction is inserted into the session body.

A batch input program can generate several consecutive batch input sessions. However, parallel generation is not possible.



Structure of the BDC Table

<i>Program name</i>	<i>Screen No.</i>	<i>Start of screen</i>	<i>Field name</i>	<i>Field value</i>
PROGRAM	DYNPRO	DYNBEGIN	FNAM	FVAL
SAPMF02D	0105	X		
			RF02D-KUNNR	Z-00-00001
			RF02D-BUKRS	0001
			RF02D-KTOKD	KUNA
			BDC_OK-CODE	/00
SAPMF02D	0110	X		
			KNA1-ANRED	Mr
			KNA1-NAME1	Treusch
			KNA1-SORTL	Axel
			KNA1-NAME2	
			KNKA1-STRAS	Heidelstrasse 1
			KNKA1-PFACH	
			KNKA1-ORT01	Walldorf

			KNKA1- PSTLZ	69190
			...	
			BDC_OK- CODE	/11

The data in a BDCDATA table is structured according to the screens (dynpros) in the transaction. Each screen that is processed during the transaction flow must be identified in a BDCDATA record. This record uses the fields PROGRAM, DYNPRO, and DYNBEGIN. A separate BDCDATA record follows for each value that can be input in a field. These records use the fields FNAM and FVAL.

The field name for the OK code is the same for all screens: BDC_OKCODE. The value for the OK code consists of a slash '/' and the function key number (example: '/11').

The **BDCDATA structure** is defined in the ABAP Dictionary as follows:

<i>Field name</i>	<i>Data Element</i>	<i>Type</i>	<i>Length</i>	<i>Short Description</i>
PRO- GRAM	BDC_PROG	CHAR	40	BDC module pool
DYN- PRO	BDC_DYNR	NUMC	4	BDC screen number
DYN- BEGIN	BDC_STARTCHAR		1	BDC begin a screen
FNAM	FNAM_____	CHAR	132	BDC field name
FVAL	BDC_FVAL	CHAR	132	BDC field value

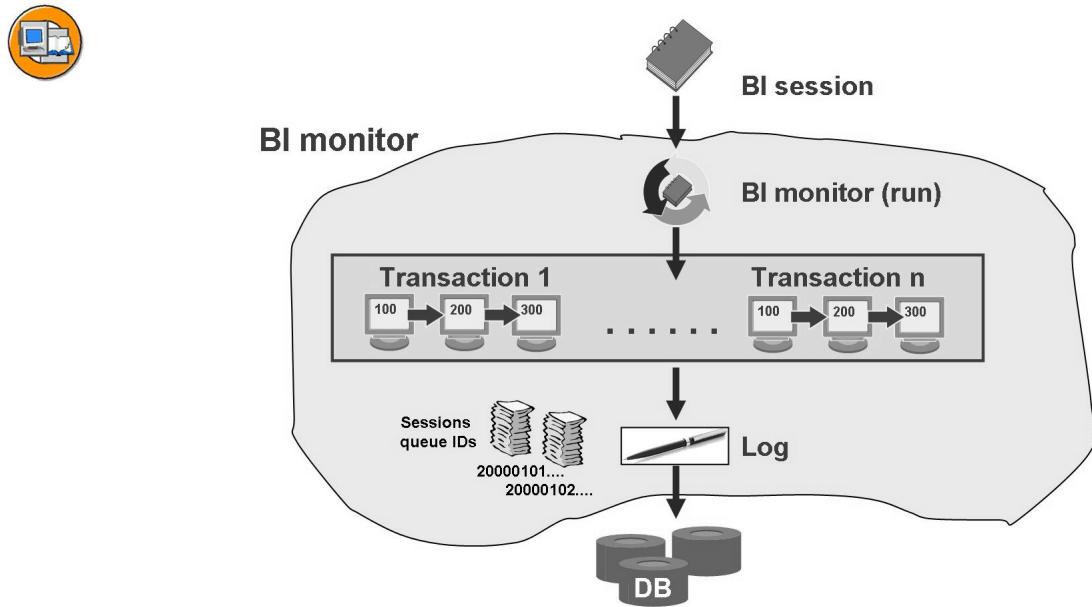


Figure 25: Importing Data in BI Processing

The batch input session combines the individual screens.

You process these sessions using the batch input monitor. As when you execute application functions (with add or change transactions) in the dialog interface, this sends data to the log file. When you subsequently update the data, it enters the relevant SAP tables.

The batch input function starts the application functions specified in the relevant session (indicated by their transaction codes).

The data from the session is now added to the screens belonging to the specified online transaction.

In the case of change transactions, first the data from the SAP database and then the data from the session is **imported** into the screens.

The data is always updated synchronously with the batch input method. With the other techniques, which will be explained in the following units, data can also be updated asynchronously.

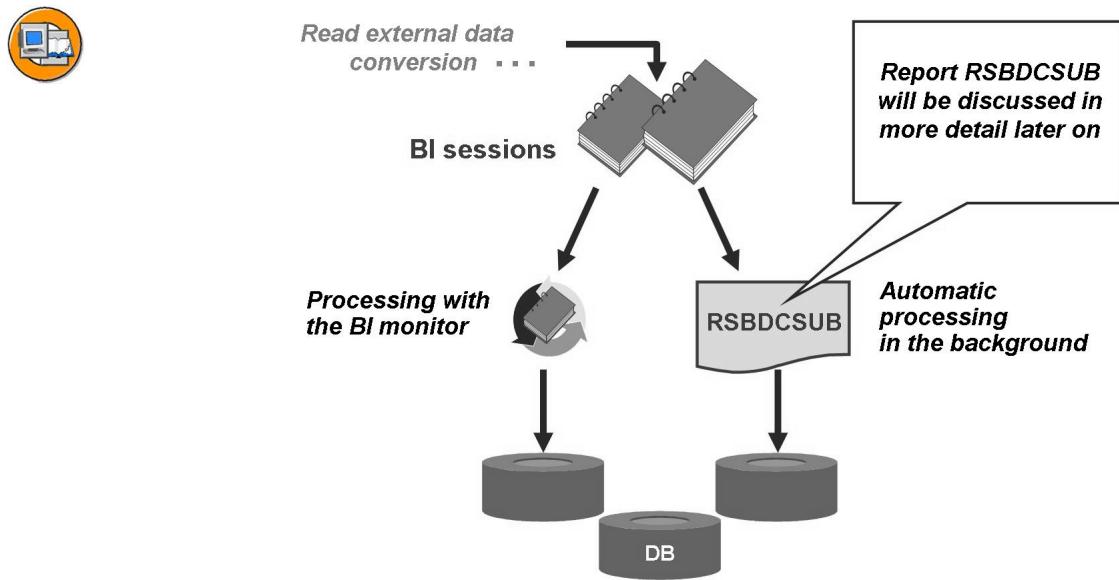


Figure 26: Overview: Batch Input Processing

There are two main methods for processing sessions:

Processing sessions using the BI-Monitor: The created session is processed manually with the batch input monitor. The data is then entered into the SAP database.

Automatic processing: BI sessions can be processed automatically using program RSBDCSUB. We will discuss this program in more detail later in the course.

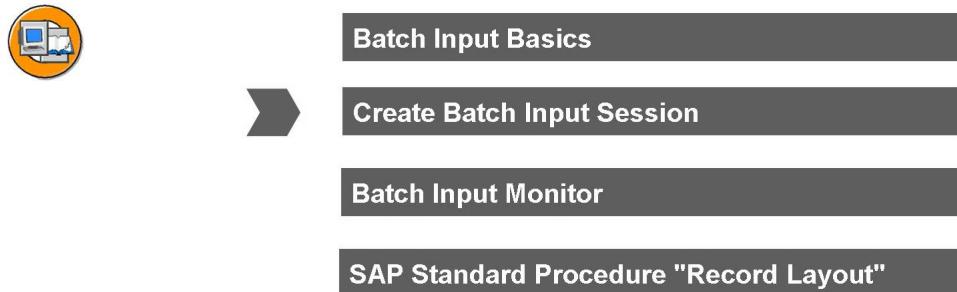
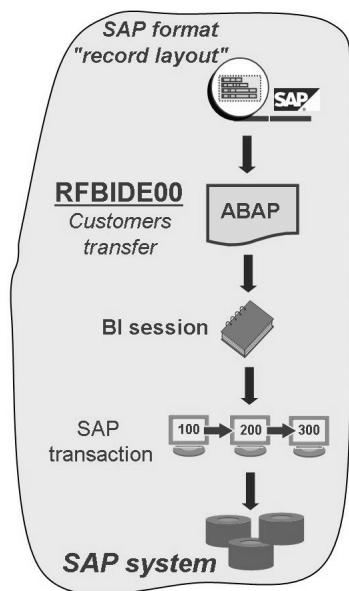


Figure 27: Basics of Batch Input Processing (2)

**Example:**

- A record layout file with converted customer data is already defined.
- This file is imported using SAP transfer method RFBIDE00 and a BI session is generated
- The BI session is processed with the BI monitor.
- By processing the transactions the customers are updated in the database.
- The updated data can be checked in the application. (Display transaction, etc.)

Figure 28: Batch Input Processing Example: Debtors Customers

The entire process of the example of the customer data transfer is going to be illustrated in the next few diagrams.

First the individual components involved will be introduced, then you will find out how these components are used within a DX-WB project.

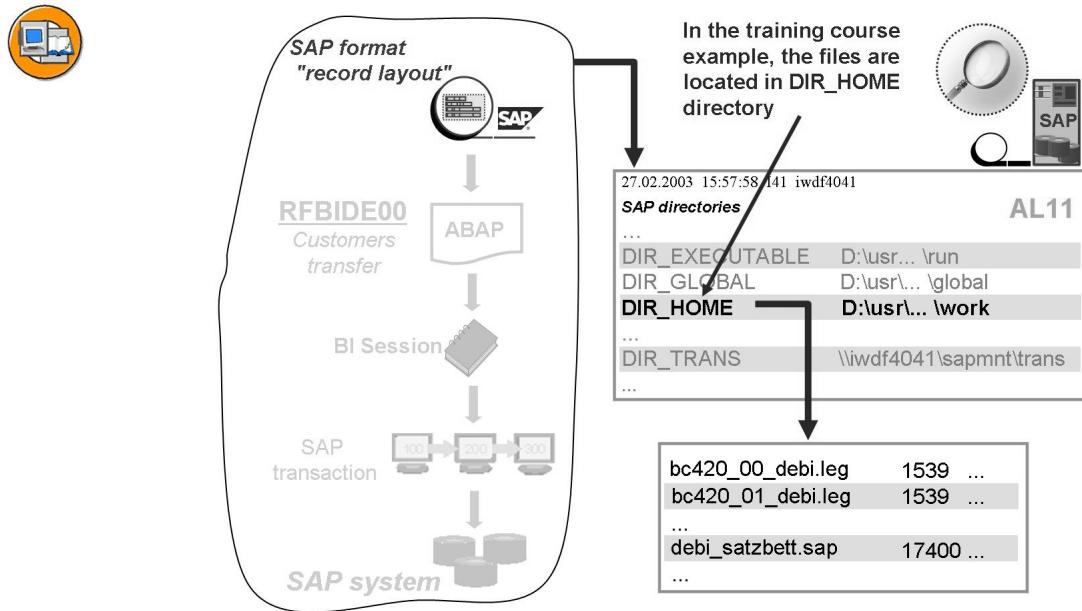


Figure 29: Directories and Files in the File Monitor

→ **Note:** Note that the appearance of the file monitor is release-dependent. In Release SAP NetWeaver AS 7.0, the data appears in an ALV list.

On the initial screen of the file monitor there are two columns:

- On the left, there are internal directory parameters.
- On the right, there are the physical directories for these parameters.

The global transport directory is hidden behind "DIR_TRANS". This is used for exchanging programs and other objects between systems in the SAP environment.

You can access the directories displayed here by double-clicking them or choosing *Display*.

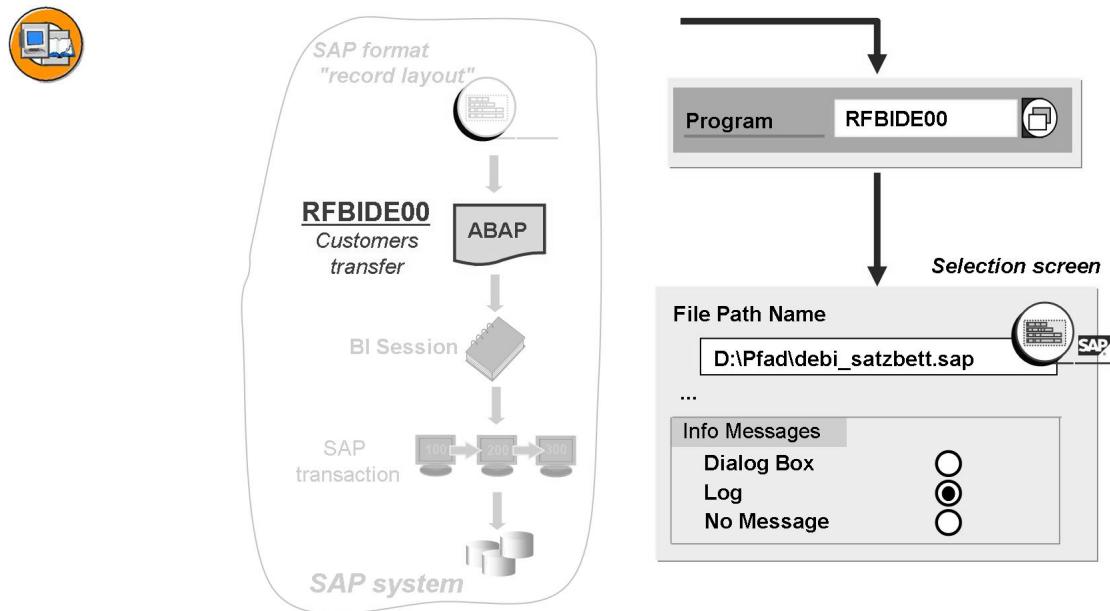


Figure 30: Starting the Transfer

RFBIDE00 was developed to transfer customers using an SAP procedure. RFBIDE00 transfers the customers internally using a batch input technique.

To illustrate how this program works, you can run it manually in SE80 / SE38 / SA38.



Hint: Transfer procedures, for example, RFBIDE00 or RFBIBL00 contain precise, detailed documentation. You should study these carefully beforehand!

The name of the record layout file (file in SAP format) must be entered on the selection screen.

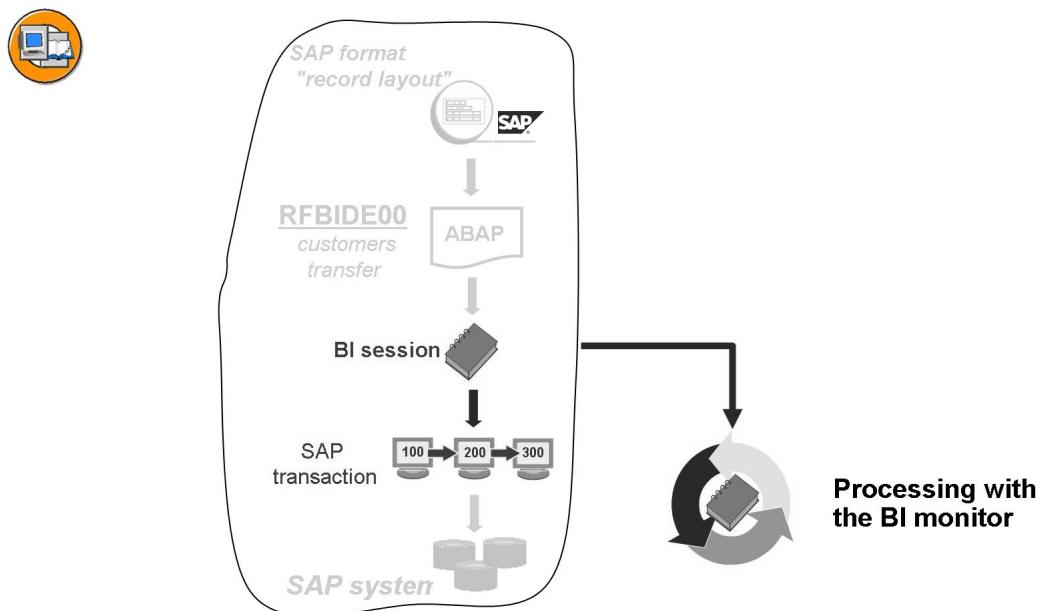


Figure 31: Processing the Session

RFBIDE00 processes the record layout and creates one or more batch input sessions (depending on the structure of the record layout).

The created session(s) can now be manually processed with the batch input monitor. While the session is running transactions are processed and data is written to the SAP database.

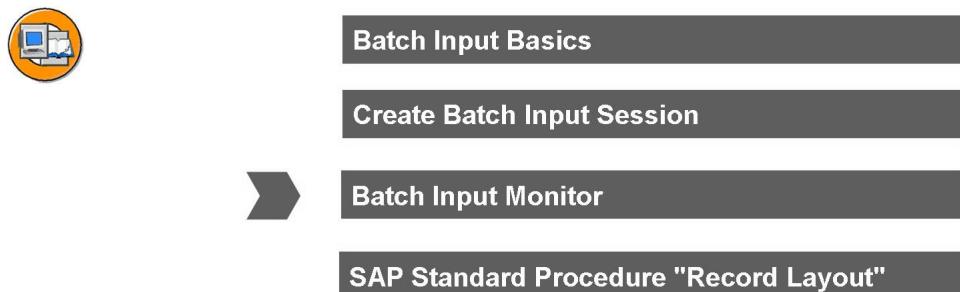


Figure 32: Basics of Batch Input Processing (3)

→ **Note:** Note that the screens or templates displayed here are release-dependent. Therefore, the terms in the menus and the icons may be different.

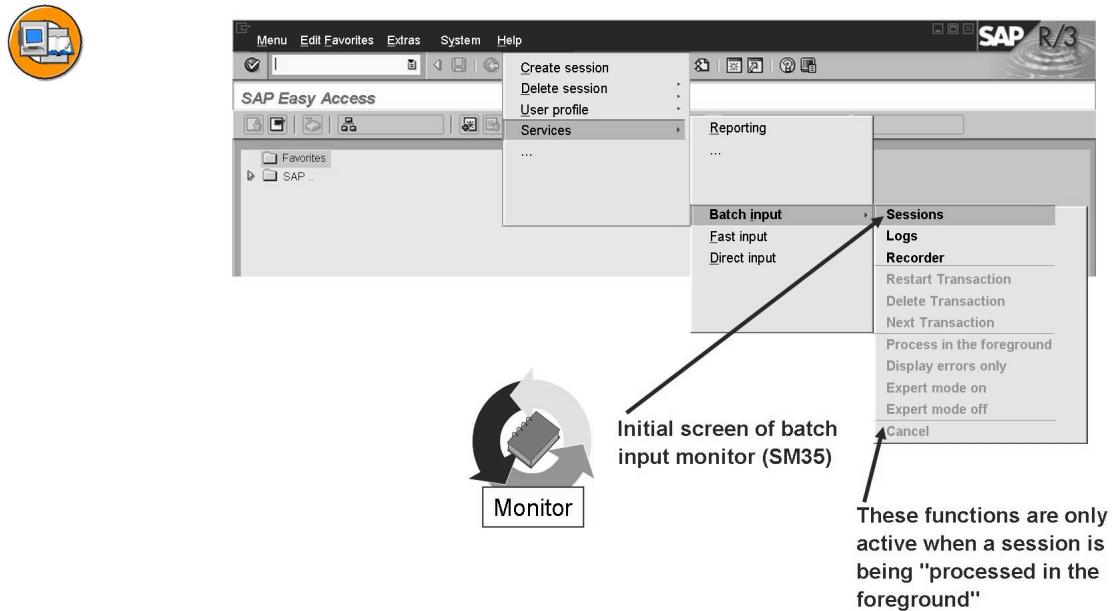


Figure 33: Batch Input Monitor

The batch input monitor can be called up from the *System* menu.

The grayed out functions are only available when batch input sessions are “processed in the foreground”.

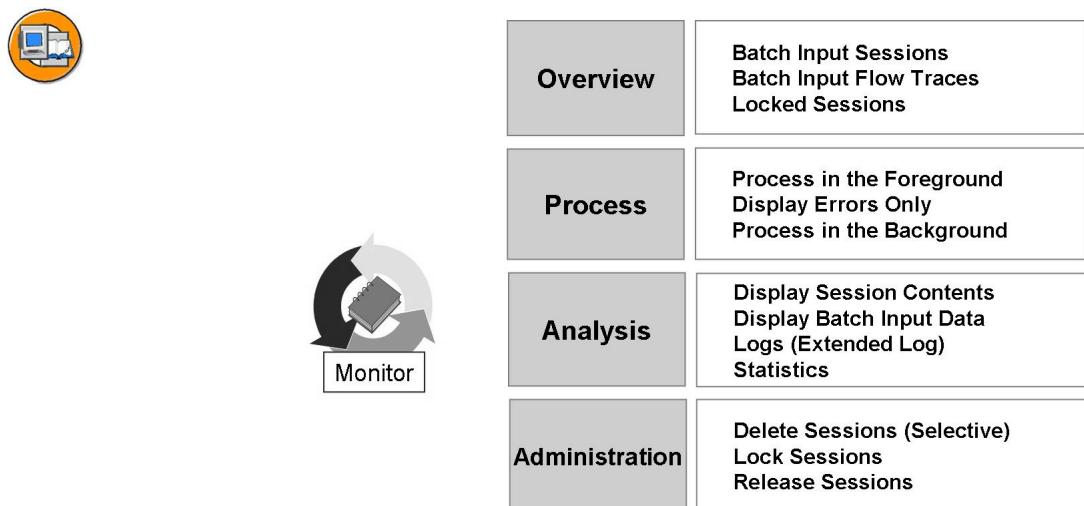


Figure 34: Functions of the Batch Input Monitor

The batch input monitor provides a series of functions for processing batch input sessions.

The sessions processed with the batch input monitor contain information about the status, screen sequence, screen contents, the user who is processing, processing mode, server, and the user who was used for the authorization check.

The system flags transactions that cannot be processed because of errors. These transactions remain in the error session. You can process these transactions again and correct the errors.

The functions in the diagram can be accessed in the monitor from the menu and using pushbuttons.

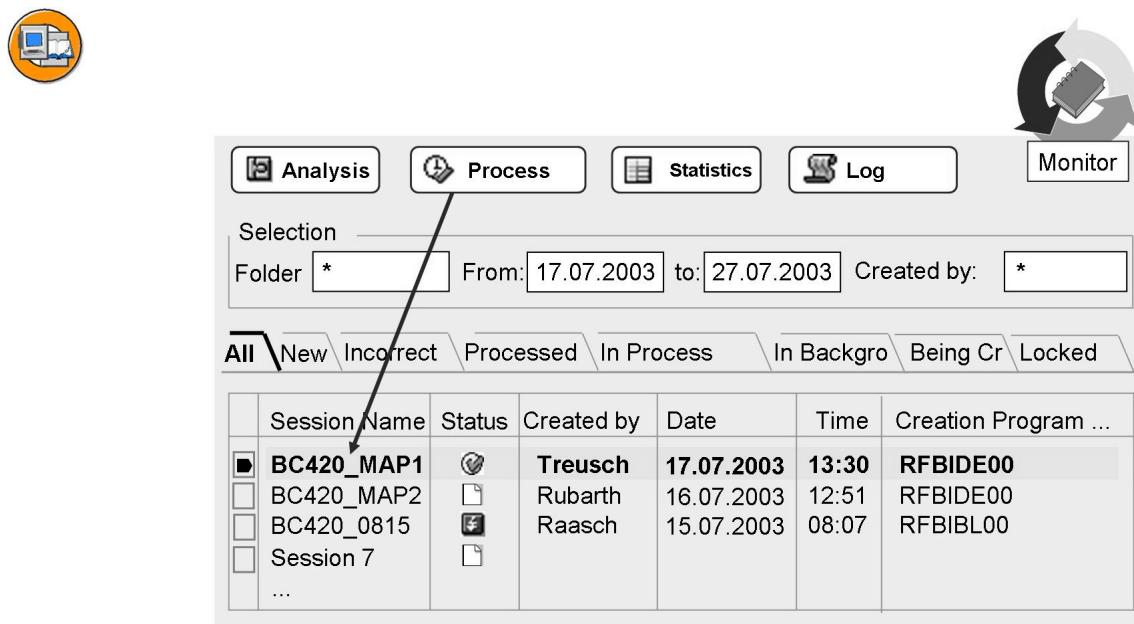


Figure 35: Initial Screen of the Batch Input Monitor

On the initial screen of the batch input monitor you can, for example, select and process individual sessions, and view the logs of sessions already processed

All important functions in the monitor can be set using authorizations. Note authorization object **S_BDC_MONI**. You can find the fields and documentation for this authorization object in transactions SU03 and SU21.

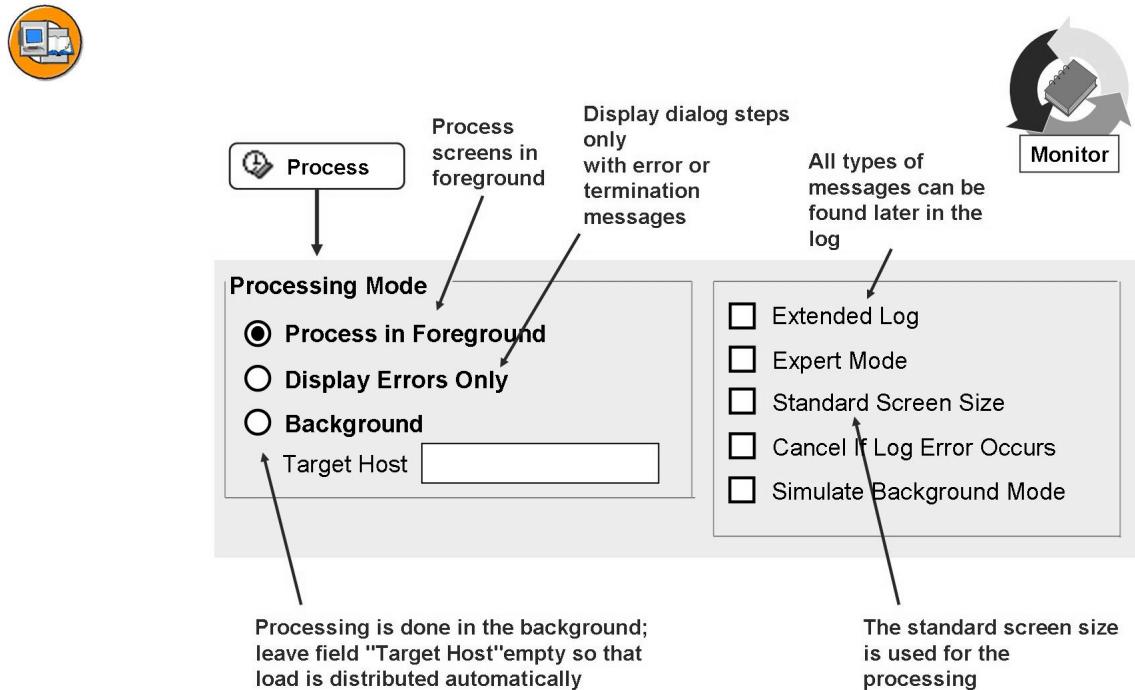


Figure 36: Processing mode

You can select the following processing modes:

Processing mode:

- Process in the foreground: Each dialog step is displayed (interactive session processing).
- Display errors only: Transactions run in the background, the screen is displayed if errors occur.
- Background: Transactions run in the background.

Target host: Specification of the background system in which you want to run your batch input session.

Additional functions:

- Extended log: In addition to “E” messages, the system can also log “W”, “I” and “S” messages. This is also possible when you process a session in the background.
- Export mode: Expert mode is only effective for interactive batch input sessions. It deactivates error message 344, that is, you can execute additional functions without the system sending you the same messages. "Batch input file not available for this screen".
- Standard screen size: Standard screen size is automatically set when processing in dialog (relevant for table controls or step loops).
- Cancel If Log Error Occurs This additional function terminates the processing of the batch input session following the transaction containing the error.
- Simulate Background Mode: The transactions are processed in the background. (Some transactions behave differently in the background and in dialog, see SY-BATCH and SAP Note 433137 about SY-BATCH).

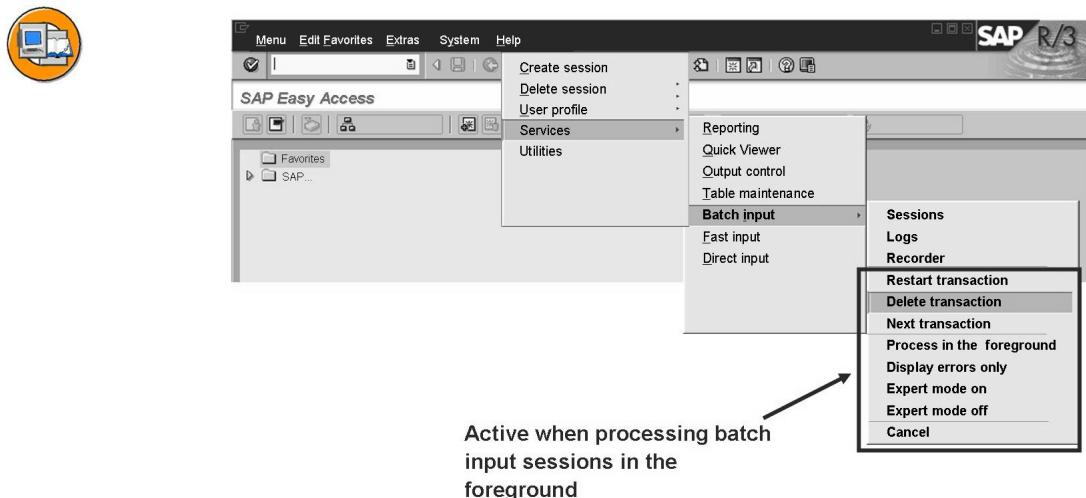


Figure 37: Menu Functions when you “Process in Foreground”

When batch input sessions are “processed in the foreground”, additional functions are available. These can also be called using special OK codes in the command line.



<i>Functions & Permitted Ok Codes</i>		<i>Meaning</i>
Next Transaction	/n	Current batch input transaction end and flag as incorrect
Delete Transaction	/bdel	Delete current batch input transaction in the session
Cancel	/bend	Terminate batch input processing and flag session as incorrect
Process in Foreground	/bda	Change display mode from "Display errors only" to "process in the foreground"
Display Errors Only	/bde	Change display mode from "Process in the foreground" to "Display errors only"
Expert Mode	/bdx	When you exit BI processing, error messages are suppressed.
Restart Transaction	/bbeg	Back to start of transaction



Figure 38: BI Functions and OK Codes

When processing batch input sessions interactively, you can correct the transaction or flag it as incorrect. The corrections will appear later in the session log.

To process a batch input session interactively (“in the foreground” or “display errors only”), choose ENTER to switch between the individual screens. You can correct the data on each screen or control the processing of your session using specific menu functions (see graphic) or OK codes.

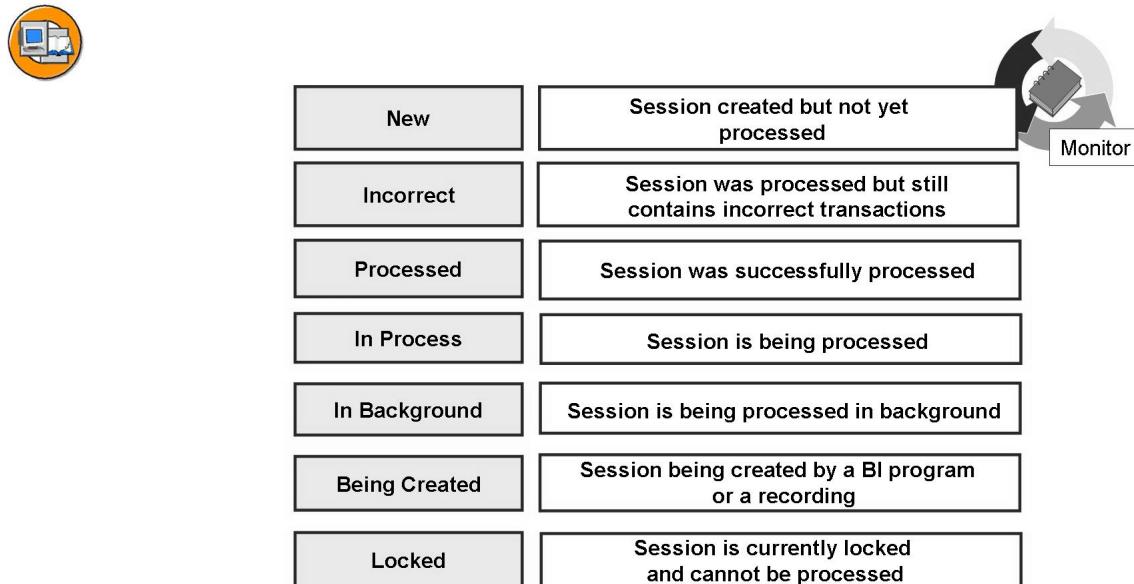


Figure 39: Session Statuses

A session in the session overview is assigned a status before, during and after processing.

The sessions are sorted according to the date and time when they were created and subdivided according to their status in the different displays.

The session overview displays the following information for each session:

- Session name: Session name
- Date and time: Date and time when the session was created
- Locked: The session is locked until this date and cannot be processed until after this date
- Created by: The user who created the session
- Transaction and screen: The number of transactions and screens
- User authorization: The user with whose authorization a session is processed, however only in the background.



- No accessing other transactions ("/\n...")
- Scrolling with ENTER key only
- Corrections during foreground processing are taken into account in the log

Note:

Data from the session, is displayed in red!

Figure 40: BI Session Behavior with “Process in Foreground”

With “foreground processing”, Enter is used for scrolling.

You should not navigate to other transactions.

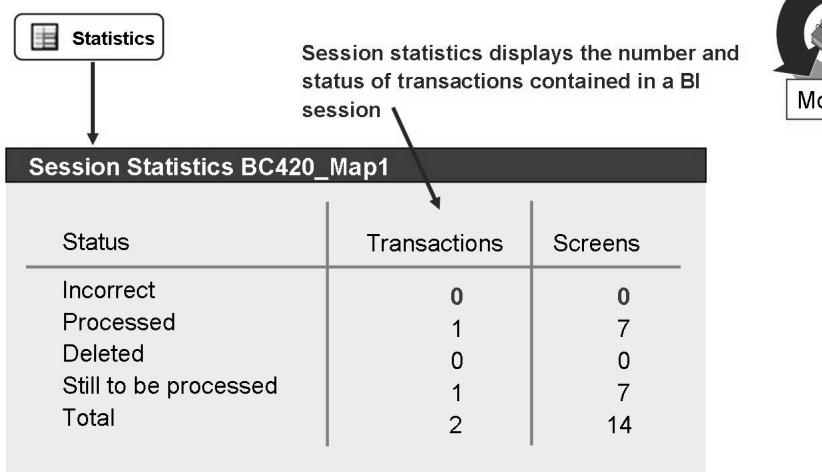


Figure 41: Session Statistics

You can request a statistic for each session. This gives you information on the number of transactions in the session that were incorrect, processed, deleted, or still to be processed.

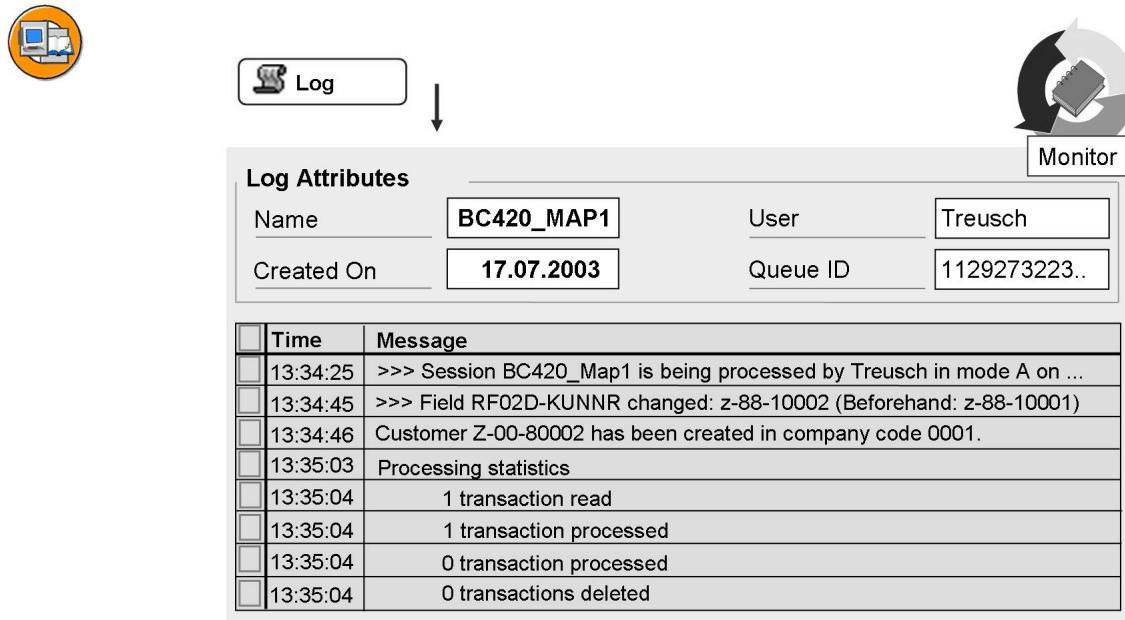


Figure 42: Session Log

The system creates a log each time a batch input session is processed.

If errors occur as the session is being processed, the session is flagged as incorrect. If you then process this session again, the system creates an additional log.

A session log contains all error messages for all transactions in a batch input session. This type of log also contains batch input error messages if problems occur when you process transactions, and the screen in which the error occurred. The log also contains overall statistics on the session.



- Overview of all transactions in a session
- Overview of **incorrect** transactions in a session
- Field list display for each screen of a transaction
- Screen display for each screen of a transaction
- Simple switch to session log
- Queue dump: Rough display of session contents
- Session header: Technical view of session header

You can also call the session analysis before “Process in Foreground”. The analysis provides a detailed technical view of the session.

You can also check all transactions and screens belonging to the session before the session is processed. The Screen Display is provided for this purpose (screens and data are displayed). The field list display provides a detailed view of the session and the transactions and screens it contains (see 8 BDCDATA structure).

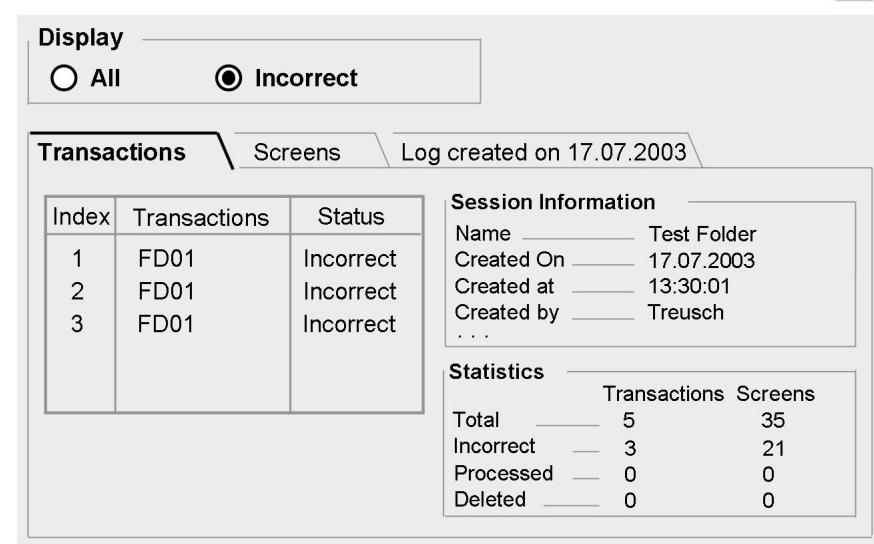


Figure 43: Session Analysis - Incorrect Transactions

The session analysis is divided into three areas:

- The tab page *Transactions* displays a list of the transactions contained in the session together with their statuses.
- The *Screens* tab page displays an overview of the screens in each transaction. You can display each screen with the data contained in it.
- The tab page *Log* displays the session log.

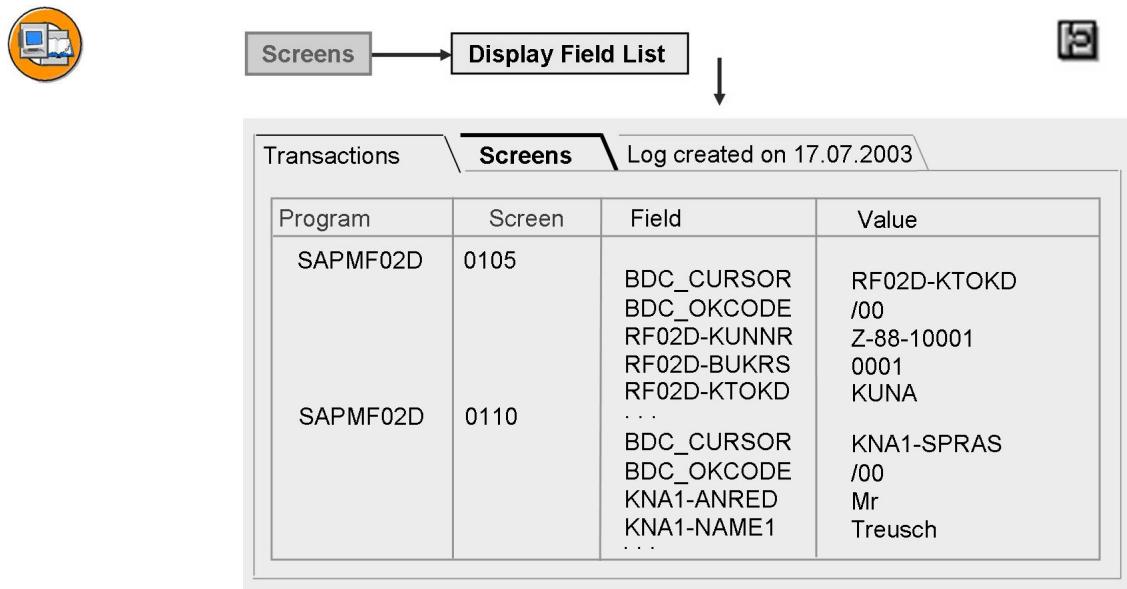


Figure 44: Session Analysis - Screen Field List

In the field list display of the tab page *Screens*, the screen data is displayed in BDCDATA structure format.

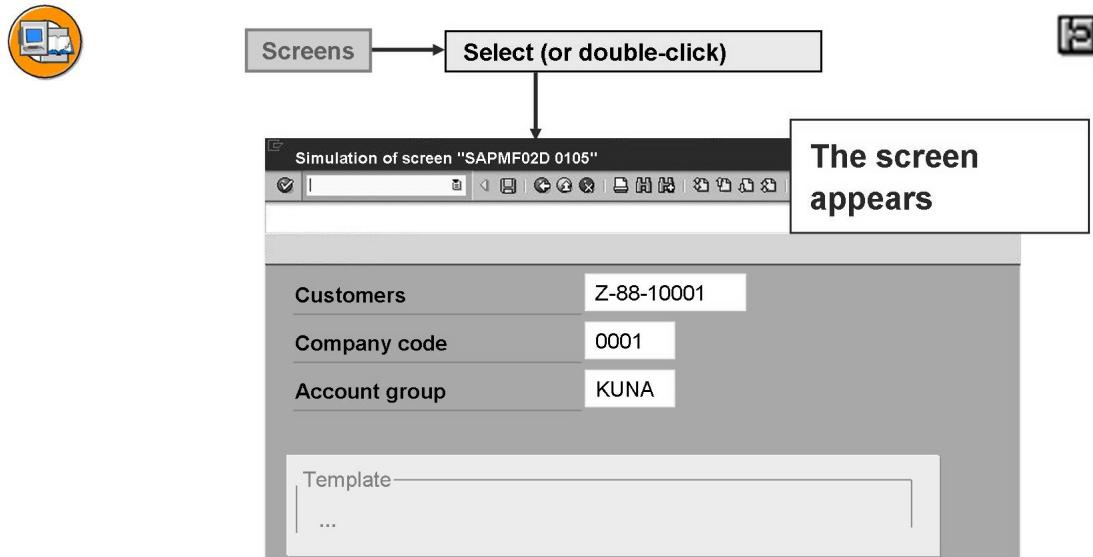


Figure 45: Session Analysis - Screen Display

If you double-click on any screen you can select the screen simulation (screen display).

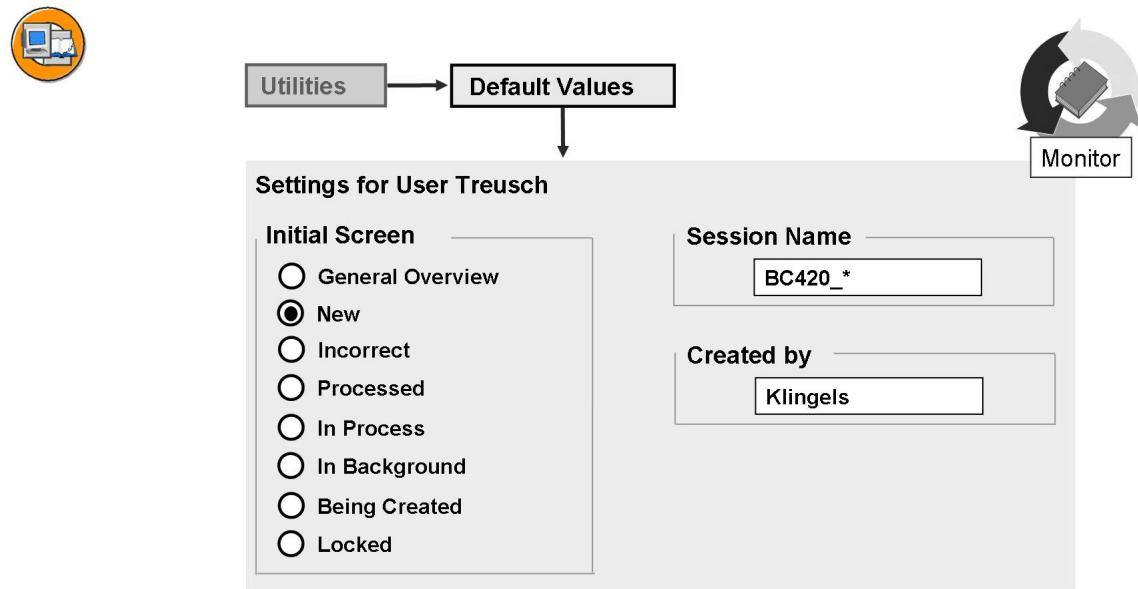


Figure 46: User Settings in the BI Monitor

All users can make their own *settings* for the batch input monitor. For example, a user can select only those sessions that start with the prefix **BC420***.

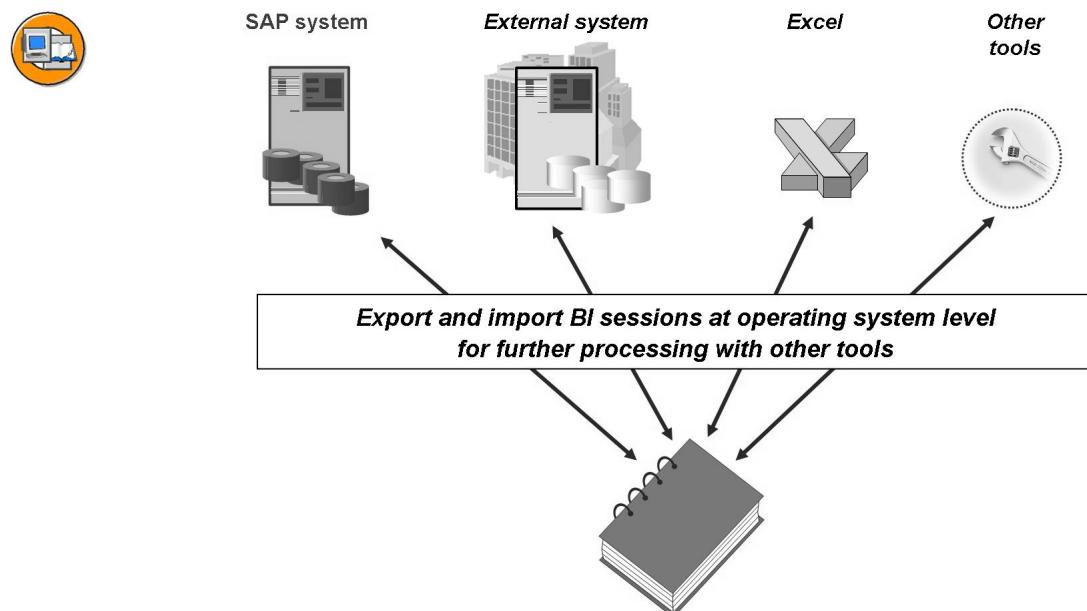


Figure 47: Exporting / Importing BI Sessions

Under *Utilities → Export/Import Session* (in batch input monitor) session contents can be exported and imported at operating system level. You can use Excel, for example, to change the data of an exported session and then to import it to another client. Note that: BI sessions are client-dependent.

This makes further processing with external software tools feasible.

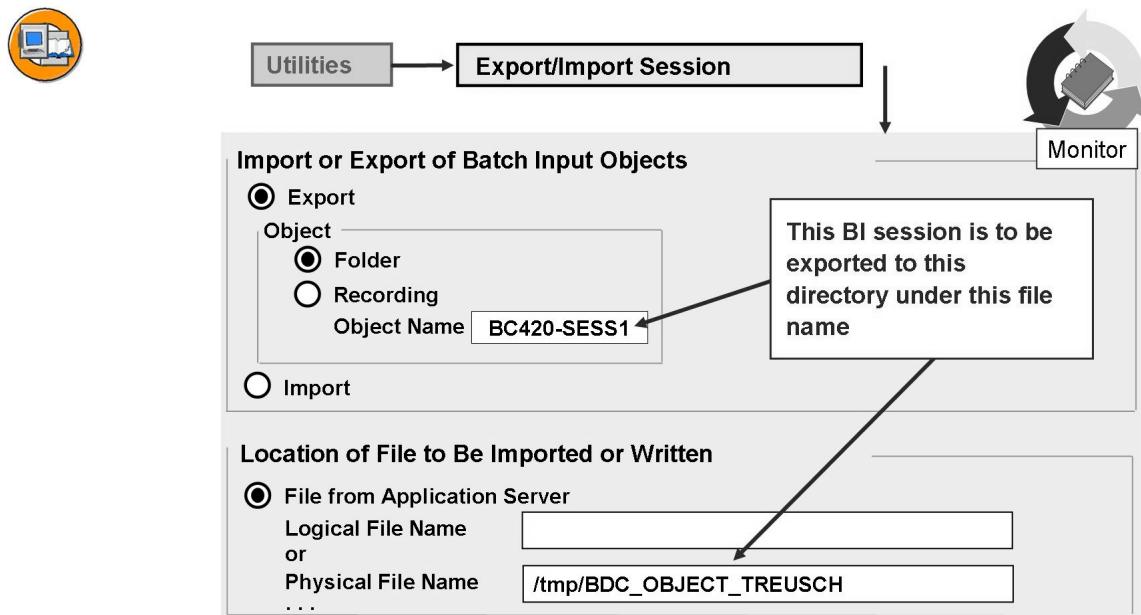


Figure 48: Export / Import Options

Physical or logical file names are used to import and export sessions on the application server. The logical file names are specified in Customizing of SAP Basis (transaction FILE). We will discuss this later.

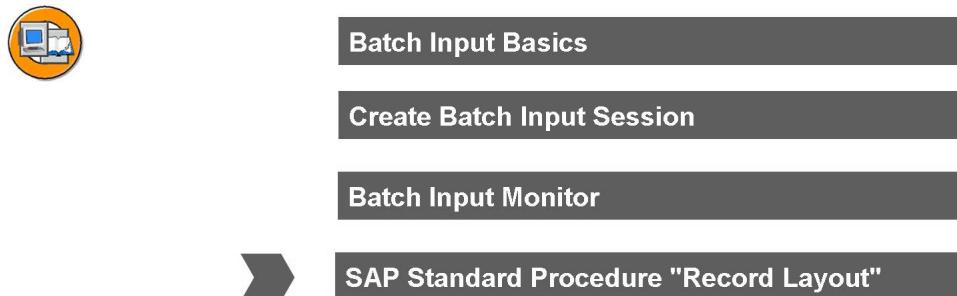


Figure 49: Basics of Batch Input Processing (4)



Example

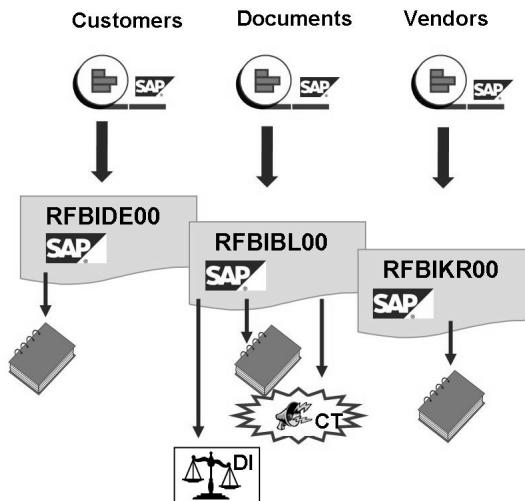


Figure 50: Example: Standard Procedure

Each application provides transfer programs. The individual programs decide which technique to use (batch input, call transaction or direct input). As a user, you cannot influence this decision.

If, however, you make the data available in the SAP record layout structure and the program you are using supports various techniques, you can use all the techniques without changing the structure and the contents of the file.

An example is program RFBIBL00, which can transfer FI documents. Here, you can use one of the three techniques: batch input, call transaction or direct input.

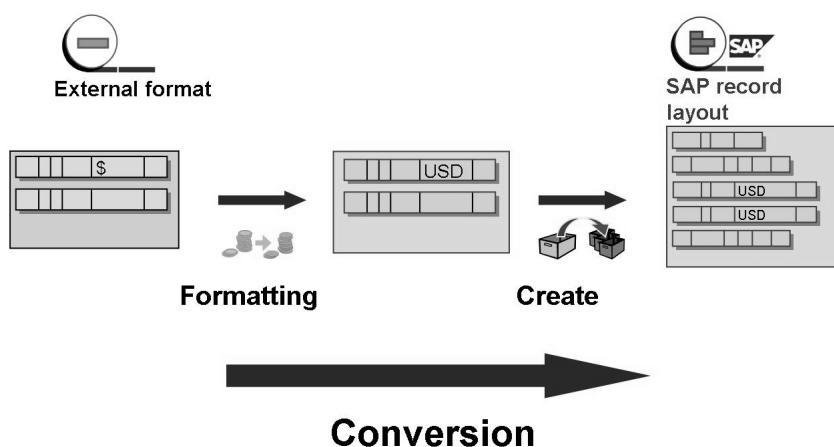


Figure 51: Converting the Data

You can convert the external data into the SAP record layout format required by the standard data transfer programs in two ways as displayed in the graphic:

1. Conversion:

First the external data format is translated into the SAP format ("\$" → "USD", for example). This can create an additional sequential file that contains the external data in the SAP format (this is not yet in the record layout format).

2. Assignment:

The sequential file in the SAP format that was previously generated is imported and the data is "mapped" to the fields in the record layout format. A sequential file is created in record layout format. This is required by the standard transfer procedure in applications.

You can also convert the external data into the SAP record layout format in one step. This step, this program (regardless of the programming language) must do the following:

- Read external data
- Convert into SAP format
- Assign the converted data to the record layout fields
- Write the sequential file in record layout format



Hint: During the course, always check where you are in the data transfer process. The introductory slides with the graphics showing the individual techniques are useful here.

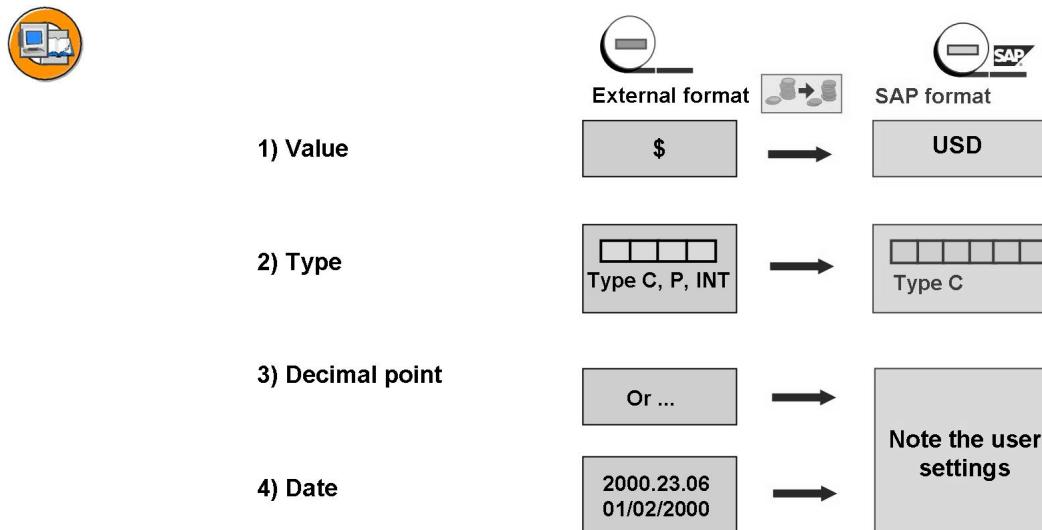


Figure 52: Formatting

When you use the record layout-orientated data transfer method, the legacy data must be formatted in exactly the same way as it is for a dialog user who inputs the values using the keyboard. The following rules apply:

- The data **must be in character format**.
- The individual fields must not be longer than the SAP field.
- If the data is shorter than the SAP field, it must be left-justified (padded with blank characters at the right end).

You must also consider the **user-specific settings** (user default settings).

The date format and the decimal point format are user-defined settings. Each user can maintain these settings themselves under *System -> User Profile -> Own Data*.

If the data is transferred online, the authorizations and default settings of the user who opened the transaction are used.

You must specify a user if you want to transfer data in the background. This user should have the same default user settings. Sometimes this user may have other authorizations, for example, in the production system for the actual data transfer.



Hint: Important: Different authorizations and settings may result in errors occurring during the data transfer (for example, no authorization for the company code).

There will be many exercises and examples to clarify this.

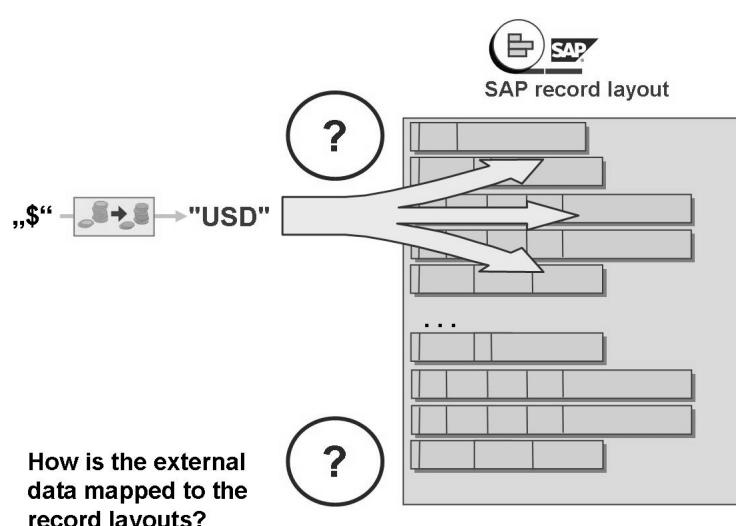


Figure 53: Assignment or “Mapping”

The standard transfer program expects a file in the appropriate SAP record layout format.

The data must match this record layout structure. However, all the record layout structures do not always have to be used.

There are some structures that must be included in the record layout, and others that are optional. This depends on the application and you can find out more from the documentation for the record layout structure in the relevant application.

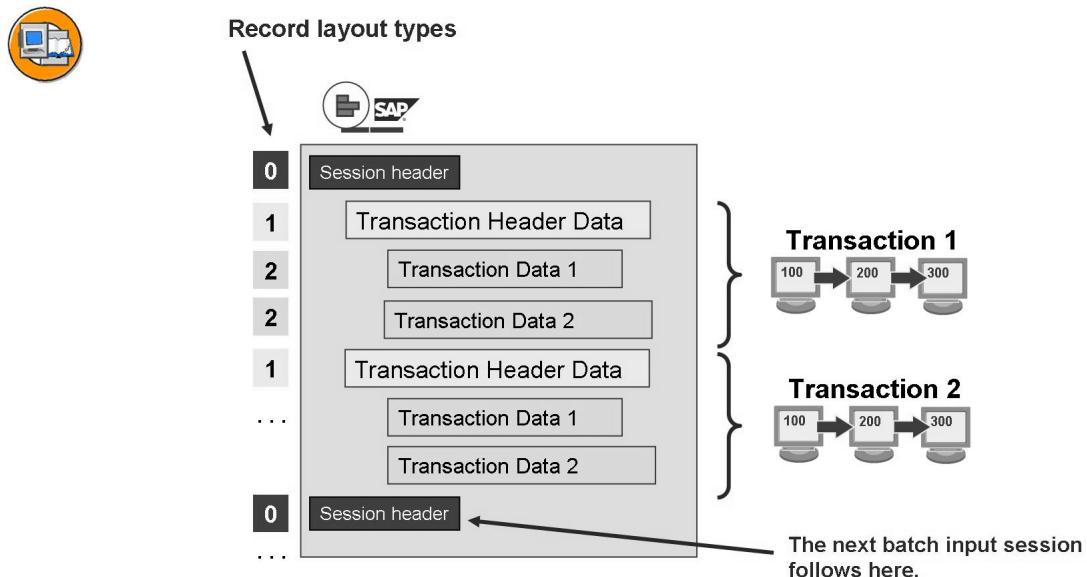


Figure 54: Example: Record Layout Structure

The sequential file data records must be in a format that can be read by the SAP standard data transfer methods.

Each transfer program expects the data precisely in this SAP record layout format.

Each record layout is made up of individual structures. Each structure is created in the ABAP Dictionary.

The structure descriptions and the structure of the sequential files depend on the application and are described in the documentation on the standard data transfer methods.

When you transfer the data, make sure that a session record is always transferred first, and then the header data and subsequent data.

If a new session header is added to the structure, the existing session will be closed and a new one created.

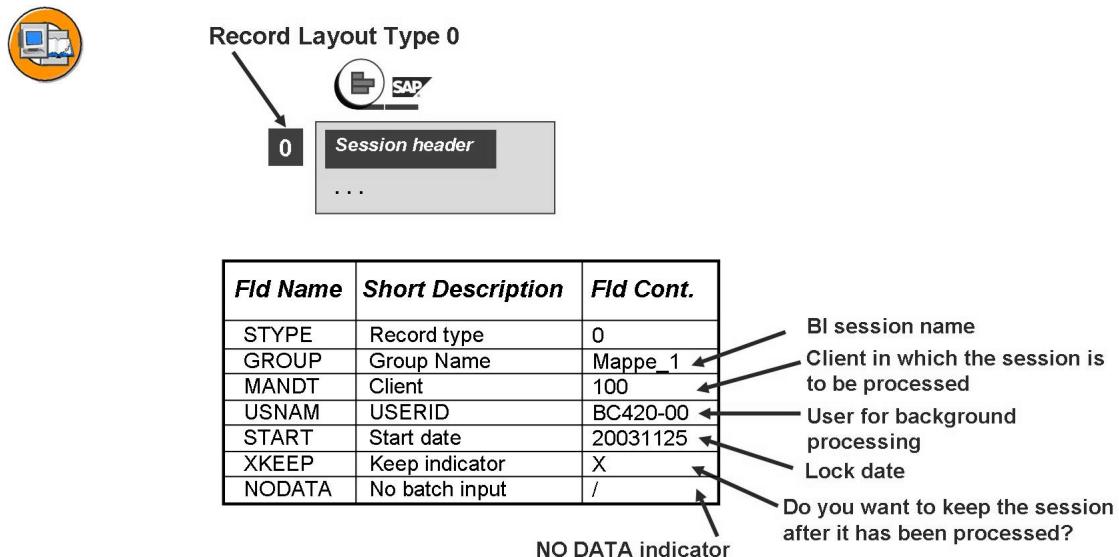


Figure 55: Example: Record Layout Type 0

Each session contains a session header (record type 0), also called a session header record. The session header record contains general administration information about the batch input session you want to create. All further records in the sequential file up to the next session record header are assigned to the current session.

Important: If the data is transferred in the background, the user specified in the session header is used for the authorization check. This user's authorizations and settings defined in the user profile are used for the check

If the keep indicator is set (= X), the processed session remains in the system. If it is not set, the session is deleted.

In field NODATA you define the character that will be later used as the NODATA indicator.



Caution: The Start field contains the lock date. Sessions are locked up to and including this date.

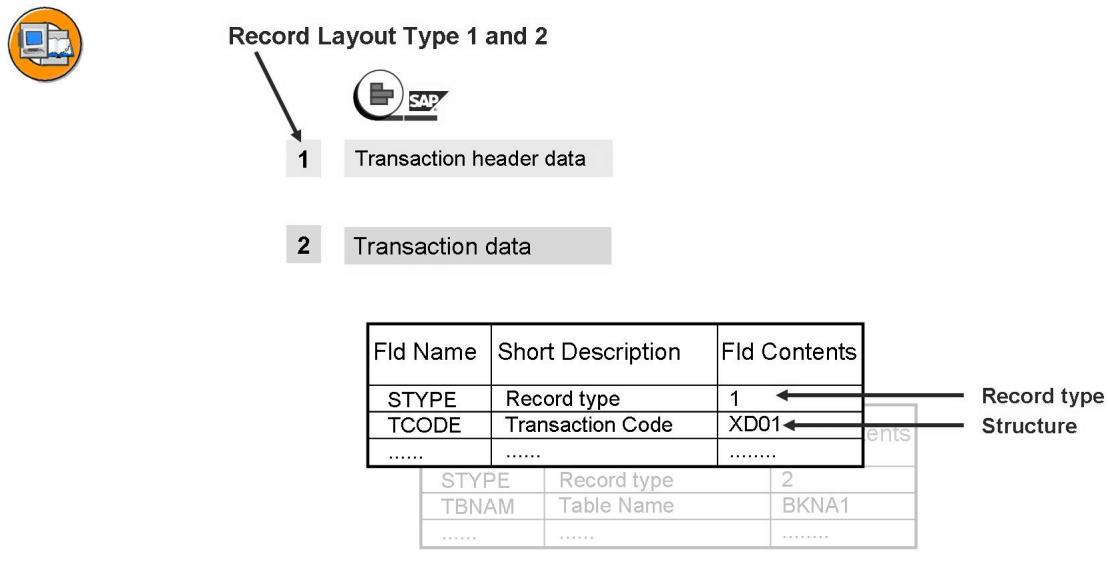


Figure 56: Example: Further Record Layout Types

Transaction Header Data

- These structures contain data for **one transaction**. You enter the data here that you want to see when you select the transaction and that you want on the initial screen. This data includes the transaction code and key fields, such as, account group, company code key, document type, account number, and document number.

Transaction data

- These structures contain master record data and document items. This includes address data of your business partners, amounts, and document items.

The different structures are indicated by record types. For the subsequent data you also need to enter the structure name in addition to the record type.

The structures used for batch input are defined in the Dictionary. You can display the individual structures there.

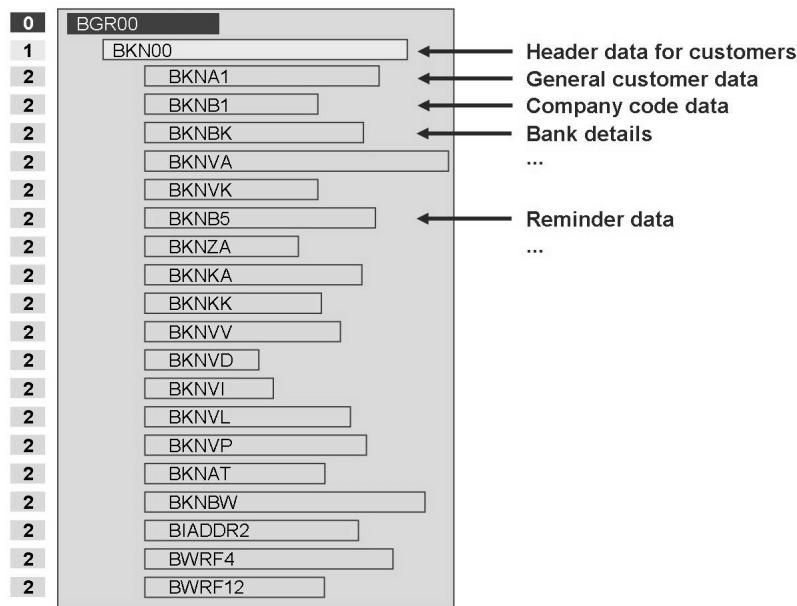


Figure 57: Example: Record Layout for Customers

Example of record layout structures for transferring customer data.

For this example, the first four record layout structures are required. In practice, all structures may be used. Clarify this beforehand using analyses and tests. The documentation describes each structure in detail.

 **Note:** Refer to the following SAP notes about the customer/vendor procedures: 738448 (Import from non-Unicode file) and 878369 (Working with files in non-Unicode format)

Exercise 2: Working with Batch Input Sessions

Exercise Objectives

After completing this exercise, you will be able to:

- Step-by-step procedure for transferring data using a record layout file.
- Working with the batch input monitor.
- Processing batch input sessions.

Business Example

Starting from a predefined record layout file, you will transfer customer data **step by step**. You will start the relevant application procedure and process the created BI session.

Task:

Find and import the record layout file containing the customer data that is already converted.

1. Open the file monitor (AL11) and in directory ...\\...**work** (parameter DIR_HOME) you can find the record layout file for your group **BC420##DEBI1.SAP**.
2. Start the SAP standard procedure for customers RFBIDE00 online to import this record layout file and create a batch input session.
3. Check the session you have just created in RFBIDE00 in the batch input monitor and then run the session.
4. Check the session log, session analysis, and session statistics.

Solution 2: Working with Batch Input Sessions

Task:

Find and import the record layout file containing the customer data that is already converted.

1. Open the file monitor (AL11) and in directory ...\\...\\work (parameter DIR_HOME) you can find the record layout file for your group BC420_##_DEBI1.SAP.
 - a) Open transaction AL11 to start the file monitor. In the left hand column is the directory parameter **DIR_HOME**. Double-click DIR_HOME and search for your record layout file in the list (you can use the Find button).
-  **Caution:** The find function is case sensitive.
2. Start the SAP standard procedure for customers RFBIDE00 online to import this record layout file and create a batch input session.
 - a) You can run RFBIDE00 in SA38, SE38 or SE80. On the selection screen you have to enter the record layout file you located in AL11. In addition, select the radio button *Log*.
3. Check the session you have just created in RFBIDE00 in the batch input monitor and then run the session.
 - a) In the batch input monitor (SM35) you will find your batch input session. The simplest solution is to filter out your own sessions under *Created by*. You can see the ten transactions in the session analysis. Select the session, choose *Process → Process in foreground*.
4. Check the session log, session analysis, and session statistics.
 - a) The button *Log* opens the session log. You can also access the log via the Analysis button.

The button *Statistics* takes you to the session statistics. Ten successfully updated customers should be listed here.

Exercise 3: Batch Input Sessions and Error Handling

Exercise Objectives

After completing this exercise, you will be able to:

- Analyze and correct errors in batch input sessions

Business Example

A predefined record layout file contains some logical errors. You have to identify and correct these errors.

Task:

Import a record layout file with logical errors and correct these errors:

- A record layout BC420_##_DEBI2 containing two logical errors has already been set up for your group.
- Start the SAP standard procedure for customers RFBIDE00 online to import this record layout file and create a batch input session. On the selection screen for *Info Messages* select *Log*.
- Check the session you have just created in *RFBIDE00* in the batch input monitor and then run the session in mode *Display errors only*. If any errors occur terminate the current transaction and continue with the procedure for running the session. Analyze the session to get to the incorrect transactions in the log. Which transactions are incorrect and why?
- Correct the data by importing the session again. Check the session log, the session statistics and the session analysis. Have your corrections been logged?
- Optional:**

Carry out step 2 above (start RFBIDE00) again using known record layout file BC420_00_DEBI1.SAP. Did errors occur? Why?

Under which circumstances would no errors occur?

Solution 3: Batch Input Sessions and Error Handling

Task:

Import a record layout file with logical errors and correct these errors:

1. A record layout BC420_##_DEBI2 containing two logical errors has already been set up for your group.

- a) Call transaction AL11 to start the file monitor. In the left hand column is the directory parameter *DIR_HOME*.

Double-click *DIR_HOME* and search for your record layout file in the list (you can use the Find button).



Caution: The find function is case sensitive.

2. Start the SAP standard procedure for customers RFBIDE00 online to import this record layout file and create a batch input session. On the selection screen for *Info Messages* select *Log*.

- a) You can run RFBIDE00 in SA38, SE38 or SE80. On the selection screen you have to enter the record layout file you located in AL11. In addition, select the radio button *Log*.

3. Check the session you have just created in *RFBIDE00* in the batch input monitor and then run the session in mode *Display errors only*. If any errors occur terminate the current transaction and continue with the procedure for running the session. Analyze the session to get to the incorrect transactions in the log. Which transactions are incorrect and why?

- a) In the batch input monitor (SM35) you will find your batch input session. The simplest solution is to filter out your own sessions under *Created by*.

You can see the ten transactions in the session analysis. Select the session, choose the button *Process -> Display errors only*. If errors occur, make a note of them and cancel the current transaction using the OK code /n or by choosing: *System → Services → Batch Input → Cancel*. The processing continues automatically with the next transaction.

The logical errors that occurred were:

1. Transaction **8**, here an **incorrect reconciliation account** was used.
2. Transaction **5**, here the **postal code used was too short**.

Continued on next page

4. Correct the data by importing the session again. Check the session log, the session statistics and the session analysis. Have your corrections been logged?
 - a) If you process the session again, you will be able to correct the errors. The corrections are recorded in the log. Once you have made the corrections, session ten contains successfully processed transactions and is assigned the status *Processed*.

5. ***Optional:***

Carry out step 2 above (start RFBIDE00) again using known record layout file BC420_00_DEBI1.SAP. Did errors occur? Why?

Under which circumstances would no errors occur?

- a) The predefined record layout file contains ten customers that were already updated. Program RFBIDE00 asks whether, for the external number assignment (account group KUNA), the customers to be imported have already been posted. If this is the case, it terminates with error messages.

If you work with internal number assignment, you can use the record layout file again. Therefore, in a productive transfer, all data that has already been successfully processed must be deleted. This will be repeated later in the course.



Lesson Summary

You should now be able to:

- Describe the structure of batch input sessions
- Process and analyze batch input sessions using the batch input monitor
- Explain what the SAP standard method is and how record layout files are used



Unit Summary

You should now be able to:

- Describe the structure of batch input sessions
- Process and analyze batch input sessions using the batch input monitor
- Explain what the SAP standard method is and how record layout files are used

Unit 3

Basic information about DX-WB

Unit Overview

This unit deals with the basic information about the DX-WB tool. The DX-WB can be considered a main program for the data transfer procedure and offers a number of partial tools for the transfer as well as a project environment and scheduling environment for the individual transfer steps.



Unit Objectives

After completing this unit, you will be able to:

- Describe the most important functions of the DX-WB
- Use the main tools of the DX-WB
- Create a data transfer project in the DX-WB and transfer customer data

Unit Contents

Lesson: DX-WB Basics	64
Exercise 4: Working with the DX-WB Tools	85
Exercise 5: Creating a New Test Record Layout	89
Exercise 6: Working with the Project Management	91

Lesson: DX-WB Basics

Lesson Overview

- Data transfer methods and DX-WB tools
- Tasks in the data transfer project



Lesson Objectives

After completing this lesson, you will be able to:

- Describe the most important functions of the DX-WB
- Use the main tools of the DX-WB
- Create a data transfer project in the DX-WB and transfer customer data

Business Example

A customer wants to use the Data Transfer Workbench (DX-WB) for a project. In particular, they want to use the DX-WB Editor.

Overview of the Functions of the DX-WB

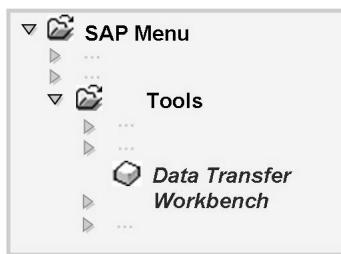


Overview

Tools

Projects

Figure 58: DX-WB Basics (1)



- **Project organization**
 - Projects, subprojects, flows, tasks
 - Execute and schedule process flows
- **Tools**
 - Create test and source files
 - Edit and check files
 - Copy, split files, etc.
 - Conversion Assistant
 - Create export/import structures
- **Program library**
 - BAPIs, reports, function modules

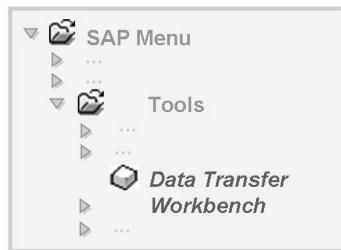
Figure 59: DX-WB: Overview of the functions

The functions of the DX-WB can be divided into three areas: Projects, tools, and library.

The DX-WB has an integrated project management system which takes care of all the steps required to transfer data into your SAP system.

The DX-WB provides various tools to support you with the data transfer.

SAP provides a range of standard data transfer programs for loading data into your SAP system. These can be called up from the DX-WB.



Another way to find documentation for the different procedures.



Help → Application help



Data Transfer Workbench

- ▷ Initial Data Transfer
- ▷ Authorizations
- ▷ Preparing the Data Transfer
-

- ▷ Data Transfer Applications

Figure 60: DX-WB: Application help

DX-WB documentation can be found in the *Application Help*. Here you can find all the information on the different techniques, ordered by application (data transfer objects by applications).

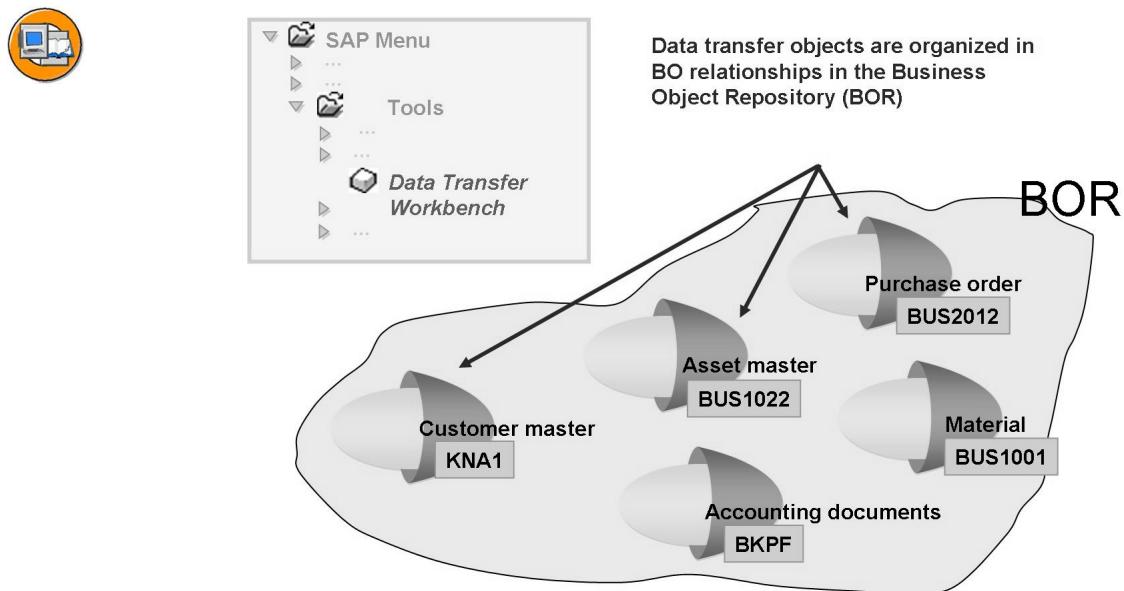


Figure 61: DX-WB: Organization of the Transfer Objects

BOR = Business Object Repository

- The BOR is the central repository in the SAP System, and contains all SAP business object types, SAP interface types, their definitions and methods, etc.
- The BOR identifies and describes the available SAP business object types and interface types, and their BAPIs.
 - For example, if you are developing an application program, in the BOR you will find all the information about SAP business object types, SAP interface types, their key fields and their BAPI methods. The BOR contains all the information you need to use the correct object type definitions and BAPI calls in your application program.
- As of Release 4.6, all transfer programs are organized using the BOR objects relevant to the application.

Various transfer methods can be used for each BOR object. These may be BAPIs or transfer programs. As BAPIs are already identified by BOR objects, it makes sense that standard transfer programs are identified in the same way. So a BOR object can identify all the transfer programs available to it.

The BOR acts as the interface between the SAP system and the outside world, so all transfer programs are assigned to the BOR objects.

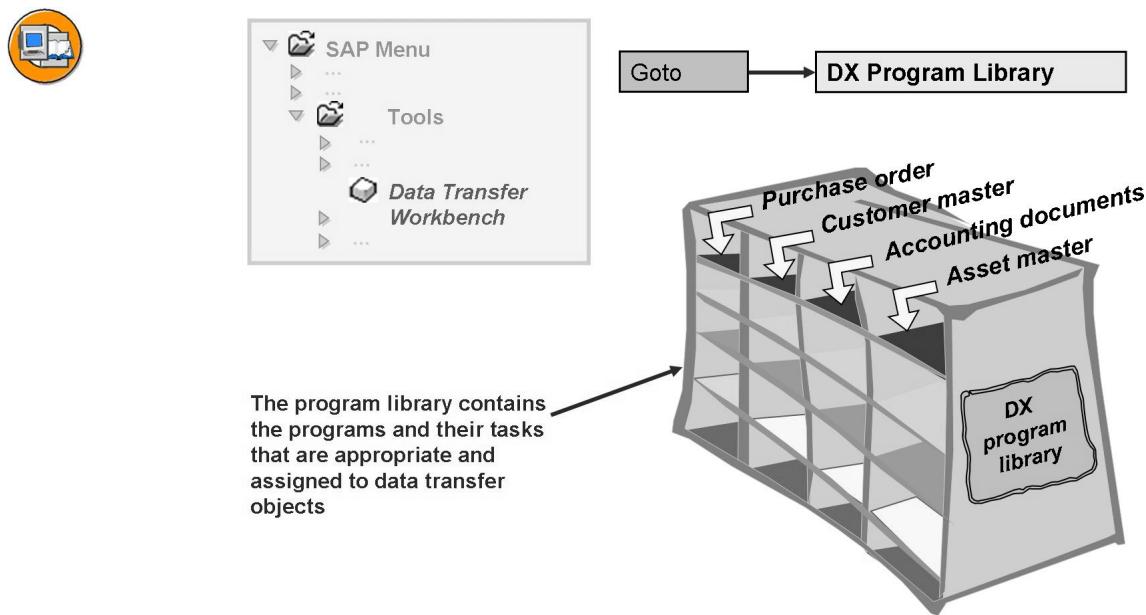


Figure 62: DX-WB: The Program Library

In the DX-WB under *Goto -> DX Program Library* you will find an overview of all business objects that already have a program registered to them.

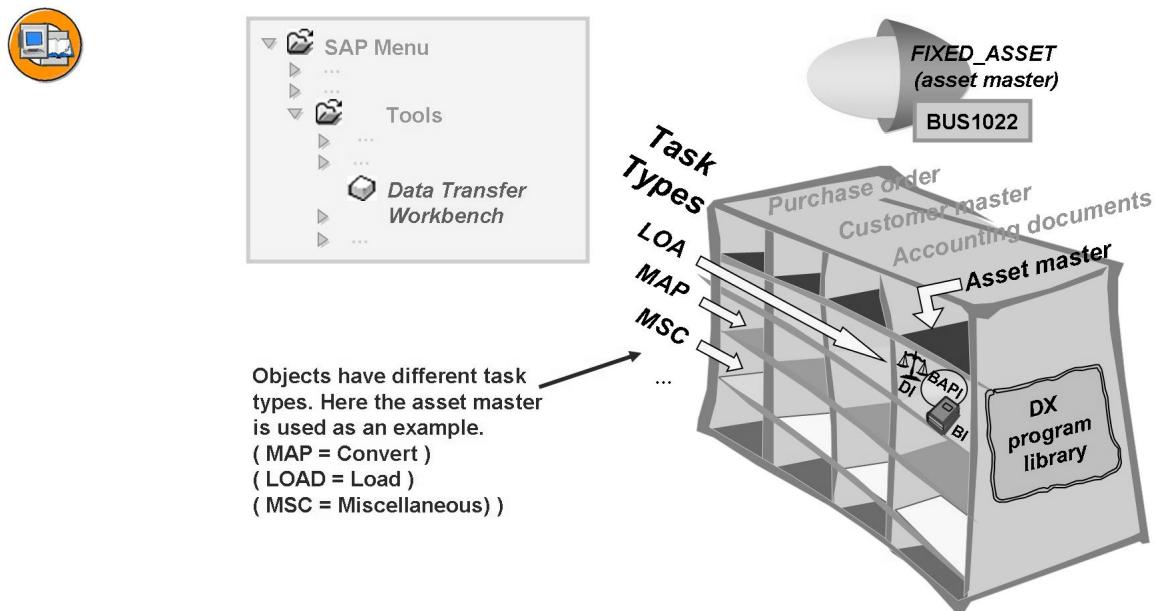


Figure 63: DX-WB: Library Task Types

SAP provides standard transfer programs or BAPIs for the majority of objects. These are registered as task type LOA (load).

If you require different data transfer programs, you will have to create them.

If you want to use your own programs in the DX-WB, you have to register them here.

Your programs will be assigned task types when you register them. For example, the task type “MAP” is for converting external data.

- **Note:** When the DX-WB was created, there was no naming convention for the names of the individual data transfer steps. Up to now, names were always handled differently in the documentation. From now on, we will use the term “conversion” to include the substeps “formatting” and “assigning” (“mapping”) the data. The DX-WB uses the name “MAP” for the whole conversion step. Please keep to this for the rest of the course.
- **Note:** The DX-WB does not have an extra step for formatting or mapping. If, for example, you want to register your own formatting program in the DX-WB (this will be shown later), you must assign it under “MSC”.



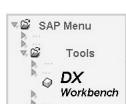
Overview



Tools

Projects

Figure 64: DX-WB Basics (2)



- **Transfer Files**

- **Check**
- **Display, create, edit**
- **copy**
- **Split and merge**

- **Create conversion proposal**
- **Display and generate import structures**
- **Create and display mapping plan**
- **Navigation in LSMW**



Figure 65: DX-WB Tools: Overview of the Functions

To use the DX-WB tools, you must decide on the data transfer program or method you want to use.

Proceed as follows:

- Select a suitable object type from your application.
- Select the supporting program type.
- Select the data transfer program or method.



Hint: Use the *F4* help.

Then you must decide on a name for the file and the directory it is created in. This can be a physical or logical path or file name. (For more information about logical file names see the unit on sequential files).

For Release 4.6, the path is: *Goto → Analyze Files and Data Structures*.

In higher releases, the menu entry is *DX-WB Tools*.

 **Note:** Note that the displays of some screens in the DX-WB may contain release-dependent components.

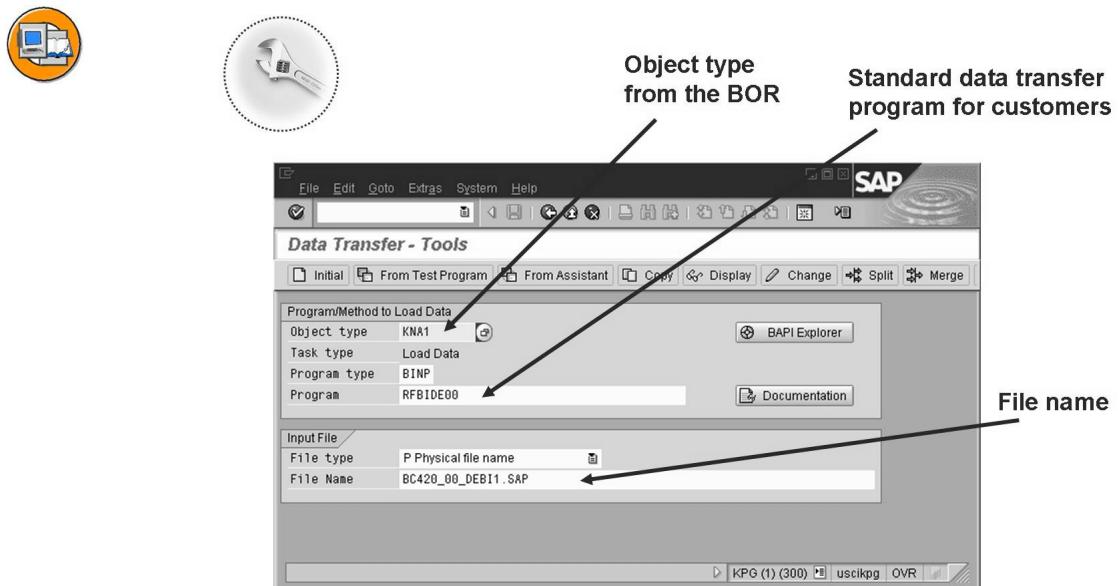


Figure 66: DX-WB Tools: Navigation

To use the DX-WB tools, you must decide on the data transfer program or method you want to use.

Proceed as follows:

- Select a suitable object type from your application.
- Select the supporting program type.
- Then choose the supported program or method (the easiest way to do this is to use the input help *F4*).

Then you must decide on a name for the file and the directory it is created in. This can be a physical or logical path or file name. (For more information about logical file names see the unit on sequential files).

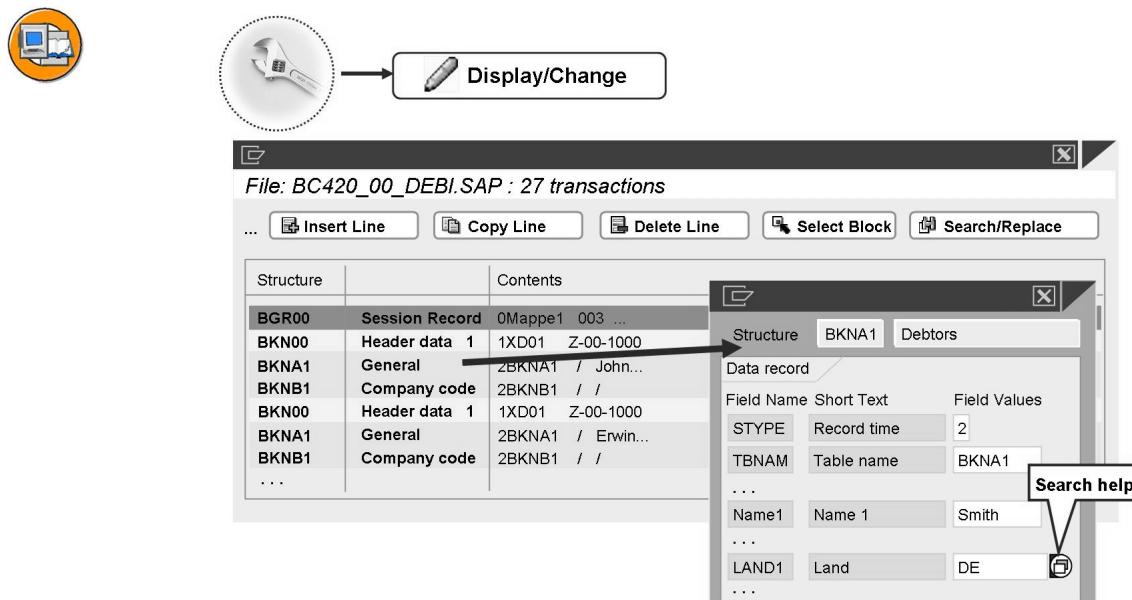


Figure 67: Displaying and Changing the Transfer File

The file display contains the following information:

- The number of data records is shown in the title. This number tells you how many transactions are contained in the transfer file.
- The fields in the transfer object are displayed in a predefined length and sequence for each structure. You can identify which structure fields of which length and in which sequence are required. If you double-click on a structure line, you can see the fields contained in the structure you selected.
- The various hierarchy levels of a data record have different colors. In record layers, hierarchy level 1 is green, level 2 yellow, level 3 blue, and level 4 is gray.
- You can print the transfer file (*Table → Print*).

Note that only the first 100 characters of each structure can be displayed in the overview.

In change mode you can enter values for the individual fields. Search help displaying a list of valid values is provided for many fields.

You can also change the field contents of records in change mode using *search and replace*.

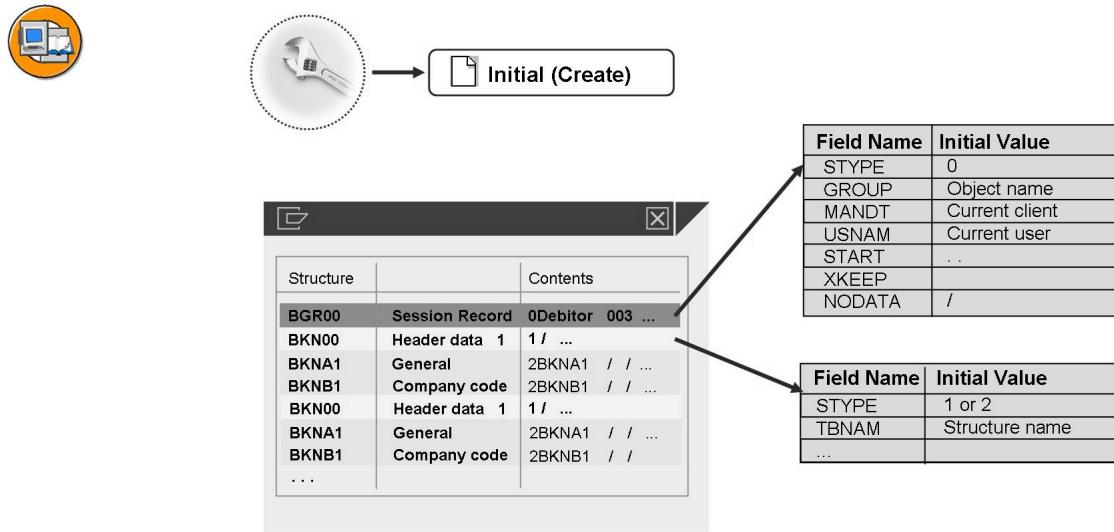


Figure 68: Creating an Initial Transfer File

If you create a file for a selected object and a method, you will get an initial file. This file contains all the structures that the object supports.

If you use a standard transfer program, the record layout will always contain a session header. This session header is automatically filled with predefined values.

Structures are only predefined for batch input or direct input with the NODATA character. This is not required if you use BAPIs or IDocs to transfer data.

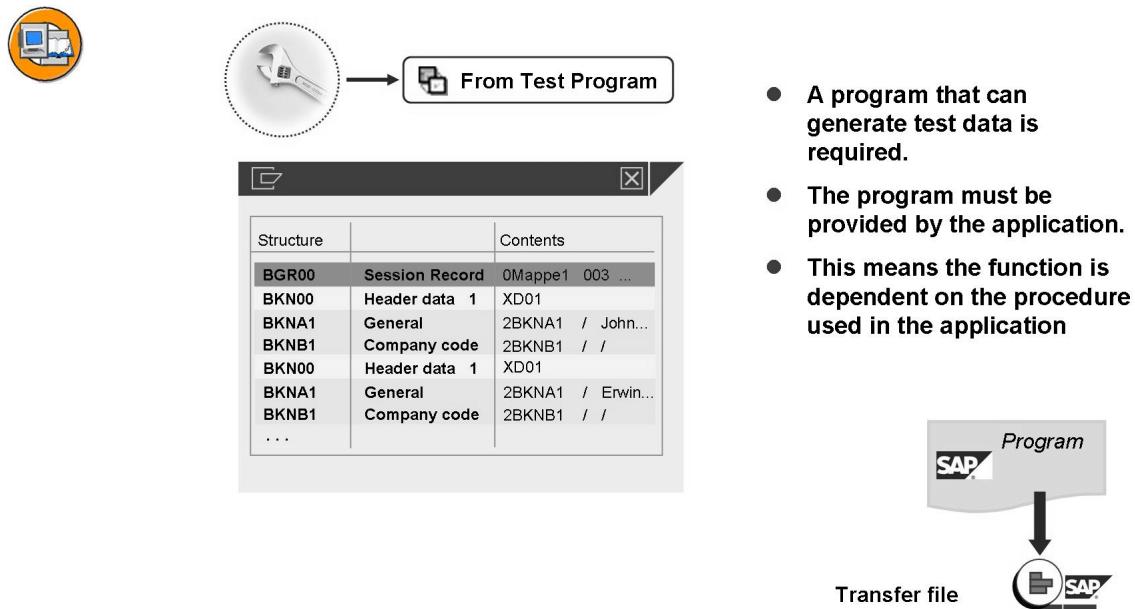


Figure 69: Creating a Transfer File with Test Data

For some data transfer projects a file containing test data can be created. To do this, data from the SAP R/3 system is copied to a file.

This function is not available for all transfer programs. It is application-specific. Standard transfer programs (batch input, call transaction and direct input) and BAPIs can contain this function.

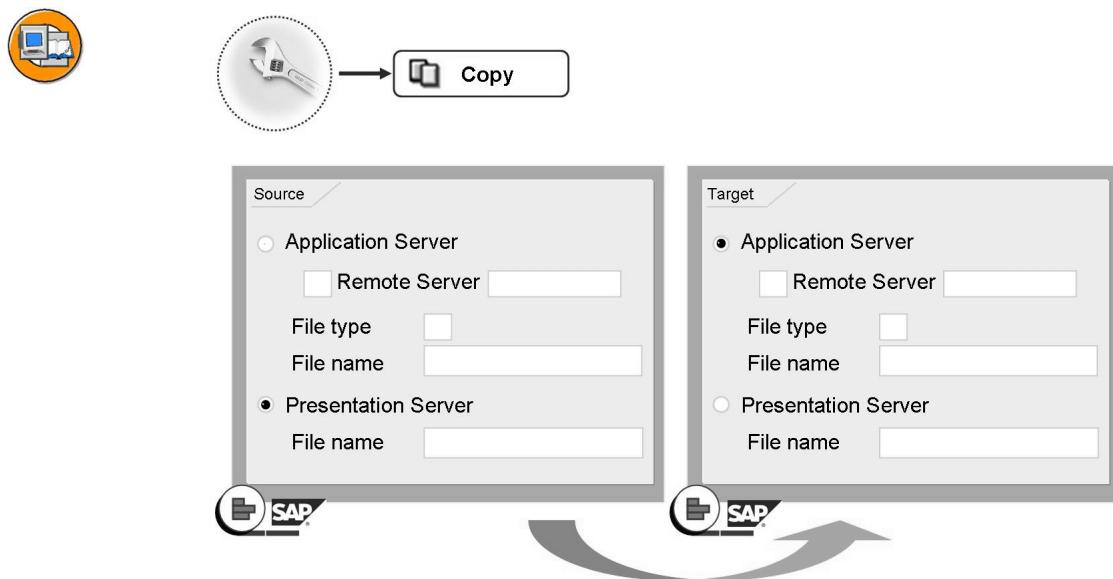


Figure 70: Copying Files for Data Transfer

The data transfer file can be copied between the application server and the presentation server.

The data transfer programs require the transfer file to have a fixed length.

The following conversions are possible:

- Copy without converting: The data transfer file is copied unmodified.
- Conversion fixed length → restricted by tabulator: This copy function can only be used when copying files from the application server to the presentation server. The data is copied to a format, in which the data is separated by a tabulator.
- Conversion restricted by tabulator → fixed length: This copy function can only be used when copying files from the presentation server to the application server. With this copy method the data is copied to a fixed length.
- You can also copy a file to another system. You can use the REMOTE SERVER to do this and you can work with a SM59 destination (see the documentation about FRCs).

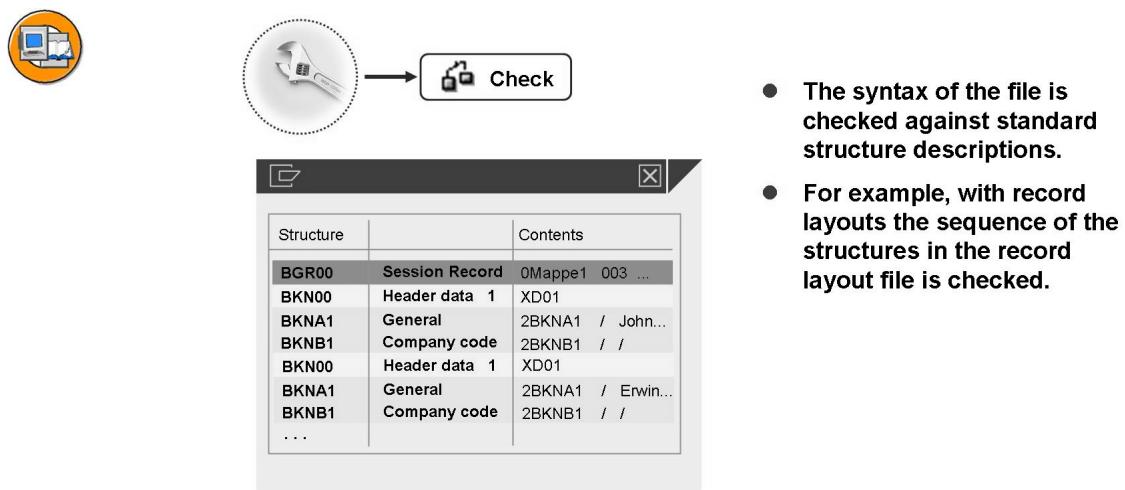


Figure 71: Checking the Data Transfer File

The syntax of the data transfer file can be checked for correctness.

- With record layout files the sequence of the record layout structure is checked.
- When BAPIs or IDocs are used for the transfer, all the EDI inbound message checks are made.

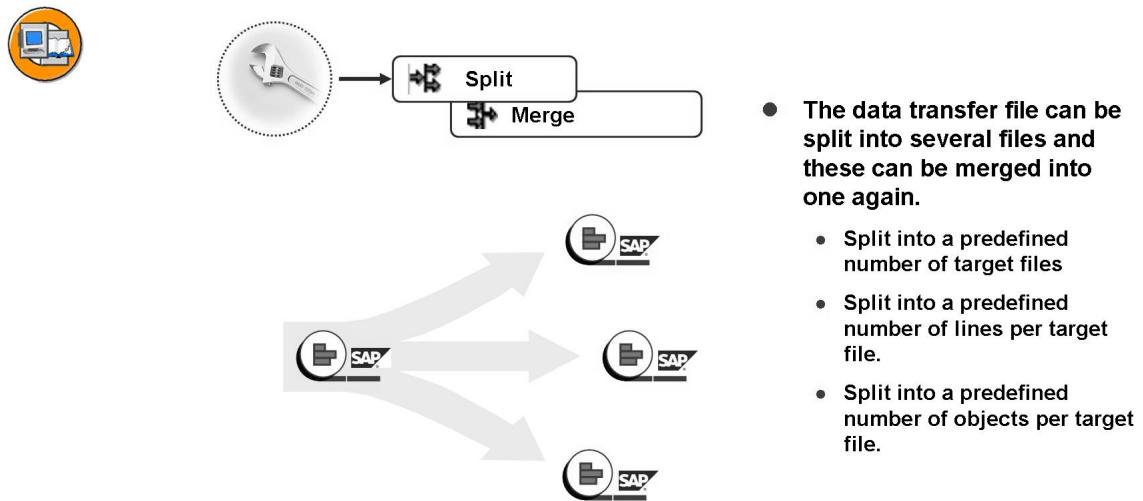


Figure 72: Splitting and Merging the Transfer File

The transfer file can be split up according to your chosen criteria:

- Into defined target files
- Into predefined number of lines per target file
- Into a predefined number of objects per target file

These target files can be merged again.

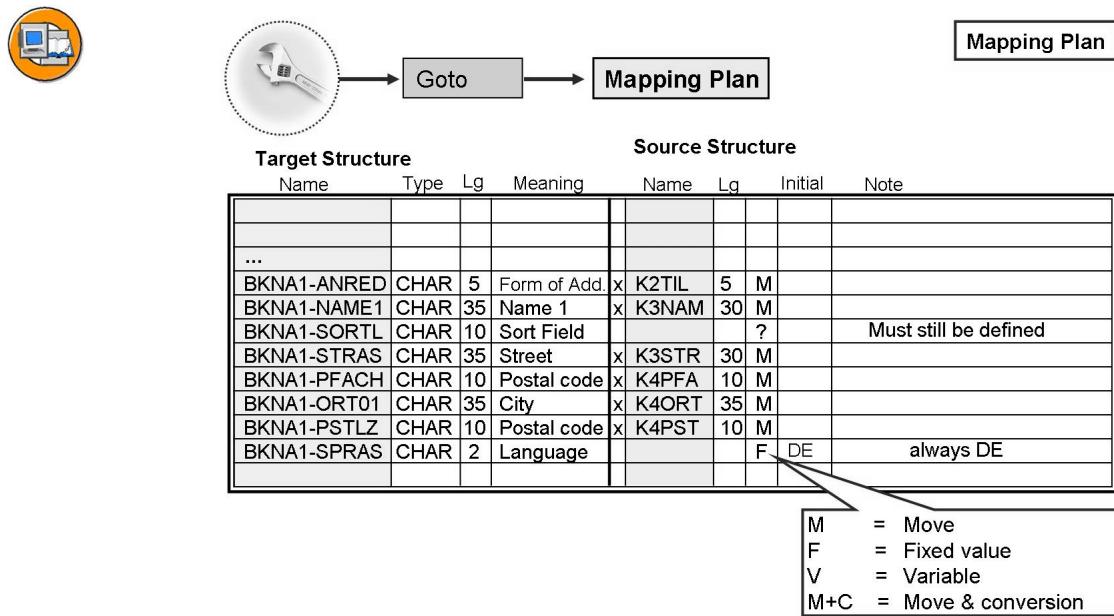


Figure 73: The Mapping Plan The Mapping Plan

→ **Note:** You can print the mapping plan and fill it from the LSMW. We recommend that you do this because then the system will provide an additional column for the “rules”. This will be discussed later in the LSMW unit. Therefore the graphic does not correspond exactly to the DX-WB.

Before you create test data, the fields in the external system should be formatted and assigned to the fields in the SAP system. The best way to do this is in a table.

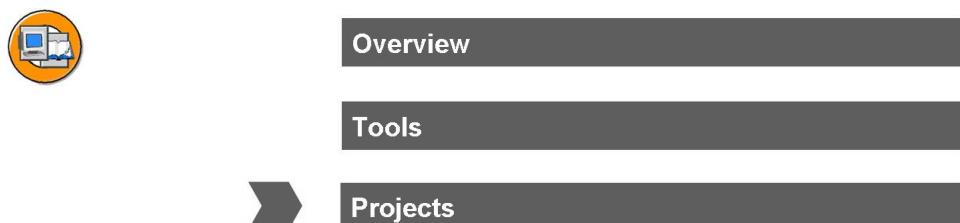
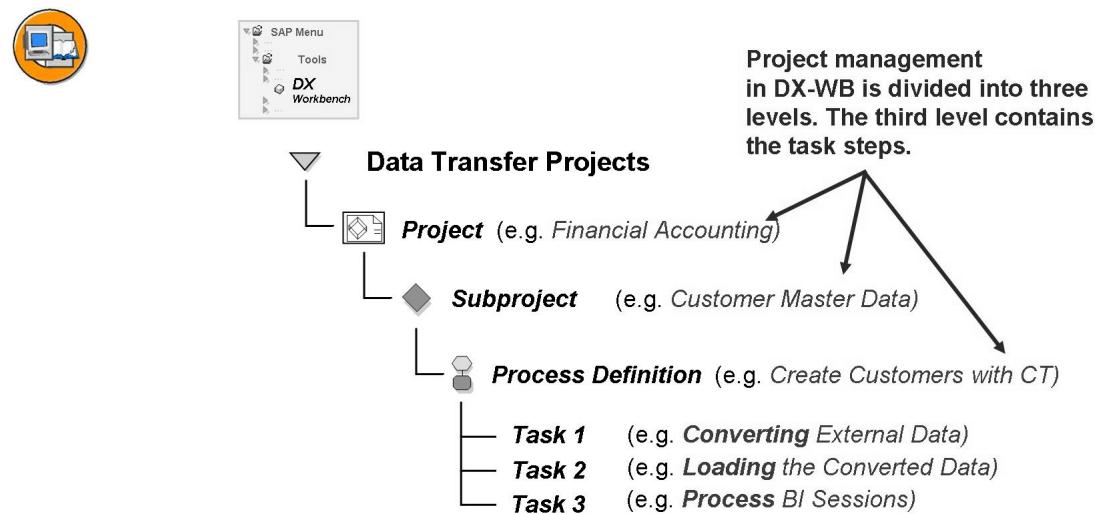


Figure 74: DX-WB Basics (3)

**Figure 75: Project Management in DX-WB**

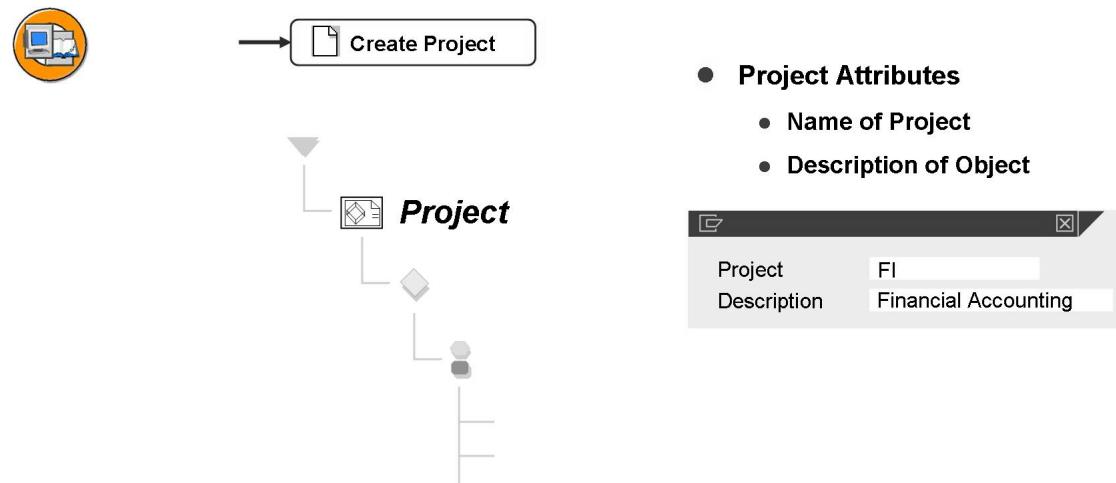
You can create a transfer project in the DX-WB. You can create, change, display, and delete projects, subprojects, process flows, and tasks.

The DX-WB functions are explained on the following slides by example of customer data transfer.

The external data must be located in the file system of the SAP application server.

The following steps are necessary:

- Convert data
- Load Data in Batch Input Sessions
- Processing Sessions Automatically

**Figure 76: DX-WB: Creating a Project**

A project is used to group business objects together.

For **each business object type** that you want to transfer, you have to create at least one **subproject** and at least one process flow definition and one task for each subproject.

The name of the project can be assigned once only in the system, as it is defined as a key and must be unique in the system.

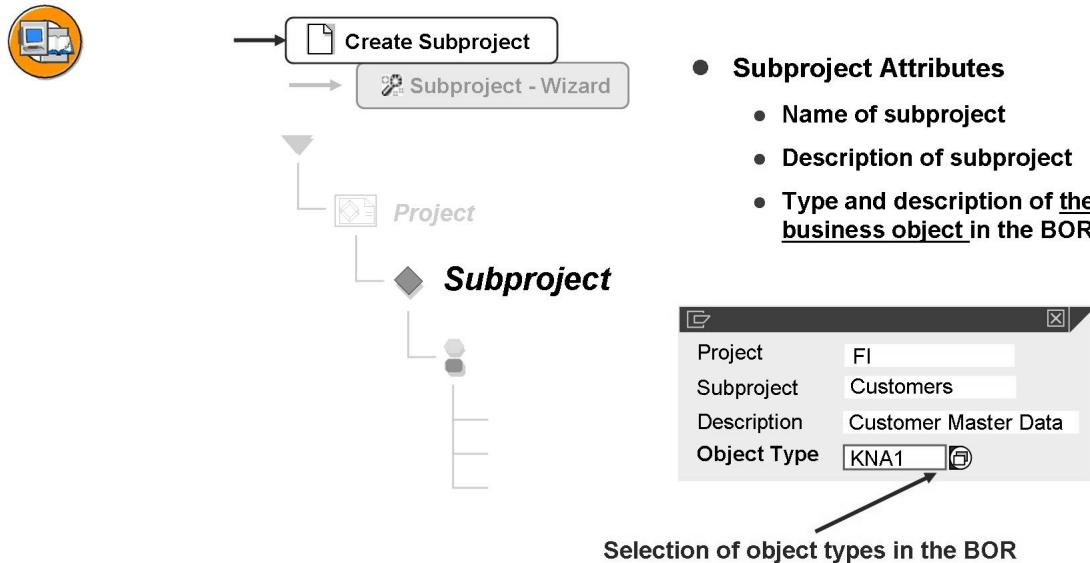


Figure 77: DX-WB: Creating a Subproject

The subproject is used to split a project up. Precisely **one business object type is assigned to each subproject**, and this business object type is transferred in the subproject. You can also create several subprojects for the same business object type within one project (for example, subproject “document data transfer from plants 1, 2 and 3”).

If you cannot find the object you require in the list of available objects by looking at the short description, you can find further information in the documentation for the application.

The name of the subproject can be assigned once only in the system, as it is defined as a key and must be unique in the system.

As of Release 6.20 the **subproject wizard** can also be used to create the subproject and all its subnodes.

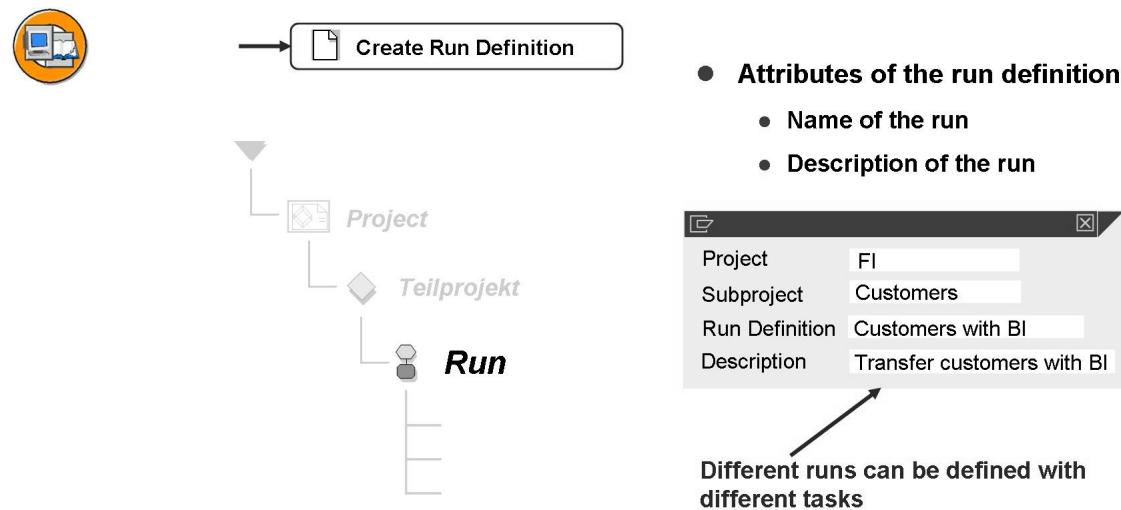


Figure 78: DX-WB: Creating a Run Definition

In the run definition you decide **how the data is transferred for a subproject**, by creating the tasks and specifying the files (see *task(s)*). You can create several run definitions for one subproject. Depending on the type different methods are possible for each run.

You can transfer data in parallel by creating several run definitions with different file names and then starting runs in parallel.

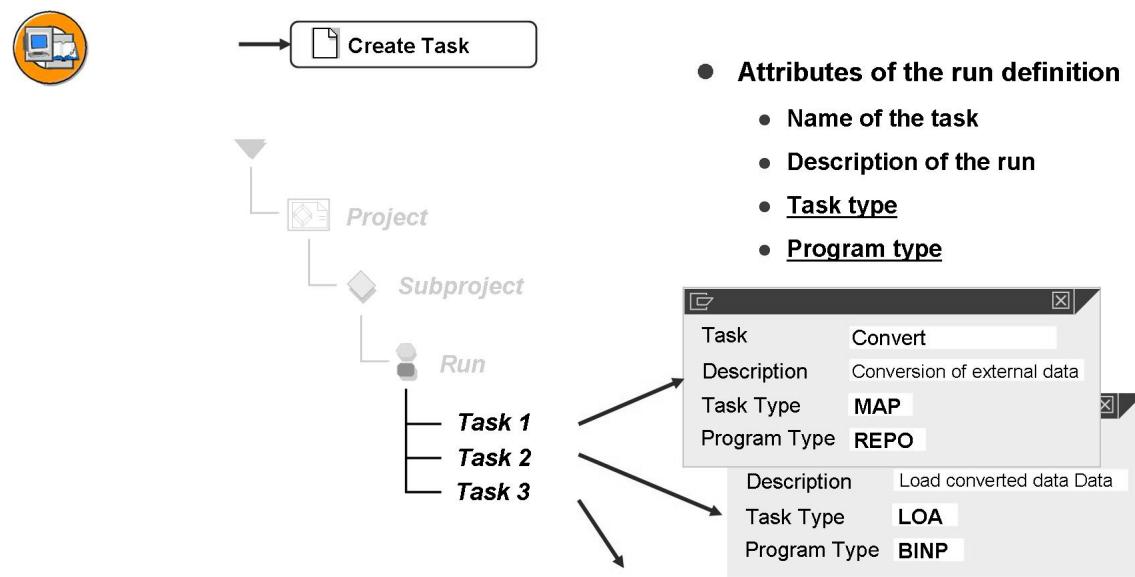


Figure 79: DX-WB: Creating Tasks

The run steps, called tasks are assigned in a run definition or run. The tasks are processed in the sequence defined.

There are different task types for different data transfer requirements, for example:

- Converting data: task type MAP
- Loading data: Task Type LOA
- Miscellaneous: task type MSC

 **Note:** The DX-WB does not recognize a conversion. Here, this step is called **MAP**.

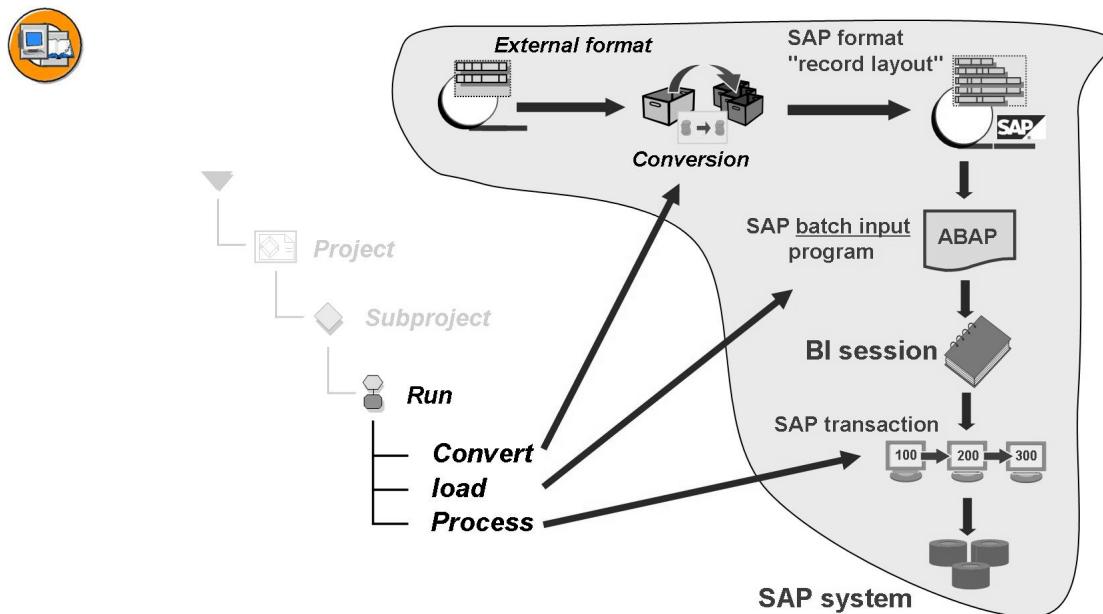
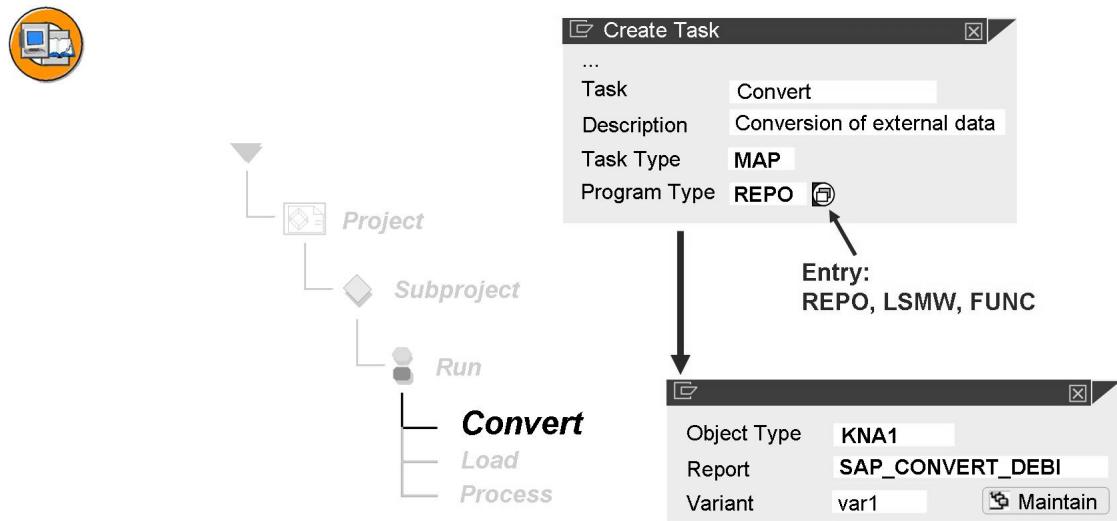


Figure 80: DX-WB: Overview of Tasks

In the example, the run definition is made up of the three task steps: Converting, loading and processing the data using batch input.

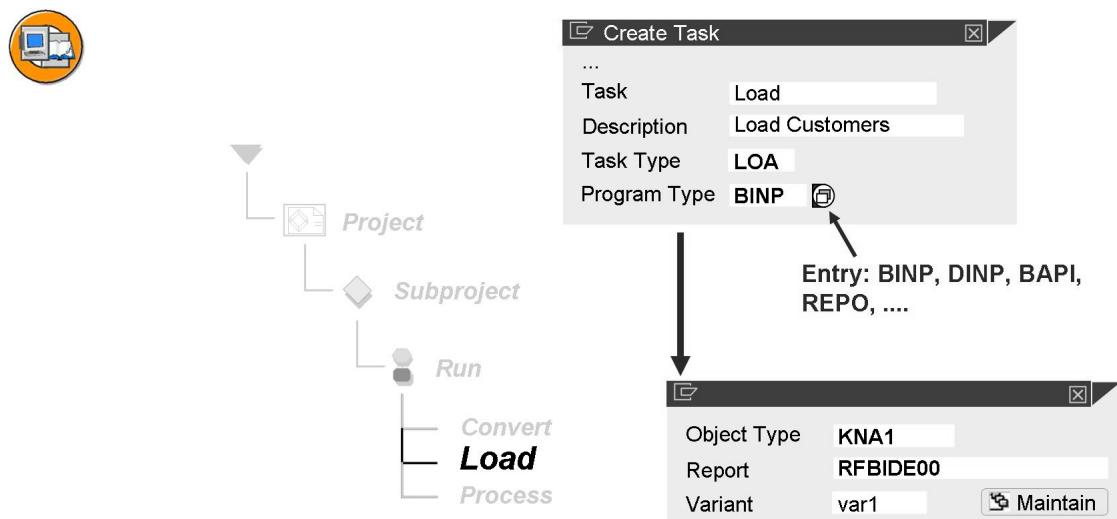
**Figure 81: DX-WB: DX-WB: Converting External Data**

To convert data the task type MAP is used in the DX-WB.

The data can be converted by a registered report (REPO), by the LSMW (LSMW), or by a function module (FUNC). A report, function module, or an LSMW project must always be integrated beforehand.

In the example shown above, report SAP_CONVERT_DEBI is entered, which converts the customers from their external format into the SAP record layout format.

This report must be already registered in the DX-WB. The relevant procedure will be mentioned later. The report is not contained in the SAP standard system, rather it has been developed exclusively for training purposes.

**Figure 82: DX-WB: Loading the Converted Data**

To load or import data, task type LOA is used in the DX-WB.

Depending on the application and the procedure used there, the data can be loaded using various methods. (batch input, call transaction, direct input, BAPI and so on). In the above example, the **program type BINP**, (batch input method) is used. The SAP program used for importing the record layout file and creating the BI sessions is **report RFBIDE00**. This report is already registered in the DX-WB as a loading program for customers and is therefore proposed here.

You can create a variant for RFBIDE00 on the maintenance template.

The path and the file name of the record layout file that is imported and processed are important parameters on the selection screen of RFBIDE00.

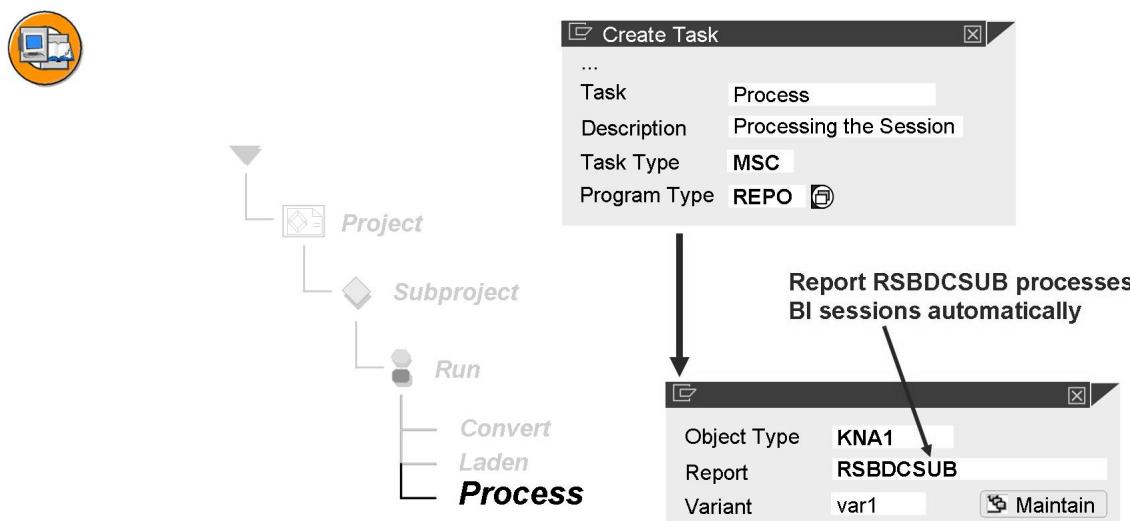


Figure 83: DX-WB: Processing the Batch Input Session

The batch input sessions created in the previous task step could be processed manually in the batch input monitor. This step can also be automated using task type MSC and report RSBDCSUB. The report must be registered beforehand, (see next course unit).

SAP report RSBDCSUB can automatically process several batch input sessions parallel in several batch work processes.

RSBDCSUB is not application-specific.

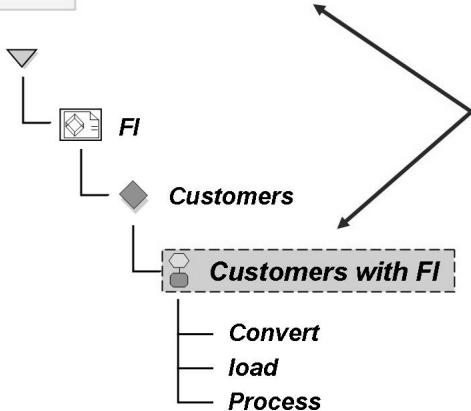
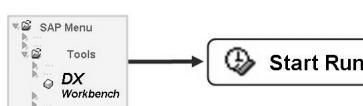


Figure 84: DX-WB: Starting a Run

The actual data transfer from a file is carried out in a run. When a run is started, the associated programs are called and all the tasks in the run definition are executed one after the other. The run is started by selecting the run within the project or subproject and choosing *Start Run*.

You can execute as many runs as you want for one run definition. Each run has a status and an application log. You can continue a run that has not been completed.

You have to define the status of a task online after a task of program type REPO, BINP or DINP has run through. All other tasks define their status automatically and are therefore also executed automatically one after the other.

The run will stop if a task error occurs.

Status information is available for each run of a run definition started:

- Task cancelled
- Task completed without errors
- Task completed with errors
- Task completed with errors, no restart
- Task ended manually

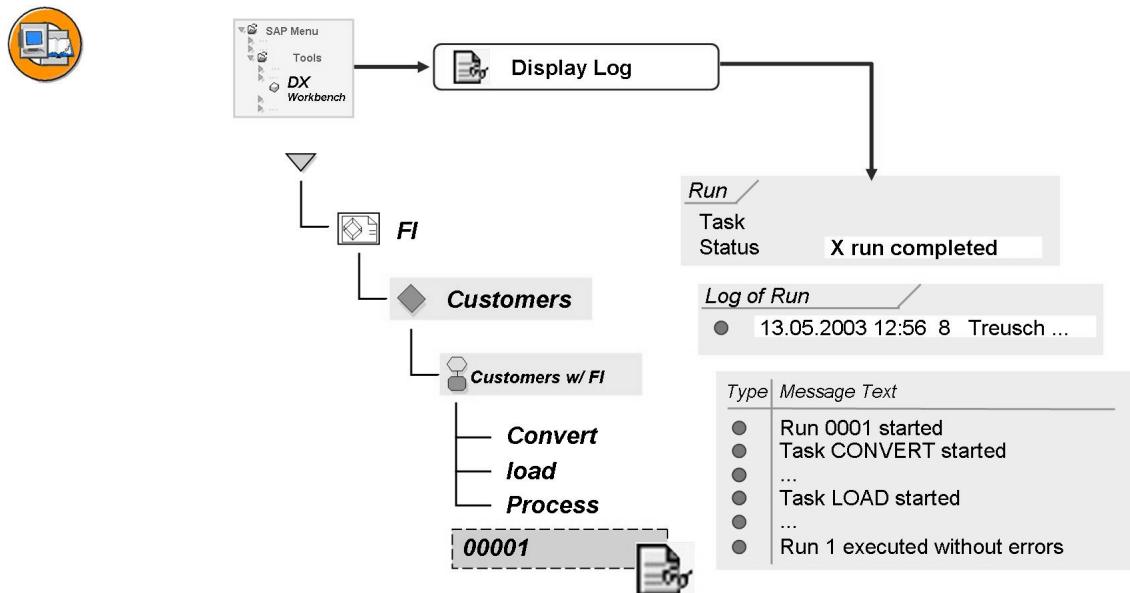


Figure 85: DX-WB: Run Log

A log is available for each run of a run definition started. You can display these by choosing *Display Log*.

All the actions of the project run are logged. Each task executed in the run may leave messages in this log. The log of a project run log is located in the application log.



Caution: The DX-WB was developed for BAPI transfer procedures. You can also use it for procedures that work with other techniques, however various restrictions apply here. Note that if you use it for other procedures, errors that occur may not be recorded in the DX-WB log. In addition, after the run has been processed, you must use transaction SM35 to search for any errors that may have occurred. The older batch input, call transaction and direct input programs were not designed to be able to write to the DX-WB log. As a result, errors during the processing of these programs always lead to green traffic lights in the DX-WB log.

Therefore, the use of the functions *Cancel Run*, *Continue Run* and so on cannot be demonstrated until later in the course when the BAPI procedure is discussed.

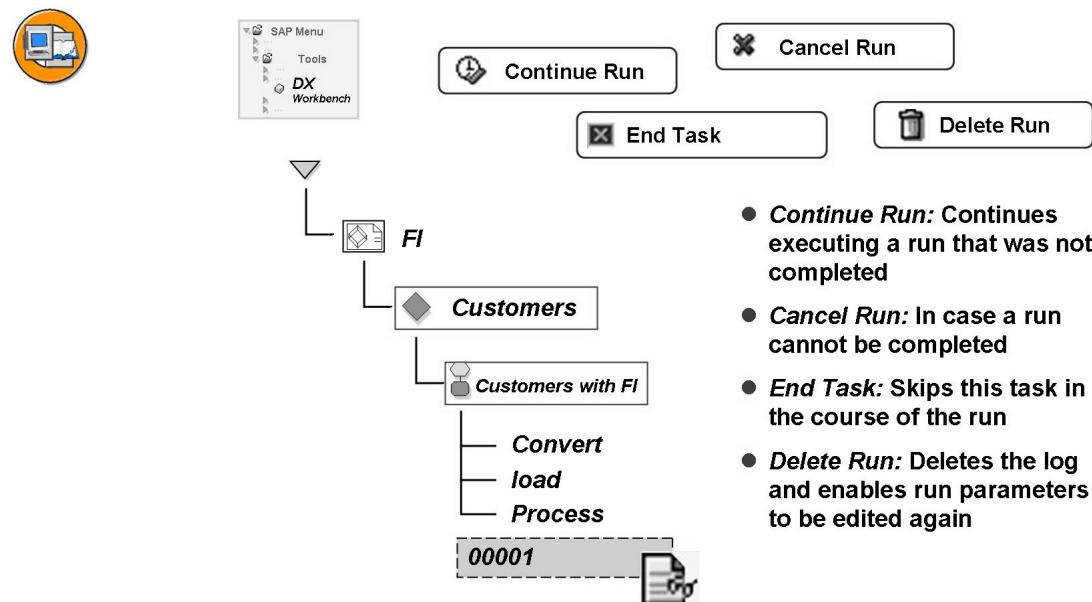


Figure 86: DX-WB: Run Errors

You have the following possibilities to handle a run that was not completed successfully:

- Continue Run:
To complete a run that was interrupted, use the log to correct the error, place the cursor on the run ID (for example, 003) and choose *Continue Run*.
- Cancel Run:
If a run cannot be completed, cancel it by choosing *Cancel Run*. Only then can other runs be started.
- End Task:
You can set a run that was not completed to completed by choosing *End Task*. The task is skipped when the run is continued.
- Delete Run:
You can delete runs that have been completed or manually cancelled by choosing *Delete*.

A log is available for each run of a run definition started.

Exercise 4: Working with the DX-WB Tools

Exercise Objectives

After completing this exercise, you will be able to:

- Work with the Data Transfer Workbench tools, and copy and edit files.
- Create a mapping plan
- Create files in SAP format and enter test data in them.

Business Example

Template Record Layout: BC420_##_DEBI1.SAP

New record layout: BC420_##_DEBI_EDIT.SAP

Task:

1. Go to the DX-WB tools and look at your old record layout file BC420_##_DEBI1.SAP from the last exercise.
2. Make a copy of this record layout file and call it *BC420_##_DEBI_EDIT.SAP*. During the course of this exercise changes will be made to this file. Check the result in the file monitor.
3. You will find ten customers in record layout format in the copied record layout file. Delete all the record layout structures for the last eight customers from this file (numbers Z-##-10003 – Z-##-10010). Afterwards there should be only two customers with their record layout structures remaining.
4. Change the customer number of these two customers in the relevant record layout structure, as follows: The sixth digit of the prefix of the number should now be a “2”. The prefixes should be: Z-##-20001 and Z-##-20002. These two new customers (only the number has changed, not the address data) should be updated with the external number assignment.
5. Verify that these customers have been updated with the external number assignment (the account group is set to KUNA in the relevant record layout structure).

Solution 4: Working with the DX-WB Tools

Task:

1. Go to the DX-WB tools and look at your old record layout file BC420_##_DEBI1.SAP from the last exercise.
 - a) The transaction code for the DX-WB is SXDA. To get to the tools choose *Goto → Tools*. Set the template parameters as follows:

Object Type:	KNA1
Program type:	BINP
Program:	RFBIDE00
File type:	P
File name:	BC420_##_DEBI1.SAP

Display the file contents by choosing the *Display* icon.

2. Make a copy of this record layout file and call it *BC420_##_DEBI_EDIT.SAP*. During the course of this exercise changes will be made to this file. Check the result in the file monitor.
 - a) You can copy the file by choosing the *Copy* icon. The original file on the application server is copied to the new file **BC420_##_DEBI_EDIT.SAP**, which is also on the application server. Check the results in the file monitor (transaction AL11).
3. You will find ten customers in record layout format in the copied record layout file. Delete all the record layout structures for the last eight customers from this file (numbers Z##-10003 – Z##-10010). Afterwards there should be only two customers with their record layout structures remaining.
 - a) You can edit the file by choosing the *Change* icon. Select the third customer by choosing the *Select Block* icon. Go to the last customer in the list and place the cursor on this customer's record layout structure, and choose *Cut*.

Then choose *Save*.

Continued on next page

4. Change the customer number of these two customers in the relevant record layout structure, as follows: The sixth digit of the prefix of the number should now be a “2”. The prefixes should be: Z-##-20001 and Z-##-20002. These two new customers (only the number has changed, not the address data) should be updated with the external number assignment.
 - a) Now there are two customers in the changed record layout file. The customer number can be changed in the record layout structure BKN00.
5. Verify that these customers have been updated with the external number assignment (the account group is set to KUNA in the relevant record layout structure).
 - a) The Account Group field is also in structure BKN00. It should be set to KUNA (external number assignment). Once you have made these changes you can check your new file in transaction AL11.



Hint: Check the results of your changes in the file monitor.

Exercise 5: Creating a New Test Record Layout

Exercise Objectives

After completing this exercise, you will be able to:

- Create a new record layout file with a test record.

Business Example

Practice using the DX-WB tools to deepen your understanding of data transfer using DX-WB.

Task:

1. Create a **new** record layout file *BC420_##_DEBI_TEST.SAP*.
2. Enter the following data into the record layout structures:
 - a) Check and extend the session header
 - b) In the additional structures, extend the data according to the following table:

Field description:	Field Value
Customer number	Z##-20003
Account group	KUNA
Company code	0001
Transaction code	XD01
Customer name	Any
Search term	Any
Street	Any
City	Any
Postal code	Any
Country	DE
Language	D
Sales tax ID	DE123456789
Reconciliation account	120000

Solution 5: Creating a New Test Record Layout

Task:

1. Create a new record layout file *BC420_##_DEBI_TEST.SAP*.
 - a) You create the new record layout file *BC420_##_DEBI_TEST.SAP* in the DX-WB tools by choosing the *Initial* icon.
2. Enter the following data into the record layout structures:
 - a) Check and extend the session header
 - b) In the additional structures, extend the data according to the following table:

Field description:	Field Value
Customer number	Z##-20003
Account group	KUNA
Company code	0001
Transaction code	XD01
Customer name	Any
Search term	Any
Street	Any
City	Any
Postal code	Any
Country	DE
Language	D
Sales tax ID	DE123456789
Reconciliation account	120000

- a) To enter data in the structures you choose *Change* to go to the editor.
To enter data in the fields of a structure, double-click on the respective structure.



Caution: Input help is available for some fields.



Hint: Check the result in the file monitor.

Exercise 6: Working with the Project Management

Exercise Objectives

After completing this exercise, you will be able to:

- Create a project with its components in the DX-WB
- Start run in the DX-WB

Business Example

To transfer external data, several work steps can be automated within the transfer project using the DX-WB. External customer master data is used here as an example.

Task 1:

Create a project FI-## in the DX-WB.

1. Under this project create a subproject **DEBI-##** to transfer external customers. Use BOR object type **KNA1**.
2. Under the subproject create a run definition **BI-##** (the name of the run should indicate the batch input method used).

Task 2:

The run definition will cover two steps or tasks. Create these tasks under the run definition and name your two tasks **LOAD-##** and **RUN-##**.

1. Define a variant for program RFBIDE00 to transfer your record layout file **BC420_##_DEBI_EDIT.SAP**.
This file that was created previously contains customers that have not yet been imported with numbers Z-##-20001 and Z-##-20002.
2. Carry out task step **LOAD-##** (load data). (Enter task type **LOA**, program type **BINP**, **RFBIDE00** with a suitable variant).
3. Carry out task step **RUN-##** (process the batch input sessions). (Enter task type **MSC**, program type **REPO**, **RSBDCSUB** with the suitable variant **SUB-##-EX**).

Continued on next page

Task 3:

1. Start the run in the DX-WB and in particular check the record layout file used for RFBIDE00. If the correct variant has been used, the correct record layout file is automatically supplied when the run is executed. If the correct variant has not been used, you can correct the name of the record layout file manually when you execute the run. (Of course, this applies only to online execution, you cannot correct it in a batch run.) The name of the batch input session in the second step to be executed must also correspond. If there is an incorrect variant of RSBDCSUB, you can also intervene here (only online).

Solution 6: Working with the Project Management

Task 1:

Create a project FI-## in the DX-WB.

1. Under this project create a subproject **DEBI-##** to transfer external customers. Use BOR object type **KNA1**.
 - a) Start the DX-WB. Select *Data Transfer Projects* in the hierarchy display. Create project *FI-##* with the name *FI ##*. Use the pushbutton *Create Project*.
Select your project in the hierarchy display. Under this project create the subproject **DEBI-##** and name it Customers. Assign object type **KNA1** to it. Use the pushbutton *Create Subproject*.
2. Under the subproject create a run definition **BI-##** (the name of the run should indicate the batch input method used).
 - a) Select your subproject in the hierarchy display. Under this subproject create run **BI-##** and call it Transfer with batch input. Use the pushbutton *Create Run Definition*.

Task 2:

The run definition will cover two steps or tasks. Create these tasks under the run definition and name your two tasks **LOAD-##** and **RUN-##**.

1. Define a variant for program RFBIDE00 to transfer your record layout file **BC420_##_DEBI_EDIT.SAP**.
This file that was created previously contains customers that have not yet been imported with numbers Z-##-20001 and Z-##-20002.
 - a) Select your run definition in the hierarchy display. Create two tasks. The tasks are for the following process steps: Create a batch input session and run it. Use the pushbutton *Create Task*.

Continued on next page

2. Carry out task step **LOAD-##** (load data). (Enter task type *LOA*, program type *BINP, RFBIDE00* with a suitable variant).

- a) Create the following task in run BI-##:

Task:	LOAD-##
Name:	Create batch input session
Task type	LOA
Program type:	BINP

Choose program RFBIDE00 with an appropriate variant, to transfer your record layout file BC420_##_DEBI_EDIT.SAP. If the variant does not yet exist, you can create it in SA38, SE38, or SE80.

3. Carry out task step **RUN-##** (process the batch input sessions). (Enter task type *MSC*, program type *REPO, RSBDCSUB* with the suitable variant *SUB-##-EX*).

- a) Create the following task in run BI-##:

Task:	RUN-##
Name:	Process batch input session
Task type	MSC
Program type:	REPO

First familiarize yourself with the selection screen of program RSBDCSUB.

Then create a suitable variant for program RSBDCSUB to process the batch input sessions that were created in the loading step. Note that the name of the batch input sessions in the second step is decisive. If the name does not correspond to the task of the first step, no sessions are processed.

- b) You should check the sequence of the tasks. Loading must occur before processing.

If you need to, use the pushbutton  or 

Task 3:

1. Start the run in the DX-WB and in particular check the record layout file used for RFBIDE00. If the correct variant has been used, the correct record layout file is automatically supplied when the run is executed. If the correct

Continued on next page

variant has not been used, you can correct the name of the record layout file manually when you execute the run. (Of course, this applies only to online execution, you cannot correct it in a batch run.) The name of the batch input session in the second step to be executed must also correspond. If there is an incorrect variant of RSBDCSUB, you can also intervene here (only online).

a) Execute the run

Select the run (mouse click). To start the run, choose *Start Run*.

Program RFBIDE00 is called. The selection screen appears. Start the program. The batch input session is created. Choose the F3 button to exit the program. Confirm the status prompt, *Task completed without errors*.

 **Note:** In lower releases (lower than 6.10), you must confirm the success or failure of the task step manually in a dialog box.

Program RSBDCSUB is called. The selection screen appears. Start the program. The batch input session is processed. Choose the F3 button to exit the program. Confirm the status prompt, *Task completed without errors*.

 **Note:** In lower releases (lower than 6.10), you must confirm the success or failure of the task step manually in a dialog box.

Select the log and display it.

 **Note:** Note that data transfer procedures that work with the techniques batch input/call transaction/direct input always lead to green traffic lights in the DX-WB. You can view the actual success or failure of the data transfer in this example in the log in transaction SM35.



Lesson Summary

You should now be able to:

- Describe the most important functions of the DX-WB
- Use the main tools of the DX-WB
- Create a data transfer project in the DX-WB and transfer customer data



Unit Summary

You should now be able to:

- Describe the most important functions of the DX-WB
- Use the main tools of the DX-WB
- Create a data transfer project in the DX-WB and transfer customer data

Unit 4

Other DX-WB Functions

Unit Overview

This unit deals with advanced functions and tools in the DX-WB.



Unit Objectives

After completing this unit, you will be able to:

- Register your own programs in the DX-WB
- Create a conversion proposal using the conversion assistant
- Generate import structures

Unit Contents

Lesson: Other DX-WB Functions	100
Exercise 7: Registering Programs	113
Exercise 8: Creating and Working with a Conversion Proposal.....	117

Lesson: Other DX-WB Functions

Lesson Overview



- Data transfer library
- Conversion Assistant
- Special functions: import structures



Lesson Objectives

After completing this lesson, you will be able to:

- Register your own programs in the DX-WB
- Create a conversion proposal using the conversion assistant
- Generate import structures

Business Example

A company wants to use the DX-WB tool to transfer data. Determine and test the target fields of the conversion procedure.

Other DX-WB Functions



Registration

Conversion Assistant

Other Functions/DX WB (internal)

Figure 87: Other DX-WB Functions (1)

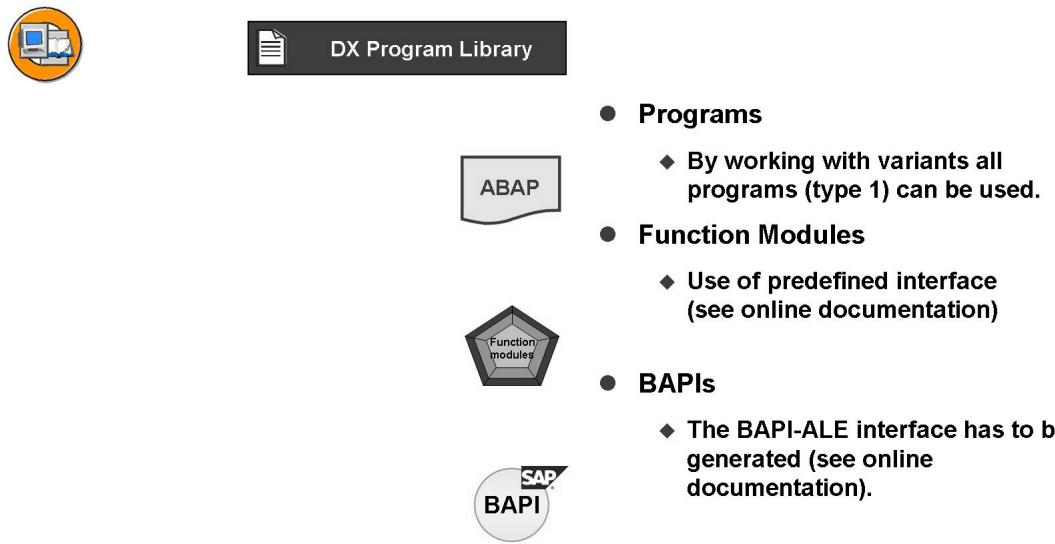


Figure 88: DX-WB: What Can Be Registered?

Data transfer methods can be used in the DX-WB only if they have been registered first. By default SAP methods are already registered and can be used in the tools or in the project environment. Methods you have developed yourself must always be registered first.

If you do register your own programs you can later start these programs with a variant from the DX-WB. In Release 4.6A/B this is only possible without variants.

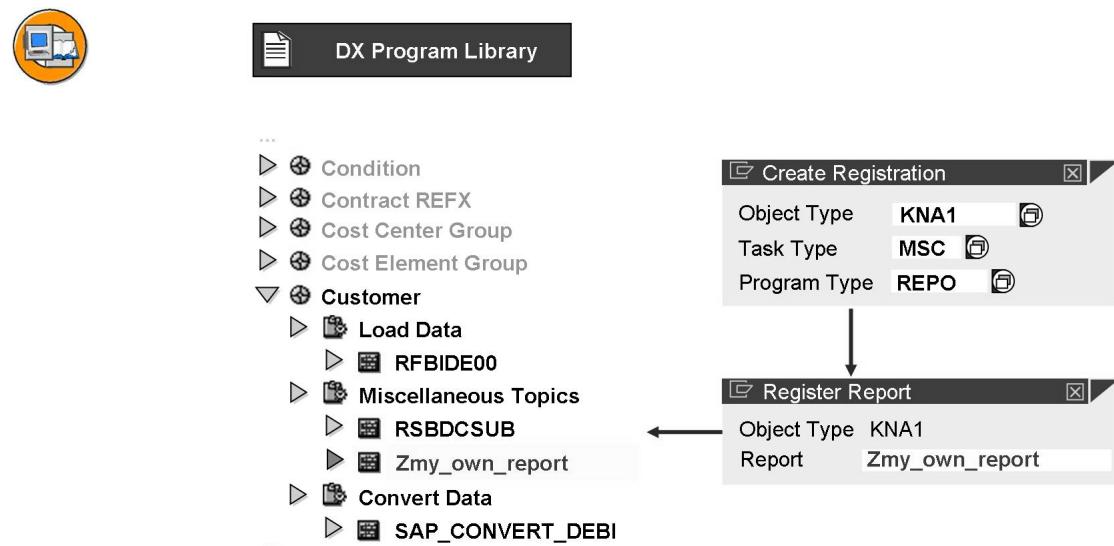
If you want to register function modules these should contain a special interface. This allows the log status to be set automatically, for example.

See the online documentation for more information.

To register a BAPI you have to:

- To use BAPIs you have developed in the data transfer workbench, you have to generate BAPI-ALE interfaces.
- If you want to supplement the BAPIs delivered by SAP for data transfer with your own BAPIs, refer to the BAPI Programming Guide.
- See the online documentation for more information.

→ **Note:** Note that some of the paths, screens, menus and icons displayed in the transactions may differ depending on the release.

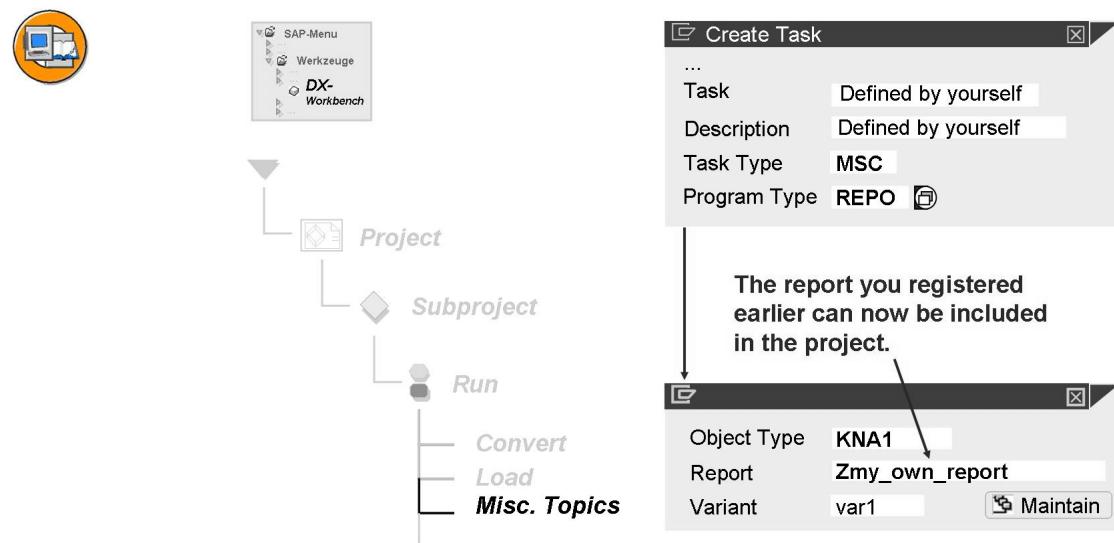
**Figure 89: DX-WB: Registration**

In the data transfer workbench choose menu path *Goto → Registration*.

- Select the relevant business object.
- Choose *Create Registration* and enter data in the required fields.



Caution: You must activate programs you have developed yourself, otherwise you cannot see them.

**Figure 90: DX-WB: Using your own Program**

The report (program) you created and registered can now be used in the DX-WB project environment or in the DX-WB tools.

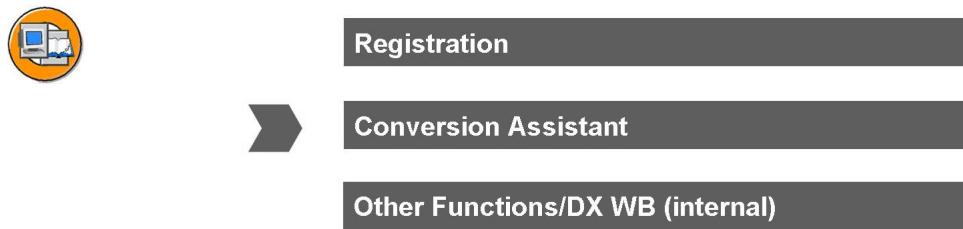
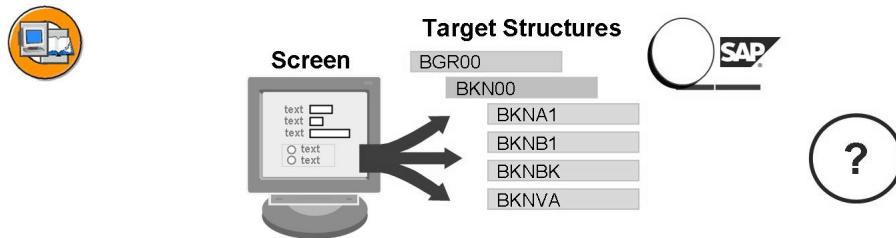


Figure 91: Other DX-WB Functions (2)



- The conversion assistant supports the conversion of external data into SAP target structures
- It supports the identification of relevant target fields.
- The transaction recorder is used for internal support.
- The fields from the recording are assigned to fields in the target structure.
- A target file with test data can be created from the result of the recording.

Figure 92: DX-WB: Conversion Assistant

The conversion assistant is provided as of Release 6.20.

It helps you to determine conversion rules, for example, rules that can be entered in the LSMW.

The conversion assistant works internally with the transaction recorder to record relevant fields on screens and afterwards reconcile them with the target structures.



Caution: The conversion proposal does not provide a 100% solution. It is intended to support you with assigning and identifying fields relevant for the target structures.

All the relevant fields for the data transfer must be entered in the transaction recorder.

All the recorded screen fields appear later in the conversion proposal and are assigned to target fields here.

The following slides show you how to create a proposal step by step.

-  **Note:** Before the proposal can be created, the *object type*, *program type*, and *import method* fields must be defined.

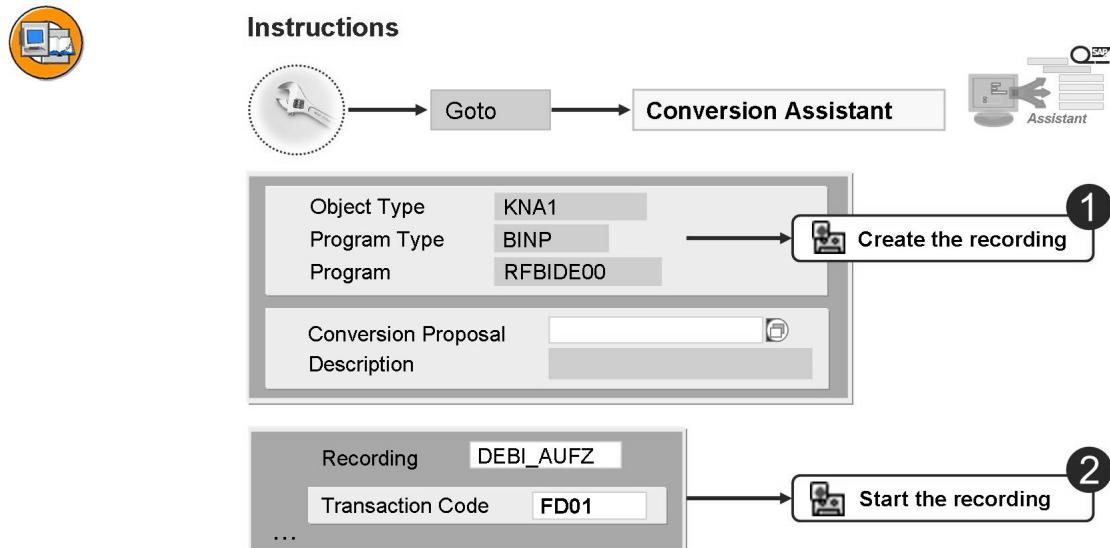


Figure 93: DX-WB: Creating a Conversion Proposal (1)

First go from the DX-WB tool environment to the conversion assistant. Before you can create a conversion proposal there, you must create a recording. See steps 1 and 2 on the first slide.

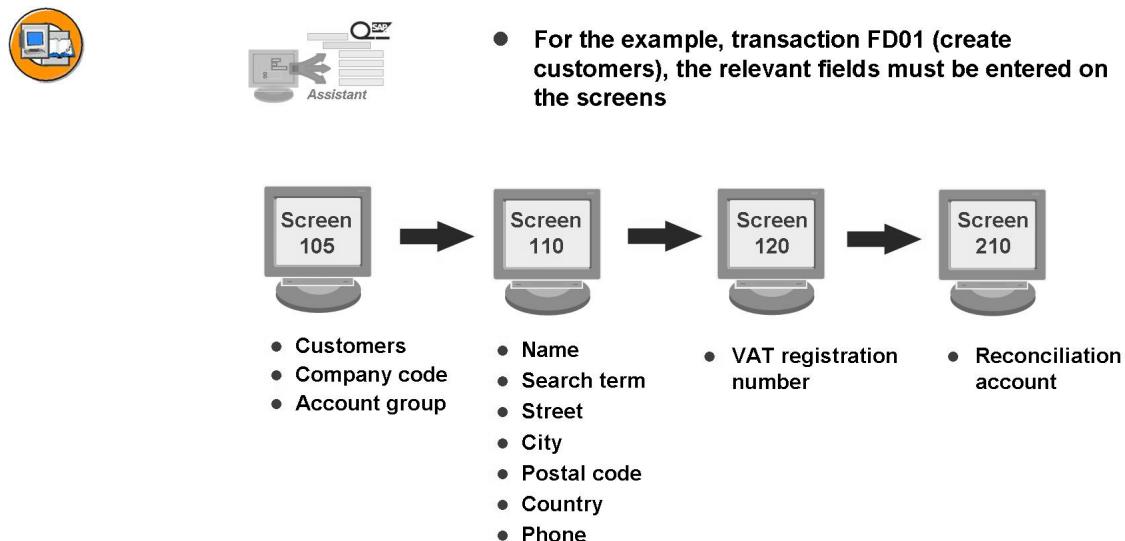


Figure 94: DX-WB: Recording XD01 - Example

Test the transaction before you create the recording. During the recording, all relevant fields are specified (all mandatory fields for the transaction and the fields from the external system). The recorder gives all recorded fields to the assistants for a later conversion proposal.

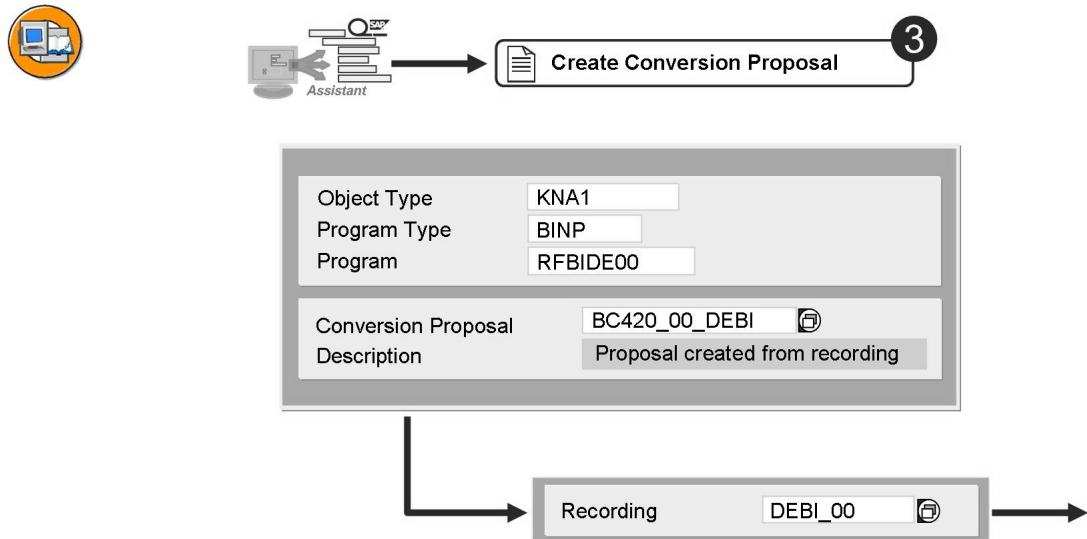


Figure 95: DX-WB: Creating a Conversion Proposal (2)

A conversion proposal is created after the recording.

The name of the recording must be specified (step 3).

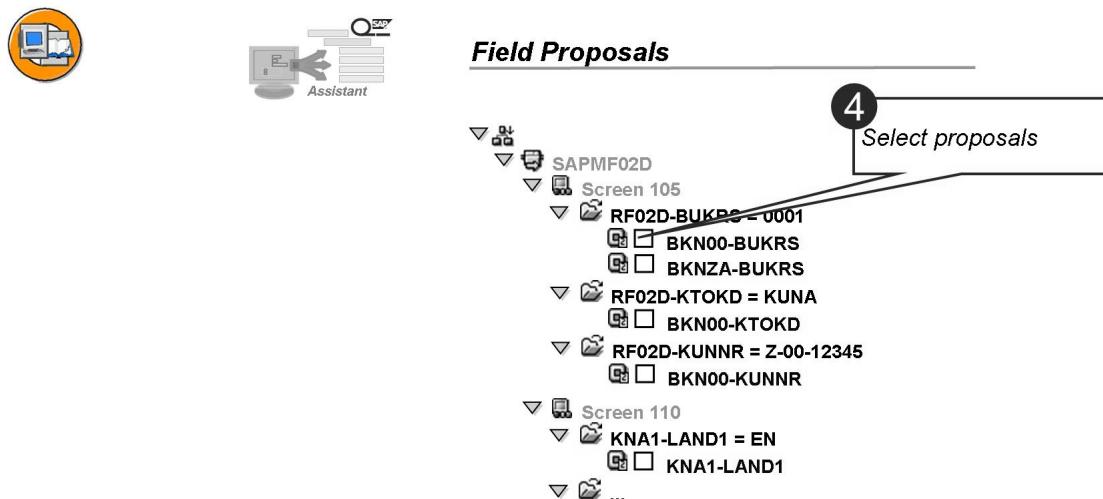


Figure 96: DX-WB: The Conversion Proposal

The conversion proposal displays the found assignments in a tree structure.

For example, for the company code one of the target fields BKN00-BUKRS or BKNZA-BUKRS can be assigned. The assistant cannot make this decision itself.

To **accept a proposal** for a field assignment the checkmark for the relevant field must be set manually.

If you do not know what to choose, you can create many different proposals here as long as you use the file created at the end of the whole procedure to post the transfer record. We recommend that you ask the user department for help and support so that you arrived at the required result as quickly as possible.

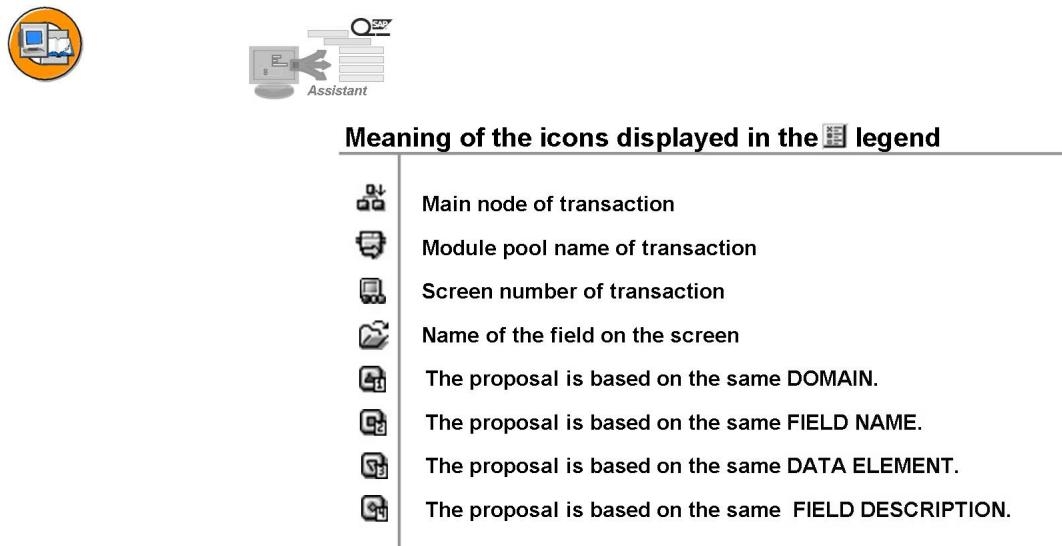


Figure 97: DX-WB: Conversion Proposal Icons

If you choose *Legend*, a description of the icons used is displayed.

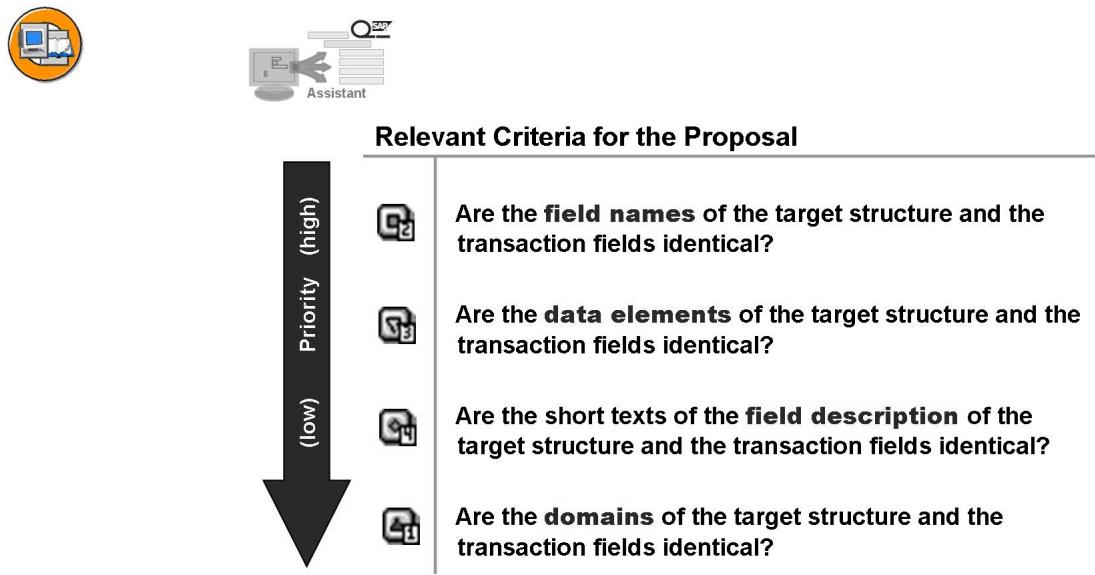


Figure 98: DX-WB: Relevant Criteria for the Proposal

The assistant prioritizes the criteria for displaying the conversion proposals as follows: Field name before data element before field description before domain.

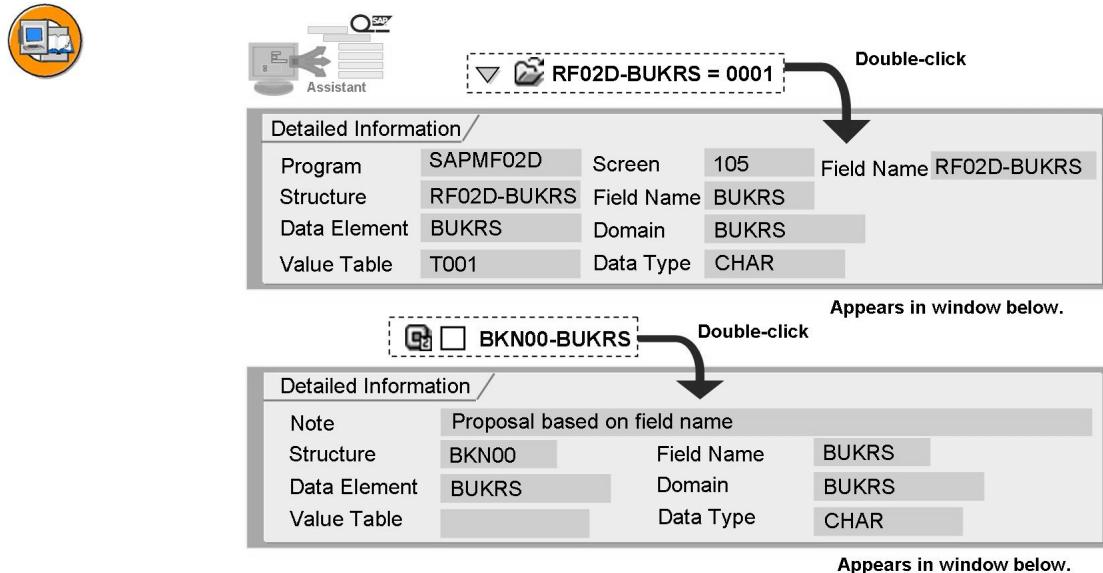


Figure 99: DX-WB: Display Details in Assistant

To display details about the fields, double-click the determined screen fields or the relevant target fields. When you do so, the system displays the module pool name, screen number, screen field, data element, value field and so on.

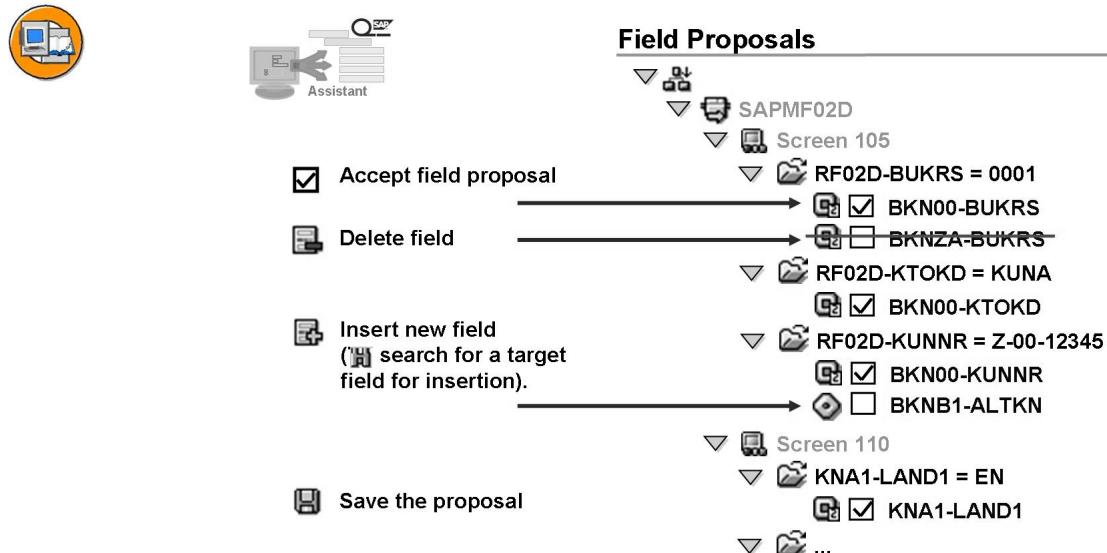


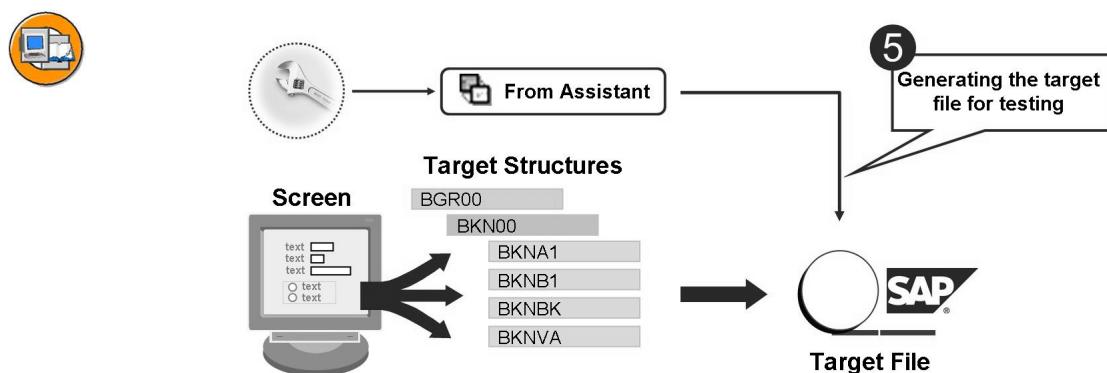
Figure 100: DX-WB: Working with the Proposal

The conversion proposal can be edited.

You can delete and add fields if you are sure that you will not need these selection fields in a later attempt.

However, you can manually insert a new field that has not been assigned by the assistant at any time.

To save the conversion proposal choose **Save**, otherwise you will lose the proposal and all changes.



- By using the conversion proposal a file can be created with the assignments from the proposal
- The created file contains the transaction data from the recording and can be used for further testing.

Figure 101: DX-WB: Creating a File with the Proposal

Finally, using the proposal created, a target file with a test data record is generated from the recording (step 5). Now you can try to carry out a transfer with this target file. If you are working with external number assignment here, use a new number to test the data transfer. If the transfer works, the assignments from the proposal are correct.

- The conversion assistant is provided as of Release 6.20.
- It helps you to determine conversion rules, for example, rules that can be entered in the LSMW.
- The assistant works internally with the recorder to assign relevant screen fields to the target fields of the target structures.
- The format of the target structure can be record layout or IDoc.
-



Caution: The conversion proposal does not generally provide a 100% solution. It is intended to support you with assigning and identifying fields relevant for the target structures.

- The result of the proposal is a file in SAP target format that you can test in a transfer.



Registration

Conversion Assistant



Other Functions/DX WB (internal)

Figure 102: Other DX-WB Functions (3)

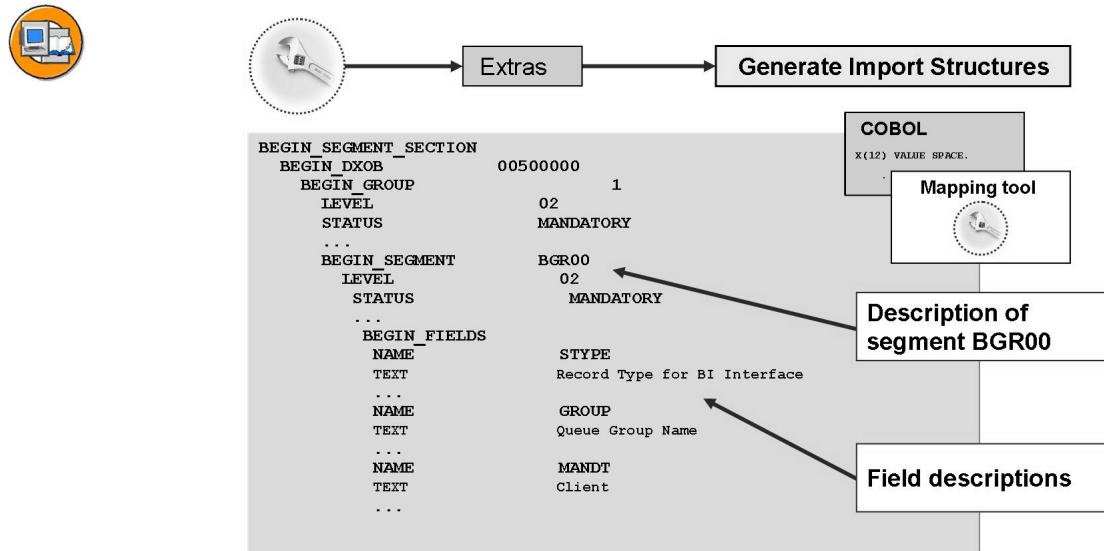


Figure 103: DX-WB: Import Structures for External Tools

This function provides a list of details of all the structures and fields.

You can download this list onto your PC.

Some external mapping tools use this list to define and generate SAP target structures within their conversion process.

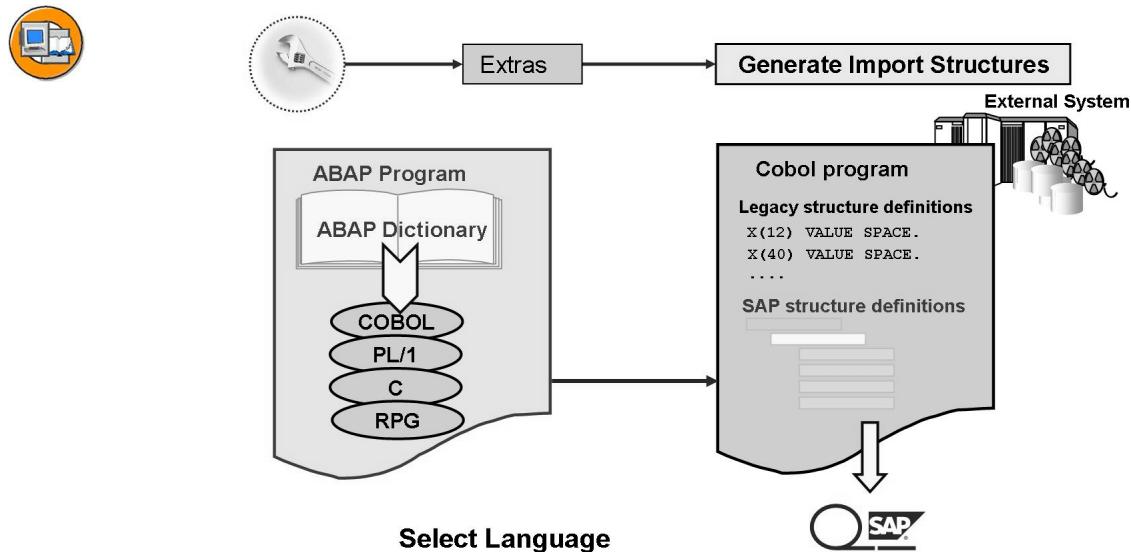
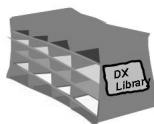


Figure 104: DX-WB: Generate Import Structures

When you choose *Extras -> Generate Import Structure* you can translate SAP structures (stored in the ABAP Dictionary) into external programming languages, such as Cobol, and store them in the file system as “include files”.

The SAP system allows you to create data transfer programs in the programming languages C, COBOL, PL/1 and P_RPG. You can use the ABAP program RDDSRCG0 to generate structure descriptions in the appropriate programming language.



System Tables of the DX-WB

- **SXDA0**
 - ◆ Contains the assignment of the BOR object names for the previous numbering of the objects
- **SXDA1**
 - ◆ Contains the registered programs for the procedures (RFBIDE00, RFBIBL00, ...)
- **SXDA2**
 - ◆ Contains the record layout descriptions
- **SXDA3**
 - ◆ Contains the proposals for record layouts

Figure 105: DX-WB (internal) - System Tables

For the record layout procedures, the DX-WB takes information from system tables SXDA0, SXDA1, SXDA2 and SXDA3. However, these tables do not contain the required definitions for the BAPI procedures. They are only used in connection with BI/CT/DI.

Exercise 7: Registering Programs

Exercise Objectives

After completing this exercise, you will be able to:

- You are to register your own program in the DX-WB.

Business Example

You will learn about the register function in the DX-WB. You can register programs you have written yourself. To simplify the task here, you will copy and register an existing program, rather than develop a new one.

Task:

RSBDCSUB is not registered in DX-WB in the standard system. If you want to process batch input sessions automatically, you have to register this program for the appropriate business object (RSBDCSUB is explained at the end of the course). So that each course participant can practice registering, in this exercise program RSBDCSUB is copied to Z_00_RSBDCSUB and then registered.

1. Create package ZBC420-## (TA:SE80). SE80
2. Copy program RSBDCSUB to Z_##_RSBDCSUB.
3. Activate program Z_##_RSBDCSUB.
4. Go to the DX-WB. Register program Z_##_RSBDCSUB (menu *Goto → DX Program Library*). Object Type: KNA1, task type: MSC, Program type: REPO, Program: Z_##_RSBDCSUB
5. Copy the run under your project FI-## and subproject DEBI-## to a new run to test your own Z_##_RSBDCSUB. The session should be processed using program Z_##_RSBDCSUB, which you have just copied. Maintain the attributes in task RUN-##. Change the program name and create a variant VAR-## for processing the batch input session (you can find the relevant session name in your record layout file in the session header record).



Hint: A run can only be changed if it has been completed and there are no logs for it. The run can only be successful for the external number assignment if a new customer number was defined in the file.

Start the run and check in the batch input monitor that the batch input session has been processed.

Solution 7: Registering Programs

Task:

RSBDCSUB is not registered in DX-WB in the standard system. If you want to process batch input sessions automatically, you have to register this program for the appropriate business object (RSBDCSUB is explained at the end of the course). So that each course participant can practice registering, in this exercise program RSBDCSUB is copied to Z_00_RSBDCSUB and then registered.

1. Create package ZBC420-## (TA:SE80). SE80)
 - a) Create the package ZBC420-##.
Application component: CA
Software component: HOME
Transport layer: Z<SID> (<SID> is your system ID)
You must then record this package in an order.
Choose *Own Requests* and select the request that the course instructor has created for you.
2. Copy program RSBDCSUB to Z_##_RSBDCSUB.
 - a) Copy program *RSBDCSUB* to *Z_##_RSBDCSUB*.
3. Activate program Z_##_RSBDCSUB.
 - a) Activate program Z_##_RSBDCSUB.
4. Go to the DX-WB. Register program Z_##_RSBDCSUB (menu *Goto → DX Program Library*). Object Type: KNA1, task type: MSC, Program type: REPO, Program: Z_##_RSBDCSUB
 - a) Create variant VAR-## of this program for processing in the batch input session that you created in the last exercise. Make sure the session name is correct. (To create the variant, start program RFBIDE00 in SE38, SA38 or SE80.)
 - b) Start the DX-WB and choose (*Goto → Register Programs*).
Register program Z_##_RSBDCSUB. Choose *Create Registration*.
Maintain the following specifications: Object Type: KNA1, task type: MSC, Program type: REPO, Program: Z_##_RSBDCSUB.
5. Copy the run under your project FI-## and subproject DEBI-## to a new run to test your own Z_##_RSBDCSUB. The session should be processed using program Z_##_RSBDCSUB, which you have just copied. Maintain the

Continued on next page

attributes in task RUN-##. Change the program name and create a variant VAR-## for processing the batch input session (you can find the relevant session name in your record layout file in the session header record).



Hint: A run can only be changed if it has been completed and there are no logs for it. The run can only be successful for the external number assignment if a new customer number was defined in the file.

Start the run and check in the batch input monitor that the batch input session has been processed.

- a) Change the run under your project FI-## and subproject DEB-##. The session should be processed using program Z##_RSBDCSUB, which you have just copied. Change the program name and variant in task MSC-##.



Hint: A run can only be changed if it has been completed and there are no logs for it.

Start the run and check in the batch input monitor that the batch input session has been processed.

Exercise 8: Creating and Working with a Conversion Proposal

Exercise Objectives

After completing this exercise, you will be able to:

- Use the conversion assistant
- Create a conversion proposal
- Use the conversion proposal to create a file with test data to support the conversion.

Business Example

When converting external data the main task is identifying the relevant fields for the transfer. The assistant helps you with this.

Task:

Enter the values below in the Data Transfer Tools screen:

1. Enter the values below in the Data Transfer Tools screen:

Object	KN1A
Type:	
Program	BINP
type:	
Program:	RFBIDE00
File type:	physically
File name:	BC420_##_PROPOSAL.SAP

2. Create a conversion proposal by creating a new recording **AUFZ-##**.
3. FD01 is recorded; create customer Z##-30001 with an address of your choice (company code 0001, account group KUNA, sales tax ID number: DE123456789 and reconciliation account 120000).



Caution: Remember to save the recording.

4. Create the conversion proposal **BC420_##_DEBI** by using your recording (tip: choose *Create* and enter recording **AUFZ_##**).
5. Change the proposal by assigning the relevant fields and selecting the checkboxes. Do not forget to save at the end.

Continued on next page

6. Generate record layout file BC420 ## PROPOSAL.SAP with the data from the conversion proposal and check the result with the record layout editor in DXWB. To prevent errors from occurring due to duplicate customers, change the customer number to Z-##-30002 and so on for security reasons before processing.

Solution 8: Creating and Working with a Conversion Proposal

Task:

Enter the values below in the Data Transfer Tools screen:

1. Enter the values below in the Data Transfer Tools screen:

Object	KN1A
Type:	
Program	BINP
type:	
Program:	RFBIDE00
File type:	physically
File name:	BC420_##_PROPOSAL.SAP

- a) The assistant is in the DX-WB tool environment. Choose *Goto → DX Tools*. Here you have to enter parameters for the object type, program type and program. The name of the file created later from the conversion proposal should be *BC420_##_DEBI_PROPOSAL.SAP* (physical file). Choose *Goto → Conversion Assistant*.
2. Create a conversion proposal by creating a new recording **AUFZ-##**.
 - a) Create the recording *AUFZ_##* in the assistant (*Create Recording*). Transaction XD01 to create customers will be recorded. Do not change the default recording parameters.

In recording, enter:

 - * On the first screen: The customer number, company code, account group
 - * On the second screen: An address of your choice
 - * On the third screen (control data): The sales ID
 - * On the last screen (accounting information): The reconciliation account.

After this last screen, choose *Save*.
3. FD01 is recorded; create customer Z-##-30001 with an address of your choice (company code 0001, account group KUNA, sales tax ID number: DE123456789 and reconciliation account 120000).

Continued on next page



Caution: Remember to save the recording.

- a) Back on the assistant input template choose *Create*, and enter *BC420_##_DEBI* as the name for your proposal. After you have selected your recording, you will see a conversion proposal.
4. Create the conversion proposal *BC420_##_DEBI* by using your recording (tip: choose *Create* and enter recording *AUFZ_##*).
 - a) Select checkboxes to assign the fields. Save your changes before you leave the input template.
5. Change the proposal by assigning the relevant fields and selecting the checkboxes. Do not forget to save at the end.
 - a) Go back to the DX-WB tool environment. Choose From Assistant to generate a record layout file with the data from the conversion proposal. Make sure that a meaningful file name is specified.
6. Generate record layout file *BC420_##_PROPOSAL.SAP* with the data from the conversion proposal and check the result with the record layout editor in DXWB. To prevent errors from occurring due to duplicate customers, change the customer number to *Z##-30002* and so on for security reasons before processing.
 - a) There is no solution.



Lesson Summary

You should now be able to:

- Register your own programs in the DX-WB
- Create a conversion proposal using the conversion assistant
- Generate import structures



Unit Summary

You should now be able to:

- Register your own programs in the DX-WB
- Create a conversion proposal using the conversion assistant
- Generate import structures

Unit 5

Sequential Files

Unit Overview

The ABAP programming language offers a number of commands to write files to the application server or presentation server and to read them. In addition, this unit deals with the options for the program-controlled conversion of data. You can use ABAP to program the conversion process of legacy data to data in SAP format.



Unit Objectives

After completing this unit, you will be able to:

- Process sequential files on the application server and on the presentation server
- Describe the structure and the components of a conversion program.
- Define and use logical file names

Unit Contents

Lesson: Sequential Files.....	124
Exercise 9: Reading a Sequential File.....	141
Exercise 10: Writing a File to the PC	145
Exercise 11: Creating a Conversion Program	149

Lesson: Sequential Files

Lesson Overview



- File monitor
- Sequential Files
- Local Sequential Files
- Logical File Names



Lesson Objectives

After completing this lesson, you will be able to:

- Process sequential files on the application server and on the presentation server
- Describe the structure and the components of a conversion program.
- Define and use logical file names

Business Example

A company wants to create its own ABAP program to convert external data.

Programming with Sequential Files



Presentation

Application

Database

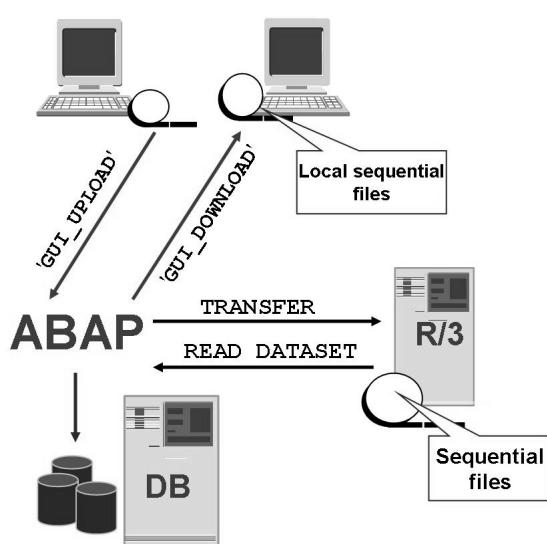


Figure 106: File transfer

The runtime environment is implemented on the application server, which executes the ABAP programs. ABAP supports the file transfer technique to the application server and the front end hosts.

The interface to the file system on the **application server** is implemented in the form of ABAP language elements.

You can process sequential files using the statements READ DATASET (read) and TRANSFER (write).

You implement the file interface on the **presentation server** using function modules. Function modules are a special type of subroutine that are stored in a library.

Files are transferred from the application server to the database server using batch input, call transaction or direct input. You use a sequential file as the data source. These techniques ensure the data is consistent.

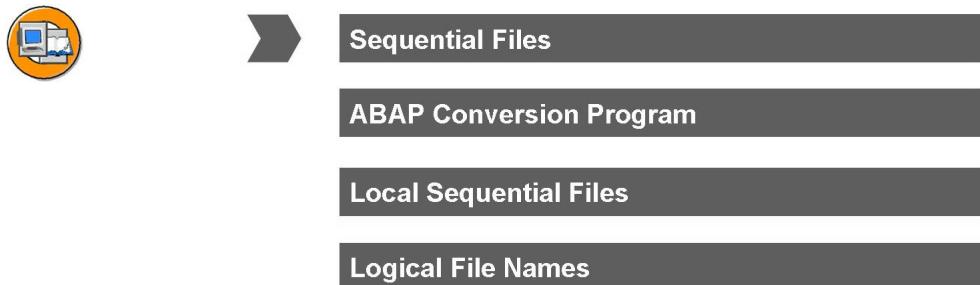


Figure 107: Sequential Files (1)

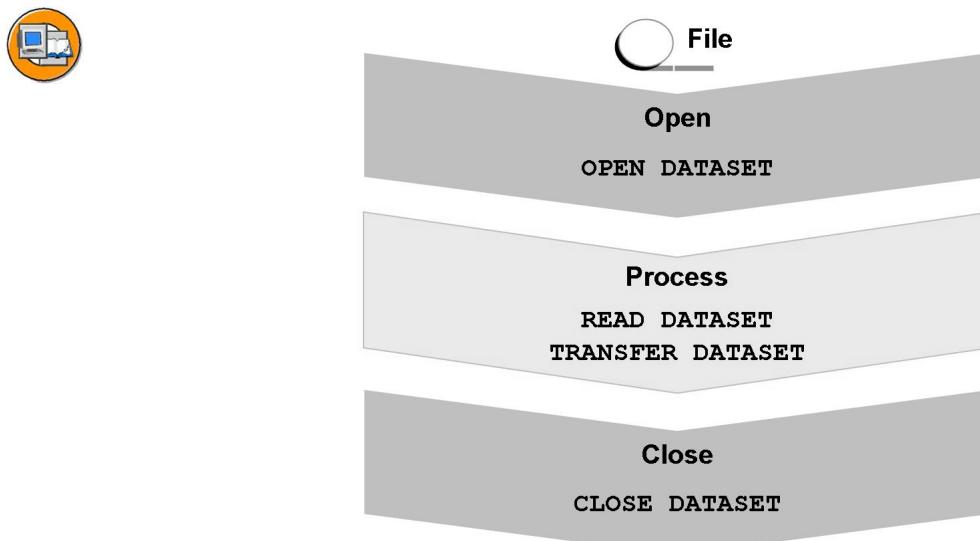


Figure 108: Processing Files

You open a sequential file before you write to or read from it. The file is closed once it has been processed.

In a program, all opened files are implicitly closed each time the screen changes, and then reopened exactly as they were before when processing resumes.

To have a good overview of what is going on, SAP recommends you always write the commands OPEN DATASET and CLOSE DATASET (even though the dataset would be closed anyway).

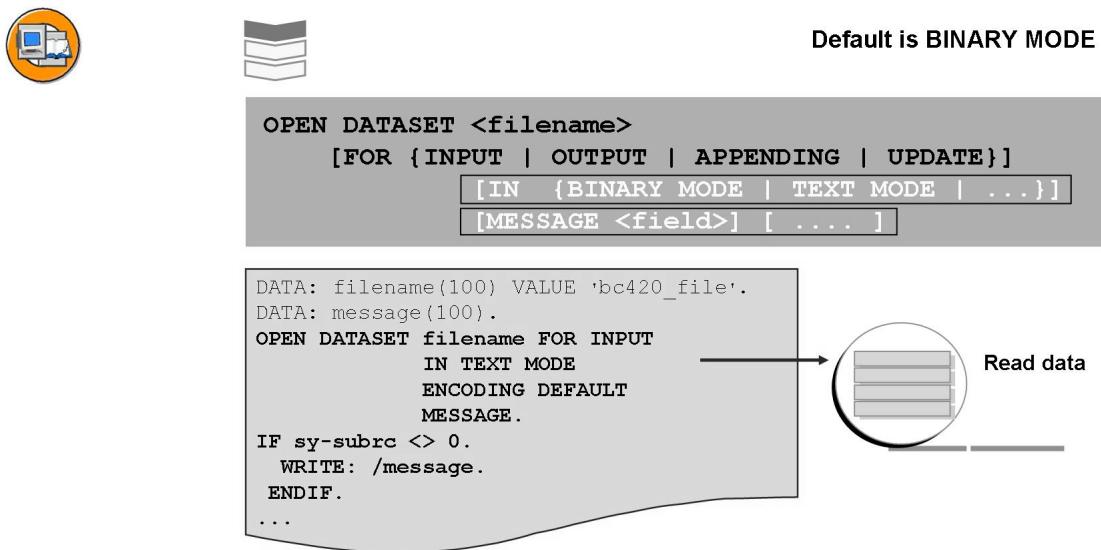


Figure 109: Open File

The OPEN DATASET command opens a file for further processing in the specified path. If the path details are not specified when the file is opened, the path from the profile parameter DIR_HOME is used.

If an error occurs opening the file, a hard termination of the program can be avoided by using the query SY-SUBRC . Operating system messages can be retrieved by using the MESSAGE addition.

You can choose to open a file in binary or text mode.

- IN BINARY MODE - The data is not interpreted by the read and write operations READ DATASET and TRANSFER. The data areas specified for these language elements are input or output directly.
- IN TEXT MODE - If you open a file with this option, the system assumes that the file has a line structure. Each time READ DATASET or TRANSFER is executed, one line is always input or output and the data is always processed up to the end-of-line selection. If the data area is too large for the line that is read, the remaining area is padded with blanks. If it is too small, the rest of the line is missing. **If the UNICODE check is active, the addition ENCODING DEFAULT must be used.** The additions IN BINARY MODE and IN TEXT MODE are mandatory as well.

If you try open a file that is already open, an exception of type CX_SY_FILE_OPEN is triggered.

For other additions of the OPEN DATASET statement refer to the online documentation.

What is Unicode:

Unicode provides a new comprehensive character set with different coding options that will replace all other code pages in the long term. The introduction of Unicode character sets should make working in the IT landscape easier because all characters used can be covered in one code page and complex code page handling and conversions between code pages can be avoided.

Unicode was designed by the international standardization organization (ISO). It is an enormous system for coding text characters (characters, syllable characters, ideograms, punctuation marks, special characters, figures). It is an attempt to combine all world-wide text characters into one character set, that is, not only the characters of the Latin alphabet, but also the Greek, Cyrillic, Arabic, Hebrew and Thai alphabets and the different Japanese (Katakana, Hiragana), Chinese and Korean scripts (Hangul). In addition, mathematical, business and technical characters can be coded in Unicode.

 **Note:** For further information about Unicode, see Notes:

- 752835 - Usage of the file interfaces in Unicode systems
- 747615 - Tool for converting files from one code page to another
- 723363 - Importing non-Unicode files in Unicode system not possible

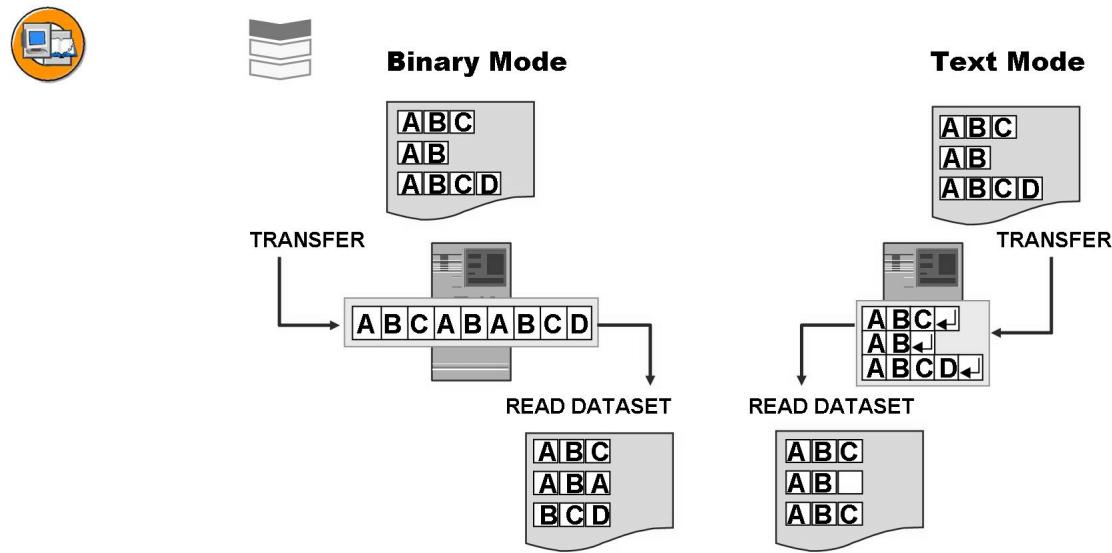


Figure 110: Binary Mode and Text Mode

The example above illustrates the difference between binary mode and text mode. Three fields of different lengths are transferred at any one time. The data is then read into three fields of the same length.

In text mode, the operating system specific at end of line character is set after every data record.

Blanks at the end of a data record are not suppressed in text mode.

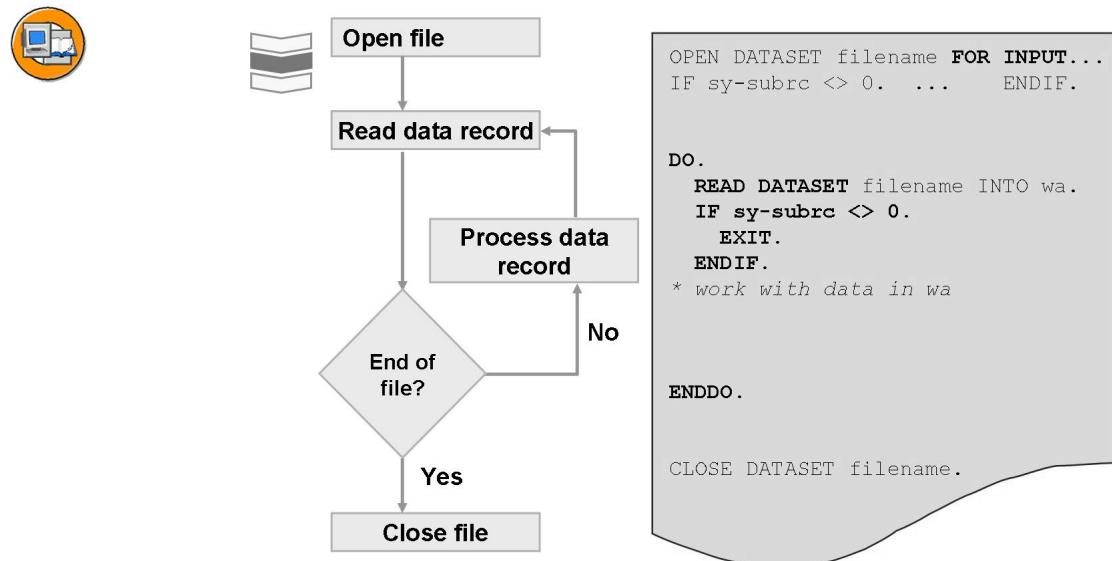


Figure 111: Overview Diagram: Read Data Read File

When the file is **read**, the structures set up for the data records with READ DATASET are read for further processing in the program. Each READ DATASET statement reads one record from a sequential file into a field or structure. Possible structures are field strings or table work areas.

The execution of the READ DATASET statement depends on the mode:

- Binary mode: Reads the length of the structure
- Text mode: Reads a line

If the specified file is not open, READ DATASET tries to open the file (IN BINARY MODE FOR INPUT or with the options of the last OPEN DATASET statement for this file).

If the end of the file is reached, SY-SUBRC is set to 4. Otherwise it is set to 0. If the file cannot be opened, SY-SUBRC contains the value 8. Errors result in program termination.

READ DATASET, like TRANSFER, does not perform conversions implicitly. The data is read in as it was written.

The READ DATASET statement, together with the additional parameter <length>, allows you to specify the length of the imported file record.

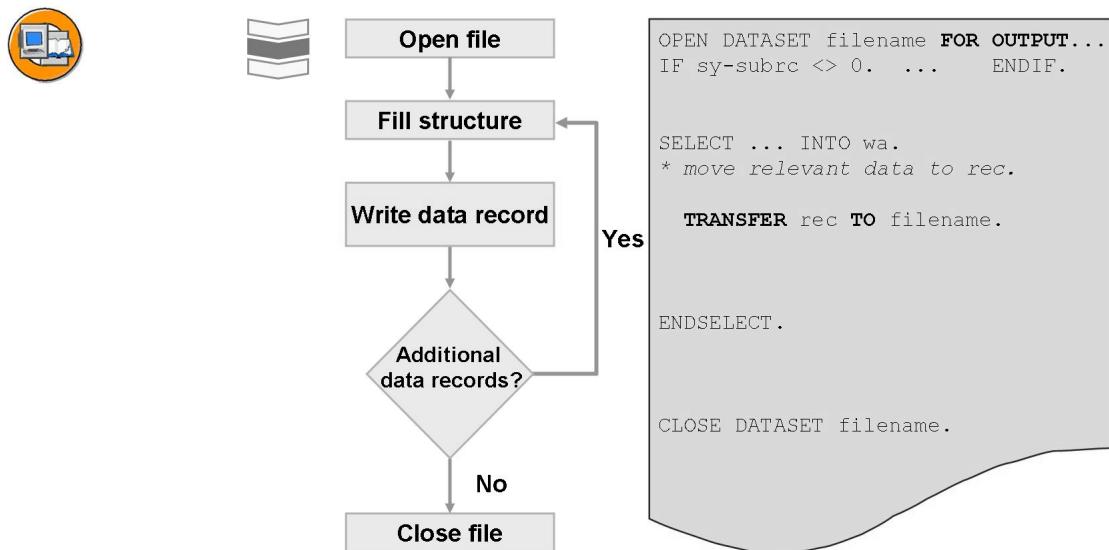


Figure 112: Overview Diagram: Read Data Write File

Each TRANSFER statement **writes** one data record to a sequential file. Before TRANSFER, you place the data record into a field or a structure. Possible structures are field strings or table work areas.

The execution of the TRANSFER statement depends on the mode:

- Binary mode: Writes the length of the field or structure
- Text mode: Writes a line

If the specified file is not open, the TRANSFER statement tries to open the file <file name> FOR OUTPUT (IN BINARY MODE) or using the additions of the last OPEN DATASET statement for this file. If this does not work, a runtime error is triggered.

Errors which occur during the TRANSFER statement result in program termination.

The additional parameter LENGTH <length> of the TRANSFER statement allows you to specify a length in bytes (in the format TRANSFER <structure> TO <file name> LENGTH <length>). In this case, the exact number of characters specified in <length> is transferred. If the structure is shorter, the record is padded (with blanks in text mode and hexadecimal zeros in binary mode). If the structure is longer, the record is truncated.

If the file was opened in TEXT MODE, and if the character representation is in file UTF-8 (Unicode system), then only character-type fields (C, N, D, T), strings, and pure character-type structures are allowed. This check is not made until runtime.

CX_SY_FILE_IO and CX_SY_FILE_AUTHORITY are examples of runtime errors that can be caught during the TRANSFER. You can find the commands and exception classes of all other runtime errors in the documentation.

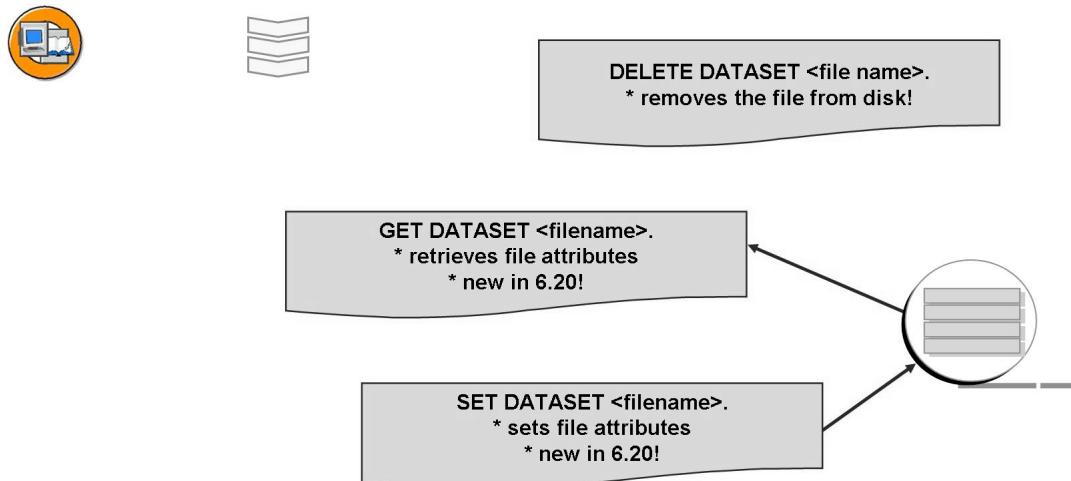


Figure 113: More Data Processing Commands

With the DELETE DATASET statement a file is physically deleted. Not just the file contents, but the file itself is deleted. If this is successful, SY-SUBRC is set to 0.

As of Release 6.20 with the statements SET DATASET and GET DATASET file attributes can be set and retrieved (retrieve or set read and write position of the file pointer).

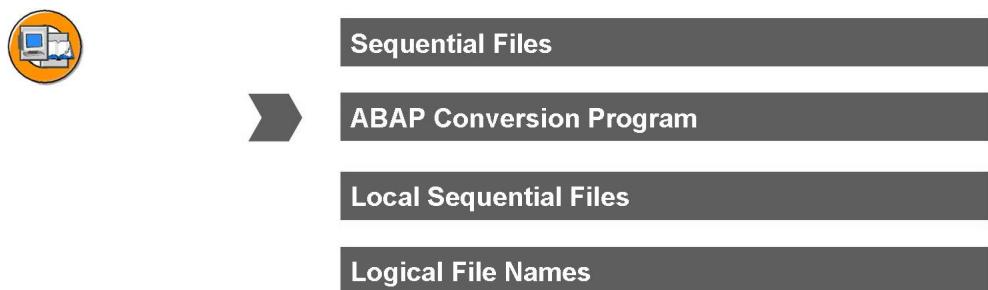
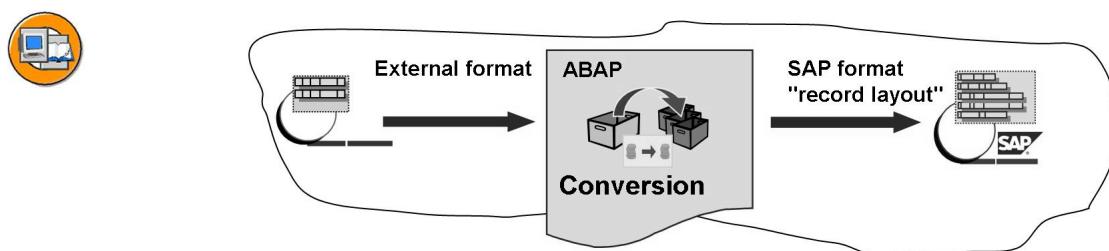


Figure 114: Sequential Files (2)



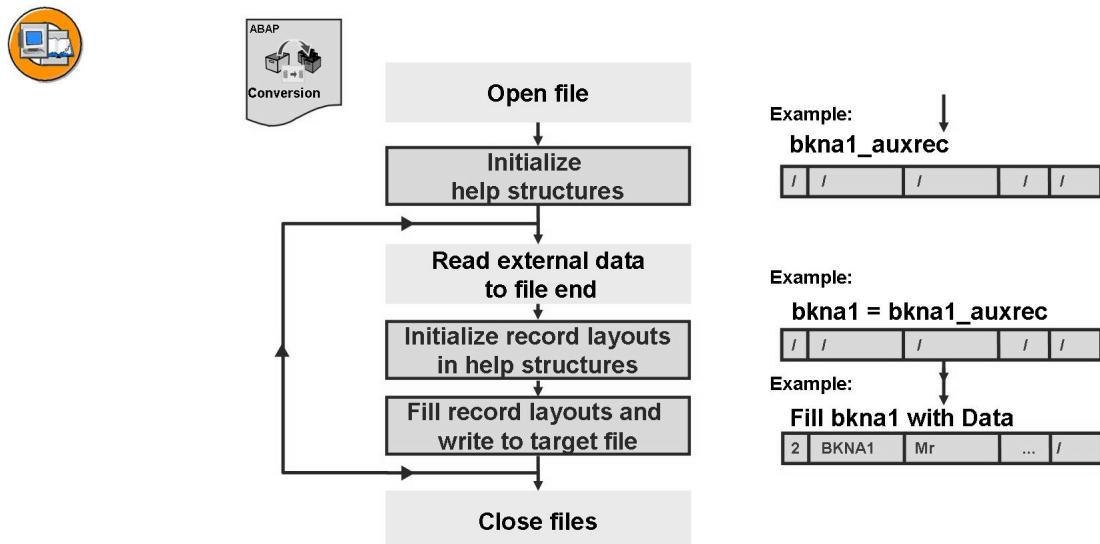
Mapping Plan

- **Read external data**
- **Change format of external fields into target format**
- **Assign external fields to target fields**
- **Write target structures to target file**

Figure 115: Conversion Program Tasks

The main tasks within the conversion program are:

- Read external data into external structures (definition of line type in program).
- Convert format of external fields into the format of the target fields. The target fields must be defined beforehand as line types.
- Enter data into target fields and target structures.
- Write the target structures into one or more target files.

**Figure 116: Flowchart of Conversion Program**

You can convert the external data into the SAP record layout format required by the standard data transfer programs in several ways:

1. Open all files
2. Initialize help structures with the NODATA character: These structures are copied to the record layouts to initialize them.
3. Read external data record by record
4. Initialize the record layouts. To do this the help structures that were initialized earlier are simply copied to the record layouts.
5. Format and assign data: First of all the external data is formatted (for example, "\$" -> "USD"). Then this formatted data must be written to the corresponding fields of the record layout.
6. Close the files after the external data has been read and the record layouts have been written.

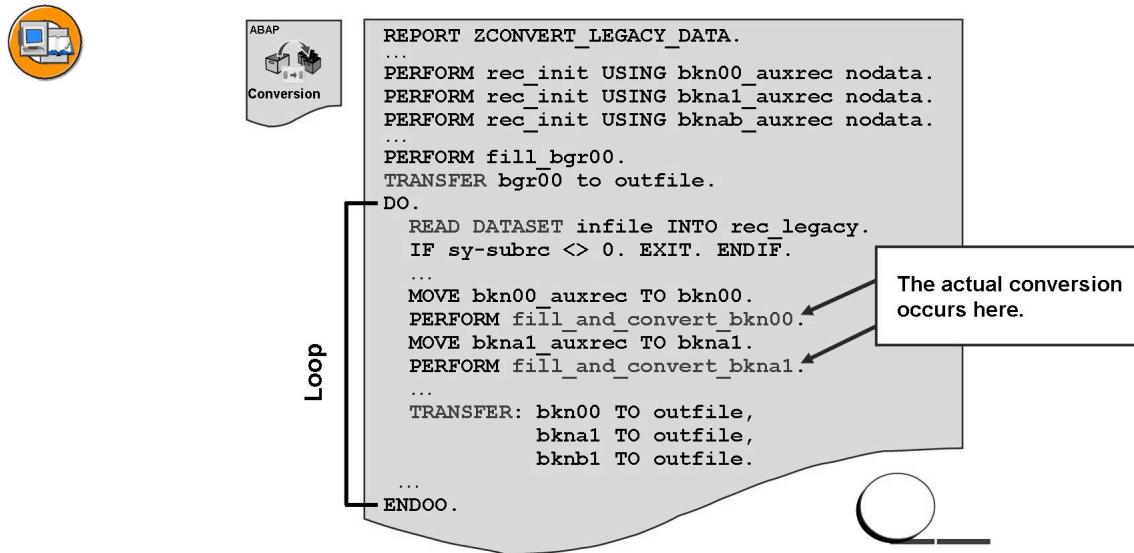


Figure 117: Example of a Conversion Program

The above program extract shows the most important steps from initializing the help structures to writing the record layouts.

The form routine REC_INIT completely fills the help record layouts with the NODATA character. Help record layouts are used to speed the process up; the MOVE statement is quicker using help record layouts rather than processing loops to initialize the actual record layouts.

The form routines “FILL_AND_CONVERT_” contain the source code for formatting and assigning the external fields to target fields.

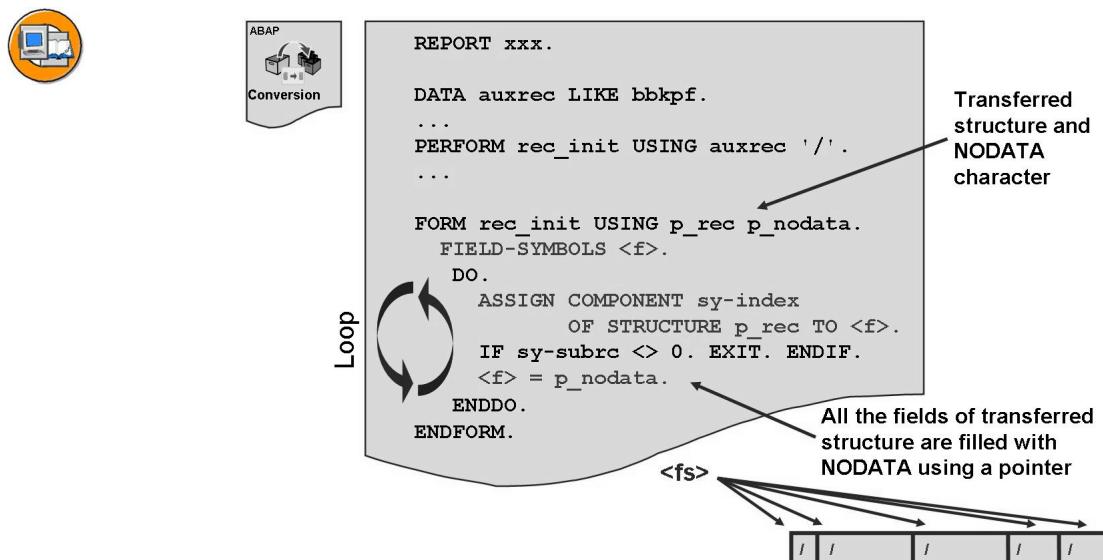


Figure 118: Initializing the Help Structures

You initialize the record layouts with the NODATA character.

You refer to and initialize all structure fields using pointer operations "ASSIGN COMPONENT ... TO <f>. MOVE ... TO <f>.

The form routine REC_INIT is structured so that any number of structures can be transferred to the formal parameter P_REC of the interface. This means that all structures can be initialized with the NODATA character of your choice (formal parameter P_NODATA).

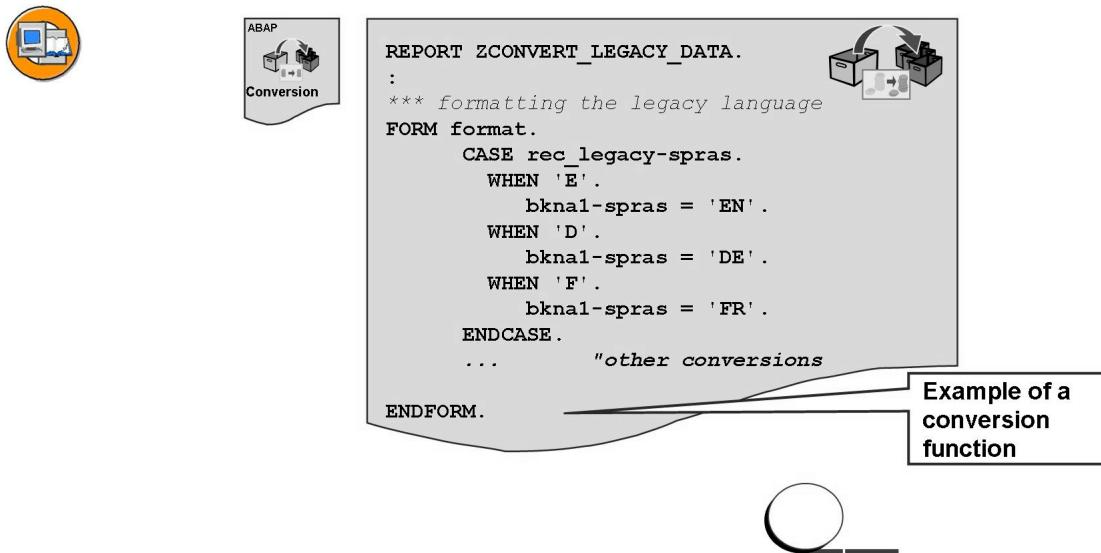


Figure 119: Example of Formatting

A formatting example is the conversion of the language REC_LEGACY-SPRAS from the old format (for example, “E”) into the new SAP format (“EN”).

The newly formatted language is assigned at the same time to the record layout field BKNA1-SPRAS.

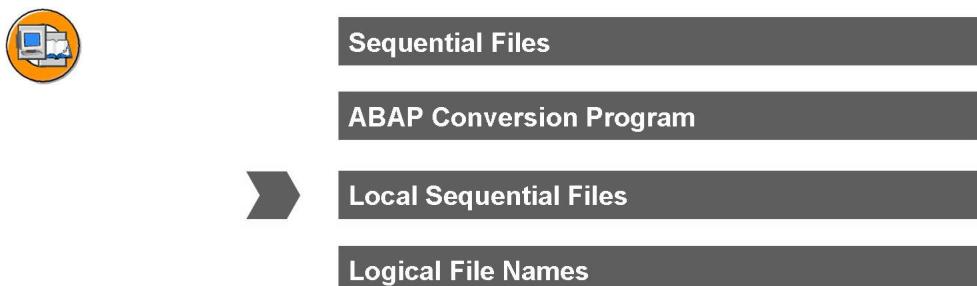


Figure 120: Sequential Files (3)

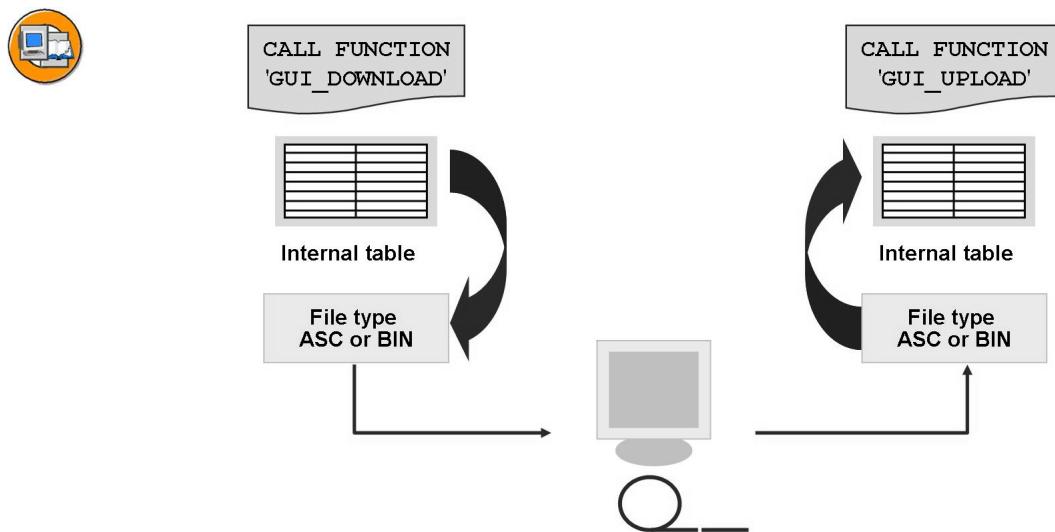


Figure 121: GUI_DOWNLOAD and GUI_UPLOAD

The function module GUI_DOWNLOAD downloads the contents of an internal table to a local sequential file.

The function module GUI_UPLOAD transfers the content of a local sequential file to an internal table.

For the local file, specify the file name (the complete path, for example, '/tmp/myfile' for a UNIX file and or 'C:\ MYFILE.TXT' for a PC file).

- The directory must be known to the presentation server.
- The choice of suitable file names is the responsibility of the customer.

The default file type is ASC.



```

REPORT sapbc420_seqd_download.
Types: BEGIN OF rectype,
      kunnr LIKE kna1-kunnr,
      land1 LIKE kna1-land1,
      name1 LIKE kna1-name1,
      ...
      END OF rectype.

DATA: itab TYPE STANDARD TABLE OF rectype
      WITH KEY kunnr WITH HEADER LINE.
SELECT kunnr land1 name1 stras ort01 pstl2
      FROM kna1 INTO CORRESPONDING FIELDS OF TABLE itab.
CALL FUNCTION 'GUI_DOWNLOAD'
      EXPORTING
        filename = 'C:\BC420_00_test.txt'
        filetype = 'ASC'
      TABLES
        data_tab = itab
      EXCEPTIONS
      ...

```



Figure 122: Example of a GUI_DOWNLOAD

The above example uses a SELECT statement to read customer data from table KNA1 and enters this data into a previously defined internal table. The function module GUI_DOWNLOAD stores the internal table on the presentation server.



```

REPORT sapbc420_seqd_upload.
Types: BEGIN OF rectype,
      ...
      END OF rectype.

DATA: itab TYPE STANDARD TABLE OF rectype
      WITH KEY kunnr WITH HEADER LINE,
      wa LIKE LINE OF itab.
CALL FUNCTION 'GUI_UPLOAD'
      EXPORTING
        filename = 'C:\BC420_00_test.txt'
        filetype = 'ASC'
      TABLES
        data_tab = itab
      EXCEPTIONS
      ...
      ...

LOOP AT itab INTO wa.
  WRITE: / wa-kunnr, wa-land1, ...
ENDLOOP.

```



Figure 123: Example of a GUI_UPLOAD

The above example program uses the function module GUI_UPLOAD to read a local sequential file from the presentation server and enters this data in a previously defined internal table. The internal table is then output with a loop.

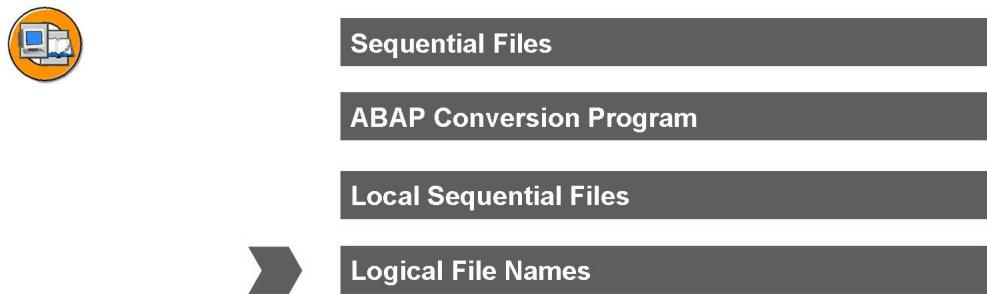


Figure 124: Sequential Files (4)

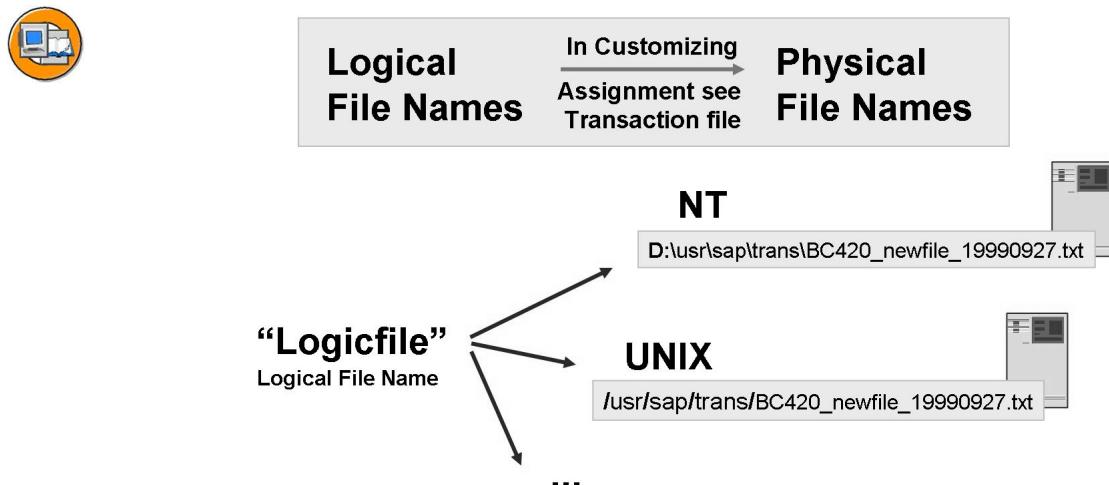


Figure 125: Customizing Logical File Names

Logical file names are often used for archiving SAP application data in transaction SARA. The logical file names technique is also used for external data transfer programs.

The platform-independent, logical file name determines what file name and path files are to be created in. Depending on the operating system used, the physical path and file name defined in Customizing (transaction FILE) are used.

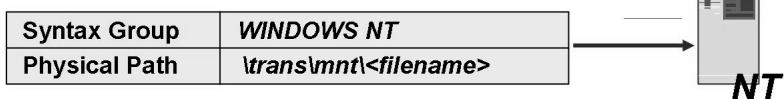
The portability of programs can be implemented using function module (FILE_GET_NAME).



1) Definition of the logical path

Name	Z_PATH
Description	Demonstration for logical path

2) Assignment of logical path → to physical path



3) Definition of logical file

Name	Z_FILE
Description	Demonstration for a logical file
Physical file	BC420_<DATE>.txt
Logical path	Z_PATH

<DATE>
<YEAR>
<MONTH>
<DAY>
...

Figure 126: Logical Path and Logical File

You can create logical paths and files in the Customizing transaction FILE.

The platform-specific physical path must contain the reserved word <FILENAME> as a placeholder for the file name. It may contain other reserved words too (see FI help documentation).



```

DATA: file TYPE filename-fileintern.
...
CALL FUNCTION 'FILE_GET_NAME'
  EXPORTING
    logical_filename      = 'Z_FILE'
  IMPORTING
    FILE_NAME             = file
  EXCEPTIONS
    FILE_NOT_FOUND        = 1
    OTHERS                 = 2.
IF sy-subrc <> 0.
  Write: / 'file not found'. EXIT.
ELSE.
  Write: / 'physical file is' , file.
ENDIF.
...

```

The logical file is an import parameter of the function module.

Depending on the operating system the physical file name is returned.

Figure 127: Function Module FILE_GET_NAME

The format of file names depends largely on the operating system. You can access portable programs by using the function module **FILE_GET_NAME**. This function module returns the physical name for a given logical file name.

You can find the function module description in the Function Builder (the function module library).

Exercise 9: Reading a Sequential File

Exercise Objectives

After completing this exercise, you will be able to:

- Write a program that reads a sequential file containing external data from the application server.

Business Example

Program: Z_##_READ_DEB

Model Solution: SAPBC420_SEQS_READ_DEB

Task:

Write program **Z_##_READ_DEB** to read the source file **BC420##DEBI.LEG** from the application server.

1. The source file containing customer data is located in the DIR_HOME directory of the application server. Create the program in the package **ZBC420##**.

Work with the internal data object *REC_LEGACY*. You can find this data object (structure) in the INCLUDE file *SAPBC420_SEQI_DEBI_LEGACYSTRUC*.



Hint: Tip: When you define the source file within the program you can omit the path details, as the DIR_HOME directory is used by default.

Solution 9: Reading a Sequential File

Task:

Write program **Z_##_READ_DEB** to read the source file **BC420_##_DEBI.LEG** from the application server.

1. The source file containing customer data is located in the DIR_HOME directory of the application server. Create the program in the package **ZBC420-##**.

Work with the internal data object *REC_LEGACY*. You can find this data object (structure) in the INCLUDE file *SAPBC420_SEQI_DEBI_LEGACYSTRU.C*.



Hint: Tip: When you define the source file within the program you can omit the path details, as the DIR_HOME directory is used by default.

a) **Solution SAPBC420_SEQS_READ_DEB**

```
*&-----*
*& Report SAPBC420_SEQS_READ_DEB *
*&-----*
*& This report reads data from the application-server *
*& *
*&-----*
```

REPORT SAPBC420_SEQS_READ_DEB

INCLUDE sapbc420_seqi_debi_legacystruc.

DATA: file(100) TYPE c VALUE 'BC420_##_DEBI.LEG',
 mess(100) TYPE c.

START-OF-SELECTION.

*#####

*** Open the file called "datei" for reading ****
OPEN DATASET file FOR INPUT IN TEXT MODE ENCODING DEFAULT
 MESSAGE mess.

IF sy-subrc <> 0.
 WRITE: / mess.
ENDIF.

*** Read data into structure rec_legacy ****

Continued on next page

```
DO.  
  READ DATASET file INTO rec_legacy.  
  IF sy-subrc <> 0.  
    EXIT.  
  ENDIF.  
  WRITE: / rec_legacy. "just to control reading...  
ENDDO.  
  
*** Close the dataset *****  
CLOSE DATASET file.
```


Exercise 10: Writing a File to the PC

Exercise Objectives

After completing this exercise, you will be able to:

- Add a data writing function to the program you have just created.

Business Example

Program: Z##_COPY_DEB_TO_PC

Model Solution: SAPBC420_SEQS_COPY_DEB_TO_PC

Task:

Extend the program you have just written so that it can write data to the presentation server (PC).

1. Use function module *GUI_DOWNLOAD*. The target file is to be located on C:\

Use tabulator characters to divide fields.

In addition, check the file using Notepad or Excel (depending on the access rights on your PC).



Hint: Tip: The definition of the internal table for the *GUI_DOWNLOAD* may look like:

DATA: itab LIKE TABLE OF rec_legacy.

Solution 10: Writing a File to the PC

Task:

Extend the program you have just written so that it can write data to the presentation server (PC).

1. Use function module *GUI_DOWNLOAD*. The target file is to be located on C:\

Use tabulator characters to divide fields.

In addition, check the file using Notepad or Excel (depending on the access rights on your PC).



Hint: Tip: The definition of the internal table for the *GUI_DOWNLOAD* may look like:

DATA: itab LIKE TABLE OF rec_legacy.

a) **Solution SAPBC420_SEQS_COPY_DEB_TO_PC**

```
*&-----*
*& Report  SAPBC420_SEQS_COPY_DEB_TO_PC
*&-----*
*& This report reads data from the application-server
*& and saves it to the PC
*&-----*

REPORT SAPBC420_SEQS_COPY_DEB_TO_PC.

INCLUDE sapbc420_seqi_debi_legacystruc.

DATA: file(100) TYPE c VALUE 'BC420_00_DEBI.LEG',
      mess(100) TYPE c.

DATA itab LIKE STANDARD TABLE OF rec_legacy.

START-OF-SELECTION.
*#####
*** Open the file called "datei" for reading ****
OPEN DATASET file FOR INPUT IN TEXT MODE ENCODING DEFAULT
      MESSAGE mess.
IF sy-subrc <> 0.
  WRITE: / mess.
```

Continued on next page

```

ENDIF.

*** Read data into structure rec_legacy *****
DO.
  READ DATASET file INTO rec_legacy.
  IF sy-subrc <> 0.
    EXIT.
  ENDIF.
*** Insert the data into internal table itab *****
  INSERT rec_legacy INTO TABLE itab.
ENDDO.

*** Close the dataset *****
CLOSE DATASET file.

*** Call the DOWNLOAD function to download data on the PC *****
*** This data can be visualized for example in EXECL
CALL FUNCTION 'GUI_DOWNLOAD'
  EXPORTING
    * BIN_FILESIZE           =
      filename                = 'C:\BC420_00_DEBI.LEG'
      filetype                = 'ASC'
    * APPEND                 =
      write_field_separator   = 'X'
    * HEADER                 =
      = '00'
    * TRUNC_TRAILING_BLANKS =
      = ' '
    * WRITE_LF                =
      = 'X'
    * COL_SELECT              =
      = ' '
    * COL_SELECT_MASK         =
      = ' '
    * DAT_MODE                =
      = ' '
    * CONFIRM_OVERWRITE       =
      = ' '
    * NO_AUTH_CHECK           =
      = ' '
    * CODEPAGE                =
      = ' '
    * IGNORE_CERR              =
      = ABAP_TRUE
    * REPLACEMENT             =
      = '#'
    * WRITE_BOM                =
      = ' '
  IMPORTING
    * FILELENGTH              =
  TABLES
    data_tab                  = itab
EXCEPTIONS
  file_write_error           =
  no_batch                   =
  gui_refuse_filetransfer   =

```

Continued on next page

```
invalid_type          =  4
no_authority          =  5
unknown_error          =  6
header_not_allowed    =  7
separator_not_allowed =  8
filesize_not_allowed  =  9
header_too_long        = 10
dp_error_create        = 11
dp_error_send          = 12
dp_error_write         = 13
unknown_dp_error       = 14
access_denied          = 15
dp_out_of_memory       = 16
disk_full              = 17
dp_timeout              = 18
file_not_found          = 19
dataprovider_exception = 20
control_flush_error    = 21
OTHERS                  = 22.

IF sy-subrc <> 0.
  WRITE: / 'SY-SUBRC = ', sy-subrc.
ENDIF.
```

Exercise 11: Creating a Conversion Program

Exercise Objectives

After completing this exercise, you will be able to:

- Write a program that reads a sequential file containing external data, converts the external data into the record layout format, and then writes it to an output file.
- Use this output file to start the data transfer in the DX-WB. The result is a batch input session to be processed later.

Business Example

Analysis of the available transfer methods as well as the analysis of the SAP record layout structures is complete.

Now write a conversion program to convert the legacy data into the SAP record layout file.

Program: Z##_CONVERT_DEBI

**Template program: SAPBC420_SEQT_CONVERT_DEBI1 or
SAPBC420_SEQT_CONVERT_DEBI2**

Model Solution: SAPBC420_SEQS_CONVERT_DEBI

Task 1:

Write a program `Z##_CONVERT_DEBI` to read and convert legacy data of customers from the source file `BC420##_DEBI.LEG`. The path is generated automatically in the template program, and does not have to be specified separately. You want to read file `BC420##_DEBI.SAP` from the output file.



Hint: Tips: The source and file structures that you need are predefined in INCLUDE files.

There are two template programs of which you can use one as a copy template for your solution.

`SAPBC420_SEQT_CONVERT_DEBI` contains tips on what statements need to be added to the program still.

`SAPBC420_SEQT_CONVERT_DEBI2` does not contain as many predefined statements, and should be used only by course participants who have a very good knowledge of ABAP.

1. Opening and closing the sequential files is specified according to the template program (the file variables `infile`, `outfile` and `errfile` are predefined with the relevant path and can be used directly).
2. Form routines to initialize and fill record layouts are already defined and contain some source code, which you will need to add to.
3. Read the external data in a loop, create the record layout structures in this loop, and save them in the target file.

Incorrect records are those with legacy customer numbers that are not filled (contain blank characters only). Write incorrect records to the error file.

Continued on next page

- Some fields must be formatted before they are assigned to record layout fields.
 - Assign the legacy customer number to field *BKNB1 ALTKN*. Create a new external number for the customer in the system. The new number is to be a 10 digit customer number. Create the prefix “**Z-##-5**”. This has already been done by the predefined variable auxkunnr(10). The last four digits of this new customer number are filled by the loop index SY-INDEX (numbers 0001-0010 are assigned using the help field NUM). The result is the following customer numbers: **Z-##-50001 to Z-##-50010**.
 - As another example of formatting change the language key (SAP languages are two digits).
- Tip: Use or write a form routine CONVERT_LANGUAGE, in which the language key is converted in a CASE statement. The following languages are to be defined: D → DE, F → FR, S → ES, J → JA, E → EN.
4. After you have written the program, check the structure of your record layout file in transaction AL11 and in the DX-WB.

Task 2:

Create a new run with two tasks in your subproject:

Project name: *use legacy project*.

Subproject name: *use legacy subproject*.

Run definition name: *BI-##, transfer customers*

1. To use your own languages in the DX-WB, you must register them first. Register your program *Z_##_CONVERT_DEBI* as task type MAP under program type REPO.
2. Create the following task:
Technical name: *CONV-##*.
Description: Map customers
Task type MAP
Select your mapping program (program type REPO)
3. Create the following task:
Technical name: *LOAD-##*.
Description: Load customers

Continued on next page

Task type LOA

Select the standard transfer program RFBIDE00 with program type BINP.

4. Start the run and analyze the run log.



Hint: Again: When you execute task LOA, test the file first (select checkbox). If no possible program termination is found, the session can be created.

5. Process the batch input session using the batch input monitor.

Solution 11: Creating a Conversion Program

Task 1:

Write a program `Z##_CONVERT_DEBI` to read and convert legacy data of customers from the source file `BC420##DEBI.LEG`. The path is generated automatically in the template program, and does not have to be specified separately. You want to read file `BC420##DEBI.SAP` from the output file.



Hint: Tips: The source and file structures that you need are predefined in INCLUDE files.

There are two template programs of which you can use one as a copy template for your solution.

`SAPBC420_SEQT_CONVERT_DEBI1` contains tips on what statements need to be added to the program still.

`SAPBC420_SEQT_CONVERT_DEBI2` does not contain as many predefined statements, and should be used only by course participants who have a very good knowledge of ABAP.

1. Opening and closing the sequential files is specified according to the template program (the file variables `infile`, `outfile` and `errfile` are predefined with the relevant path and can be used directly).
 - a) There is no solution.
2. Form routines to initialize and fill record layouts are already defined and contain some source code, which you will need to add to.
 - a) There is no solution.
3. Read the external data in a loop, create the record layout structures in this loop, and save them in the target file.

Incorrect records are those with legacy customer numbers that are not filled (contain blank characters only). Write incorrect records to the error file.

Continued on next page

- Some fields must be formatted before they are assigned to record layout fields.
 - Assign the legacy customer number to field *BKNBI ALTKN*. Create a new external number for the customer in the system. The new number is to be a 10 digit customer number. Create the prefix “**Z-##-5**”. This has already been done by the predefined variable auxkunnr(10). The last four digits of this new customer number are filled by the loop index SY-INDEX (numbers 0001-0010 are assigned using the help field NUM). The result is the following customer numbers: **Z-##-50001 to Z-##-50010**.
 - As another example of formatting change the language key (SAP languages are two digits).

Tip: Use or write a form routine *CONVERT_LANGUAGE*, in which the language key is converted in a CASE statement. The following languages are to be defined: D → DE, F → FR, S → ES, J → JA, E → EN.
- a) There is no solution.
4. After you have written the program, check the structure of your record layout file in transaction AL11 and in the DX-WB.
- a) There is no solution.

Task 2:

Create a new run with two tasks in your subproject:

Project name: *use legacy project*.

Subproject name: *use legacy subproject*.

Run definition name: *BI-##, transfer customers*

1. To use your own languages in the DX-WB, you must register them first. Register your program *Z_##_CONVERT_DEBI* as task type MAP under program type REPO.

a) There is no solution.
2. Create the following task:

Technical name: CONV-##
Description: Map customers
Task type MAP

Continued on next page

- Select your mapping program (program type REPO)
- a) There is no solution.
 3. Create the following task:
Technical name: LOAD-##.
Description: Load customers
Task type LOA
Select the standard transfer program RFBIDE00 with program type BINP.
 - a) There is no solution.
 4. Start the run and analyze the run log.



Hint: Again: When you execute task LOA, test the file first (select checkbox). If no possible program termination is found, the session can be created.

- a) There is no solution.
 5. Process the batch input session using the batch input monitor.
- a) **Solution SAPBC420_SEQS_CONVERT_DEBI**

```
*&-----*
*& Report  SAPBC420_SEQS_CONVERT_DEBI
*&-----*
*& Source file contains debtors in legacy format.
*& This data is converted into SAP record layout format for data
*& transfer with RFBIDE00
*&-----*
REPORT  sapbc420_seqs_convert_debi MESSAGE-ID bc420 LINE-SIZE 200.

TABLES: bgr00, bkn00, bkna1, bknbl.
* include contains structur of file with SAP-formatted data
INCLUDE sapbc420_seqi_debi_legacystruc.
INCLUDE bc420_fill_nodata.

* help structure (structure equals BKN00)
DATA bkn00_nodata LIKE bkn00.
* help structure (structure equals BKNA1)
DATA bkna1_nodata LIKE bkna1.
* help structure (structure equals BKNB1)
DATA bknbl_nodata LIKE bknbl.

DATA: num(4) TYPE n,
```

Continued on next page

```

text(100).

PARAMETERS:
  newkunrn(10) DEFAULT 'Z-##-5',           "will be new debtor-no.
  infile(70) default  'BC420_##_DEBI.LEG'    LOWER CASE,
  outfile1(70) default 'BC420_##_DEBI.SAP'  LOWER CASE,
  errfile(70) default 'BC420_##_DEBI.ERR'  LOWER CASE,
  session(20) DEFAULT 'BC420SES-##'        LOWER CASE,
  nodata      DEFAULT '/'                  LOWER CASE.

AT SELECTION-SCREEN.
*-----
* open files
  OPEN DATASET: infile FOR INPUT IN TEXT MODE MESSAGE text.
  IF sy-subrc NE 0.
    MESSAGE e101 WITH infile.
  ENDIF.

  OPEN DATASET: outfile1 FOR OUTPUT IN TEXT MODE MESSAGE text.
  IF sy-subrc NE 0.
    MESSAGE e100 WITH outfile1.
  ENDIF.

  OPEN DATASET: errfile FOR OUTPUT IN TEXT MODE MESSAGE text.
  IF sy-subrc NE 0.
    MESSAGE e100 WITH errfile.
  ENDIF.

START-OF-SELECTION.
*-----
* fill structure with nodata
  PERFORM init USING bkn00_nodata nodata.
  PERFORM init USING bknal_nodata nodata.
  PERFORM init USING bknbl_nodata nodata.

* fill structure bgr00
  PERFORM fill_bgr00.
* write data to file
  TRANSFER bgr00 TO outfile1.
  WRITE / bgr00.
* fill structure bkn00, bknal and bknbl
  DO.
    READ DATASET infile INTO rec_legacy.
    IF sy-subrc NE 0. EXIT. ENDIF.
    " just to do some cleansing of legacy data!
    IF rec_legacy-kunrn IS INITIAL.
      TRANSFER rec_legacy TO errfile.
      WRITE: / text-001, rec_legacy-kunrn.

```

Continued on next page

Internal Use SAP Partner Only

```

ELSE.                                     " record is okay !
    MOVE bkn00_nodata TO bkn00.
    PERFORM fill_bkn00.
    MOVE bknal_nodata TO bknal.
    PERFORM fill_bknal.
    MOVE bknb1_nodata TO bknb1.
    PERFORM fill_bknb1.

                                         " write data to file
    TRANSFER: bkn00 TO outfile1,
               bknal TO outfile1,
               bknb1 TO outfile1.
    WRITE: / text-002, rec_legacy-kunnr.
    WRITE: text-003, bkn00-kunnr.

ENDIF.
ENDDO.

* close files
CLOSE DATASET: infile, outfile1, errfile.

*-----*
*&      Form  FILL_BGR00
*      fill structure bgr00
*-----*
FORM fill_bgr00.
    MOVE: '0'           TO bgr00-stype,
          session     TO bgr00-group,
          sy-mandt   TO bgr00-mandt,
          sy-uname    TO bgr00-usnam,
          nodata     TO bgr00-nodata,
          'X'         TO bgr00-xkeep.

ENDFORM.                                     " FILL_BGR00
*-----*
*&      Form  FILL_BKN00
*&      fill structure bkn00
*&      New debtor-numbers are created; they start with "Z-##-1"
*&      the old debtor number is concatenated --> "Z-##-1xxxxx"
*-----*
FORM fill_bkn00.
    MOVE: '1'           TO bkn00-stype,
          'XD01'        TO bkn00-tcode,
          '0001'        TO bkn00-bukrs,
          'KUNA'         TO bkn00-ktokd.

num = sy-index.                           "sy-index is set in DO-ENDDO-Loop
MOVE num TO newkunnr+6(4).                "move new number
MOVE newkunnr TO bkn00-kunnr.

```

Continued on next page

Internal Use SAP Partner Only

```

ENDFORM.                                     " FILL_BKN00
*-----*
*&      Form  FILL_BKNA1
*      fill structure bknal
*-----*
*-----*
FORM fill_bknal.

MOVE: '2'          TO bknal-stype,
      'BKNA1'        TO bknal-tbnam,
      rec_legacy-name1 TO bknal-name1,
      rec_legacy-sortl TO bknal-sortl,
      rec_legacy-stras TO bknal-stras,
      rec_legacy-ort01 TO bknal-ort01,
      rec_legacy-pstlz TO bknal-pstlz,
      rec_legacy-land1 TO bknal-land1,
      rec_legacy-telf1 TO bknal-telf1.

PERFORM convert_language.           "D->DE; E->EN and so on

IF rec_legacy-stcег IS INITIAL.
  MOVE nodata      TO bknal-stcег.
ELSE.
  MOVE rec_legacy-stcег   TO bknal-stcег.
ENDIF.
ENDFORM.                                     " FILL_BKNA1
*-----*
*&      Form  FILL_BKNB1
*      fill structure bknb1
*-----*
*-----*
FORM fill_bknb1.

MOVE: '2'          TO bknb1-stype,
      'BKNB1'        TO bknb1-tbnam,
      '120000'        TO bknb1-akont,
      rec_legacy-kunrn TO bknb1-altkn.
*      write: / 'Alte Nummer = ', rec_legacy-kunrn.
ENDFORM.                                     " FILL_BKNB1

*-----*
*      FORM Convert_langu
*-----*
*-----*
FORM convert_language.

CASE rec_legacy-spras.
  WHEN 'E'.
    bknal-spras = 'EN'.
  WHEN 'D'.

```

Continued on next page

```
bknal-spras = 'DE'.
WHEN 'F'.
  bknal-spras = 'FR'.
WHEN 'S'.
  bknal-spras = 'ES'.
WHEN 'J'.
  bknal-spras = 'JA'.
ENDCASE.
ENDFORM.
```



Lesson Summary

You should now be able to:

- Process sequential files on the application server and on the presentation server
- Describe the structure and the components of a conversion program.
- Define and use logical file names



Unit Summary

You should now be able to:

- Process sequential files on the application server and on the presentation server
- Describe the structure and the components of a conversion program.
- Define and use logical file names

Unit 6

Legacy System Migration Workbench

Unit Overview

This unit gives you an introduction to the migration tool LSMW for converting external data into the format of the SAP system. It also shows you the basic functions of the LSMW.



Unit Objectives

After completing this unit, you will be able to:

- Use the basic functions of the LSMW
- Define rules in the LSMW
- Convert data in the LSMW

Unit Contents

Lesson: LSMW - Legacy System Migration Workbench.....	164
Exercise 12: Converting and transferring documents using the Legacy System Migration Workbench.....	209

Lesson: LSMW - Legacy System Migration Workbench

Lesson Overview

Contents:

- Principles of the LSMW
- Defining source and target structures
- Creating field mapping and rules
- Import files
- Transfer data to the SAP system



Lesson Objectives

After completing this lesson, you will be able to:

- Use the basic functions of the LSMW
- Define rules in the LSMW
- Convert data in the LSMW

Business Example

A company wants to use the LSMW tool to convert external data into the SAP target structures. They want to program as little as possible themselves.



Basics

Structure Definitions

Field Mapping and Rules

Read and import data

Administration

Additional useful LSMW functions

Figure 128: Basics

LSMW: Properties



- Availability
 - For SAP R/3 Release 4.0 – 4.5 the LSMW 1.7.2 is available as an add-on
 - For SAP R/3 Release 4.6 the LSMW 1.8.0 is available as an add-on
 - For Web Application Server 6.10 the LSMW 3.0 is available as an add-on
(Functions for 1.7.2 identical)
 - As of Application Server 6.20 the LSMW 4.0 is available built-in
- One-off and periodic transfers from legacy systems into the SAP system.
- LSMW uses standard techniques

The LSM Workbench is a tool that supports one-off and periodic data transfer from non-SAP systems (“legacy systems”) to the SAP system.

The tool supports the easy conversion of data from the legacy system. This data can then be imported into the SAP system using batch input, call transaction, direct input, BAPIs, or IDocs.

The LSMW has a recording function, enabling you to generate a “data migration object” from an entry transaction or a change transaction.

The LSMW has been available since Release 3.0 F. (LSMW-Version 1.0)

Version 4.0 of the LSMW is used for the SAP NetWeaver Application Server.



<http://service.sap.com/lsmw>

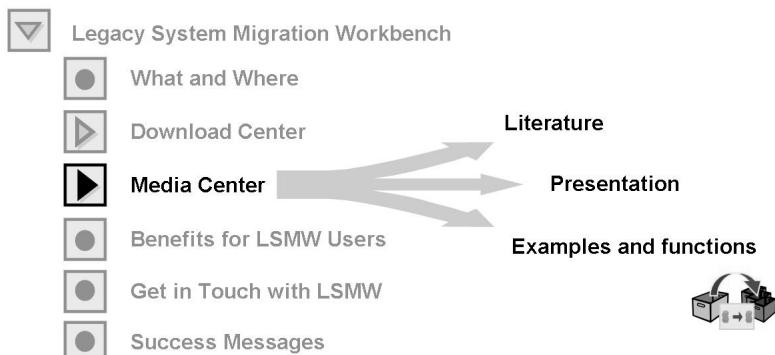


Figure 129: LSMW Information and Documentation

Documentation and tools, such as the tool download are provided for the LSMW.

- On the SAPNET page <http://service.SAP.com/lsmw>
- In OSS or in the notes on SAPNET under component **BC-SRV-DX-LSM**
- Email: lsm@sap.com

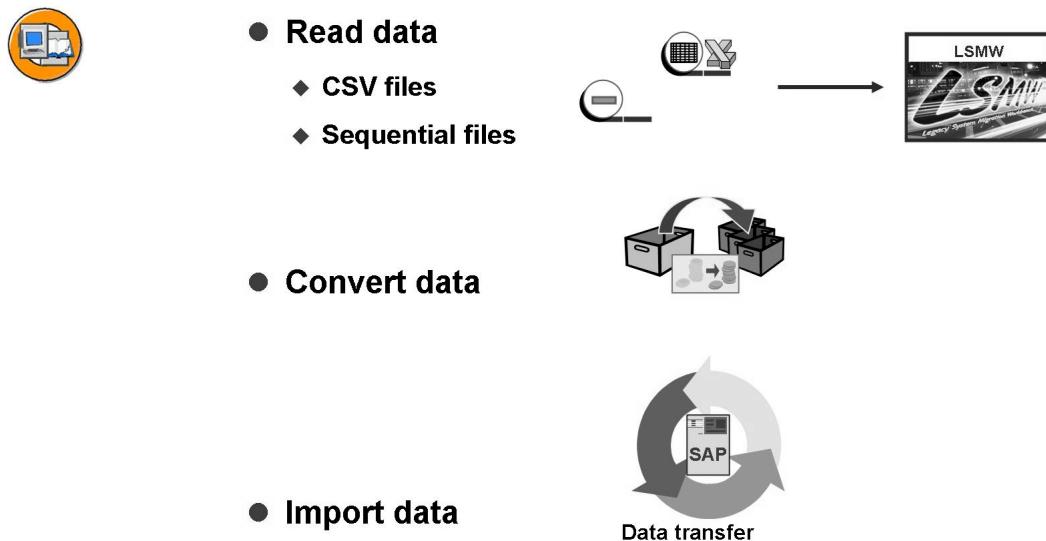


Figure 130: LSMW Core Functions

The central LSMW functions are:

- Import of files containing data from the legacy system into an LSMW format.
- Conversion of data, that is, formatting and assigning it to the appropriate structure.
- Calling standard data transfer programs or BAPIs or IDocs to actually transfer the data. This step also recognizes the DX-WB. There, it is called “loading”. Both the LSMW and DX-WB tools offer this step. When you should use which tool and how the tools work together will be discussed later on.

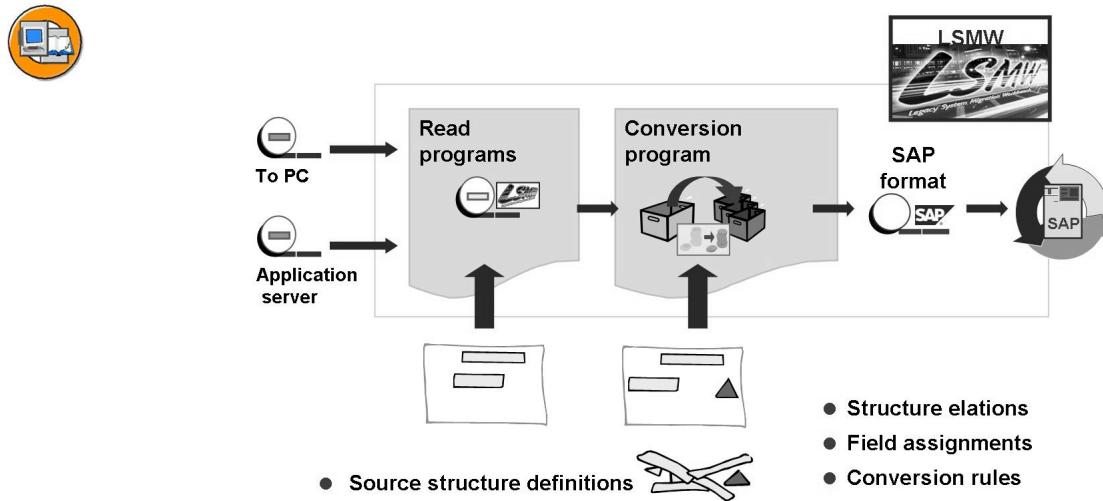


Figure 131: LSMW Concept

The LSMW reads files and converts their contents into corresponding target structures and fields, and creates a file from this in the required SAP format. This file can then be used to transfer data. SAP does not deliver **any standardized conversion programs**. The conversion programs **are generated from predefined rules**.

Follow the steps below:

- The LSMW enables data to be read from presentation servers (PC) or from application servers. To read files, the source structure of the files must be defined.
- The file content is then read into an LSMW internal format and stored as an LSMW file.
- Afterwards structure relationships and field assignments must be defined. These determine which field from **the source structure** is mapped to which field in **the target structure**.
- For each source field that is mapped to a field in the target structure, a conversion rule has to be defined. Here you define how the field value of the source structure is transferred to the field of the target structure.
- This definition is used to convert the contents of the LSMW file into the target structure.
- The LSMW generates two executable ABAP programs from the structure definitions and from the conversion rules. The result is a file in SAP format, which is processed by starting the transfer program (also possible within LSMW).

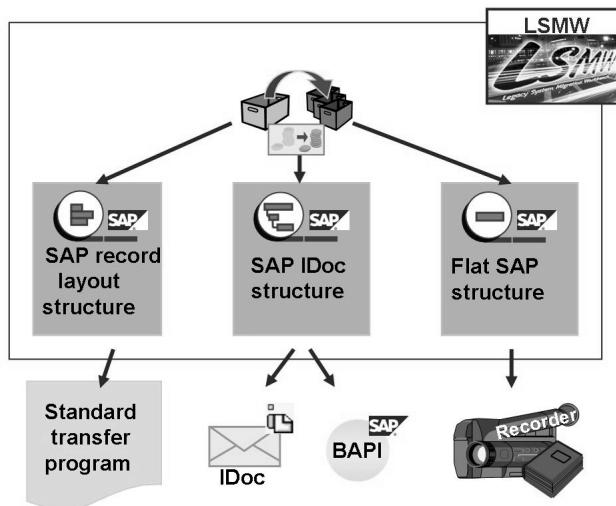
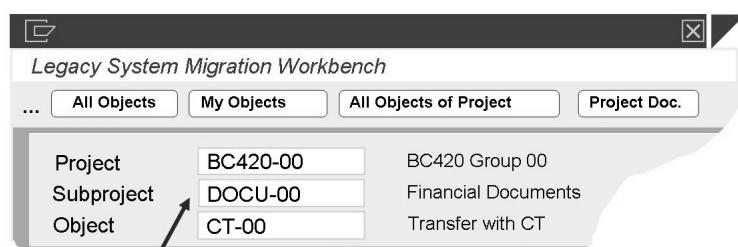


Figure 132: LSMW Transfer Procedure

The LSMW supports the following transfer techniques:

- Standard transfer procedures that work with batch input, call transaction, and direct input.
- Data transfer using IDocs.
- Data transfer using BAPIs.
- Creation of a recording using the transaction recorder and generation of a batch input session



- All Objects
 - ◆ Overview of objects created in the system
- My Objects
 - ◆ Overview of objects the user has created
- All Objects of Project
 - ◆ Overview of objects of current project

Figure 133: Project Overview

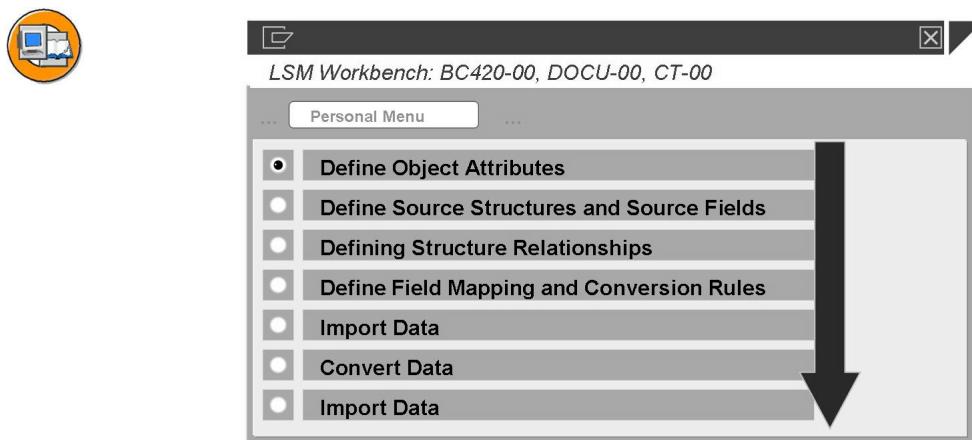
The LSMW is started with transaction code **LSWM**.

On the initial screen you can create subprojects and objects by choosing *Create New Projects*.

- **Project:** Maximum of 10 digits for the name for your data transfer project. If you want to copy data from more than one legacy system, you can, for example, create a new project for each legacy system.
- **Subproject:** Maximum 10 digit ID that will be used for a grouping characteristic.
- **Object** Maximum 10 digit ID for the name of the business data object.

The following functions are available:

- **All Objects:**
Creates an overview of all available projects.
- **My Objects:**
Displays an overview of all objects you create.
- **All Objects of Project:**
Displays a tree structure with all objects contained in the selected project.
- **Project documentation:**
If documentation was created, this displays all the documentation for the individual dialog boxes and steps. You can print and send the documentation, or save it in various file formats.



The work steps are processed from top to bottom.

Figure 134: The Most Important Work Steps

A total of 26 different work steps are provided. The work steps displayed are **dependent on the selected object**.

You can create a customized **personal menu** of the work steps. Here, you can choose which of the object-dependent process steps you want to be displayed. The diagram shows the most important work steps.

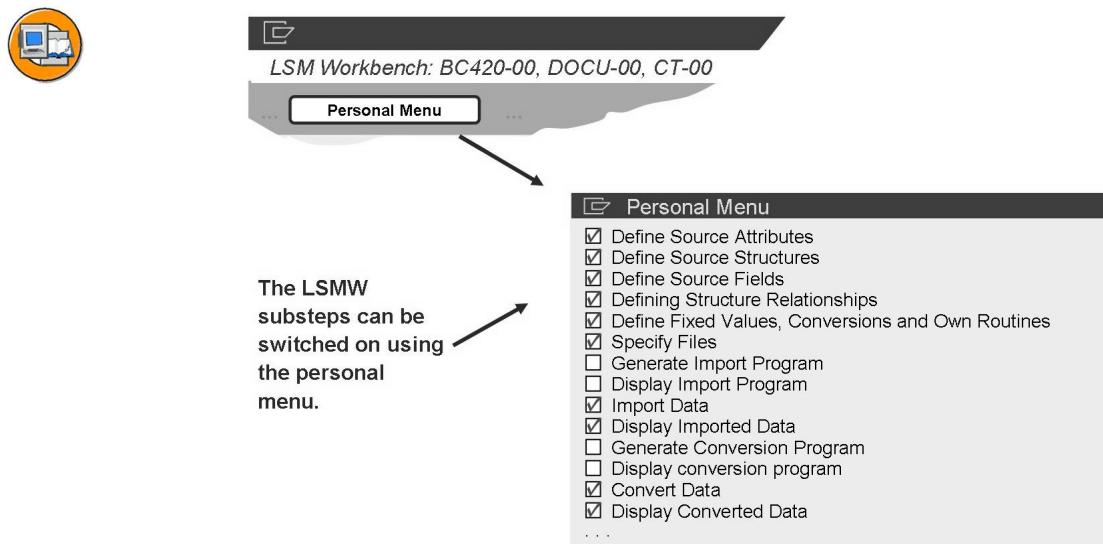


Figure 135: LSMW - Substeps

In the LSMW basic settings only the main substeps are displayed. You can add to these by selecting your required substeps in the **Personal Menu** (for example, *Display conversion program*).

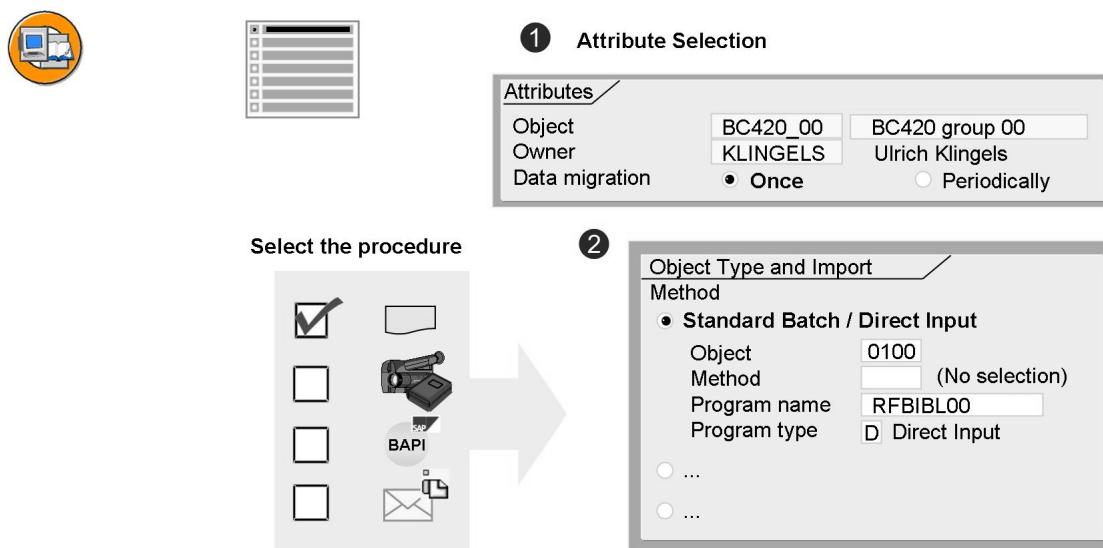


Figure 136: Object Type and Import Method

The first step is to define the object attributes.

You can give the object a name. Name your object. By entering your name in the *owner* field, the project will be added to the list of all the projects you have created. You can find it afterwards in the initial screen under *My Objects*.

Choose whether the data transfer is a one off or to be done periodically.

With periodic transfers the PC cannot import files. (See the unit about sequential files, background processing)

Choose the object type and import method. Possible entries, (F4) help is available for the input field.

- For batch input, call transaction, and direct input, program documentation is available under the program name (choose the Glasses icon.)
- Do not be irritated if, during the record layout procedure, the LSMW displays the following: "Standard batch/direct input". This means all standard transfer procedures that work with batch input, call transaction or direct input. Call transaction is also possible if it is provided by the procedure.

If you have selected batch input recording, you can add recordings by clicking the arrow. Recordings in the LSMW are presented at the end of the course.



Caution: Caution: If you choose the import method BAPI or IDoc, the system checks, when saving, whether a partner profile already exists for the preset partner (see IDoc unit) and the selected message type. If not, the system attempts to create a profile. This will become clear later on in the relevant unit.

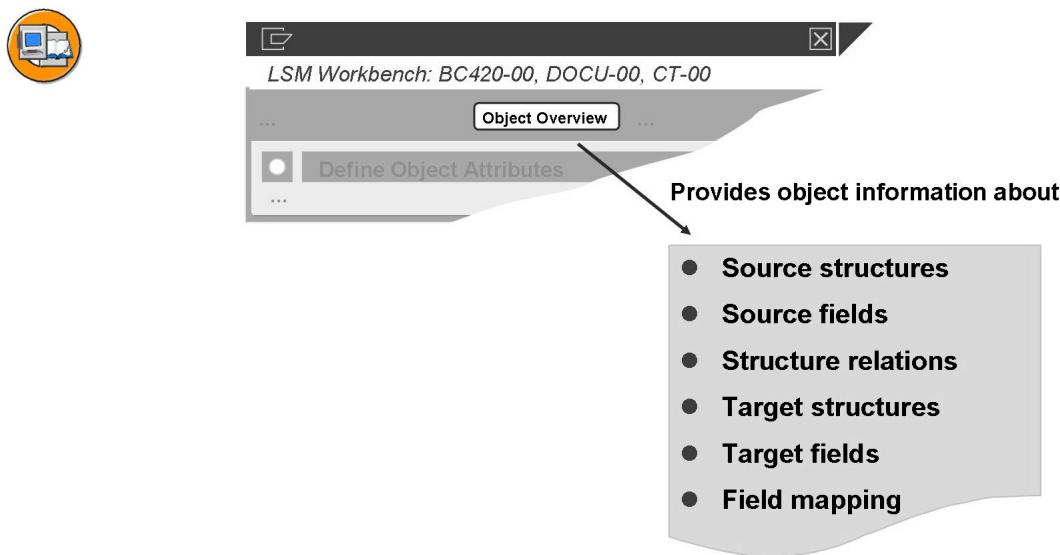


Figure 137: Creating Object Overview

To create a list of information about the points mentioned above, choose *Object overview*.

This list can also of course be saved on your PC and also used for project documentation.

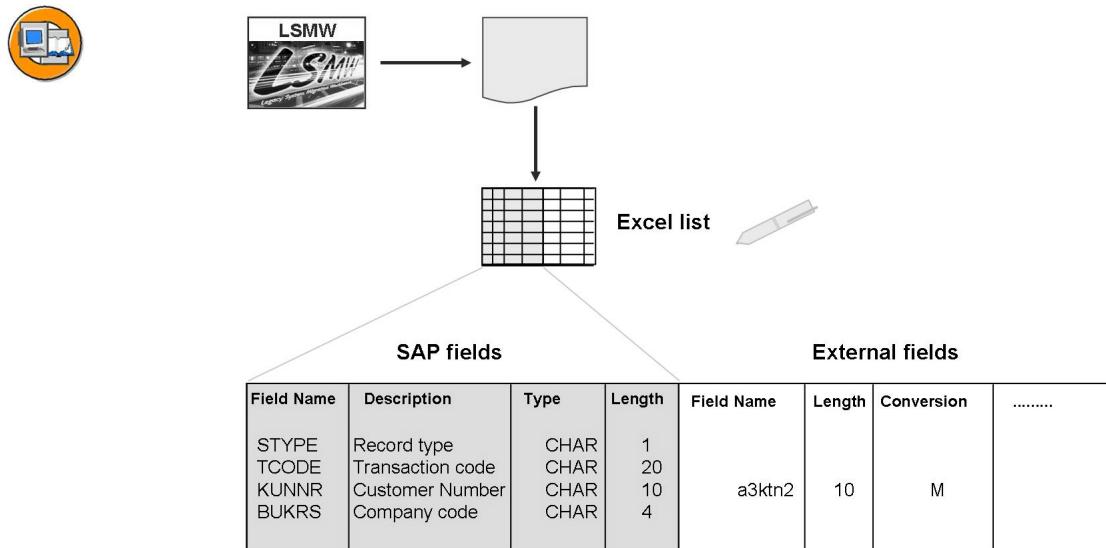


Figure 138: Converting to Paper

Before you start using the LSMW, create a “**mapping on paper**” for the object. Create and print an **object overview**.

At this time, the overview only contains the design and description of the SAP structures and the fields they contain. You can use this overview to help you map these target structures and fields to the corresponding source structures and fields.

Note: The rules **M**, **M+C**, **V**, **F** are SAP proposals. Feel free to define your own rules and abbreviations and use them in your project. You must ensure that each project team member knows and understands the rule set and the agreed rules and keeps to the specifications.

Emphasize that in practice, you must work with a mapping plan. If you spend time doing this, it will pay off later.

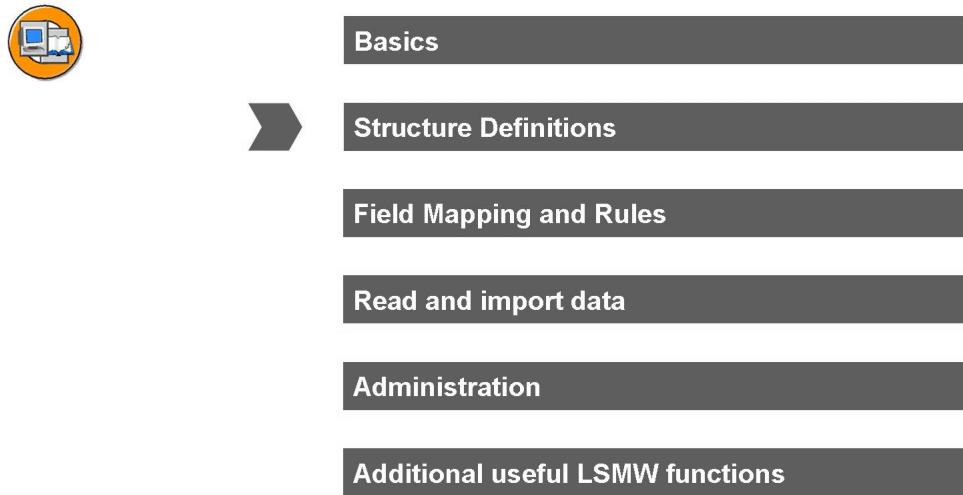


Figure 139: LSMW Structure Definition

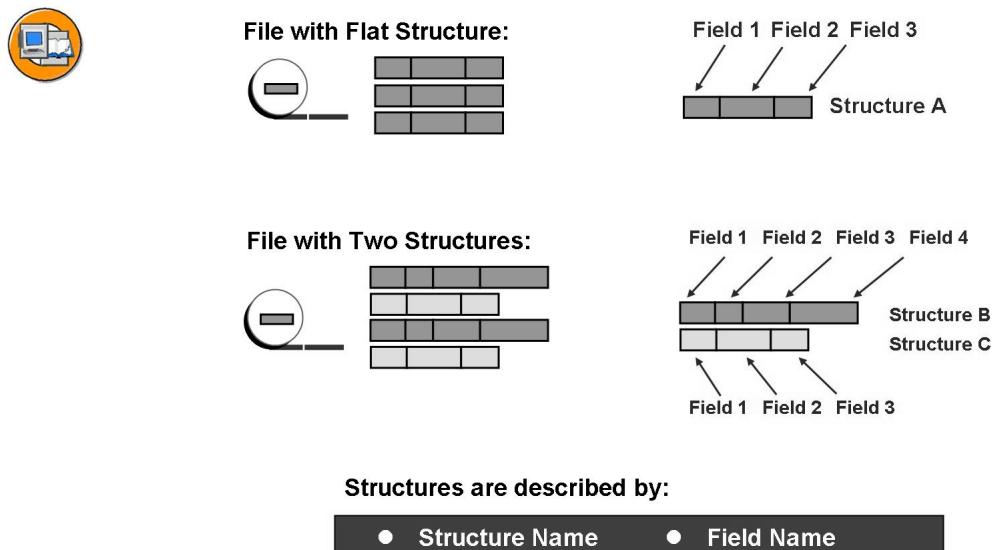


Figure 140: Source Structures I

So that the LSMW can import the data from the external system, the structure of the data must be determined. This is known as **defining the source structure**.

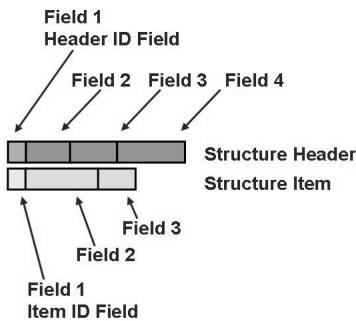
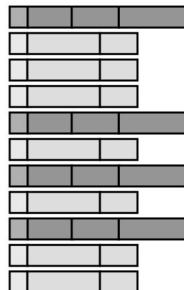
The source structure may be structured in different ways.

If the source structure is a flat structure, each line in the file has the same structure.

If the file consists of two different structures with a ranking order, a source structure (flat structure) must be created for each structure.

For the definition, the structure name and the field names, as well as the technical characteristics of the fields, must be defined.

In principle, any number of structures can be imported.

**File with Header and Positions:****Structures are described by:**

- **Structure name**
- **Field Name**
- **Field ID**

Figure 141: Source Structures II

If the file consists of two structures with different ranking orders, each position in the first line in the first position must contain information describing the record type. This is called the **ID field**.

For the definition, the structure name and the field names, as well as the technical characteristics of the fields, must be defined. In addition, when defining the fields, the ID field must be identified as such.

For the example displayed here for FI documents, a file like this with header entries and item entries will be predefined.

To read different external structures with different ranking orders, the data may come from several files. For example, for finance documents, the document headers may come from one file and the items may come from another file. The assignment then takes place using a field with the same name (the document number in the example). You can find an example of this in the attachments.



Define Source Structures

- Name of source structure
- Description of the source structure

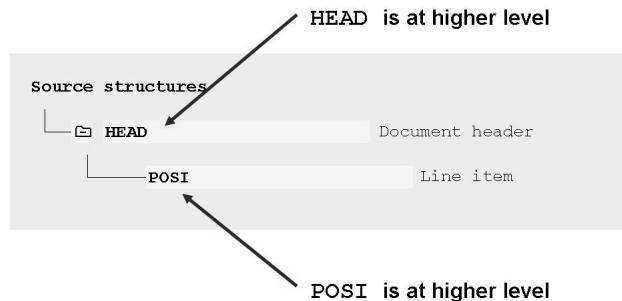


Figure 142: Define Source Structures

In this step, you define the object structures including name, description, and hierarchical relationships.

In the dialog box, choose Change. You can now choose to create new structures, or change, rearrange, or remove these. Use the relevant pushbuttons to execute these functions.

If you are creating more than one structure, a dialog box appears asking you to define the relationship between the structures: On the same level or on a lower level?

This is important for assigning the source structures to the SAP target structures later.



Caution: You can create only one structure per recording for migration objects created through a recording of a transaction as there is only one flat target structure available per recording.



Define Source Fields

- Field name
- Field length
- Field ID
- Field type

Assignment of Source Fields	Help functions
<p>Source Fields</p> <ul style="list-style-type: none"> HEAD <ul style="list-style-type: none"> SET1 C(001) record indicator REF_NO C(009) Reference number : ... POSI <ul style="list-style-type: none"> SET2 C(001) Record indicator DATE DDMY(010) Posting date : ... 	<p> ● Create a single field</p> <p> ● Copying fields from files or other objects</p> <p> ● Creating/editing fields using an editor matrix</p>

Figure 143: Define Source Fields

In this process step fields are defined for the structures defined in the previous step.

There are three ways of creating fields:

- By choosing “*Create*”, fields can be created with their type and length step by step.
- By choosing “*Copy from Another Object*” source fields, for example, from an Excel file, a special data file or from another LSMW object, can be imported. If you are interested in this option, we recommend that you read the special LSMW documentation under service.sap.com.
(For more information see the LSMW online documentation).
- By choosing “*Table Maintenance*”, fields can be created in one work step. Unlike the function “*Create*”, each field is created separately.

Field Types



- Fields with alphanumeric contents (C field)
- Fields with numeric contents (N field)
- Date in date format (DDMY, DMDY, DYMD)
=> Internal ABAP format YYYYMMDD
- Amount fields (AMT1, AMT2, AMT3, AMT4)
=> Internal ABAP format with a period as a decimal place
- Fields with packed contents (PAC0PAC9)

When the data is imported you can specify how the data is converted:

- Date values are converted into the internal date format (YYYYMMDD)
- Amount fields are converted into the calculation format (1234.56, no grouping characters, only the decimal point). This format can only be used for amounts with two decimal places.
- Packed fields with and without decimal places. The number for PAC“X” specifies the number of decimal places. In most cases, you will work with CHAR fields (in the legacy data files) in practice. Refer to the LSMW documentation for more information.

 **Note:** Note that the usage of these special types is designed for reading the data in the LSMW and therefore, certain conversions of the data can be carried out automatically by the LSMW during reading. This can be an advantage for the rest of the data transfer. In particular, note the date here. As you continue, monitor how the date from the external file is eventually converted and transferred. Then the advantage of these read types becomes obvious!

The following exercises in this course have been designed so that the advantages of these types become more obvious.

Under *Selection parameters for importing/converting data*, a flag can be set for fields of structures from the highest hierarchy level. If you set this flag, the field in question is provided as a selection parameter when you import and convert the data. This is usually used for tests.

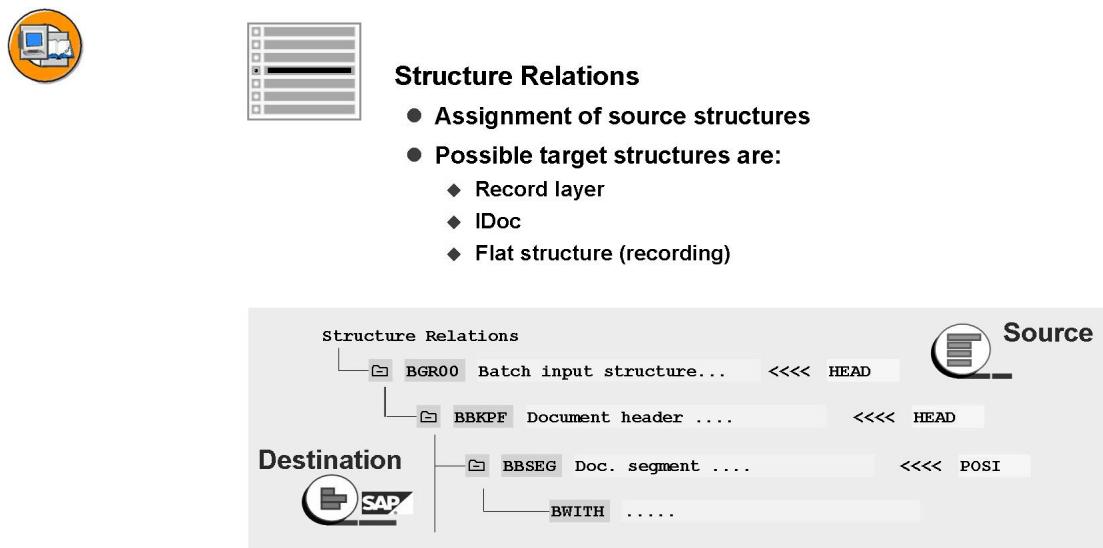


Figure 144: Defining Structure Relationships

The structure relationships define the relationships between the source and target structures.

The possible target structures are set when the object type and import method are selected.

Usually there are target structures that must be selected (“required segments”). In this case the system informs you: “This structure must be selected.”

To set structure relationships, place the cursor on a field in the SAP structures or target structures. Choose **Relationship**. A dialog box appears with a list of the source structures you have created. Select a structure from this list.

If you want to change the relationship, you must first delete the existing relationship using the relevant pushbutton.

A **check function** is available to check the structure relationships for errors. Messages for this check are displayed in the status bar: “No structure relationship errors found.”

 **Note:** Many batch input, call transaction and direct input programs use a control record called BGR00 or BI000. Always assign the highest hierarchy level (“header structure”) of the source structure to this record.

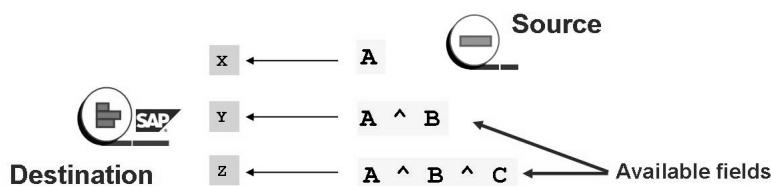
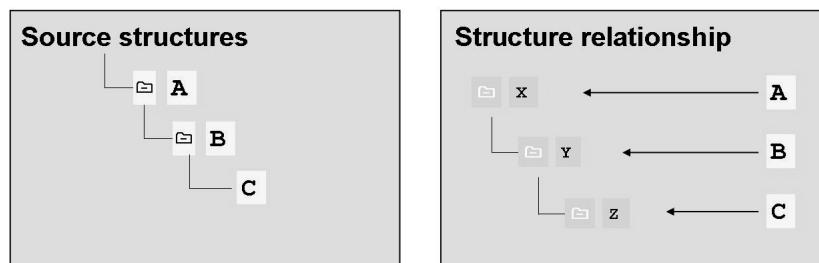


Figure 145: Hierarchical Display

In certain circumstances you may want to assign several source structures to one target structure.

- In this case,
create your target structures.

Assign the subordinate source structure to the target structure.

The fields from both source structures are available for the fields of the target structure.

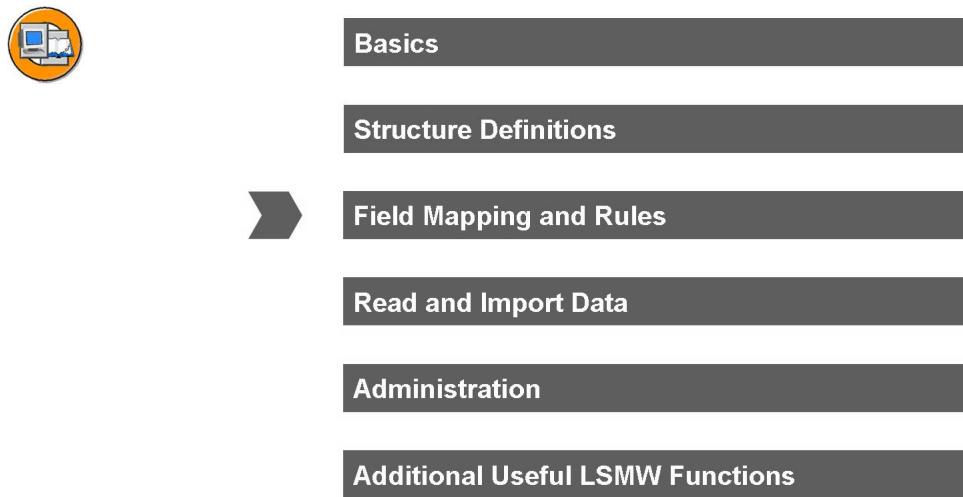


Figure 146: LSMW Field Mapping and Rules

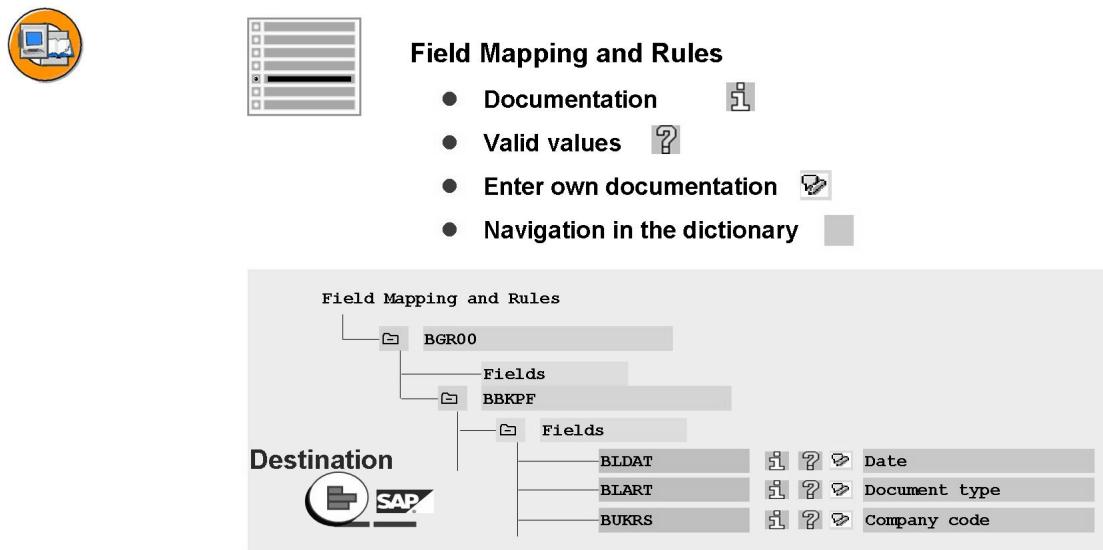


Figure 147: Overview: Field Mapping

In this step, you assign source fields to the target fields and specify how the field contents are to be converted. The rules of the mapping plan are now used in the LSMW. The aim is the converted file, the **record layout**.

The system displays all fields for all target structures you selected in the previous step.

The following information is displayed for each target field:

- Field ID
- Source fields that may be assigned
- Rule type (for example: default value, conversion)
- Code



Hint: Some fields are prefilled by the system. These are *technical fields* with the **default** rule type. Changes to the default may severely affect the data conversion process. You can reset a value to default that you have changed by choosing *Extras → Restore Default*.

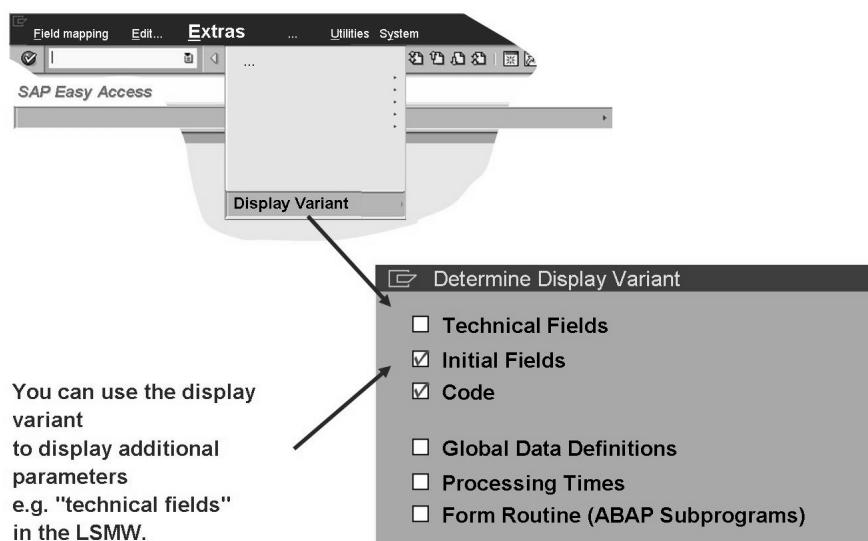


Figure 148: Display Variant in the LSMW

You can specify the field assignment using the display variant.

The visibility of objects can be determined by the “display parameters”. You can get a detailed view of the processing logic and global data definitions of the LSMW. (Display processing times and Global data definitions). You could also add your own ABAP code for specific processing times.

(See documentation on LSMW).

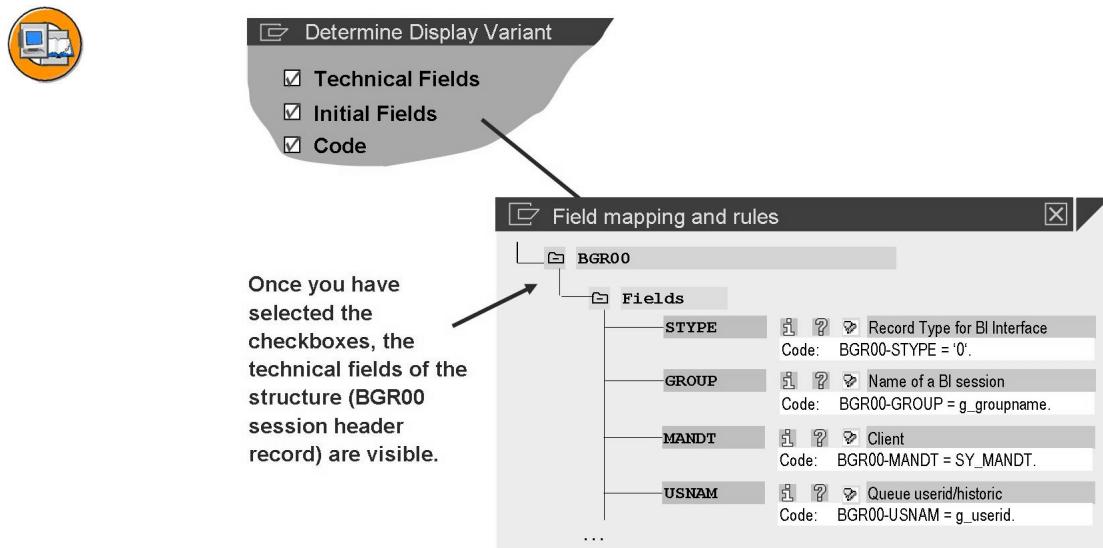


Figure 149: Display Variant: Technical Fields

In the LSMW basic settings technical structures are grayed out (for example: session header record, control record of IDoc.)

To make the parameters of the session header record BGR00 visible, choose *Extras → Display variant* and activate **Technical fields**.

These technical fields are partially preassigned by the LSMW and do not usually have to be changed. The functions of the display variants are not required for this unit dealing with the basics of the LSMW. However, later on in the course, you will be working closely with display variants!

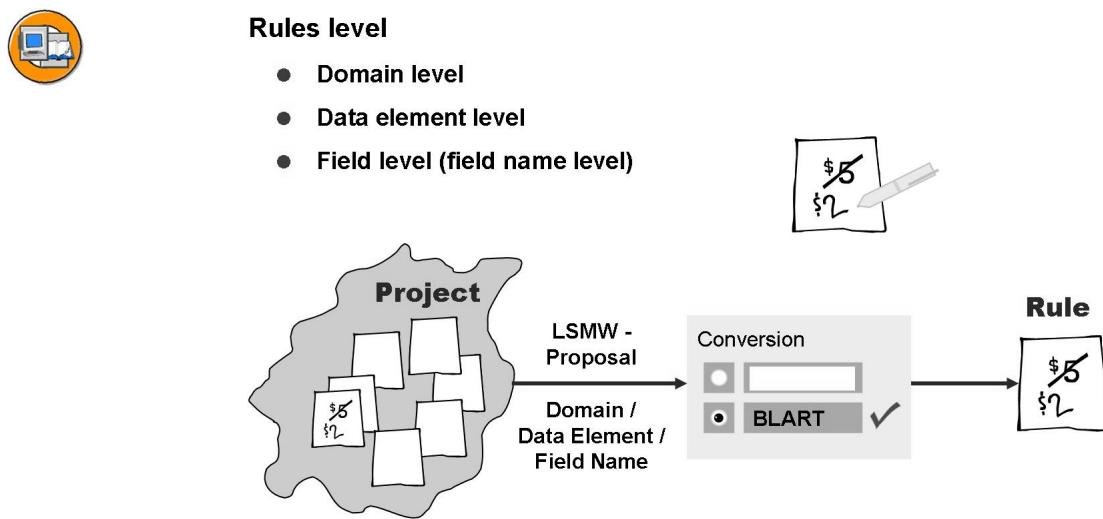


Figure 150: Reusability of Rules

Reusable rules are those that are available to **all parts of a project**, and can be used in all objects in a project. Reusable rules are: *Default values, translations and own routines*.

When you are assigning a reusable rule to the target field, the system suggests between one to three names.

These will be names from one of the three levels: Domain, data element or field level. They are distinguished as follows:

- If the rule is set at domain level, this rule is suggested when the target field is defined using this domain.
- If the rule is set at data element level, this rule is suggested when the target field is defined using this data element.
- If the rule is set at field level, this rule is suggested when the target field is defined using the same field.

We recommend that you accept the name proposed by the system. An exception to this recommendation is if the domain is of a very general type, for example, “CHAR1”.

This naming procedure keeps the number of conversion rules small and guarantees data consistency in the data transfer



Sequence of use in the assignment process

- 1) Assignment of the source field to target field
- 2) Definition of the conversion rule

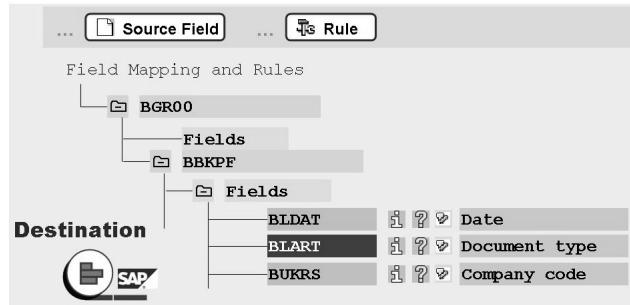
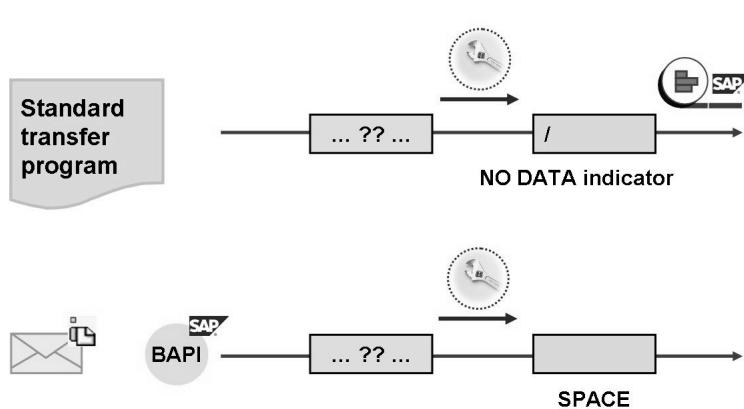


Figure 151: Assigning the Source Field

- Assigning source fields:
To assign a source field, place the cursor on a target field in the tree structure and choose *Assign source field*. A list of available source fields appears, from which you can select the desired source field by double-clicking it.
- Deleting source field assignments:
To remove the assignment from a source field, place the cursor on a target field in the tree structure and choose *Remove source field*. If only one source field is assigned, this assignment is removed. If more than one source field is assigned, a list of all assigned source fields appears. To select the desired field, double-click it.
- After you have assigned the source fields, you define the conversion rules. The default is “MOVE”. Various standard techniques are available by choosing the relevant pushbutton.

**Use**

- Normally:

**Figure 152: Rule: Initial**

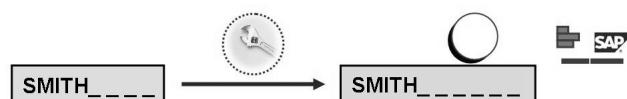
In this step, **code** assigned to the target field is deleted. Source fields assigned to the target field are also deleted. In addition, source fields assigned to the target field are deleted.

- For standard batch input, call transaction and direct input: Nodata characters (defined in session header records, for example, BGR00, BI000)
- For batch input recording: / as NODATA characters
- For BAPIs and IDocs: Type-specific initial value of the field (that is, character field → blank; numeric field → 00...0)

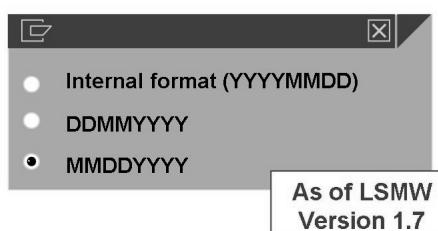
**Use**

- Target field is filled with ABAP command **MOVE**

... ...

**Exceptions**

- Packed fields
- Date fields
- Amount fields

**Figure 153: Rule: Transfer**

Data is transferred using the ABAP command “MOVE”. For source fields not of type C or N, this means:

- **Packed field:** Unpack in target field WRITE...TO...
- **Date Field** If target field is at least ten characters: Output format is set according to the settings in the user master. If you do not change anything here, the data is left in the internal format; for example, 01.10.1998 (YYYYMMDD).

As of LSMW Version 1.7, the date format can be determined for the data conversion. Note that it is worth using the output format when you are processing templates.

For standard procedures, format the date according to the user master record. Otherwise, errors will occur when you import data. Procedures that work with IDocs or BAPIs (interfaces without dialog) do not process any templates and therefore you must always use internal format for these. This will become clear as the course goes on and you carry out the relevant exercises. It will all fall into place!

- **Amount field:**

For batch input/direct input, that is, for the standard procedures: The amount value is prepared in output format according to the settings in the user master. The LSMW does this automatically.

For BAPIs and IDocs: The amount value is always left in internal calculation format.

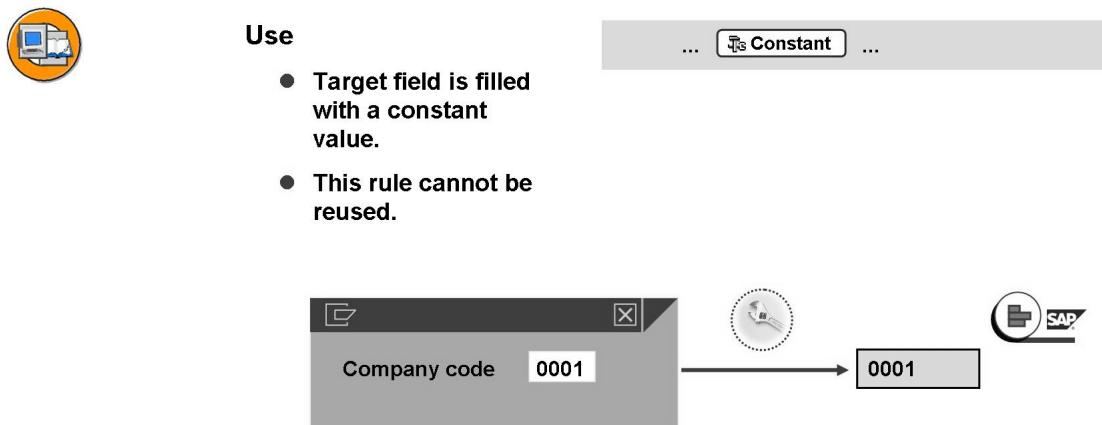
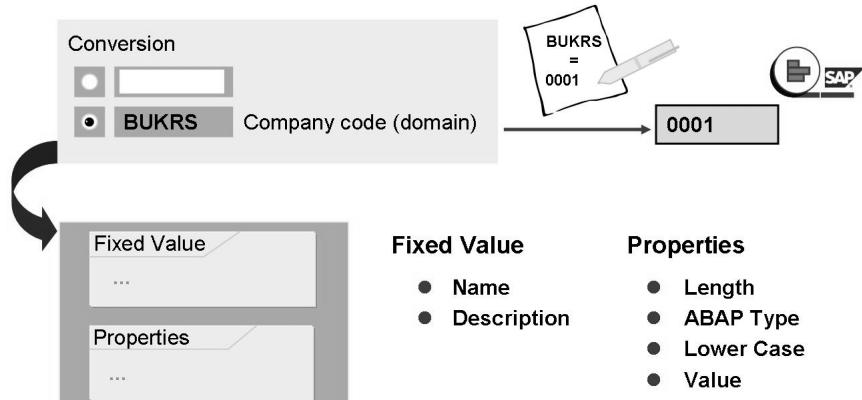


Figure 154: Rule: Constant

The target field is assigned a constant.

**Use**

- Target field is assigned to a variable

**Figure 155: Rule: Fixed Value**

Fixed Value: The system displays default values for the name of the fixed value. These are domains, data elements, or field names.

Choose a proposal or enter a new name.

You can specify the field description, length, type, value, and whether uppercase or lowercase must be used.

**Use**

- Select source field



- Assign conversion

**Figure 156: Rule: Creating the Conversion**

Using conversion you can convert field values in the external system to valid values in the SAP system.

Conversion: The system displays default values for the name of the conversion.

These are domains, data elements, or field names.

Select a proposal. You can also specify a new name, however you do not need to do so in the exercises and examples here.

How these proposals are used and particularly how they are reused has an effect on other objects of this project in the LSMW. Some rules can be reused in all parts of a project. The selection by domain, data element and field name influences later conversion proposals of the LSMW in other objects of the project.

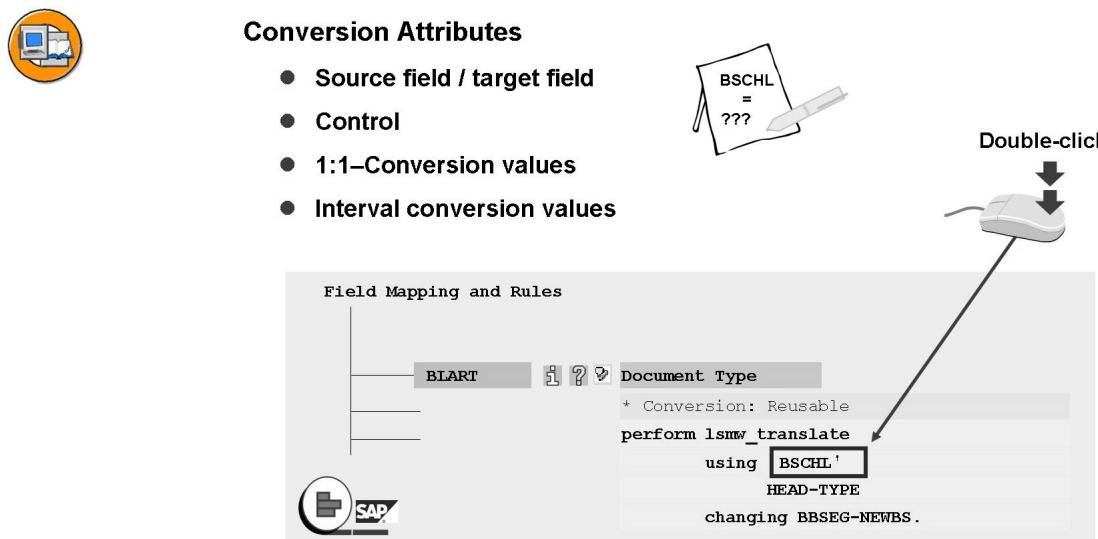


Figure 157: Rule: Change Conversion

To create a conversion you must define the source and target fields. You must specify how the conversion is controlled.



Hint: If you choose the **conversion** rule, you can double-click twice to go to the rules of the conversion (the conversion will be defined there). Note that you must make the second double-click on the character literal of the target field in the coding editor (see the slide).

You can choose “1:1 conversion” or interval conversion.



Conversion attributes

- Control



	Variant	First Alternative	Second Alternative
1:1 Conversion	<input checked="" type="radio"/>	<input type="radio"/>	
Interval Conversion	<input type="radio"/>	<input checked="" type="radio"/>	
Initial Value		<input type="radio"/>	<input type="radio"/>
Constant		<input type="radio"/>	<input type="radio"/>
Transfer (MOVE)		<input type="radio"/>	<input type="radio"/>
Your Own Routine		<input type="radio"/>	<input type="radio"/>
No Action		<input type="radio"/>	<input checked="" type="radio"/>

Other

Add 1:1 Conversion Values Automatically

Figure 158: Conversion Control

Here you determine the type of conversion. You can specify which conversion table the system should search first for a value, and which alternative should be selected if no matching entry is found.

Above example: The 1:1 conversion variable has been set. That is, the conversion is carried out using the 1:1 conversion. If the system does not find a matching entry, the first alternative is used. Here the interval conversion is used as an example. If the system still cannot find a matching entry, the second alternative is used.

With **Other** you can decide whether to add the value of the source file automatically to the 1:1 conversion, if there is no matching entry for this value.



Hint: If you do not define any values in the conversion table, but you start the *Data Conversion*, all the representative values of the source file are added to the conversion table.

Note also that the LSMW does not recognize any numbers, rather it works with characters. Therefore, "02" is not the same as "2".



1:1 Conversion values

- Old value
- New value



Old Value	New Value	OK
0004	40	<input checked="" type="checkbox"/>
0005	50	<input checked="" type="checkbox"/>
		<input type="checkbox"/>
		<input type="checkbox"/>

Don't forget to select checkbox

Figure 159: 1:1 Conversion Values

Here you specify which value (old value) in the non-SAP system is to be replaced with the new value.



Caution: Only those values for which the **OK indicator** is set are converted.

One of the most common errors is to forget this indicator! The LSMW then ignores this conversion.



Interval Conversion Values

- Old value from ... to
- New value



Old Value From	Old Value To	New Value	OK
0001	0004	40	<input checked="" type="checkbox"/>
0005	0008	50	<input checked="" type="checkbox"/>
			<input type="checkbox"/>
			<input type="checkbox"/>

Figure 160: Interval Conversion Values

Interval Conversion Values Here you specify which interval value (old value) in the non-SAP system is to be replaced with the new value. Specify the value table with the value intervals to use for value conversion. You can upload the values from a PC file (text separated by tabs) to the table. F4 Possible entries help is available in the “New value” column.

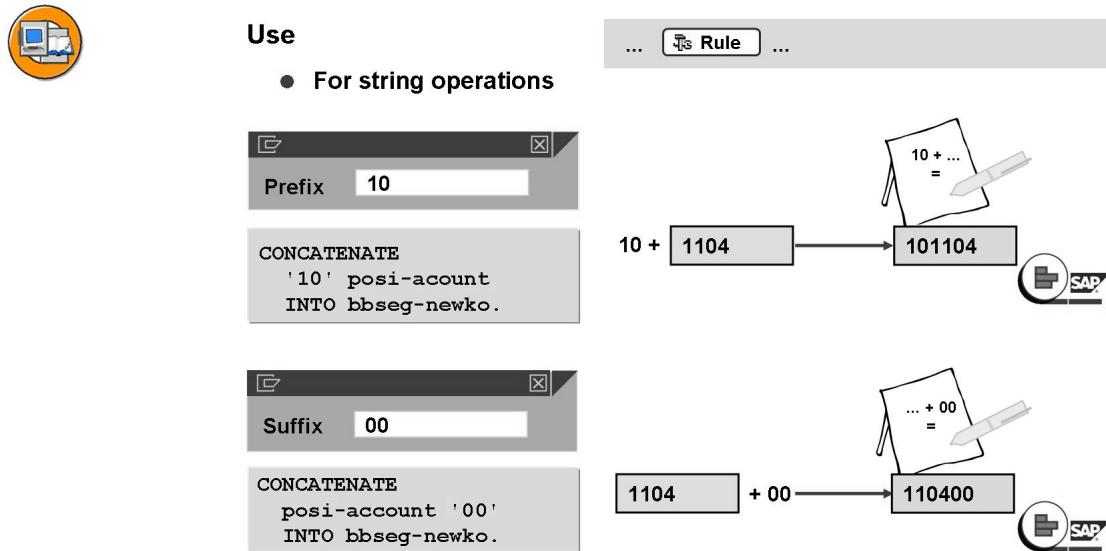


Figure 161: Rule: Prefix / Suffix

You can specify a prefix of your choice, which will be placed in front of the contents of the source field.

You can specify a suffix of your choice, which will be placed at the end of the contents of the source field.

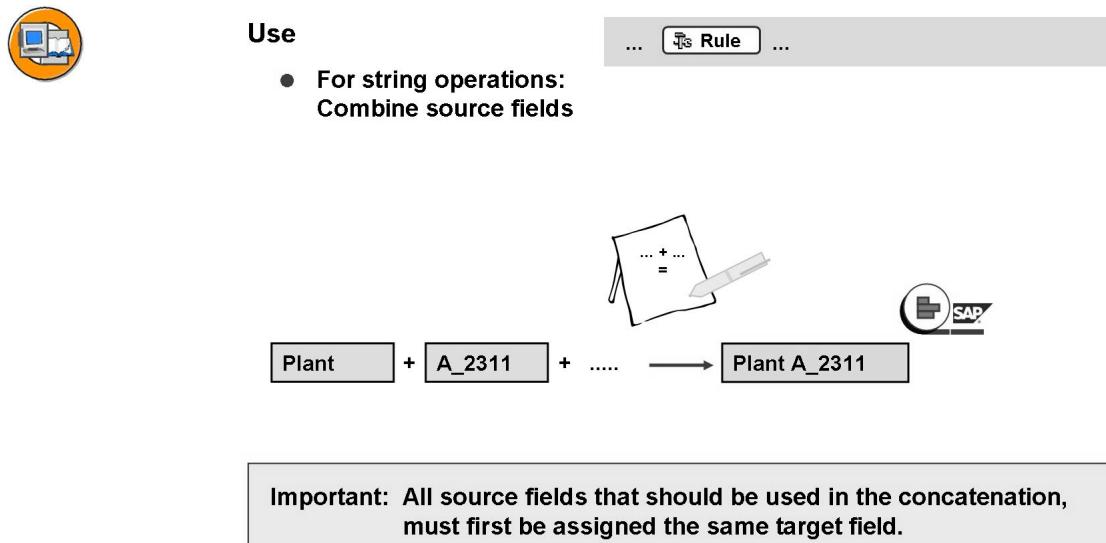


Figure 162: Rule: Concatenation

Concatenation: Concatenation: You can combine two or more source fields.

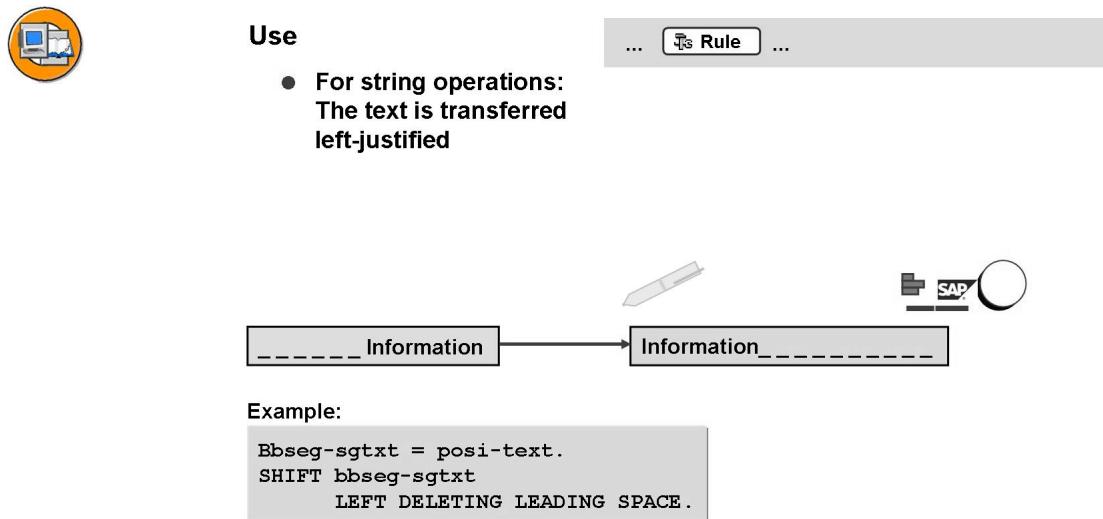


Figure 163: Rule: Transfer Left-Justified

Transfer Left-Justified: This transfers the field contents left-justified.

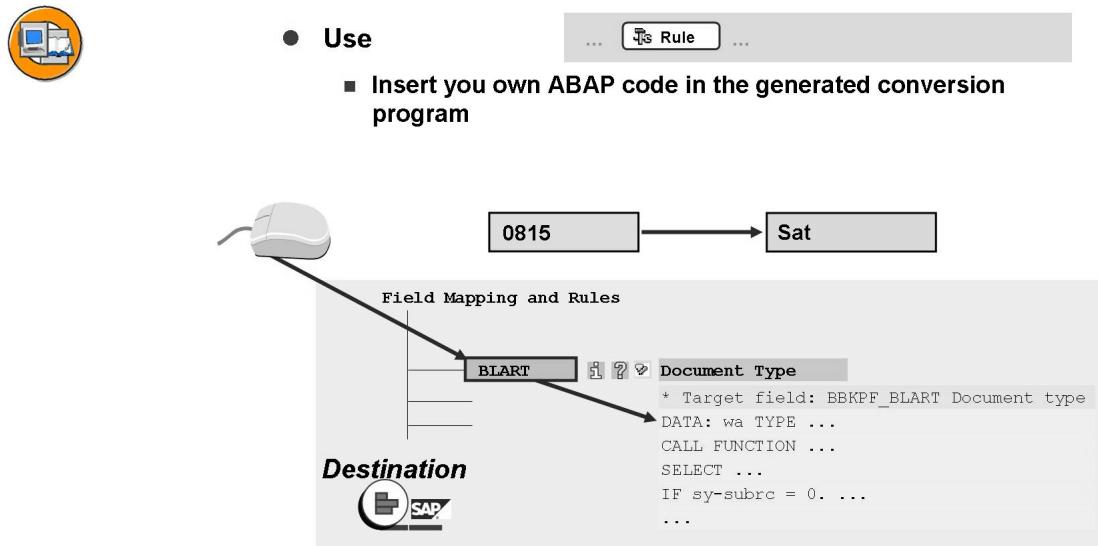


Figure 164: Rule: ABAP code

You can add your own ABAP statements for field conversion to the program. Your statements are then processed for every data record.

By double-clicking on the target field you can switch to the ABAP editor.

The ABAP statements are executed directly at this point in the conversion program.



- **Use**

- **Include subprograms**

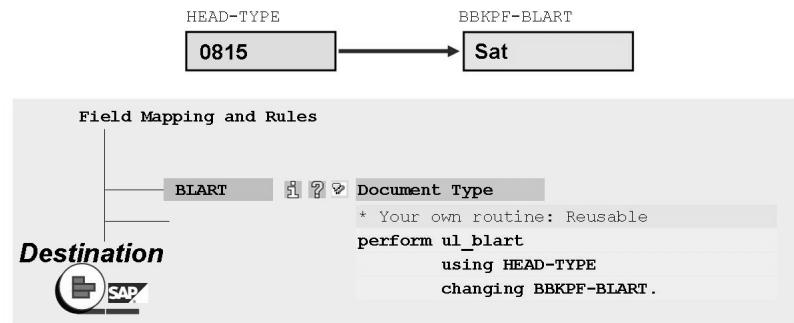


Figure 165: Rule: Your Own Routine

You can include a subprogram and a subprogram call. The subprogram can include your statements for the field conversion. The subprogram will be executed for every data record.

You can reuse subprograms.



- **Fixed Values, Conversions, ...**

- **Overview of the reusable rules:**
Fixed values, conversions, own routines

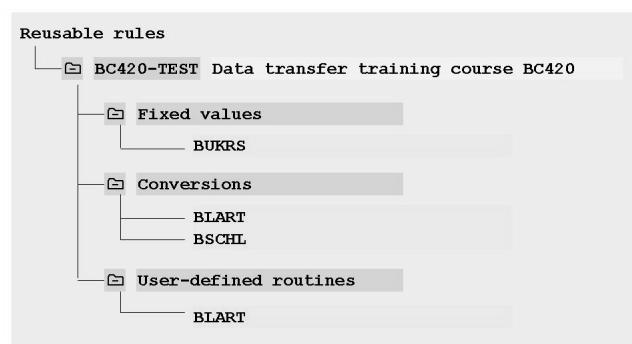


Figure 166: Field Mapping and Conversion Rules

Under the menu entry *Field mapping and conversion rules* you can get an overview of all the defined reusable rules. You can check or change details for the rule by double-clicking.

Rules are displayed in three categories:

- Fixed values
- Conversions
- User-defined routines

They can be edited here again.

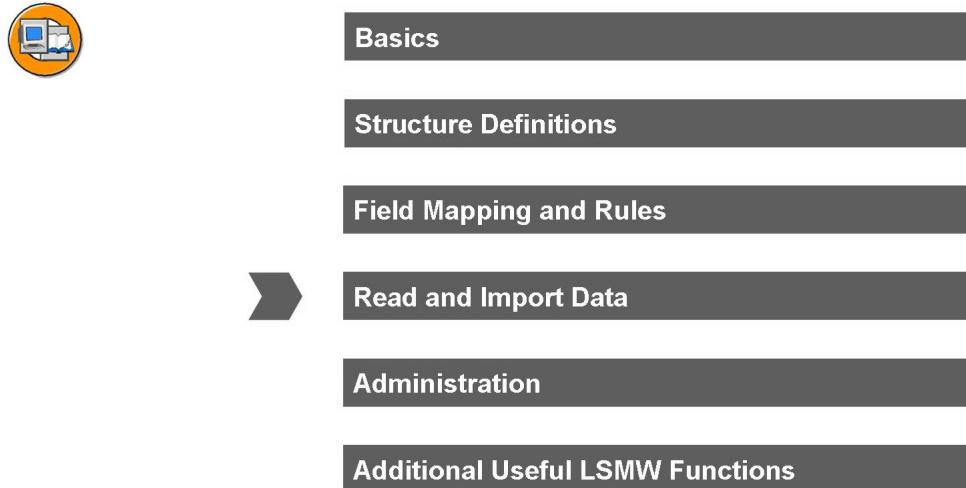


Figure 167: Import LSMW Data and Transfer it to the SAP System

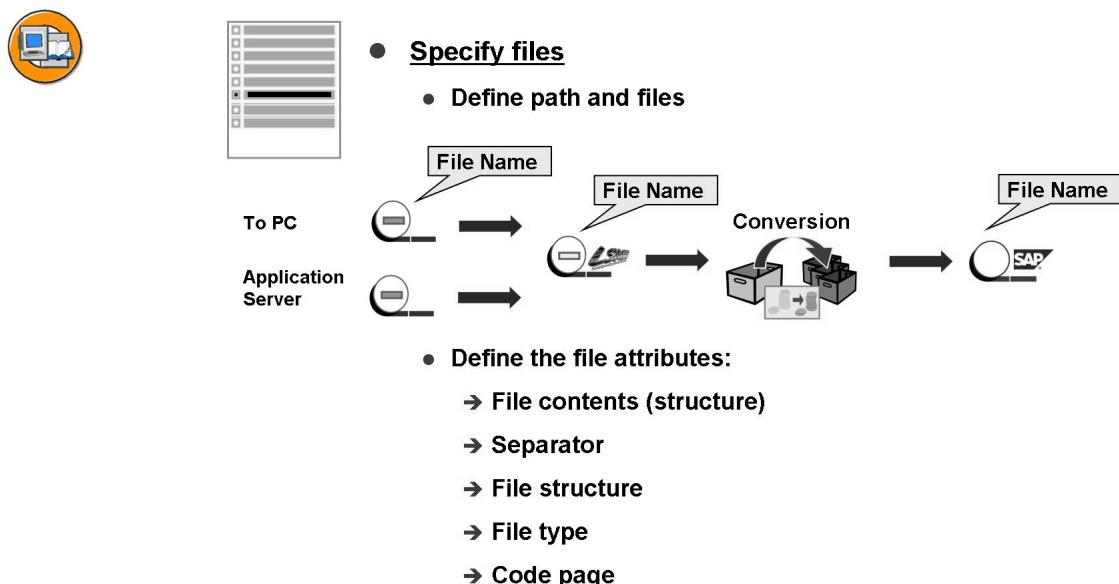


Figure 168: Specify Files

Specify all the files that are to be used in the following steps:

- Your legacy data on the PC and/or SAP R/3 server (place cursor on relevant data and choose *Create*).
- The internal LSMW file for the read data. (This is used internally by the LSMW. Do not confuse it with the external data file).
- The file for the converted data (in SAP format, for example, record layout).

Enter the file attributes:

- File path, file name, and file description.
- Determine whether the file contains data belonging to several source structures or data belonging to one source structure.
- Define the technical record description and the separator, In practice, these are, for example, a tabulator or a comma for CSV files that are created using Excel.



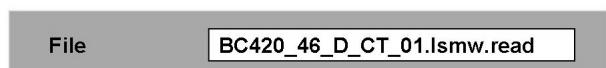
Hint: Application server: Ensure that the name of the SAP system on the operating system is under the name user-ID <sid> adm. Make sure the selected directory has read or write authorization, as appropriate.

If you work with a file on the front end, the transfer in the background is not possible. (See also GUI_DOWNLOAD / GUI_UPLOAD function modules).

- Under “code page ID” enter the code page ID of the external system. If the external file is not in the SAP system code page, in many cases, the external data can still be read if you choose the external code page. For further information, see the notes about this topic under service.sap.com/notes. In addition, refer to the notes from the earlier unit “Sequential Files” about Unicode.



- **LSMW import file: Path and file names**



File name: <path><file name>.lsmw.read

File name = project name
+ subproject name
+ object name

- **LSMW conversion file: Path and file names**

- Define physical path and file name
- Define logical path and logical file if this is supported by the transfer object



File name: <path><file name>.lsmw.conv

File name = project name
+ subproject name
+ object name

Figure 169: LSMW Import and Conversion Files

The system proposes a name for the file containing the imported data. If you want, you can change the name.

For the LSMW file containing the converted data, you can define physical or logical path and file name independently of the object.

You can only enter data in the fields *Logical Path* and *Logical File Name* if the batch input or direct input program called later supports this.



Hint: 1:

You are free to choose the names for the paths and files, within the naming conventions of the operating system.

2:

If your data is stored in more than one set, you can enter a wild card (*) as your file name. You can enter the possible values for ,* under *Values of Wild Card*.



- **Assigning place holders (wildcards) in the file names**

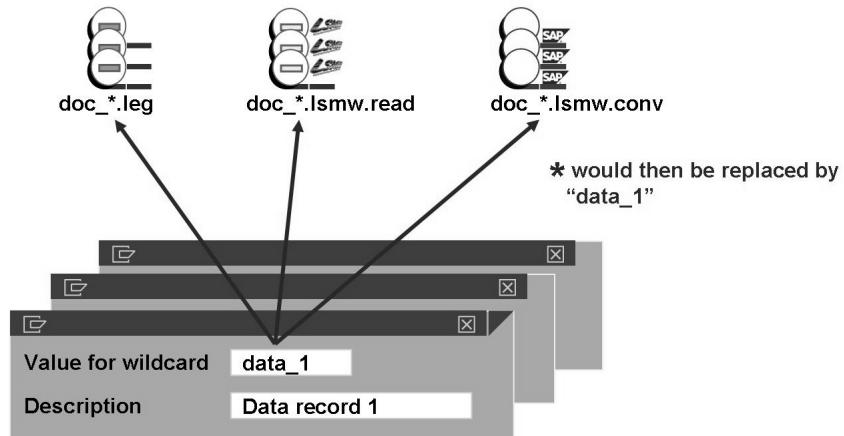


Figure 170: Wildcards in File Names

Example of the use of wildcards in file names: Assume the legacy data is contained in the following four files:

- File 1: D:\Mig\Purchase Orders\PO Header 1.txt
- File 2: File 4: D:\Mig\Purchase Orders\PO Position 1.txt
- File 3: D:\Mig\Purchase Orders\PO Header 2.txt
- File 4: File 4: D:\Mig\Purchase Orders\PO Position 2.txt

Each two files (*1.txt and *2.txt.) form one “set”. File 2 contains the item data for the header records in file 1, and file 4 contains the item data for the header records in file 3.

During data import, first import file 1 and file 2, and then file 3 and file 4.

Use the wildcard “*” in the file names and define the values 1 and 2 for the wildcard.

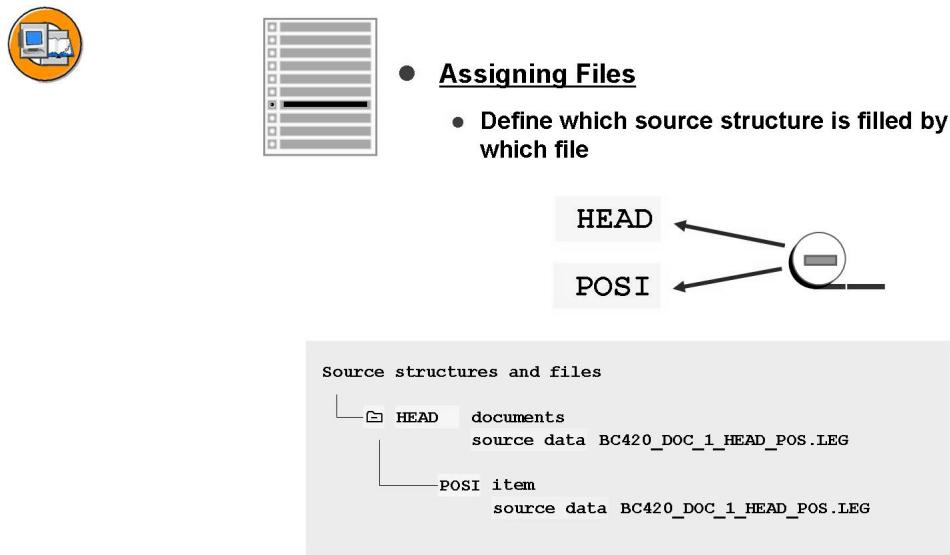


Figure 171: Assigning Files

In this step you assign the defined data to the source structures.

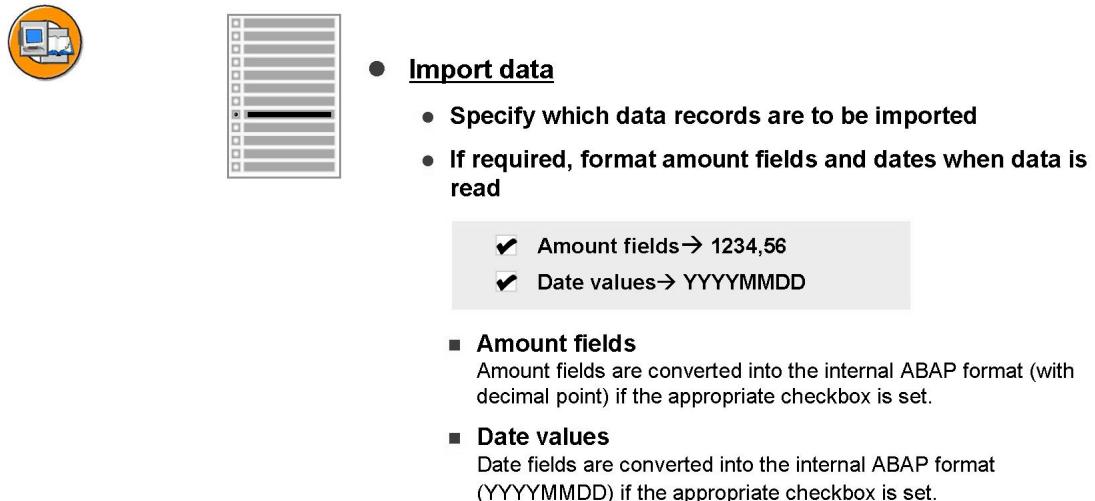
A file that contains data for several source structures can be assigned to several source structures.

A file that contains data for only one source structure can be assigned to only one source structure.

If you later change the file name or the file attributes, the file assignment remains the same.



Hint: It may also be useful to read the data from several files. Example: The document headers could be read from a header file and the document items could be read from an item file. A field with the same name in both files will then act as a key for assigning the records. The LSMW will then automatically assign all corresponding records (for example, document number in the header and document number in the items) to each other. No further measures are required.

**Figure 172: Importing Data**

You have the following options with the function *Read data*:

- If you want to process all the data belonging to an object, execute the function.
- If you just want to migrate part of the dataset, you can restrict the amount of data that you want to migrate in the field

General Selection Parameters.

Select your data in “Transaction number” “from ...to ...” Your data selection. You can make more than one selection

If you have selected one or more source fields as the selection parameters when defining the source fields, these fields are also available as selection parameters.

Two checkboxes are provided for you:

- **Amount field:** Amount fields are converted into the internal ABAP format (with a decimal point.)
- **Date values:** Date fields are converted into the internal ABAP format (YYYYMMDD.)

If you are using a wildcard in the file name of the input files and you have defined at least one value for the wildcard, you will also get a selection parameter for the wildcard. If you do not specify anything, all the defined wildcard values are processed.

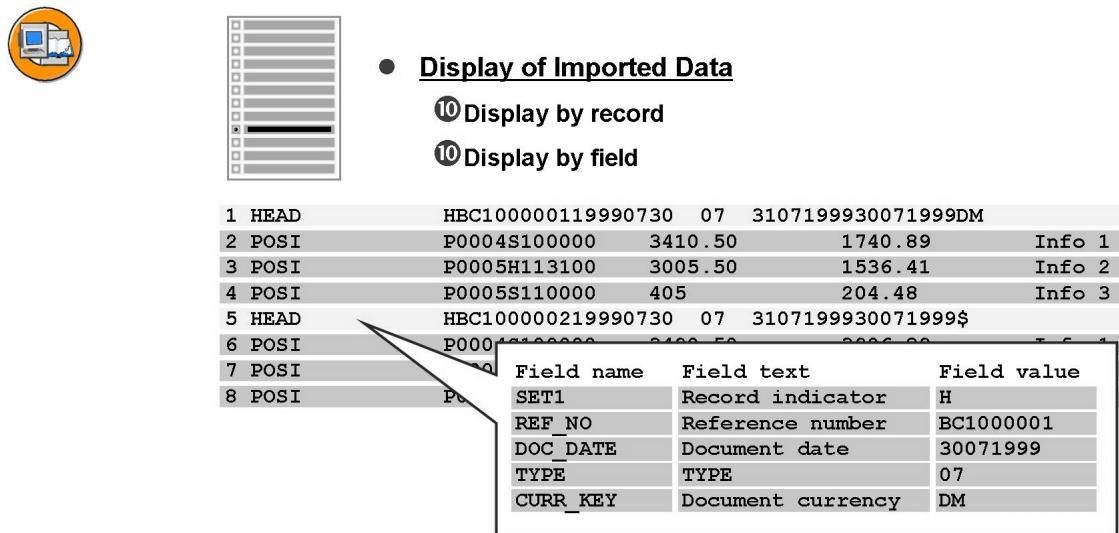


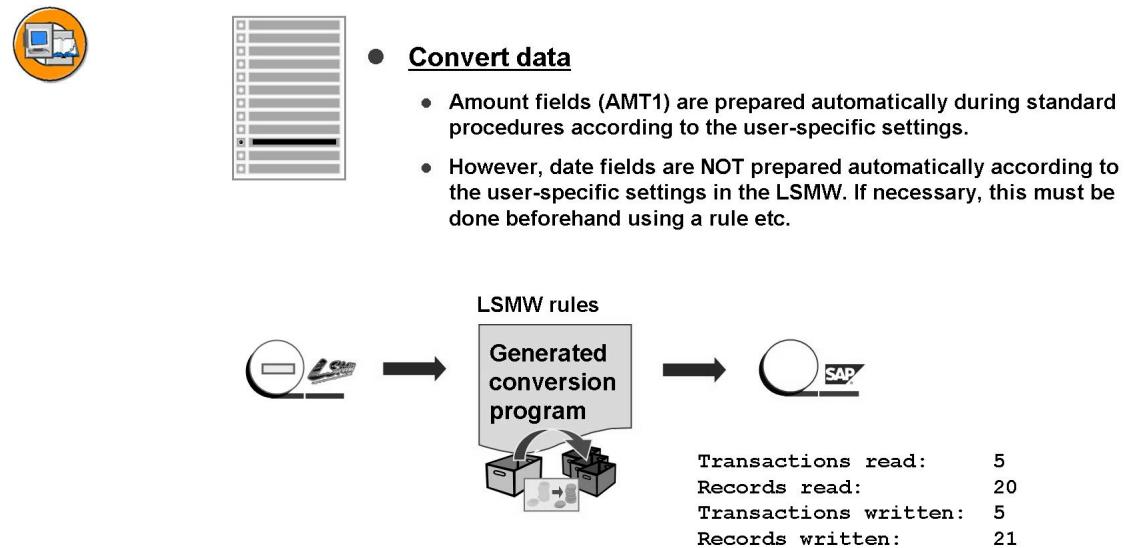
Figure 173: Displaying Imported Data

In this step you can display a part of or all of the imported data in a table. By selecting a table row (or selecting Field contents), you can display all the information in this row. You can also do this by choosing *Field Contents*.

Carry out this step as a control before you continue with the conversion. If source fields are defined incorrectly, for example, or if structure assignments are made incorrectly, it does not take long for these errors to become visible.

You can use the *Change Display* function to choose between a single line or multiline view.

The colors of the individual hierarchy levels can be displayed using *Display color legend*.

**Figure 174: Convert Data**

This work step is similar to the work step **Read data**.

The result of the rules and therefore the main focus of all functions in the LSMW is delivered here. The imported data is converted into the target file using the defined rules.

If you do not select any data, confirm by executing the process. Otherwise select the data in “transaction number” “from...to...”. You can also select several transaction numbers.

If you have selected one or more source fields as the selection parameters when defining the source fields, these fields are also available as selection parameters.

If you are using a wildcard in the file name of the input files and you have defined at least one value for the wildcard, you will also get a selection parameter for the wildcard. If you do not specify anything, all the defined wildcard values are processed.



Hint: The system first checks whether the data conversion program is still up-to-date. If it is not, it is automatically regenerated. If errors occur, for example, due to own program rows in rules, the system automatically goes to the generated conversion program. You can use the function *check/syntax check*, to go directly to where the error occurred. Always correct the problem in the rules and never in the generated program.




- **Display of Converted Data**

⑩ Display by record
⑪ Display by field

1 BGR00	OKNA1_A	100KLINGELS	/
2 BBKPF	1FB01	30071999SA0001/	/ DEM
3 BBSEG	2BBSEG	40/	//
4 BBSEG	2BBSEG	50/	//
5 BBSEG	2BBSEG	50/	//
6 BBKPF	1FB01	30071999SA0001/	/ USD
7 BBSEG	2BBSEG	10/	//
8 BBSEG	2BBSEG	10/	//
9 BBSEG	2BBSEG	10/	//

Fld name	Field text	Field value
STYPE	Record type	1
TCODE	Transaction code	FB01
BLDAT	Date	07071999
BLART	Document type	SA
BUKRS	Company code	0001
BUDAT	Date	/
WAERS	Currency key	DEM

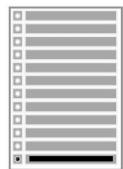
Figure 175: Display Converted Data

After the conversion, we recommend that you carry out this step to discover any errors and correct them before you import the data.

In this step you can display a part of or all of the imported data in a table. By selecting a table row (or selecting Field contents), you can display all the information in this row. You can also do this by choosing *Field Contents*.

You can use the *Change Display* function to choose between a single line or multiline view.

The colors of the individual hierarchy levels can be displayed using *Display color legend*.



- **Start the transfer (import/load)**

⑩ Start the transfer procedure according to the technique used
(BI, CT, DI, IDoc, BAPI)

⑩ Important: For BI, after this step, the data is available in BI sessions and must then be imported.

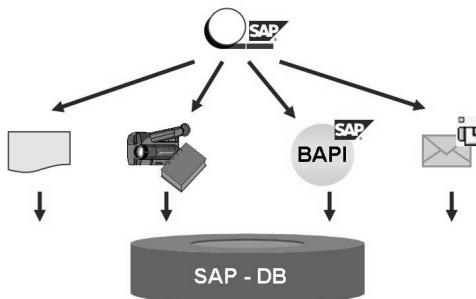


Figure 176: Start the Transfer

The steps depend on the procedure and data transfer method used.

That is, it depends on the selected object type.

- Standard batch input or recording:
 - Create batch input session
 - Process batch input session

Standard direct input:

- Start direct input program

BAPI or IDoc:

- Start IDoc Creation
- Start IDoc processing
- Create IDoc overview
- Start IDoc postprocessing

Note that this last step of the data import can also be carried out from the DX-WB. The tool that is recommended depends on your experience and the technique. We will discuss the differences and recommendations later on in the course.

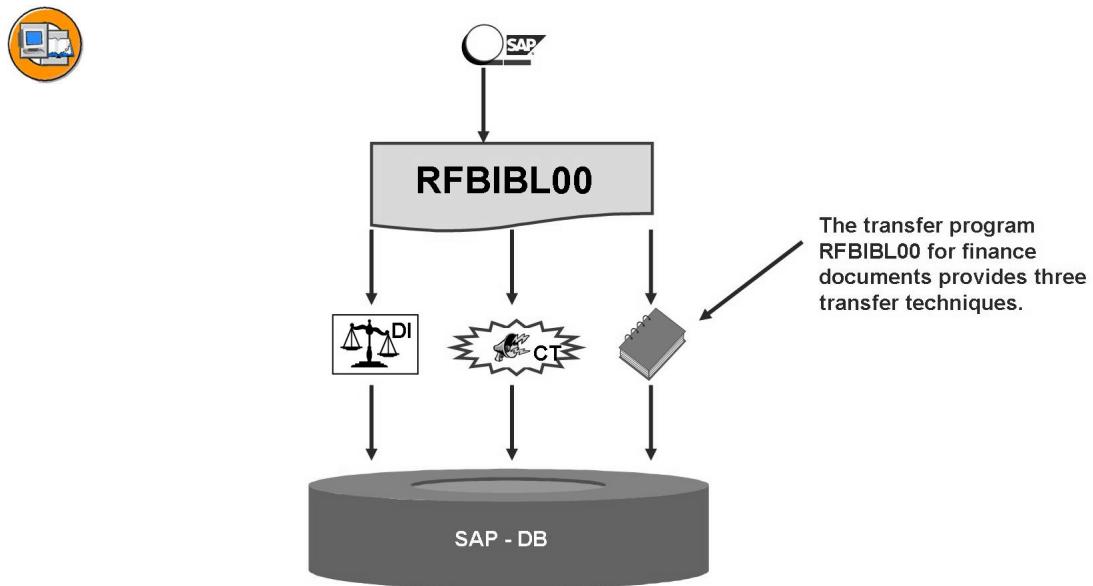


Figure 177: Start the Standard Transfer Program

- If you have selected a standard SAP transfer program, it may support batch input, call transaction, or direct input or some combination of them.
- If you use batch input:

You must create the session and then process the session using the batch input monitor.

You can go to the batch input monitor. However, only the batch input sessions for the selected object will be displayed.

For batch input, note that after a session has been created, the data is only stored on the database in a temporary format. The data is not posted from the session to the application tables until the sessions are processed.

- If you use call transaction:

You can start the program directly and import the data immediately. If an error occurs, batch input sessions are created.

- If you use direct input:

You can start the direct input program directly (this should only be used for testing purposes), or start the data transfer using the direct input monitor. Direct input procedures differ considerably from each other in their operation and usage. Study the documentation for this topic carefully.

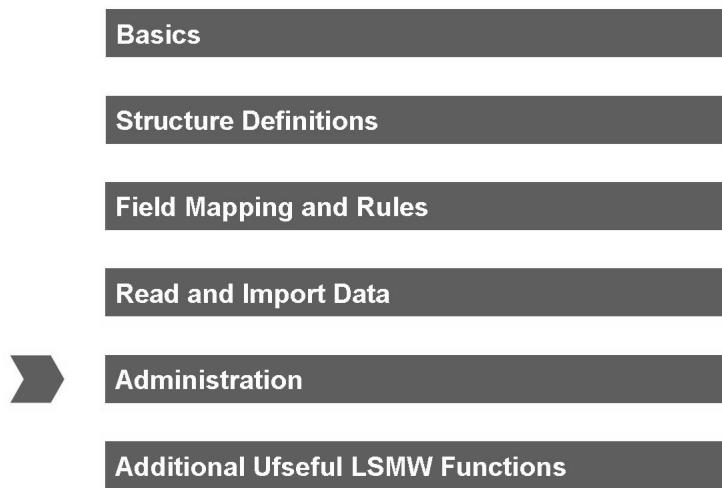


Figure 178: LSMW Administration



- **Administration**

- Rename, copy, etc. projects, subprojects, and objects
- Write documentation for projects, subprojects, and objects

- **Administration**

- Create change request to transport LSMW project
- Export/import subcomponents of an LSMW project
- Display LSMW version

Figure 179: Administrative Functions

From the initial screen select *Goto* → *Administration* to go to the administration function. Here you will find an overview of all the existing projects.

You can create, edit, display, delete, copy or rename projects, subprojects, objects and reusable rules.

By double clicking on an entry you can go to the entry display.

If you position the cursor on an entry, you can create a personal remark in the documentation. With each processing the name and date of the last change is saved.

By choosing *Extras* → *Generate Change Request*, you can create a change request (see SE09) and transport the entire LSMW project into another SAP system.

By choosing *Extras → Export/Import*, you can transport the entire LSMW project as well as just parts of it in a file into another system.



- **Authorization Objects: B_LSMW and B_LSMW_PRO**

- Define who is allowed to display, create or edit the LSMW projects
- Define who is allowed to edit which LSMW project.

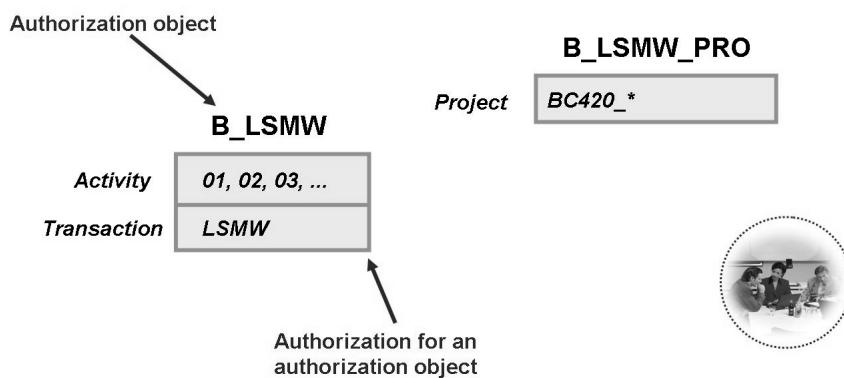


Figure 180: Authorizations

As of LSMW Version 1.7 there is a new authorization object, **B_LSMW_PRO**, in addition to **B_LSMW**. **B_LSMW_PRO**.

With this object you can assign authorizations at project level.

Therefore, you can control who can display or edit which project.

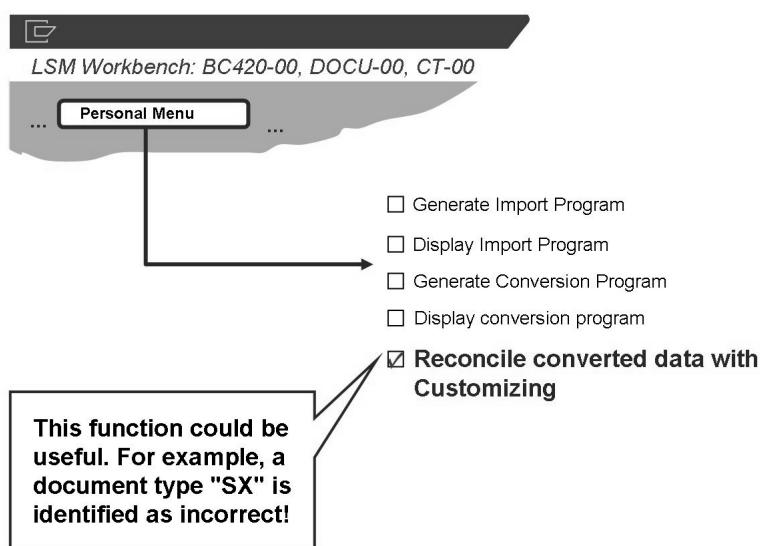
With authorization object **B_LSMW** you can assign basic authorizations such as, display, create, and edit.

There is also a standard SAP authorization profile for LSMW: **B_LSMW_ALL**, **B_LSMW_CHG**, **B_LSMW_EXEC**, **B_LSMW_SHOW**.

Specific authorizations are usually assigned using the profile generator (this is not part of this training course).

**Basics****Structure Definitions****Field Mapping and Rules****Read and Import Data****Administration****Additional Useful LSMW Functions****Figure 181: LSMW - Additional Functions and Tips**

The following slides contain additional useful functions of LSMW.

**Figure 182: Additional Tips and Tricks for LSMW - 1**

We recommend that you execute the function for *checking the converted data against Customizing* with some representative data from the external system because otherwise the execution will take a very long time. This function can determine incorrect rules or rules that have not been defined completely.

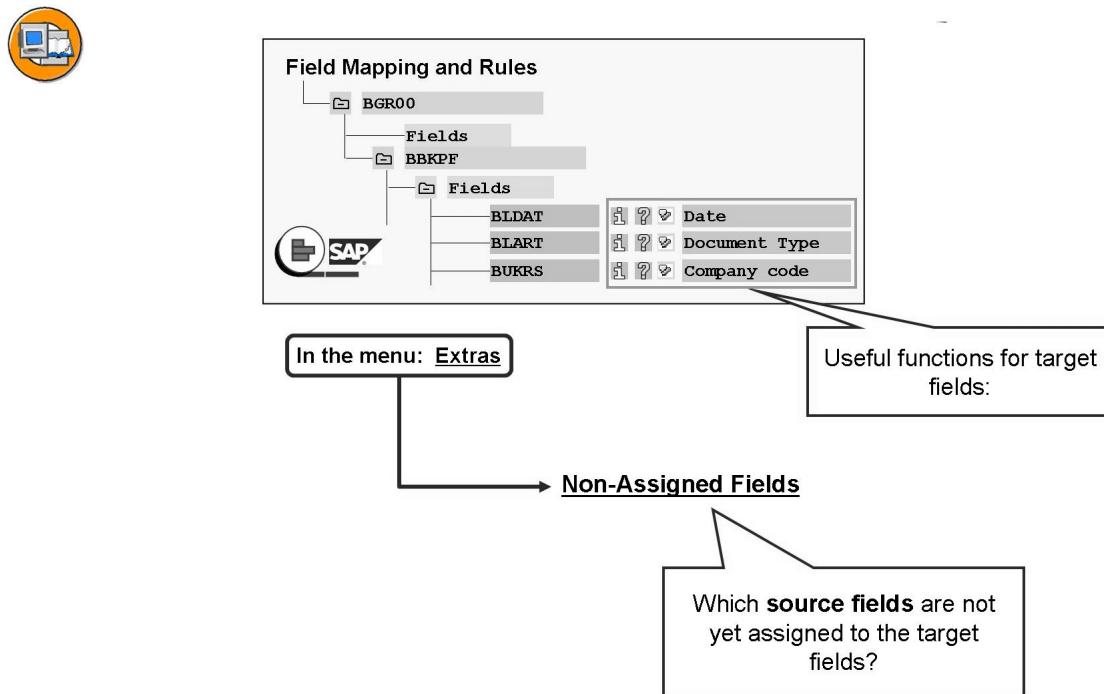


Figure 183: Additional Tips and Tricks for LSMW - 2

If you have a large number of source fields that have to be assigned, some source fields may get overlooked. You can use the function for *fields that have not yet been assigned* to locate these source fields.

→ **Note:** Remember the *Object List* function. At the end of your project work with the LSMW, we recommend that you choose the object list display in the **List** display in the LSMW. Then you will receive comprehensive project documentation with all chosen definitions: Source structures, source fields, mapping and rules, files and so on.

This function is particularly suited to **subsequent project documentation** and logs all relevant definitions.

Exercise 12: Converting and transferring documents using the Legacy System Migration Workbench

Exercise Objectives

After completing this exercise, you will be able to:

- Create a mapping plan for FI documents.
- Define the conversion rules in the LSMW for the documents.
- Map the document data using the LSMW.
- Transfer documents with the call transaction technique

Business Example

You want to import external FI documents into the SAP system using the LSMW with the call transaction technique.



Hint: LSMW project: BC420-##

Subproject: DOCU-##

Object: CT-##

File with documents in external format:
BC420_DOC_1_HEAD_POS.LEG

Task 1:

The FI documents are available in file BC420_DOC_1_HEAD_POS.LEG on the application server. This file contains five documents, each with three items. You want to use the LSMW to transfer this data to the relevant SAP record layout structure. You then want to use program RFBIBL00 with the call transaction technique to transfer this data.

1. As a test, use transaction FB01 to create an FI document online.

Document header

Document date:	Today
Document type:	Sat
Company code:	0001
Currency:	EUR

Continued on next page

First line item

Posting key:	50
Account:	100000
Amount:	200

Next line item

Posting key:	40
Account:	113100
Amount:	200

Then save the document.

Task 2: LSMW Activities:

Start the LSMW (transaction LSMW) to define the transfer object and display the target structure.

1. In the LSMW, choose project BC420-## with the name “Finance data”. Create the subproject DOCU-## with the name “Documents”. Create the object CT-## with the name “Documents with CT”.
2. Execute the process step **Maintain Object Attributes**. Under *Object Type and Import Method*, choose Standard Batch/Direct Input. Use the F4 help to choose object 100 (financial documents).

Position the cursor in the *Method* field and choose the F4 help to choose the method and program. Save your settings and return to the project overview (list of process steps).



Hint: Change the maintenance mode from double-click=display to double-click=change.

3. Display the object overview for your object (pushbutton *Object Overview* in the application toolbar) in table form. Save the list as a local file in spreadsheet format. (*System → List → Save → Local File*). Choose spreadsheet format. Enter C:\temp\FI_DOC-##.xls. as the file name.

Continued on next page

4. Start Microsoft Excel and open file C:\temp\FI_DOC##.xls. You can now use this Excel file to print out the target structures.



Hint: Do not generate a printout as this is already available in the attachment.

Task 3:

Create a **mapping plan** to transfer the external documents. An extract from the Excel table is available on paper.

1. The data from the external system is displayed in the following table:

Field Name	Field Length	Typical Field Value	Description
HEAD -REF	9	123456789	Old document number
HEAD-DATE	10	20.01.2006	Document Date
HEAD-TYPE	4	07	Document type
HEAD -CU	2	\$ or EU	Document currency
POSI-KEY	4	0004 or 0005	Posting key
POSI-ACCOUNT	10	100000	Document account
POSI-AMOUNT	15	1000,10	Amount in document currency
POSI-TEXT	6	info	Text for document item

2. Which field that you need to create a document does not exist in the table?
 3. Use the value of the field from point 1. Value: _____
 4. Now use the attached table to create a mapping plan. For the source fields, maintain the following: Field name, length, conversion method and coding/comment.



Hint: The document type (07) is G/L account. The posting key (0004) is the debit posting and (0005) is the credit posting.

Continued on next page

5. Check the mapping plan by comparing it with the solution (Excel list).

Task 4:

1. File BC420_DOC_1_HEAD_POS.LEG with the document data from the external system is on the application server in the DIR_HOME directory. Start the data monitor (transaction AL11) in a new session and display the data. How many FI documents exist in this file? _____
2. Go to the LSMW to create the source structures of the source file.

Execute the process step **Maintain Source Structures**. The source file consists of two structures: The document header structure and the document item structure. Create structure HEAD and a level under the structure POSI.

Task 5:

Execute the process step **Maintain Source Fields**.

Create the source fields for the relevant structures: (The number in brackets is the length of the field).

1. The source fields for the **document header** are:
SET1(1) Record indicator for header: **H (H for header)**,
REF(9) Reference number in the external system,
DATE(10) Document date **Important: Field type DDMY**,
TYPE(4) Document type,
CU(2) Document currency.
2. The source fields for the **document item** are:
SET2(1) Record indicator for items: **P (P for item)**
KEY(004) Posting key,
ACCOUNT(10) Account number,
AMOUNT(15) Amount in document currency **Important: Field type AMT1**,
TEXT(6) Description of the document item.

Task 6:

1. Execute the process step **Maintain Structure Relations**.

Continued on next page

Structure relations: Assign BGR00 and BBKPF to the document header and BBSEG to the document item.

Task 7:

Execute the process step **Maintain Field Mapping and Conversion Rules**.

1. Execute mapping according to your mapping plan. You must assign the following fields:

Document date: Move. Maintain the date according to your user-specific settings.



Hint: This is possible as of LSMW Version 1.7. In versions lower than 1.7, the date is always converted in internal ABAP format.

Document type: Conversion

Company code: Constant (0001)

Currency key: Conversion (for example, \$ -> USD)

Reference document number: Move

Posting key: Conversion (for example, 0004 -> 40)

Amount in document currency: Move

Item text: Move

Account: Move

Task 8:

1. Execute the process step **Specify Files**.

The source file is BC420_DOC_1_HEAD_POS.LEG and you can find it on the application server. Maintain the file name under *Legacy Data* [on the R/3 server (application server)]

Under 'File Contents', choose *Data for Multiple Source Structures (Seq. File)*.

Choose code page 1100.

2. Execute the process step **Assign Files**. Assign the files to:

Document - file: BC420_DOC_1_HEAD_POS.LEG

Items - file: BC420_DOC_1_HEAD_POS.LEG

3. Execute the process step **Read Data**.

Continued on next page

4. Execute the process step **Display Read Data**. Display the read data as a structure and as a field display. Check the data.
5. Execute the process step **Convert Data**.
6. Execute the process step **Display Converted Data**. Display the converted data as a structure and as a field display. Check the data.
7. Execute the process step **Start Direct Input Program**. Choose program RFBIBL00 here.
Use the call transaction technique.
First check the file. If there is no termination situation, the transfer can go ahead.
Deactivate the *Check File Only* function. Start the data transfer. The data is transferred with call transaction.
Make a note of the first five document numbers.
8. You can use transaction FB03 to view the resulting documents.

Task 9: Optional Task

1. There is the new file **BC420_DOC_1A_HEAD_POS.LEG** with documents in external format. However, this file contains an error in the data records.
Execute points 9 to 13 with this file again. If you transfer data, the system issues an error message. This incorrect data record is placed in a batch input session. Process this session and correct the error.

Solution 12: Converting and transferring documents using the Legacy System Migration Workbench

Task 1:

The FI documents are available in file BC420_DOC_1_HEAD_POS.LEG on the application server. This file contains five documents, each with three items. You want to use the LSMW to transfer this data to the relevant SAP record layout structure. You then want to use program RFBIBL00 with the call transaction technique to transfer this data.

1. As a test, use transaction FB01 to create an FI document online.

Document header

Document date:	Today
Document type:	Sat
Company code:	0001
Currency:	EUR

First line item

Posting key:	50
Account:	100000
Amount:	200

Next line item

Posting key:	40
Account:	113100
Amount:	200

Then save the document.

- a) Call transaction FB01 in another session and maintain the relevant fields as described in the table above. The G/L account posting must have a cleared balance otherwise the document cannot be posted.

Continued on next page

Task 2: LSMW Activities:

Start the LSMW (transaction LSMW) to define the transfer object and display the target structure.

1. In the LSMW, choose project BC420-## with the name “Finance data”. Create the subproject DOCU-## with the name “Documents”. Create the object CT-## with the name “Documents with CT”.
 - a) Proceed as usual and use the Create pushbutton in the menu bar.



Caution: Even though fields *Project*, *Subproject* and *Object* are ready for input, you must use the Create pushbutton in the menu bar so that the creation works correctly.

2. Execute the process step **Maintain Object Attributes**. Under *Object Type and Import Method*, choose Standard Batch/Direct Input. Use the F4 help to choose object 100 (financial documents).
Position the cursor in the *Method* field and choose the F4 help to choose the method and program. Save your settings and return to the project overview (list of process steps).



Hint: Change the maintenance mode from double-click=display to double-click=change.

- a) If you select a standard procedure, the LSMW will usually not display batch input/call transaction/direct input under the object settings, rather it will display any text registered for the procedure. For RFBIBL00, “DI” appears under program type. Do not get annoyed. You can still use batch input and call transaction.
3. Display the object overview for your object (pushbutton *Object Overview* in the application toolbar) in table form. Save the list as a local file in spreadsheet format. (*System → List → Save → Local File*). Choose spreadsheet format. Enter C:\temp\FI_DOC-##.xls. as the file name.
 - a)
4. Start Microsoft Excel and open file C:\temp\FI_DOC-##.xls. You can now use this Excel file to print out the target structures.



Hint: Do not generate a printout as this is already available in the attachment.

- a) If you cannot start Excel, consult your instructor.

Continued on next page

Task 3:

Create a **mapping plan** to transfer the external documents. An extract from the Excel table is available on paper.

- The data from the external system is displayed in the following table:

Field Name	Field Length	Typical Field Value	Description
HEAD -REF	9	123456789	Old document number
HEAD-DATE	10	20.01.2006	Document Date
HEAD-TYPE	4	07	Document type
HEAD -CU	2	\$ or EU	Document currency
POSI-KEY	4	0004 or 0005	Posting key
POSI-ACCOUNT	10	100000	Document account
POSI-AMOUNT	15	1000,10	Amount in document currency
POSI-TEXT	6	info	Text for document item

- a) Try to assign as many fields as possible. Use the online transaction and the document for RFBIBL00 to help you. You can also use the DX-WB conversion assistant.
2. Which field that you need to create a document does not exist in the table?
 - The company code.
3. Use the value of the field from point 1. Value: _____
 - 0001

Continued on next page

4. Now use the attached table to create a mapping plan. For the source fields, maintain the following: Field name, length, conversion method and coding/comment.



Hint: The document type (07) is G/L account. The posting key (0004) is the debit posting and (0005) is the credit posting.

- a) Example of a rule: 07 becomes SA, the conversion rule is therefore M+C. Posting key 0004 becomes 40, the rule is therefore M+C, and so on.
5. Check the mapping plan by comparing it with the solution (Excel list).
 - a) You can find the mapping plan in the attachment. You can also use the instructor solution as a guide. As described previously, it is when errors occur that is interesting. Therefore, in mapping, you actually want errors to occur.

Task 4:

1. File BC420_DOC_1_HEAD_POS.LEG with the document data from the external system is on the application server in the DIR_HOME directory. Start the data monitor (transaction AL11) in a new session and display the data. How many FI documents exist in this file? _____
 - a) Proceed as usual to use the data monitor (transaction AL11)
2. Go to the LSMW to create the source structures of the source file. Execute the process step **Maintain Source Structures**. The source file consists of two structures: The document header structure and the document item structure. Create structure HEAD and a level under the structure POSI.
 - a) Define header structure HEAD and under this (place the cursor on HEAD), create POSI as a structure for the document items.

Task 5:

Execute the process step **Maintain Source Fields**.

Create the source fields for the relevant structures: (The number in brackets is the length of the field).

1. The source fields for the **document header** are:

SET1(1) Record indicator for header: **H (H for header)**,
REF(9) Reference number in the external system,

Continued on next page

- DATE(10) Document date **Important: Field type DDMY**,
TYPE(4) Document type,
CU(2) Document currency.
- a) Proceed as described on the course slides. You can enter the fields individually using the CREATE pushbutton or using table maintenance (see the icon in the menu bar).
 2. The source fields for the **document item** are:
SET2(1) Record indicator for items: **P (P for item)**
KEY(004) Posting key,
ACCOUNT(10) Account number,
AMOUNT(15) Amount in document currency **Important: Field type AMT1**,
TEXT(6) Description of the document item.
a) Proceed as described on the course slides. You can enter the fields individually using the CREATE pushbutton or using table maintenance (see the icon in the menu bar).

Task 6:

1. Execute the process step **Maintain Structure Relations**.
Structure relations: Assign BGR00 and BBKPF to the document header and BBSEG to the document item.
a) You can create the relations using the CREATE pushbutton in the menu bar. BGR00 and BBKPF are assigned to the document header and BBSEG is assigned to the document items.

Task 7:

Execute the process step **Maintain Field Mapping and Conversion Rules**.

1. Execute mapping according to your mapping plan. You must assign the following fields:
Document date: Move. Maintain the date according to your user-specific settings.



Hint: This is possible as of LSMW Version 1.7. In versions lower than 1.7, the date is always converted in internal ABAP format.

Continued on next page

Document type: Conversion
Company code: Constant (0001)
Currency key: Conversion (for example, \$ -> USD)
Reference document number: Move
Posting key: Conversion (for example, 0004 -> 40)
Amount in document currency: Move
Item text: Move
Account: Move

a) Use your own mapping plan or the solution as a guide. If errors occur during mapping, this is actually good because you want to practice dealing with errors.

Task 8:

1. Execute the process step **Specify Files**.

The source file is BC420_DOC_1_HEAD_POS.LEG and you can find it on the application server. Maintain the file name under *Legacy Data* [on the R/3 server (application server)]

Under 'File Contents', choose *Data for Multiple Source Structures (Seq. File)*.

Choose code page 1100.

- a) Note that you can find all the files on the application server.
Double-click the correct file.

2. Execute the process step **Assign Files**. Assign the files to:

Document - file: BC420_DOC_1_HEAD_POS.LEG

Items - file: BC420_DOC_1_HEAD_POS.LEG

- a) Both the header structures and item structures originate from the same file. Therefore, in this process step, you cannot cause any errors in this example.

3. Execute the process step **Read Data**.

- a) Double-click this process step.

4. Execute the process step **Display Read Data**. Display the read data as a structure and as a field display. Check the data.

- a) Double-click this process step.

Continued on next page

5. Execute the process step **Convert Data**.
 - a) Double-click this process step.
6. Execute the process step **Display Converted Data**. Display the converted data as a structure and as a field display. Check the data.
 - a) Pay attention to particularly important fields such as the document type, the data and the amounts. If target fields are not populated, you may have forgotten the checkbox during mapping (this is a common error).
7. Execute the process step **Start Direct Input Program**. Choose program RFBIBL00 here.

Use the call transaction technique.

First check the file. If there is no termination situation, the transfer can go ahead.

Deactivate the *Check File Only* function. Start the data transfer. The data is transferred with call transaction.

Make a note of the first five document numbers.

 - a) It sounds confusing: If you select the last step **Start direct input program**, you cannot use batch input or call transaction as the technique. Note that RFBIBL00 is compatible with all three techniques. In practice, we recommend that you use direct input in this case.
8. You can use transaction FB03 to view the resulting documents.
 - a) Call transaction FB03 in another session.

Task 9: Optional Task

1. There is the new file **BC420_DOC_1A_HEAD_POS.LEG** with documents in external format. However, this file contains an error in the data records.

Execute points 9 to 13 with this file again. If you transfer data, the system issues an error message. This incorrect data record is placed in a batch input session. Process this session and correct the error.

 - a) Tip: Check the amounts for the document to be imported. Can you find the error?



Lesson Summary

You should now be able to:

- Use the basic functions of the LSMW
- Define rules in the LSMW
- Convert data in the LSMW



Unit Summary

You should now be able to:

- Use the basic functions of the LSMW
- Define rules in the LSMW
- Convert data in the LSMW

Unit 7

Direct Input

Unit Overview

Direct input is a transfer technique that you can use to transfer data to the SAP system quickly and without using transactions. The input format used here is the SAP record layout.



Unit Objectives

After completing this unit, you will be able to:

- Transfer FI documents into the SAP system using the direct input method
- Scheduling jobs
- Work with the direct input monitor

Unit Contents

Lesson: Direct Input	226
Exercise 13: Transferring FI Documents using Direct Input	235

Lesson: Direct Input

Lesson Overview



- Direct Input Concept
- Direct Input Monitor



Lesson Objectives

After completing this lesson, you will be able to:

- Transfer FI documents into the SAP system using the direct input method
- Scheduling jobs
- Work with the direct input monitor

Business Example

A company wants to use the technique that is available most quickly for a procedure (here finance documents) that supports direct input. For performance reasons, procedures are developed in certain places in the direct input applications.

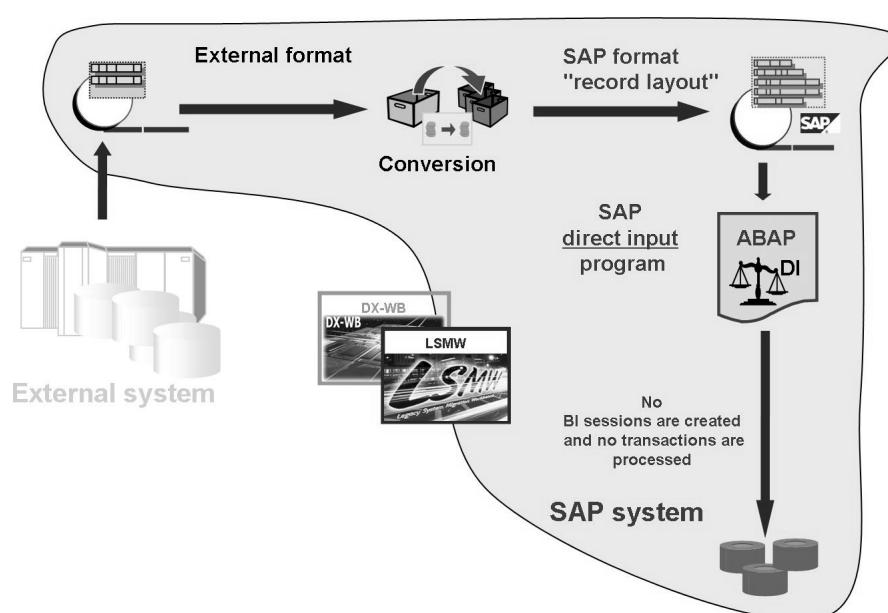


Figure 184: Direct Input Concept

An alternative to the batch input method is the direct input method. This method is more efficient than the other methods making it especially useful for transferring large datasets. Unlike batch input, no sessions are created as the data is updated

directly. No screens are involved in this procedure. **No screens are involved** in this procedure. The data is entered directly into database tables by calling function modules that carry out the necessary checks.

In case errors occur during the data transfer, direct input has a **restart mechanism**. To ensure this, the programs can only be run as background jobs. Program RBMVSHOW or transaction BMV0 (the direct input monitor) are provided to manage and start direct input programs.

If you are working with test data, you can start direct input in foreground. Make sure that neither error logs are created nor that the possibility of a restart exists in error situations.

SAP strongly recommends that you use transaction BMV0 for the actual data transfer for some selected procedures. This transaction ensures the data transfer can be restarted.

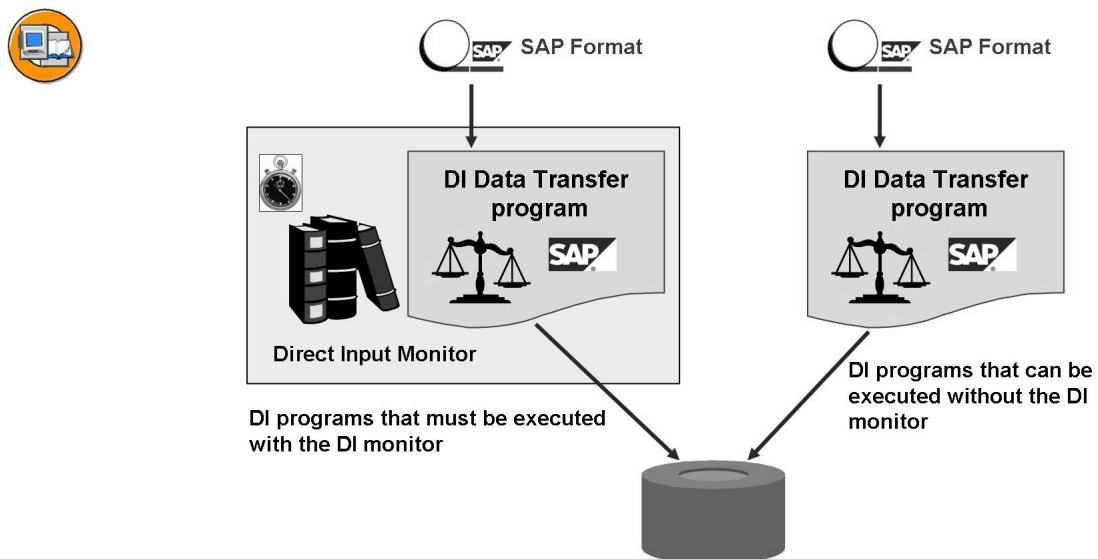


Figure 185: Direct Input Programs

Direct input programs that must be executed with the direct input monitor (transaction BMV0):

- The advantage is that you can restart the data transfer program, if logical errors (for example, missing material) or terminations occur. This restart capability guarantees that no duplicate entries will be added to the database if the program is restarted at the point it terminated. A processing log is available to help you identify, correct, and repost any errors.
- Using direct input in the background has the advantage of better performance.
- The following direct input programs are included in the direct input monitor:
 - FI documents (RFBIBL00)
 - Material master (RMDATIND)
 - Environment data (SAPLC131..132...133)
 - Sales documents (RVINVB10)

Direct input programs that can be executed without the direct input monitor:

- These programs partly support the restart capability. This depends on the application and is not always necessary.

For all direct input procedures: Study the documentation about the details of the individual procedures thoroughly and test them in your test system before you transfer them to the production system.

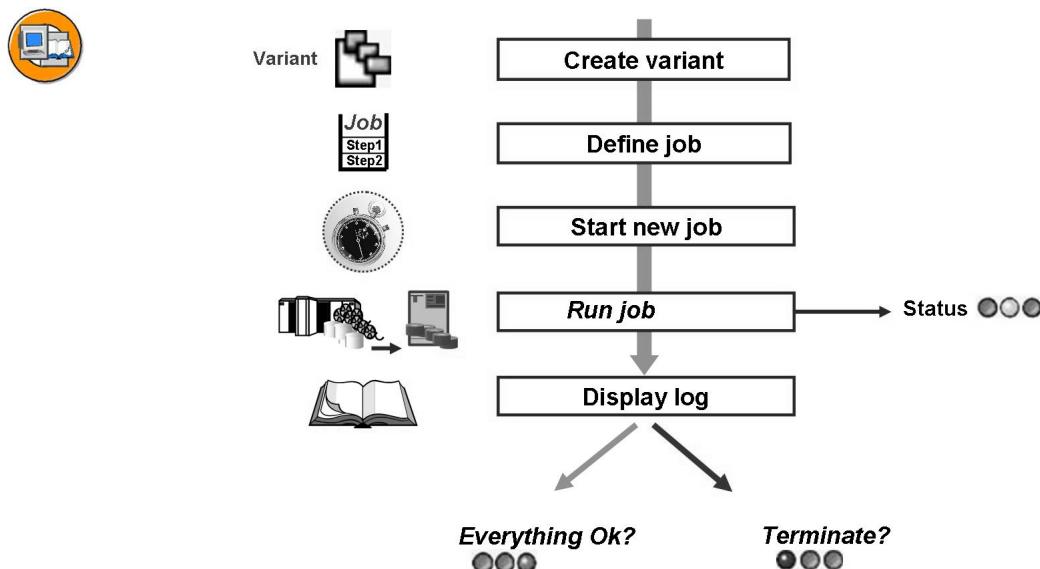


Figure 186: Tasks in the Direct Input Monitor

Direct input run:

- First create a variant for the transfer program.
- Define a job using this variant.
- Schedule the job (start: immediately, date, time, after job, after event and so on).
- Analyze the job log after the job run.
- You can restart the job run if it is terminated.

Requirements for the restart mechanism:

- The input file must not contain any formal errors. You can choose *Check file* to establish this. If the job is terminated due to a formal error, you cannot restart the method. You then determine which data records were updated before the termination, and reduce the file accordingly.

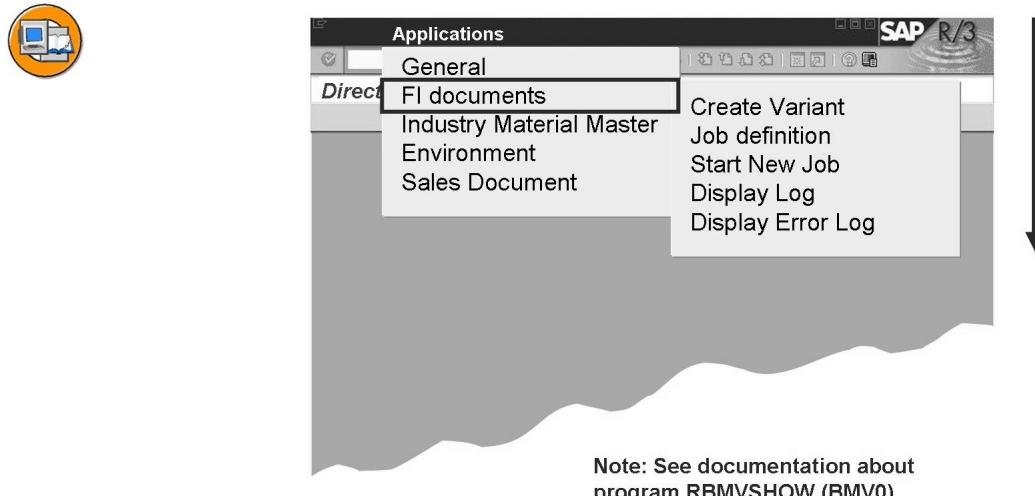


Figure 187: Example: FI Documents

You have to create a variant for each direct input program. The standard rules for creating variants apply.

In the menu path *Job administration → Define new job*, you define control data for a job. A job is uniquely identified for each client by its name. You have to specify the report name (for example, RMDATIND) and a variant name. If you want, you can also specify a server for the background job. This can be overwritten later. You can also specify a user name under which the direct job runs. This user then needs all authorizations required for the application.



Use the function *Start new job* to start the job you defined earlier (by choosing F4 in the first input field). You can change the default background server or user name. On the next screens, enter the print parameters and start time.



Note: As of SAP NetWeaver AS 6.10:

For the sales and distribution documents, there is a BAPI for transferring data in addition to the existing direct input program (RVINVB10). CustomerOrder.CreateFromDat2 (business object BUS1022 “SalesOrder”).

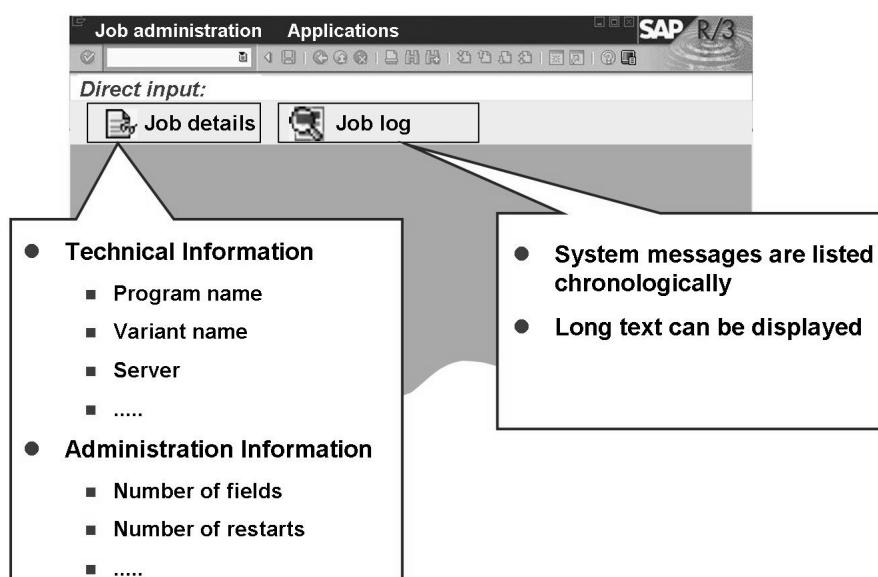


Figure 188: Job Analysis

You can display details of the selected job in the detail display.

When you “update” and select the job, the current status of the transfer is always displayed. You can follow what happens. If the job is terminated, due to a database error, the job is assigned status “Batch: job terminated.” In this case you can analyze the error (function *Display log*) and remove the error. Then choose *Job administration → Reset job*. The job is restarted and the data transfer is continued from the point at which it terminated.

To assure data integrity, each direct input program has a restart mechanism. Using some function modules the program writes synchronization information to table TBIST. If there is a termination, then it is guaranteed that the data transfer can be **reset to the correct place**. This means, for example, that no document is posted twice or not posted at all.

System table TBIST contains entries:

- Number of data records with errors
- Last number processed when program terminated

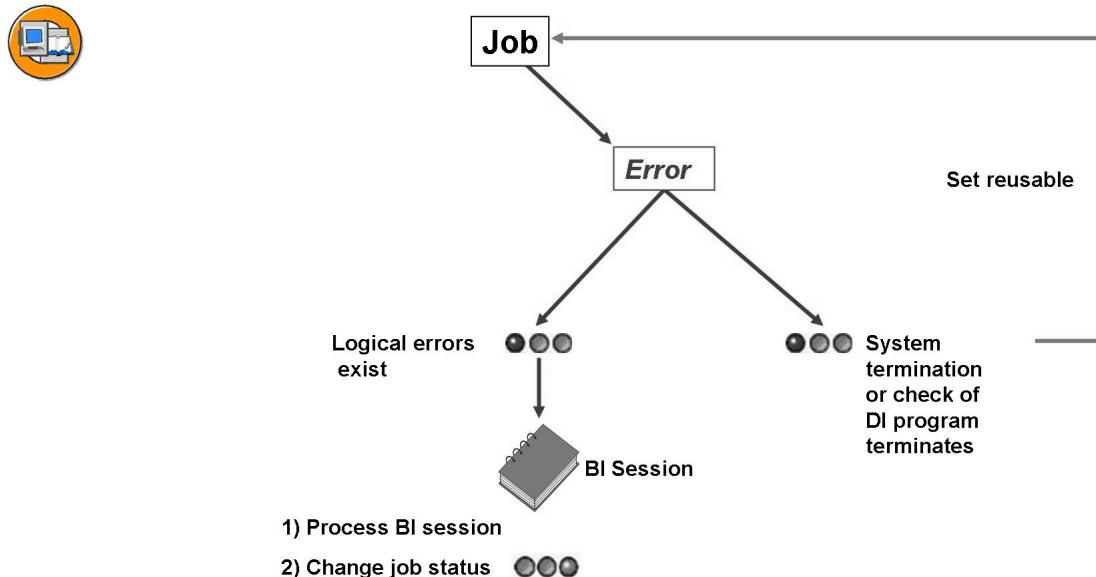


Figure 189: Error in FI Documents

Each batch input program carries out checks before the data is transferred. If an error occurs (for example, in RFBIBL00: data record has no posting key), this terminates the job without posting a single data record. The job is assigned the status “Background job terminated”.

If the file has only logical errors, that is, errors occur when the data is checked in the system (for example, the currency amount is missing from the data record in RFBIBL00), this does not terminate the job.

The job is completed and is assigned status “Completed”: job executed. However, it contains logical errors. Records containing errors are placed in a batch input session.

The session name of the first BGR00 record is used. After you have corrected the error, you can process the batch input session and complete the data transfer. Set the status of the job with the function for *subsequent processing* to *Set to done*.

For more information, see the application help documentation for the FI documents under the DX-WB documentation.

Note: The program RFBIBL00 can process only 20 data records without the BMV0 for the direct input or call transaction technique. It should be executed within transaction BMV0 because of the restart capability.

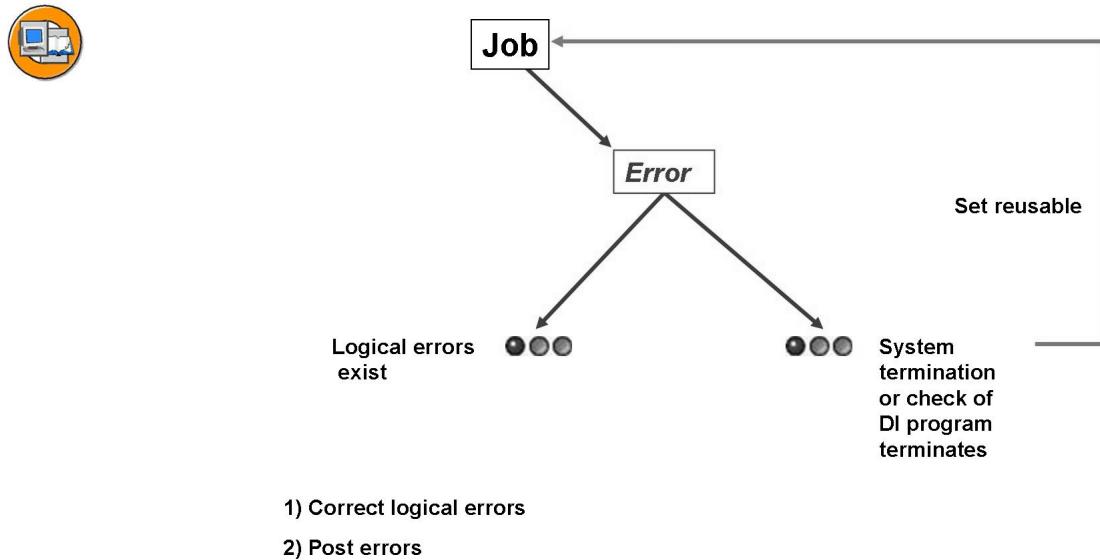


Figure 190: Errors in Material Master or SD Document

An error file is created for material master records. You can use RMDATIND (or *Applications → Industry mat master → Display logical errors*) to display and process the errors. After you have processed this file (or corrected a Customizing setting), you can trigger the subsequent processing of the error by choosing *Subsequent Processing → Correct errors*. Then the data is not read from the original file, rather from the file being processed and the system tries to subsequently post the data after the relevant check.

For Sales and Distribution Documents: If logical errors occur during your job, you can use a detailed job log to identify these errors and to eliminate them and then carry out a subsequent processing in the SAP system.

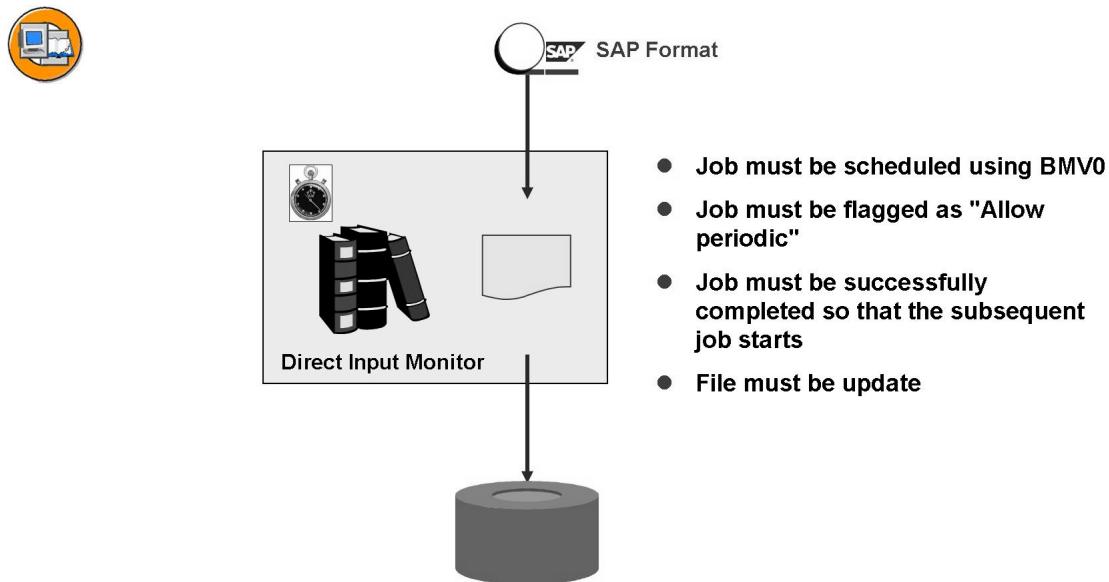


Figure 191: Periodic Jobs

If you want to use the direct input method for a periodic job, you must adhere to the following:

- In the job definition, select “Periodic allowed” for the job.
- You must schedule the job using the direct input monitor.

If you want to add additional steps to the background job, save the start data for the job by choosing *Start data transfer using direct input method*. Choose a start time in the future. After saving, start transaction SM37 and edit the job step list. The job name corresponds to the job name from the job definition.

Make sure you check that the job has been completed correctly each time it runs. If a job terminates, the subsequent job will not start! You can use the *History* function to check which jobs were started periodically for a job definition. If a job does not start because the previous job was terminated, this is flagged in the history list by “Invalid”.

If you want to restart a job that terminated, first check to make sure the specified input file is still correct. The input file is usually switched each time the periodic job runs. If you use the *Restart* function for a job that terminated, and the input file has been switched in the meantime, data integrity will probably be affected.

Exercise 13: Transferring FI Documents using Direct Input

Exercise Objectives

After completing this exercise, you will be able to:

- Use the direct input monitor to transfer the data.

Business Example

Use the direct input technique to transfer financial accounting documents to the SAP R/3 system.

Job: Job-##

Task:

Optional: This is because only one user is allowed to use the direct input monitor at a time.

Do not transfer the documents using call transaction, rather using the direct input procedure. You can use the same record layout structure for this. As a result, you can reuse project BC420-## with subproject BELE-## or DOCU-## (depending on what you have called it).

In the LSMW, start the *Direct input program*. Choose transaction BMV0 for the transfer using direct input.

1. Start *direct input management*.
2. You can find the path for the direct input steps under: *Applications → FI documents*.
 - Create variant VAR## for program RFBIBL00. To do this, enter the LSMW output file (converted data) as the file name. Choose the direct input technique as the transfer technique.
 - Now create the job Job-## for program RFBIBL00 and the variant VAR##.
 - Start the new job immediately and follow its flow. (Update the overview).

Solution 13: Transferring FI Documents using Direct Input

Task:

Optional: This is because only one user is allowed to use the direct input monitor at a time.

Do not transfer the documents using call transaction, rather using the direct input procedure. You can use the same record layout structure for this. As a result, you can reuse project BC420## with subproject BELE## or DOCU## (depending on what you have called it).

In the LSMW, start the *Direct input program*. Choose transaction BMV0 for the transfer using direct input.

1. Start *direct input management*.
 - a) Work as usual, proceed as in the last exercises.
2. You can find the path for the direct input steps under: *Applications → FI documents*.
 - Create variant VAR## for program RFBIBL00. To do this, enter the LSMW output file (converted data) as the file name. Choose the direct input technique as the transfer technique.
 - Now create the job Job-## for program RFBIBL00 and the variant VAR##.
 - Start the new job immediately and follow its flow. (Update the overview).
 - a) If you do not have any knowledge of background processing or jobs, ask the instructor for help. Batch processing is part of ADM100.



Lesson Summary

You should now be able to:

- Transfer FI documents into the SAP system using the direct input method
- Scheduling jobs
- Work with the direct input monitor



Unit Summary

You should now be able to:

- Transfer FI documents into the SAP system using the direct input method
- Scheduling jobs
- Work with the direct input monitor

Unit 8

Data Transfer Using IDocs

Unit Overview

You can also use the ALE/IDoc concepts for a data transfer. However, they are considerably different from the standard procedure or record layout procedure mentioned previously.



Unit Objectives

After completing this unit, you will be able to:

- Describe data exchange using IDocs
- Transfer data in LSMW using IDoc technology
- Perform the required administrative steps for IDoc transfer

Unit Contents

Lesson: Data Transfer Using IDocs	240
Exercise 14: Transferring FI Documents Using Direct Input.....	269

Lesson: Data Transfer Using IDocs

Lesson Overview



- ALE/IDoc basics
- Data transfer in LSMW with IDoc technology
- IDoc Administration



Lesson Objectives

After completing this lesson, you will be able to:

- Describe data exchange using IDocs
- Transfer data in LSMW using IDoc technology
- Perform the required administrative steps for IDoc transfer

Business Example

A company wants to perform a periodic data transfer with a high interval rate. In addition, the company wants to add customer-specific fields to the IDoc basic types provided by SAP. The LSMW supports this plan.

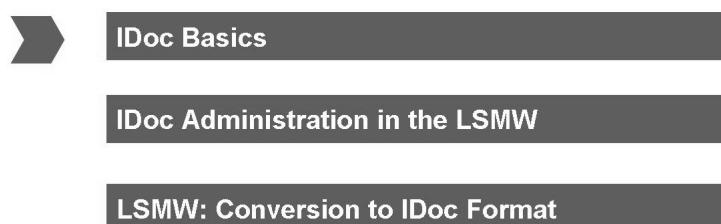


Figure 192: Data Transfer Using IDocs

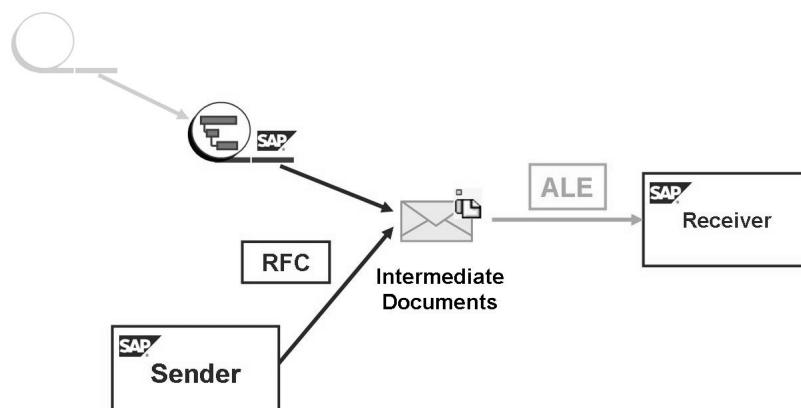


Figure 193: IDoc Concept

IDoc is an SAP standard format for data transfer between systems. “IDoc” stands for intermediate document.

In SAP systems, Application Link Enabling (ALE) includes the techniques, structures, and Customizing required for data exchange based on IDocs. There are two ways to exchange IDocs: as files or by Remote Function Call (RFC).

Data transfer focuses on how IDocs can be generated from sequential files in the SAP system and then posted as a document in the system.

IDoc technology is:

- **Message-oriented** - Data is also stored in applications, only in other formats (the application documents). The IDoc communicates between these application documents, as the language spoken by both applications. It is not important whether the application is programmed by SAP or by another software manufacturer.
- **Asynchronous** - Data can be stored in IDocs before an application document is created. This is important, for example, if incorrect data is transferred. In this case, the application document is only created when the data is corrected.

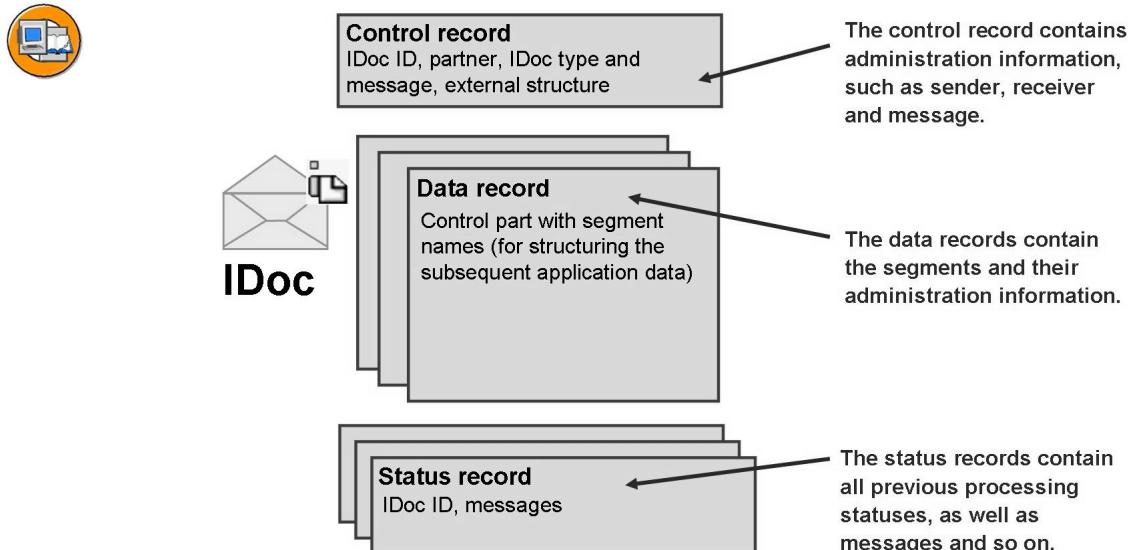


Figure 194: Structure of an IDoc - WE60

Each IDoc in the SAP database consists of one **control record**, **data records** that include application data in the segments and describe the hierarchy of these segments in the IDoc, and **status records** that determine the processing steps for the IDoc.

However, an IDoc that is exchanged to or from an external system does not contain any status records.

The important part of the **control record** is the IDoc ID, which is assigned internally in the system. It is unique. Status confirmation messages (notifications of the external system) always refer to this number.

The control record also contains the key fields of the partner profiles, and the last processing status.

The **data records** consist of a control part and an application part. As a result of the segment name in the control part, a structure is assigned to the unstructured part of the application data. The “network of application fields” is applied to it. The data type of the segment fields is “character”.

The **status records** log the statuses that the IDoc has passed through, for example, “created” or “ready for sending”. Consequently, they contain important data for monitoring and communication. The external system (receiving system) provides information about the processing statuses of a received IDoc by sending status confirmation messages. These contain status records only (for port type “file”). If the status confirmation reports communication errors, the system starts the exception handling. This may trigger a workflow, for example, during the process of which the persons responsible can react.

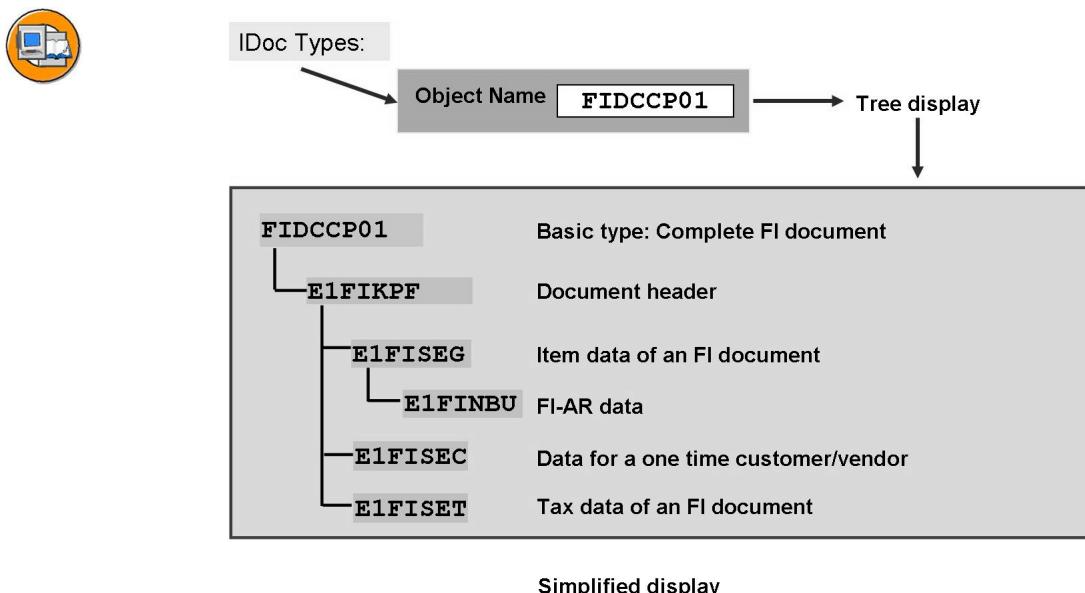


Figure 195: Structure of an IDoc Data Record.

Each **business process** usually corresponds to a particular **IDoc type** that can include the relevant data.

An IDoc type is defined by the segments, their hierarchy, sequence and frequency of use. This information is contained in the control part of the data records.

The hierarchy of the segments can be displayed. Segments can be children of the parent segments. This enables application data to be assigned.

The IDoc types supplied by SAP are known as **basis types**. Basic types can be combined with customer-defined enhancements according to strict rules.

You can display the structure of an IDoc type from the EDI Basis menu (*Tools → Business Communication → IDoc Basis → Documentation → IDoc Types*) using the *tree display*.

The example shows a segment (simplified display) of the basic type “FIDCCP01” (FI document).

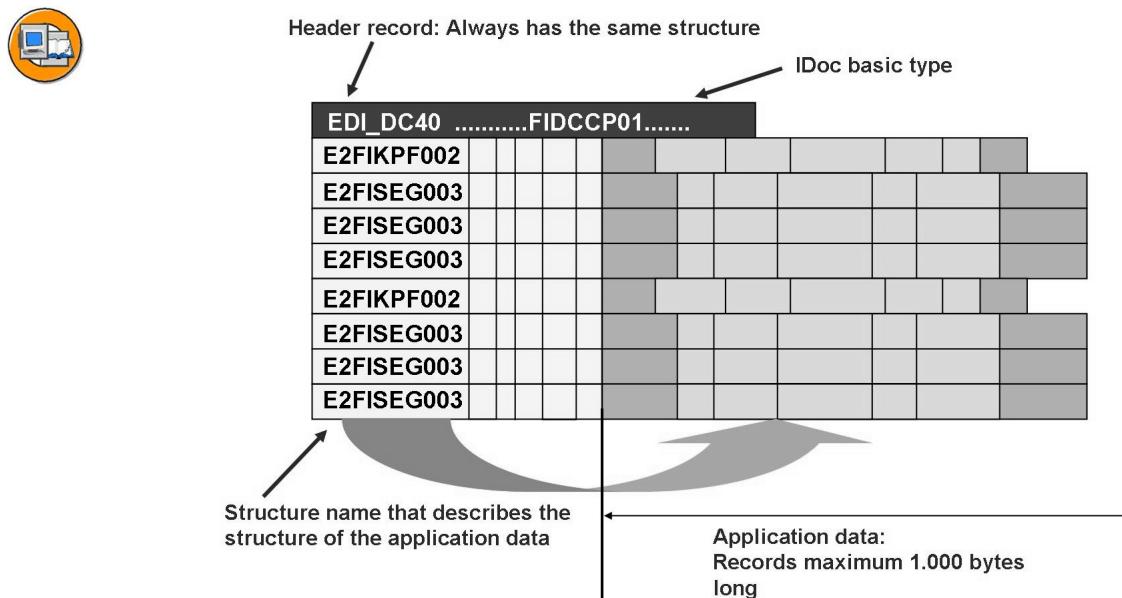


Figure 196: Structure of an IDoc file

An IDoc always contains a header segment.

Each of the records below contains six fields.

SEGNAM	Segment
MANDT	Client
DOCNUM	Number of the IDoc
SEGNUM	Segment number
PSGNUM	Number of the parent segment
HLEVEL	Hierarchy Level

The structure of the application data is described for each record in the SEGNAM field.

IDoc basics:

- The fields in IDoc segments are of type CHAR.
- Naming convention: A segment consists of one SAP release-independent segment type and at least one SAP release-dependent segment name. Segment types are structures in the ABAP Dictionary. They begin with “E1”. Segment names begin with “E2”. From the segment name, an external system can recognize the version of the segment in question.

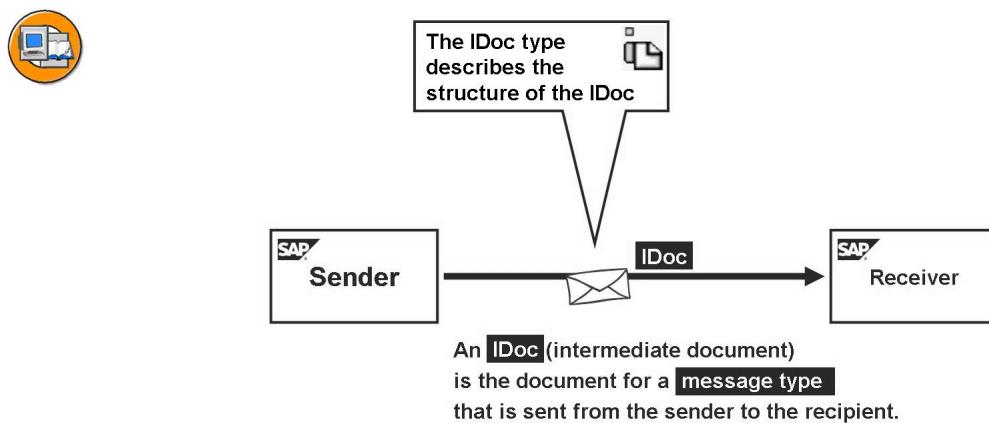


Figure 197: IDoc and Message Type

If a business process extends across two SAP Systems, information can be sent in IDocs. The structure of an IDoc is defined by an IDoc type. Among other things, the IDoc type contains information about which data is saved where (line and offset). IDoc types are assigned to message types.

Whereas the message type contains only the semantics of a message (for example, material master data), the IDoc type contains the exact structure of the document for a message type.

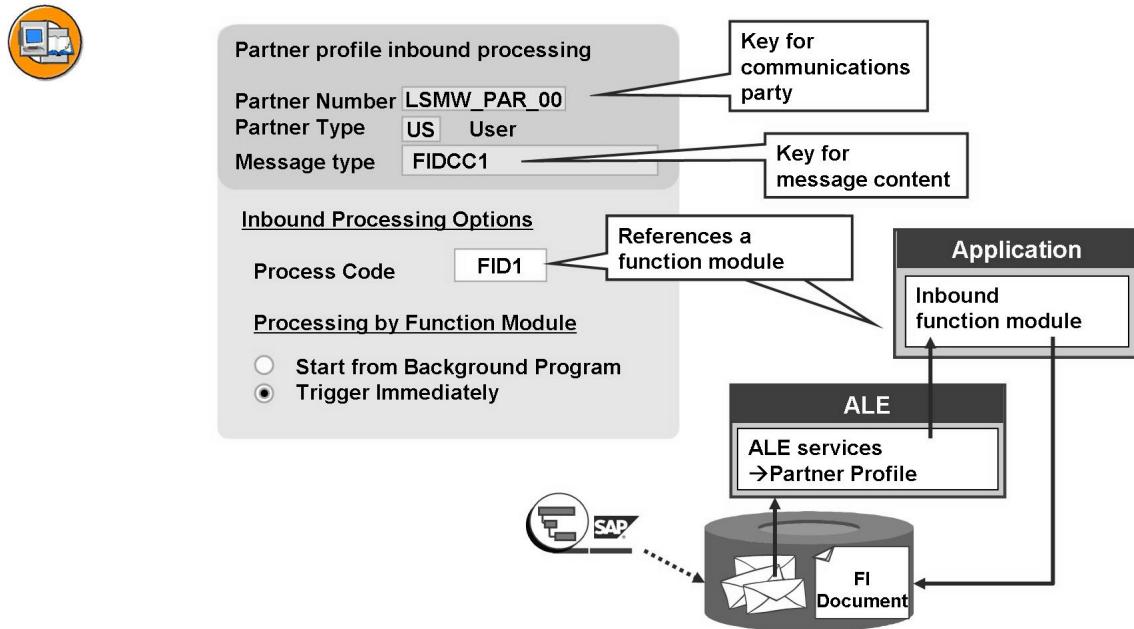


Figure 198: Partner Profiles: Inbound Processing - WE20

IDoc files are processed in two steps:

1. The data from the IDoc file is stored as an IDoc in the database.
2. The ALE inbound processing service transfers the IDocs from the database to a function module for processing. This function module checks and posts the data.

For the second step a partner profile must be set up in Customizing, which defines the processing for the key data from the IDoc. Key fields for the processing include the message type of the message content and the partner number of the sender of the data.



What can I do if an IDoc is not processed correctly?

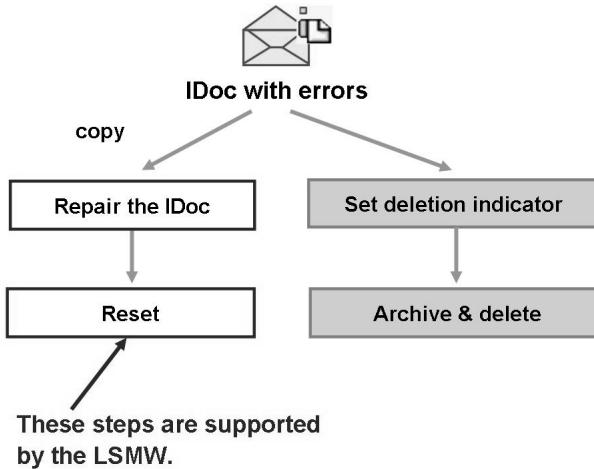


Figure 199: Error Handling for IDocs

If it is not possible to successfully post the data of an IDoc, you can repair the IDoc and reset it. A new status (IDoc status 69) documents the repair of an IDoc. As a result of this process, the old, original IDoc receives a new number. The old number is assigned to the new, copied IDoc.

You can delete IDocs only by “detouring” through archiving. After you set the deletion indicator, IDocs can be archived and therefore deleted from the system. This is possible only from the integrated inbox or the Business Workplace.

The “foreground processing” of IDocs (transactions are called internally using *Call Transaction*) is possible for certain messages only. The decisive factor here is the process code, which mostly points to a function module. This “foreground processing” is available only from the integrated inbox or the Business Workplace as described above.



IDoc Basics



IDoc Administration in the LSMW

LSMW: Conversion to IDoc Format

Figure 200: Data Transfer Using IDocs - IDoc Administration

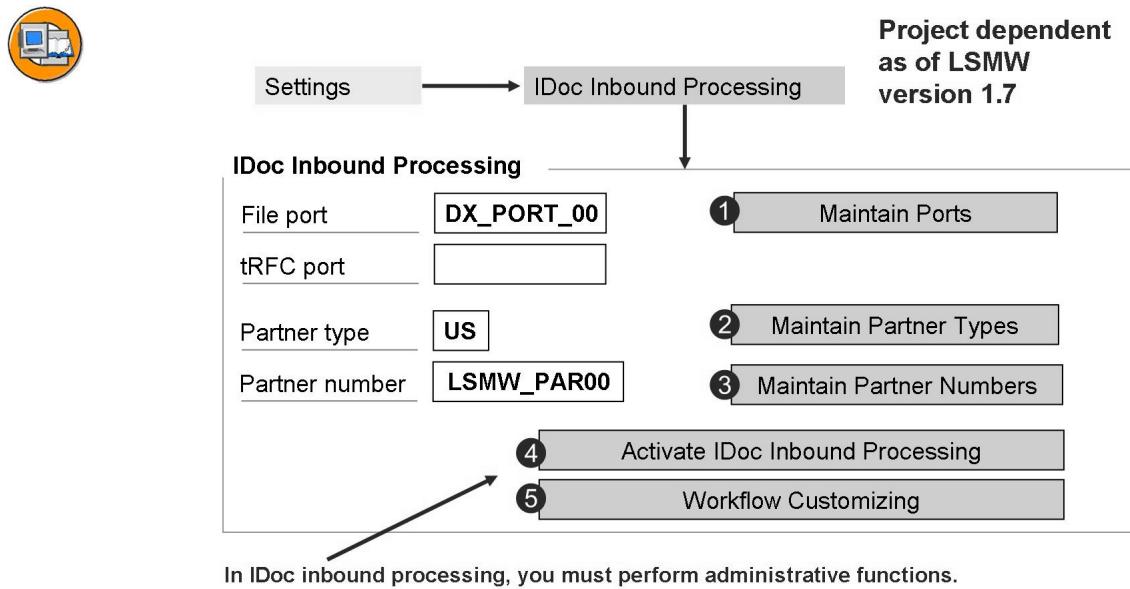


Figure 201: IDoc Inbound Processing from LSMW

By choosing *Settings* → *IDoc Inbound Processing* to go to the LSMW administration environment for IDocs. From here, go to the relevant maintenance transaction of IDoc processing.

As of LSMW Version 1.7, the settings of *IDoc inbound processing* are project dependent; that is, settings are maintained for the selected LSMW on the entry screen. In earlier versions, the settings are valid for all projects.

The screenshot shows the 'Maintain Ports' screen. Step 1, 'Maintain Ports', is highlighted. On the left, a tree view shows 'Ports' expanded, with 'File' selected, showing 'DX_PORT_00'. Other options include 'CPI-C', '...', 'Internet', and 'XML'. On the right, details for 'DX_PORT_00' are shown:

- Port: DX_PORT_00
- Description: Demo LSMW
- Version: ...
 - IDoc record types SAP Release 4.x
- Outbound file: ...
 - Physical directory: D:\usr\sap\trans
- Inbound file: ...
 - Function module:
- Status file: ...
 - Outbound file: Outbound_file.txt

Figure 202: Maintain Ports

Ports are channels through which the IDocs are exchanged. The IDoc Interface supports five different transmission methods. These are the following port types: File, transactional RFC, CPI-C, Internet and programming interface PPS.

For the file port type, IDocs written to files at operating system level. The receiving system can read the files here. The receiving system can also be started using the synchronous RFC. Apart from IDocs, you can also exchange data records using files.

The port definition file comprises the name and the directory of the file. Only the outbound file is important because the place and name of the file are determined by the external system during inbound processing of IDocs or of a status confirmation. However, if you enter a parameter for the inbound IDoc and status file, test tools can generate default values. It is important that the port exists, even if it is used in inbound processing only, because the IDoc Interface accepts only ports that it recognizes.

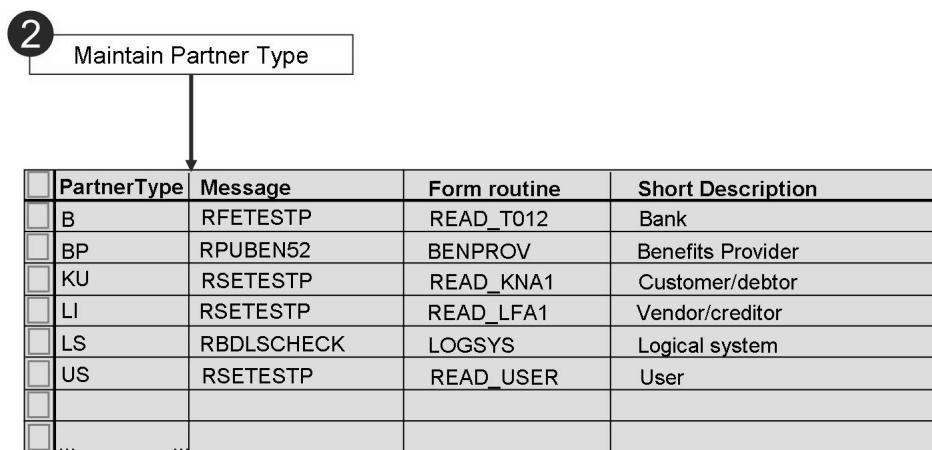


Figure 203: Maintain Partner Type

For the *Partner*, choose a “US” SAP user (partner type US = user). In this way, you can forward all messages (errors) that occur during data transfer using LSMW to suitable processors. These processors are in the partner profiles of this SAP user.

Historically speaking, the partner types “LI” and “KU” are defined next to enable the system to separate the same numbers of customer and vendors. During IDoc development, other partner types were added, for example “B” for banks and “US” for users.

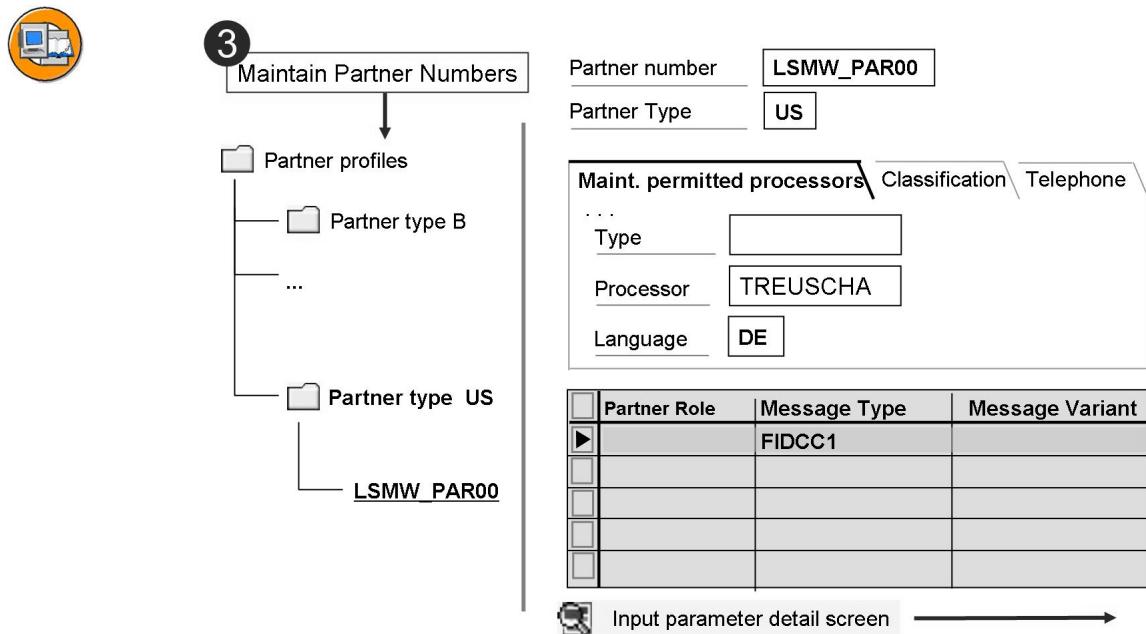


Figure 204: Maintain Partner Numbers.

The administrator entered under Processor is notified if an error occurs.

For each partner profile, you can control who is notified if an error occurs.

Partner Number	LSMW_PAR00	Partner
Partner Type	US	User
Partner function		
Message type	FIDCC1	Sending entire FI documents
Message variant		
Message function		

Inbound Options	Post processing allowed ...	Telephony
Process Code	FID1	FIDCC1: FI entire documents
...		
Processing by Function Module		
<input checked="" type="radio"/> Start from Background Program <input checked="" type="radio"/> Start Immediately		

Inbound IDocs are processed immediately.

Figure 205: IDoc Inbound Parameter

By choosing *Detail Screen → Inbound Parameters* you reach a screen in which you can set the inbound processing option “Trigger immediately”, among others. Inbound IDocs are then directly transferred to the application.

If you choose “Trigger by background program”, the system does not transfer the IDoc to the application. For this purpose, choose the postprocessing function from LSMW (select status 64 - IDocs).

The message type FIDCC1 is assigned to the process code “FID1”. Behind the term **process code** is a method of processing, such as a function module or workflow, for example. In this processing, the system reads IDocs. By using process codes, you can, in one step, replace an old method of processing with a new one for as many partners as required.

In the top node of the IDoc Interface, you can display the process codes for a message type. (Transaction WE64, *Docu → Process codes*).

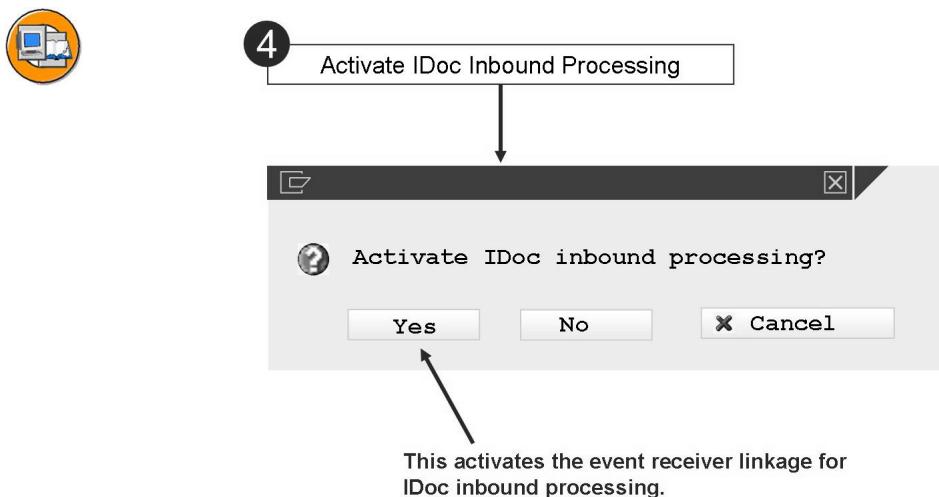


Figure 206: Activate IDoc Inbound Processing

You can also call this function using Customizing for the IDoc Interface in the IMG (Implementation Guide: *Basis → Basis Services → IDoc Interface*).

IDocs in the inbox are first saved to the database and then transferred to the application-specific inbound processing.

This transfer is transferred by an event, except in the case of port type “tRFC”. The processing standard task (the event receiver) must therefore be linked to this event. In addition, the link must be activated.

Basic Principles of Event Receiver Linkage:

If IDocs are received, the system first saves them to the database. In a second, independent step, the system processes them further (for port types “file”.) This is made possible by the workflow concept of the event: If IDocs are saved to the database, the system triggers an event that waits for the event receiver in the system. The event receiver (a function module) finds the event and triggers event processing. As a result, the receiver has used the event and the event no longer

exists in the system. The workflow determines when the event receiver starts to search for events: There is therefore an interval between when the system saves data and when it processes data further (asynchronous processing).

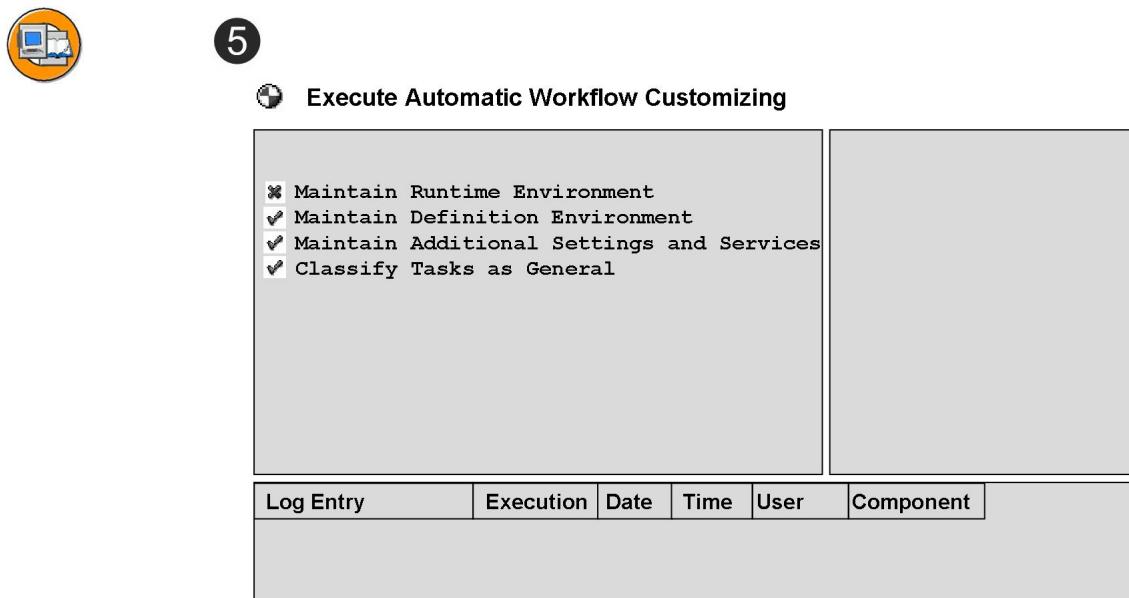


Figure 207: Workflow Customizing

You must set a green checkmark for the following entries of Workflow Customizing.

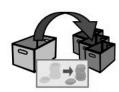
- Runtime environment:
 - Configure RFC destination
 - Maintain a system administrator for workflow
 - Generally classify decision tasks
- Maintain additional settings and services:
 - Sending to objects and HR objects is activated

For this purpose you can start automatic Customizing.

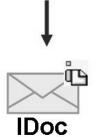
You should deactivate “monitoring jobs for incorrect workitems” (set to “Not selected”). Otherwise, the SAP system would constantly try to post incorrect IDocs that are created during data migration. For more information, see SAP Note 149368.

**IDoc Basics****IDoc Administration in the LSMW****LSMW: Conversion to IDoc Format**

Figure 208: Data Transfer Using IDocs - Conversion in LSMW



SAP
IDoc
structure



- Define object attributes
- Define source structures and source fields
- Define structure relationships
- Define field mapping and conversion rules
- Import data
- Convert data
- Start IDoc creation
- Start IDoc processing
- Create IDoc overview
- Start IDoc postprocessing

Figure 209: Process Steps for Transferring IDocs

To transfer an IDoc using the LSMW, you require the process steps shown in the sketch.

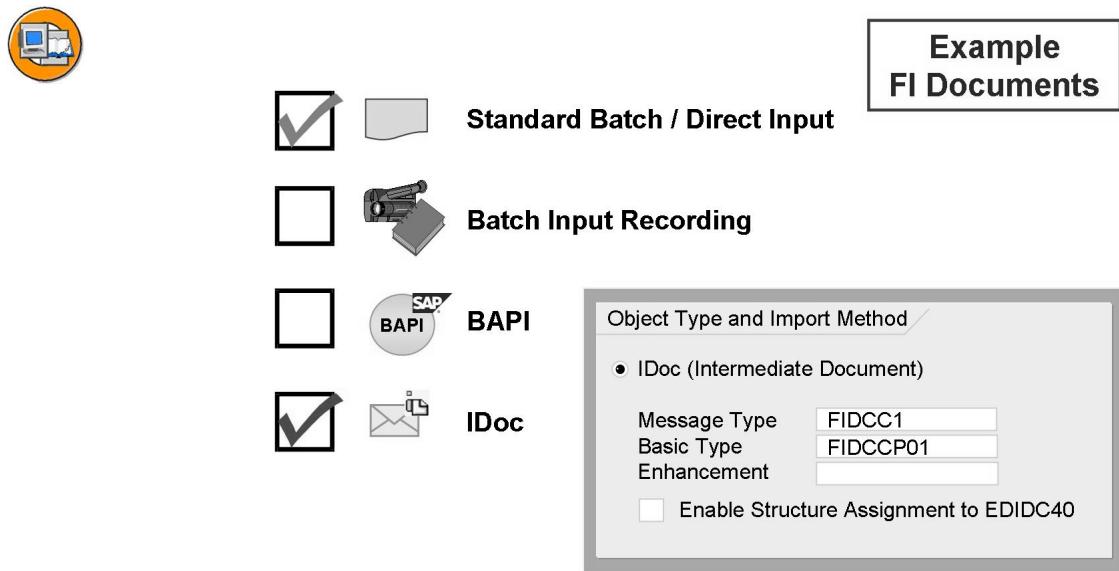


Figure 210: Object Type and Import Method IDoc

Choose the object type and import method. The input field provides an *F4* help (input help) with highlighted lists from which you can select the relevant objects.

If you choose IDoc as the import method, the system checks (when saving) whether a partner profile already exists for the default partner and the selected message type. If not, the system attempts to create a profile.

Port profiles and partner profiles will be discussed later in the unit.

Using the choice field *Allow Structure Assignment for EDIDC40*, you can assign source fields to the IDoc control structure EDIDC40. Use this field only if you want to import old LSMW objects to Release 6.40 or higher in order to manually set EDI_DC40-TABNAM = 'EDI_DC40_U' in the control structure of the IDoc control record. The LSMW automatically triggers this when creating a new LSMW object with the type IDoc. Therefore, you can normally ignore this choice field.

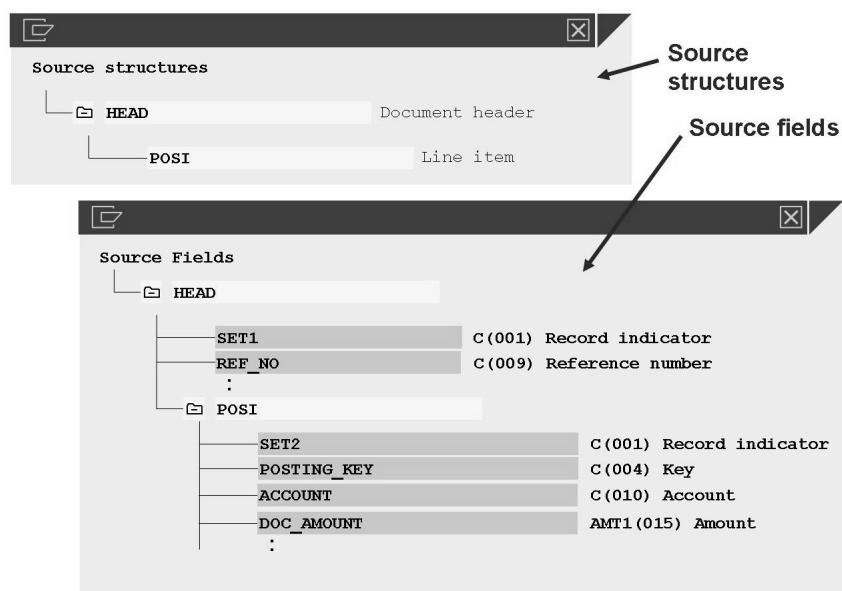


Figure 211: Define Source Structures and Source Fields

In this step, you define the object structures including name, description, and hierarchical relationships. Afterwards, fields are defined for the structures defined in the previous step.

In the dialog box, choose *Change*. You can now choose to create new structures, or change, rearrange, or remove these. Use the relevant pushbuttons to execute these functions.

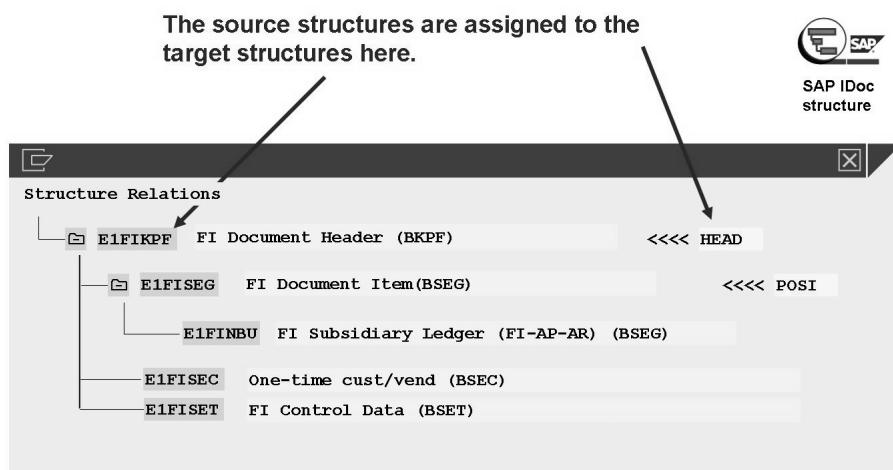


Figure 212: Defining Structure Relationships

The structure relationships define the relationships between the source and target structures. The possible target structures are defined when you select the object type and import method.

Some of the target structures must be selected (“mandatory segment”). In this case the system informs you: “This structure must be selected.”

To set structure relationships, place the cursor on a field in the SAP structures or target structures. Choose “Relationship”. A dialog box appears with a list of the source structures you have created. Select a structure from this list.

If you want to change the relationship, you must first delete the existing relationship using the relevant pushbutton.

A check function is available to check the structure relationships for errors. Messages for this check are displayed in the status bar: “No structure relationship errors found.”

IDoc Features with FI Documents

When IDocs are used to transfer data there are always special characteristics to pay attention to. For document transfer using IDoc, these are as follows:

- A global company code must exist (ALE)
- Specify transaction type
- Fill G/L account number with leading zeros
- Set debit/credit indicator
- Enter transaction type of the general ledger
- Determine G/L account number

Unlike data transfer using batch input, when you use IDocs there are extra fields that must be filled. The IDoc type and the application determine which fields these are.



- **Cross-system company codes** are used for ALE distribution in financial accounting.
- In a distributed environment there is one central system for each **cross-system company code**.
- A company code must be assigned to this **cross-system company code** on each system in the distributed environment.

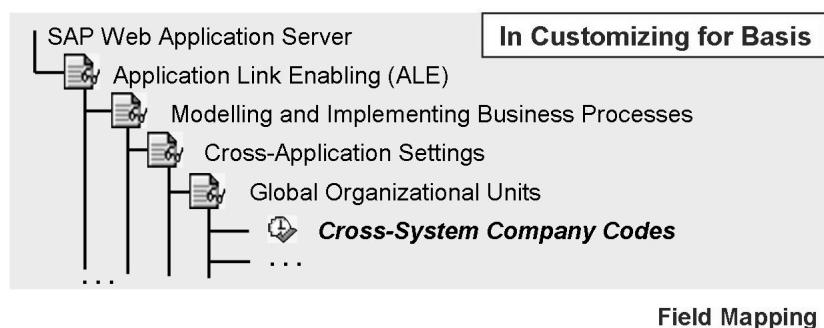


Figure 213: Global Company Code

When a message with company-code-specific data (in an ALE scenario) is sent, the company code is replaced with the global company code in all company code fields. When a message with company code-specific data is received, the reverse conversion takes place in the target system.

You can also call the node *Application Link Enabling (ALE)* directly using the transaction code SALE.

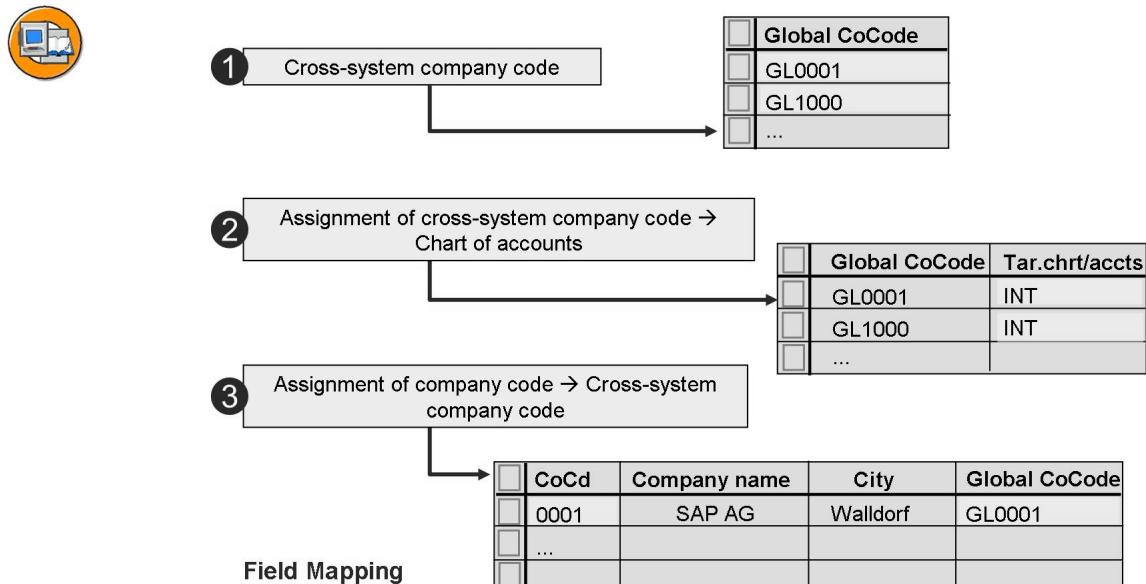


Figure 214: Set Up Cross-System Company Codes

Activities:

1. Create the cross-system company code.
2. Assign the cross-system company code to the chart of accounts to which it is also assigned in the central system (see the definition of a central system in ALE). You must also make this setting in the decentral system to determine the chart of accounts of the company code in the central system.
3. Assign cross-system company codes to the local company codes. For the individual company codes, assign the descriptions of the relevant cross-system company code.

Recommendation: We recommend that you choose matching descriptions for the company code and the cross-system company code.

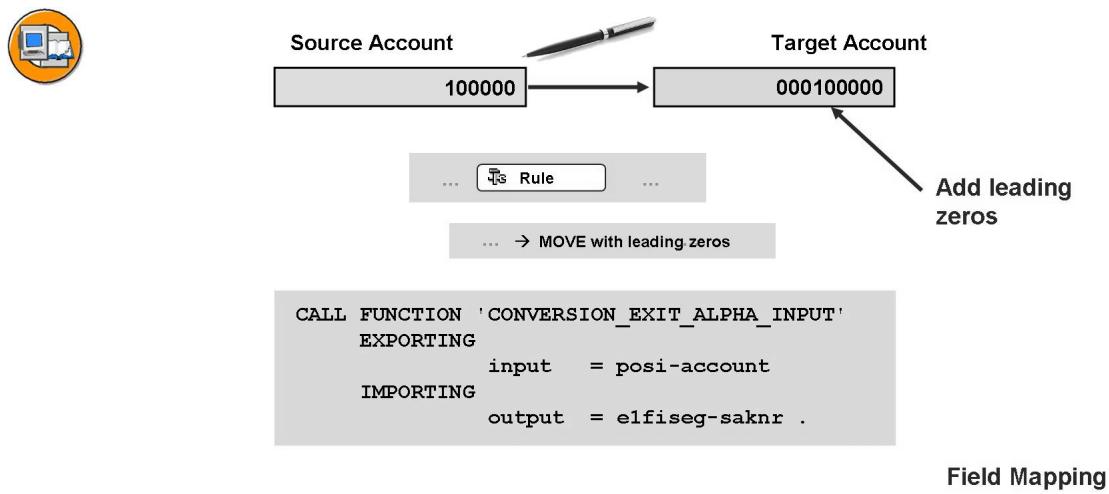


Figure 215: Convert G/L Account Number

Conversion exit ALPHA, external → internal:

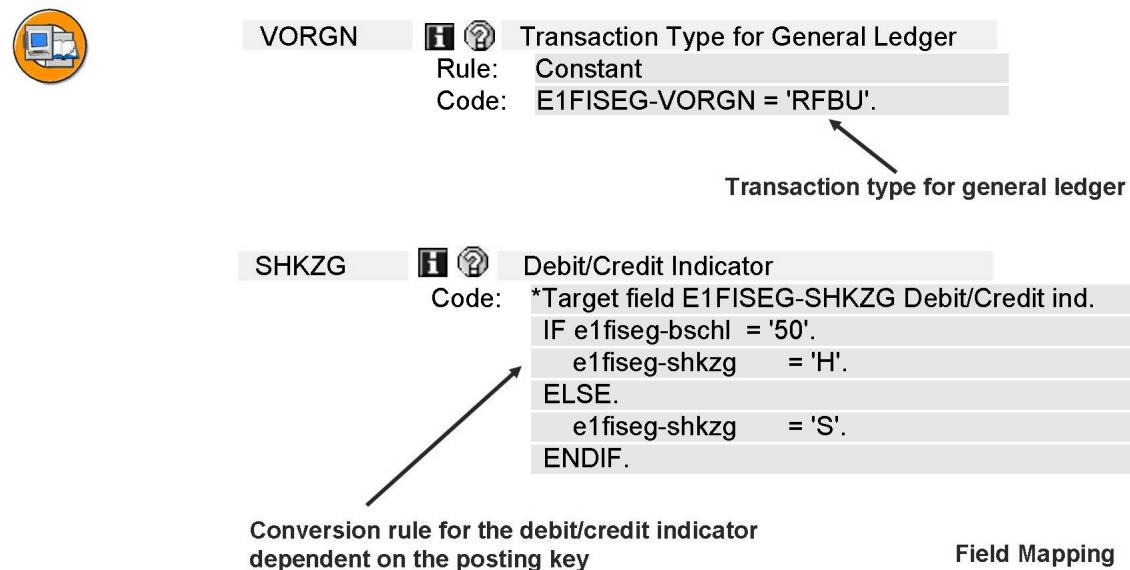
The ALPHA conversion is used especially for account numbers. When account numbers are converted into the internal format, the system checks that the *Input* consists of numbers only (numeric), possibly with blank characters before or after the numbers. If this is the case, the numeric string is placed into the *Output* field, right-justified, and any spaces to the left are filled with zeros. If the input field contains characters too, this character string is transferred to the OUPUT field, left-justified, and any space to the right is filled with blank characters.

Examples (input field and output field are both eight characters long):

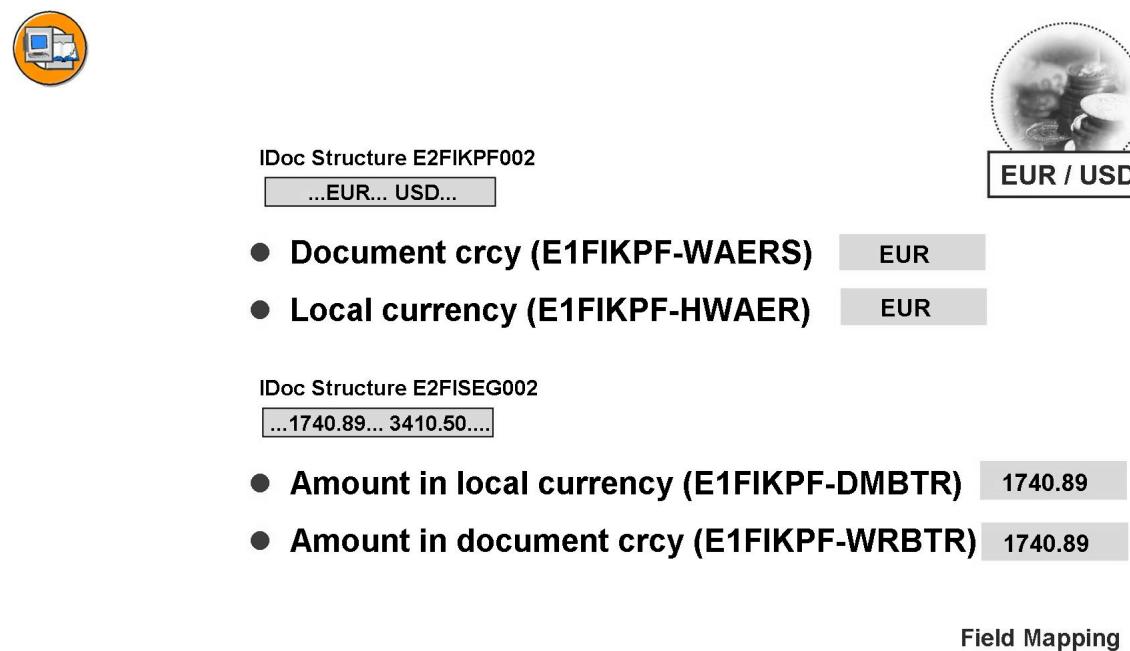
1. '1234' > '00001234'
2. 'ABCD' > 'ABCD'
3. '1234' > '00001234'

The conversion from the internal format to external format (function module **CONVERSION_EXIT_ALPHA_OUTPUT**) is done exactly in the reverse order.

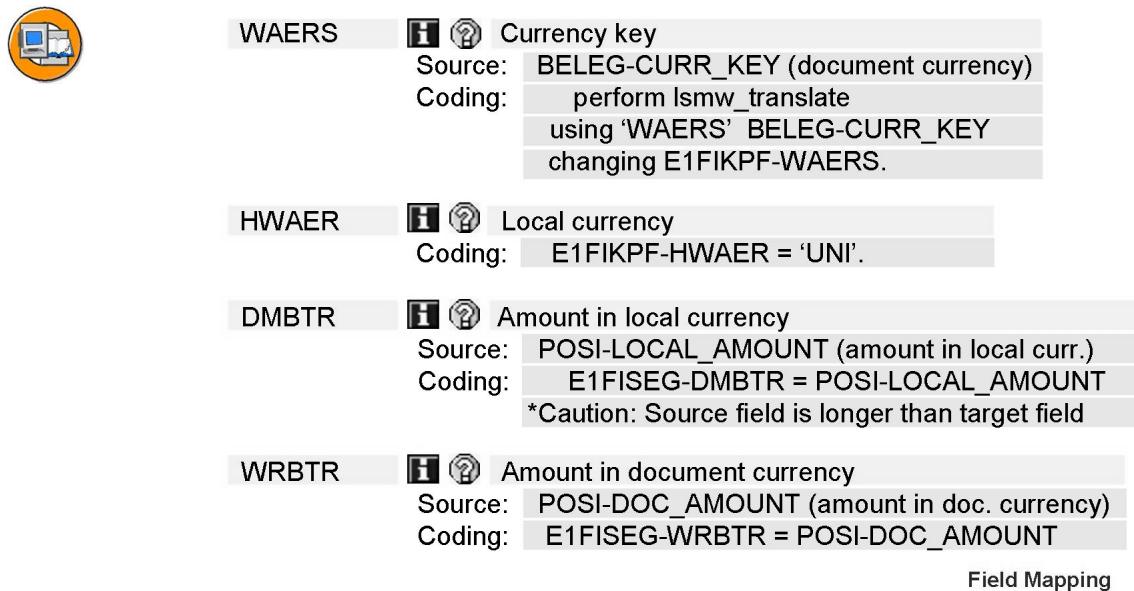
In the LSMW account numbers are converted using the rule “**Move With Leading Zeros**”.

**Figure 216: Determine Transaction Type or Debit/Credit Indicator**

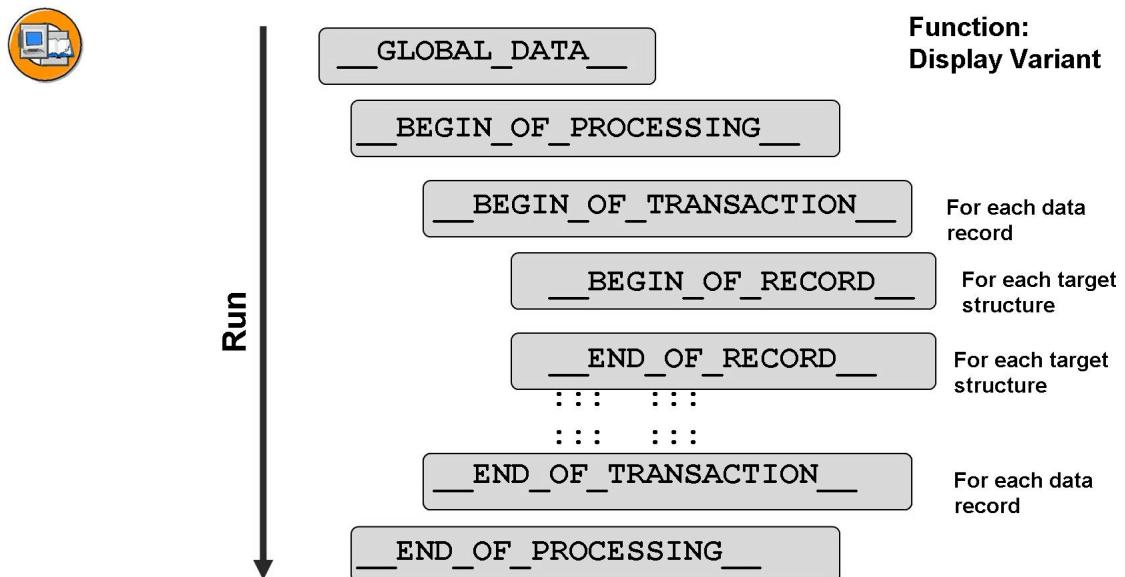
To enable document transfer using IDoc, you must enter the process code and the debit/credit indicator.

**Figure 217: Document Currencies**

To enable document transfer using IDocs, you must enter two currency fields and amount fields for both.

**Figure 218: Currencies for IDoc.**

The following fields in the target structure must be filled with the amounts and currencies.

**Figure 219: Processing times in the LSMW**

Global data definitions: The system displays the label **__GLOBAL_DATA__** for global data definitions and declarations. There you can define a variable, structures, tables and so on that you want to use in your own source code in the field mapping.

Processing times: Provide you with the option of inserting your own source code for particular processing times.

Processing time - Meaning:

- BEGIN_OF_PROCESSING (before the start of the data processing)
- BEGIN_OF_TRANSACTION (before the start of data processing for a transaction)
- BEGIN_OF_RECORD (before applying the conversion rules for a source structure)
- END_OF_RECORD (after applying the conversion rules for a source structure)
- END_OF_TRANSACTION (after the end of the transaction processing)
- END_OF_PROCESSING (after the end of the data processing)

In practice, it is possible (at a particular processing time) to read additional data from SAP tables or, for example, to access Customizing. This additional data can be included in the rules. However, when programming this type of ABAP extra, you must always consider performance. Therefore, if you enhance the rules of a field with an additional SELECT SINGLE, this will certainly impair the performance of the transfer. Identical selects using identical accesses would impair the performance even more. Therefore, you should always use performance-optimizing programming. Keep in mind that the source code for a field will normally run at least as often as there are external records in the external file.



Under *Extras → Display Variant:*

Activate technical fields and times of processing.

GLOBAL_DATA Global Data Definitions...

Code: DATA: pos_counter like
E1FISEG-BUZEI value 0.

E1FIKPF Segment

BEGIN_OF_RECORD Before Using Conversion Rules

Code: pos_counter = 0.

E1FISEG Segment

BEGIN_OF_RECORD Before Using Conversion Rules

Code: pos_counter = pos_counter + 1.

BUZEI Number of Posting Items Within the Document

Code: E1FISEG-BUZEI = pos_counter

Field Mapping

Figure 220: Item Counter

To determine the number of the document item, use the counter pos_counter. For each document item, this counter must be incremented and assigned to the field *EIFISEG-BUZEI*.

In a classic ALE scenario with inbound processing and outbound processing, this counter is automatically set by ALE services. Since IDoc objects must be “rebuilt” manually for data transfer, you must program the incrementation of the counter yourself (using the LSMW).

Further Steps



- Define Object Attributes
- Define Source Structures and Source Fields
- Defining Structure Relationships
- Define Field Mapping and Conversion Rules
- Importing Data
- Convert Data
- Start IDoc Creation
- Start IDoc processing
- Create IDoc overview
- Start IDoc postprocessing

You must now run through the specified steps.



- Import data
- Display the imported data

Check the imported data.

LSMWBC420-00	DOCU-00	IDOC-00	1011..	
1 HEAD	HBC100000120060730	07 20060731	20030730	
2 POSI	P0004S100000	3410.50	1740.89	Info 1
3 POSI	P0005H113100	3005.50	1536.41	Info 2
4 POSI	P0005S110000	405	204.48	Info 3
5 HEAD	HBC100000220060730	07 20060731	20030730	EU
6 POSI	P0			
7 POSI	5H	Field name	Field text	Field contents
		SET1	Record indicator	H
		REF_NO	Reference number	BC1000001
		DOC_DATE	Document date	20060730
		TYPE	Document type	07
		POSTING_DATE	Posting date	20060731
		TRANS_DATE	Translation date	20060730
		CURR_KEY	Document currency	EU
		...		

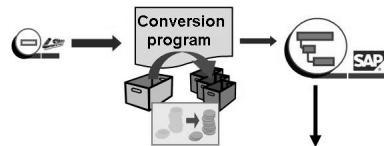
Figure 221: Importing Data

After you import the external data, you can display this data from the LSMW. Colored labeling emphasizes the document header and the item data.

When you choose a line, the system displays the individual fields with their contents.



- Convert data
- Display the converted data



1	EDI_DC40_U	...	100	...700	...	FIDCCP01
2	E1FIKPF	E2FIKPF002		100		00001... 1GL1000
3	E1FISEG	E2FISEG003	100			00002... 2
4	E1FISEG	E2FISEG003	100			00003... 2
5	E1FISEG	E2FISEG003	100			00004... 2
6	EDI_DC40_U	...	100	...700	...	FIDCCP01
7	E1FIKPF	E2FIKPF002	100			00005... 1GL1000
8	E1FISEG	E2FISEG003				
9	E1FISEG	E2FISEG003				

Field name	Field text	Field value
TABNAM	Table name	EDI_DC40_U
MANDT	Client	100
DOCNUM	IDoc Number	
DOCREL	SAP Release	700
STATUS	IDoc Status	
...		
IDOCTYP	Basic Type Name	FIDCCP01

Check the converted data.

Figure 222: Convert Data

The previously imported data is now converted to the IDoc format. This does **not** mean that an IDoc is already generated on the SAP database.

The IDoc segments are emphasized using colors.

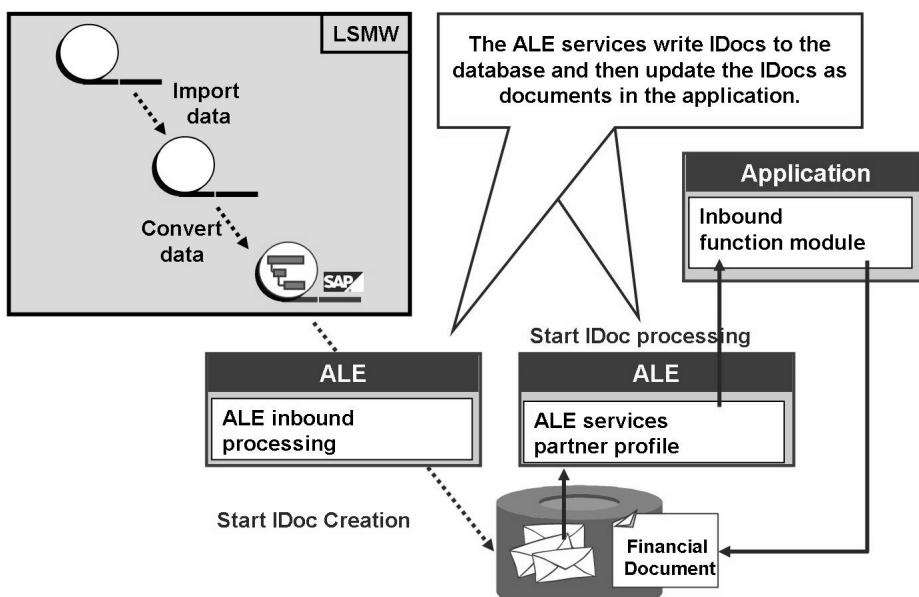


Figure 223: Start IDoc Generation and Processing

After a file has been created in IDoc format, you can use the ALE functions of the SAP system to triggers further processing. This can also be triggered from the LSMW.

Start IDoc creation:

The data converted earlier is now passed to ALE inbound processing. An IDoc is created in the database for each data record (for example, an FI document).

Start IDoc processing:

The created IDocs are read one by one and the inbound function module referenced in the partner profile is called. This checks the data and if the check is successful it writes the data in the application tables. The data (for example, an FI document) is created.

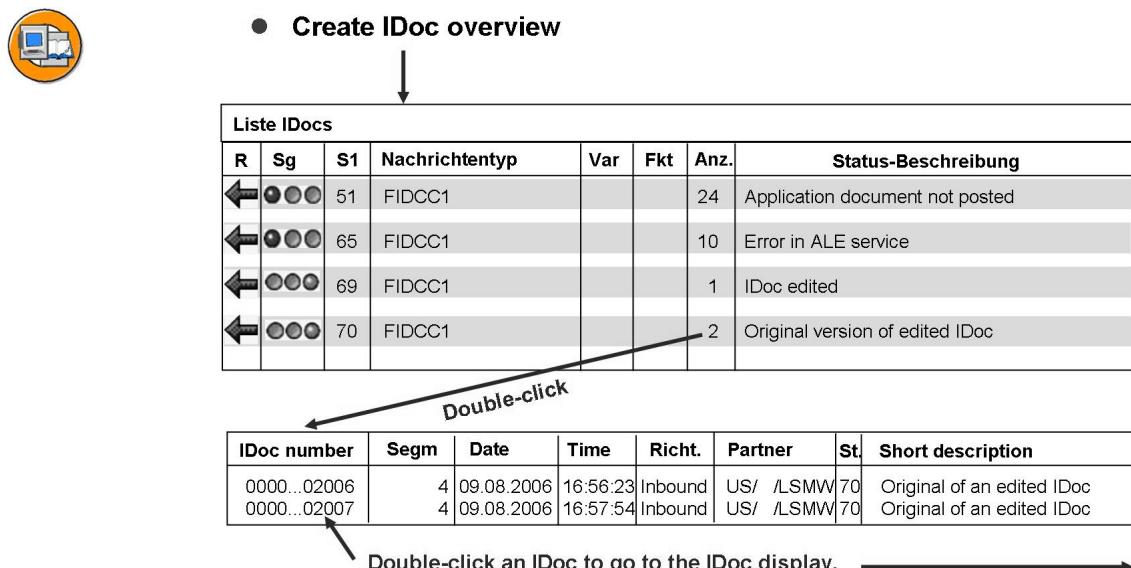


Figure 224: IDoc Lists

You can display the status confirmations of the created IDocs in the IDoc overview.

The list displays IDocs according to status number. By double-clicking on a status number you can display the details of individual IDocs.

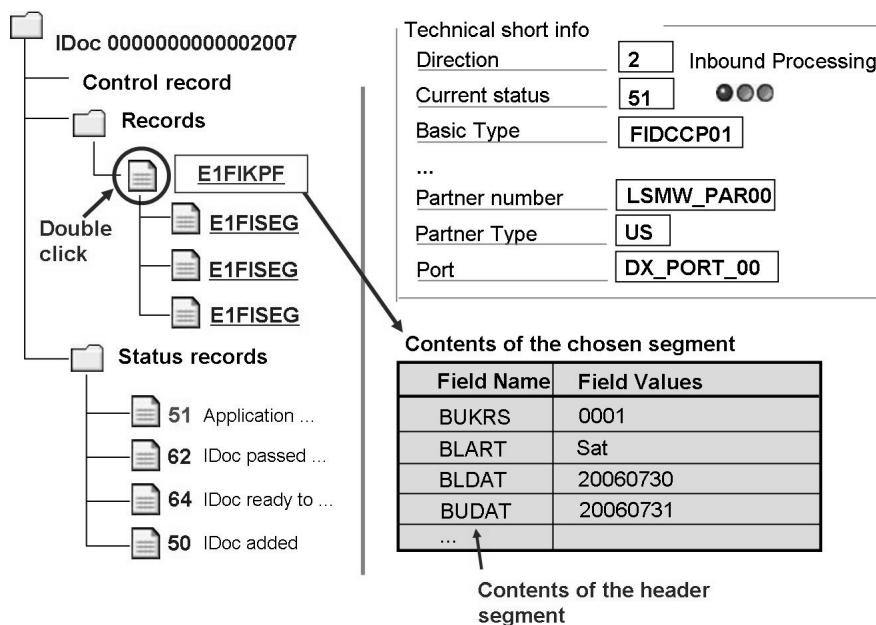


Figure 225: IDoc display

Here the IDoc with the unique number 2007 and all the status messages created up to now are displayed. The status is arranged chronologically meaning that the last status created (in this case number 51) was the last message reported.

On the right side of the screen, you can monitor the contents of the selected segment E1FIKPF. By double-clicking on the segment icon, you can display a detailed list with the fields of E1FIKPF.



IDoc number	2007	
Segment Type	E1FIKPF	FI IDoc: Document header: (complete document)
Number	1	
Superior segment no.	0	Hierarchy level 1

Field Name	Field Values	Short Description
BUKRS	0001	Cross-system company code
BLART	Sat	Document Type
BLDAT	20060730	Document date in document
BUDAT	20060731	Posting date in the document
WWERT	20060730	Translation date
XBLNR	BC1000002	Reference document number
WAERS	EUR	Currency Key
GLVOR	RFBU	Business Transaction
HWAER	EUR	Local Currency
...		

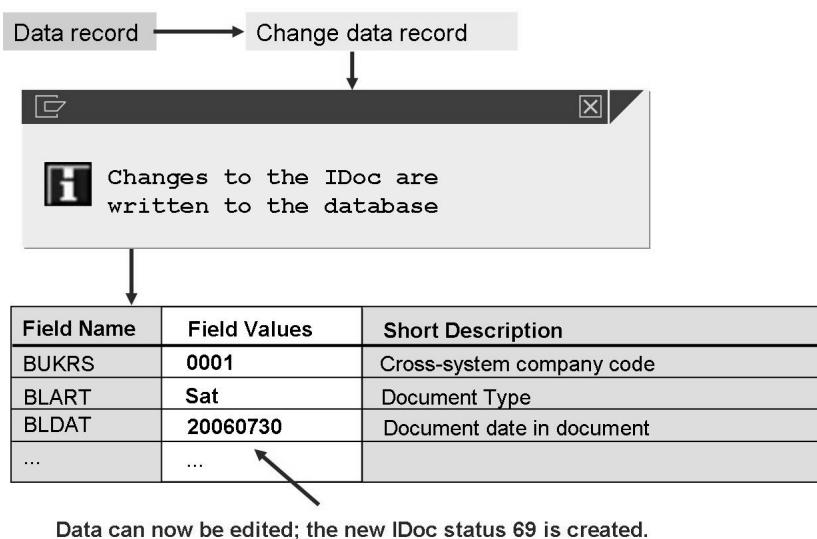
When you double-click the header segment icon, the system displays this detail screen.

Figure 226: Displaying an IDoc Record



List with fields of the segment E1FIKPF.

Based on this screen, you can edit the IDoc data.



Data can now be edited; the new IDoc status 69 is created.

Figure 227: Edit IDoc

You can edit the data of the IDoc by choosing *Data Record → Display/Change Data Record*.



Hint: What happens when I change an incorrect IDoc?

- 1) The system generates a copy of the old IDoc with a new number, for example 2008.
- 2) The copied, new IDoc receives the old number, in this case 2007.
- 3) After the editing process, the IDoc with the number 2007 is the new copied IDoc.
- 4) You can now process this edited, corrected IDoc 2007 again.

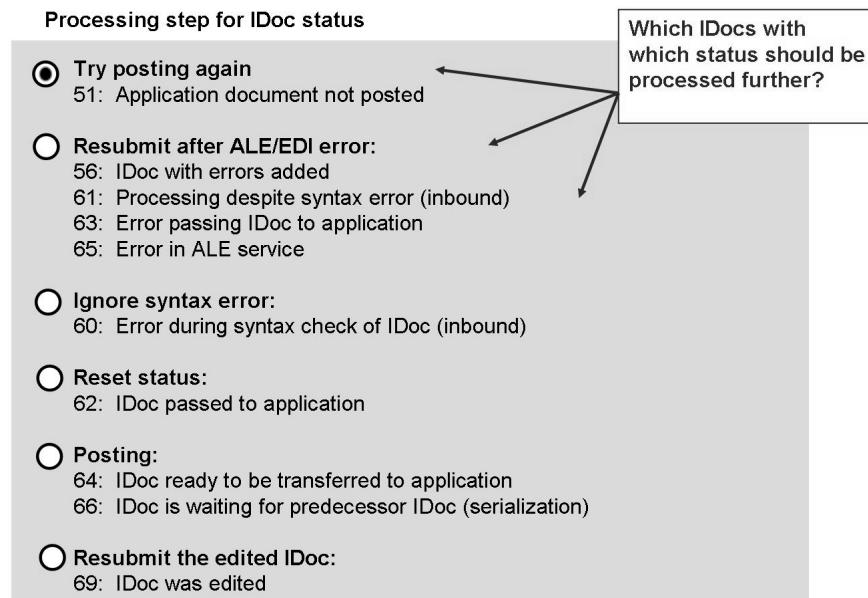


Figure 228: Start IDoc postprocessing

In IDoc postprocessing, the system processes IDocs with a particular status; that is, the IDocs are sent to inbound processing again. Therefore, you can process the previously edited and corrected IDocs (all in status 69) again. The incorrect original IDoc has status record 70. This IDoc is used for storage and for tracing or documenting the entire process.

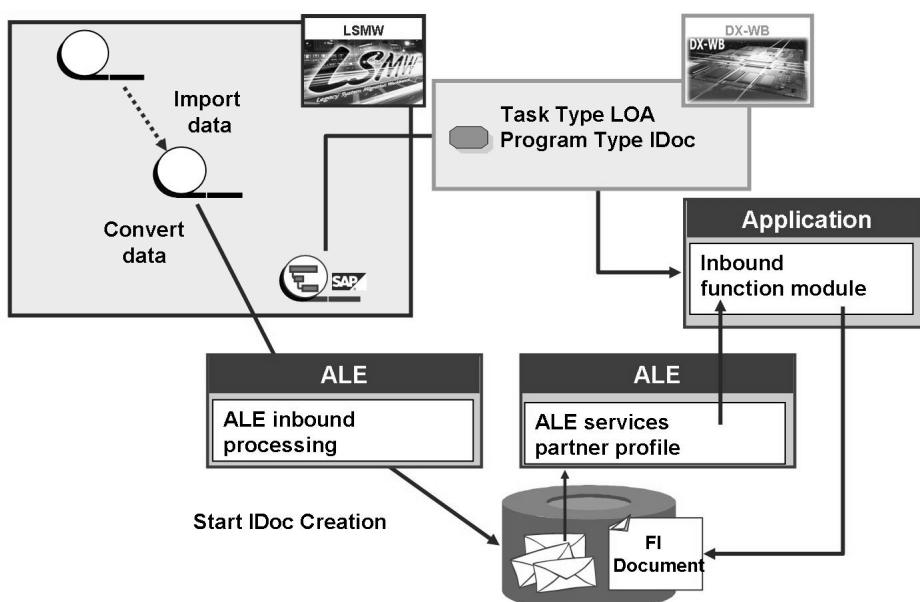


Figure 229: Outlook: Special Processing Methods

Special processing methods (possible as of 6.20):

1. In LSMW, IDocs can be created directly in the process step *Convert Data*. This avoids the process step *Create IDocs*.

If you select the option “Trigger Immediately” in the partner profile, IDocs created this way are transferred directly to ALE inbound processing - this also avoids the process step *Process IDocs*.

2. In the DX Workbench, you can trigger the data transfer from the converted file directly to ALE inbound processing. This circumvents the step for creating IDocs on the database.

To do this, the inbound function module (in this case IDOC_INPUT_FIDCC1) must have been registered (task type LOA, program type IDoc). This method avoids ALE inbound processing, which converts the global company code, for example. This is important to consider when you define the field mapping in LSMW.

Exercise 14: Transferring FI Documents Using Direct Input

Exercise Objectives

After completing this exercise, you will be able to:

- IDoc Processing Basics
- Transferring FI Documents Using IDoc Technology

Business Example

You want to use IDocs to transfer external FI documents to an SAP system. As a tool you use the LSMW.

LSMW project: BC420-##

Subproject: DOCU##

Object IDOC##

File with documents in external format: BC420_DOC_2_HEAD_POS.LEG

Task 1:

The FI documents are available in the file BC420_DOC_2_HEAD_POS.LEG in the transport directory. This file contains five documents, each with three items. You want to use the LSMW to transfer this data to the relevant IDoc structure. You then want to transfer this file to ALE inbound processing. This generates IDocs in the database and processes them.



Hint: Tip: Refer to the course diagrams while you are working through this exercise.

1. The mapping plan is already created for you. The required mapping rules are in the Excel list. Supply the following target fields in the IDoc:

- Cross-system company code with GL0001
- Document type (conversion: 07 -> SA)
- Document date
- Posting date,
- Translation date
- Old document number
- Currency key (conversion: EU -> EUR, \$ -> USD)

Continued on next page

- Transaction type (RFBU)
- Local currency (EUR)
- Number of posting items within the document (you must create a separate field for this)
- Posting key (conversion: 0004->40 and 0005->50),
- Debit/credit indicator (if greater than 50, set to H; otherwise set to S)
- Amount in local currency
- Amount in document currency
- Document text
- Transaction type for general ledger (RFBU)
- G/L account for general ledger accounting (take account of leading zeros)
- G/L account (see G/L account above)

Task 2:

In the LSMW, in the project BC420-## and the subproject DOCU-##, create the IDOC-## for document conversion.

1. Maintain the object attributes: IDoc with message type *FIDCC1* and basis type *FIDCCP01*.
2. The source structure must consist of the document header HEAD and the document item POSI.
- 3.

Source Fields:

The source fields for the **document header** are:

- SET1(1) Record indicator for header: H
- REF_NO(9) Reference number in the external system
- DOC_DATE(10) Document date **Field type DDMY**
- TYPE(4) Document type,
- POSTING_DATE(10) Posting date **Field type DDMY**
- TRANS_DATE(10) Translation date **Field type DDMY**
- CURR_KEY(2) Document currency

The source fields for the **document item** are:

Continued on next page

- SET2(1) Record indicator for items: P
- POSTING_KEY(004) Posting key
- ACCOUNT(10) Account number,
- DOC_AMOUNT(15) Amount in document currency, field type AMT1
- LOCAL_AMOUNT(15) Amount in local currency, field type AMT1
- TEXT(30) Description of the item.

Task 3:

Structure relations:

1. You want to assign E1FIKPF to the document header and E1FISEG to the document items.

Task 4:

Field mapping:

1. Perform the mapping according to your mapping plan. Consider the special features of the document transfer using IDocs (see the training course material).
2. The document item requires a sequence number. Because this is not provided in the external data, this must be generated in the LSMW. It must be initialized with 0 for each new document header and incremented for each item. Follow the instructions below:

Continued on next page

- Under *Extras → Display Variant*, activate the global data definitions and the processing times.
- Under Global Data, create a counter for the items:

```
DATA: Pos_counter LIKE E1FISEG-BUZEI VALUE 0.
```
- At the event __BEGIN_OF_RECORD__ of the header record, reset the counter to zero.

```
Pos_counter = 0.
```
- At the event __BEGIN_OF_RECORD__ of the item record, increase the counter by one.

```
Pos_counter = Pos_counter + 1.
```
- Assign the field Pos_counter to the field E1FISEG-BUZEI.

```
E1FISEG-BUZEI = Pos_counter.
```

Task 5:

Execute the process step **Specify Files**.

1. Execute the process step **Specify Files**.

Maintain the path and file name: Source file:

BC420_DOC_2_HEAD_POS.LEG

2. Execute the process step **Assign Files**:

Document: BC420_DOC_2_HEAD_POS.LEG

Items: BC420_DOC_2_HEAD_POS.LEG

3. Execute the process step **Import Data** and view the results.
4. Execute the process step **Convert Data** and view the results.
5. Execute the process step **Start IDoc Generation**.
6. If required: Execute the process step **Start IDoc Processing**.

Make a note of the IDoc numbers:

-
7. In the IDoc display (double click) you view the status confirmation messages. Select an IDoc. What is the last status that was created for one of your IDocs? _____
 8. If the IDocs contain an error, correct the error in the LSMW rules and execute the relevant steps in the LSMW again.

Continued on next page

9. Make a note of the first five document numbers.

10. You can use transaction FB03 to view the documents.

Solution 14: Transferring FI Documents Using Direct Input

Task 1:

The FI documents are available in the file BC420_DOC_2_HEAD_POS.LEG in the transport directory. This file contains five documents, each with three items. You want to use the LSMW to transfer this data to the relevant IDoc structure. You then want to transfer this file to ALE inbound processing. This generates IDocs in the database and processes them.



Hint: Tip: Refer to the course diagrams while you are working through this exercise.

1. The mapping plan is already created for you. The required mapping rules are in the Excel list. Supply the following target fields in the IDoc:
 - Cross-system company code with GL0001
 - Document type (conversion: 07 -> SA)
 - Document date
 - Posting date,
 - Translation date
 - Old document number
 - Currency key (conversion: EU -> EUR, \$ -> USD)
 - Transaction type (RFBU)
 - Local currency (EUR)
 - Number of posting items within the document (you must create a separate field for this)
 - Posting key (conversion: 0004->40 and 0005->50),
 - Debit/credit indicator (if greater than 50, set to H; otherwise set to S)
 - Amount in local currency
 - Amount in document currency
 - Document text

Continued on next page

- Transaction type for general ledger (RFBU)
 - G/L account for general ledger accounting (take account of leading zeros)
 - G/L account (see G/L account above)
- a) Note the following: Familiarize yourself with the mapping plan. You can also check the instructor solution later. Keep in mind that the aim is not to supply all mapping fields 100% without errors in one go. Instead, the aim is to understand the functions and procedure of the LSMW when working with IDocs. Errors are actually good in this exercise. Take a note of the number of an incorrect IDoc so you can correct this error later.

Task 2:

In the LSMW, in the project BC420-## and the subproject DOCU-##, create the IDOC-## for document conversion.

1. Maintain the object attributes: IDoc with message type **FIDCCI** and basis type **FIDCCP01**.
 - a) Carry out this step in the usual manner.
2. The source structure must consist of the document header HEAD and the document item POSI.
 - a) Carry out this step in the usual manner.
- 3.

Source Fields:

The source fields for the **document header** are:

- SET1(1) Record indicator for header: H
- REF_NO(9) Reference number in the external system
- DOC_DATE(10) Document date **Field type DDMY**
- TYPE(4) Document type,
- POSTING_DATE(10) Posting date **Field type DDMY**
- TRANS_DATE(10) Translation date **Field type DDMY**
- Curr_KEY(2) Document currency

The source fields for the **document item** are:

Continued on next page

- SET2(1) Record indicator for items: P
 - POSTING_KEY(004) Posting key
 - ACCOUNT(10) Account number,
 - DOC_AMOUNT(15) Amount in document currency, field type AMT1
 - LOCAL_AMOUNT(15) Amount in local currency, field type AMT1
 - TEXT(30) Description of the item.
- a) Proceed as usual or copy the field definitions from the instructor's project. The instructor will also give you instructions.

Task 3:

Structure relations:

1. You want to assign E1FIKPF to the document header and E1FISEG to the document items.
 - a) Proceed in the usual manner.

Task 4:

Field mapping:

1. Perform the mapping according to your mapping plan. Consider the special features of the document transfer using IDocs (see the training course material).
 - a) Use the mapping plan or the instructor solution as a guide.
2. The document item requires a sequence number. Because this is not provided in the external data, this must be generated in the LSMW. It must be initialized with 0 for each new document header and incremented for each item. Follow the instructions below:

Continued on next page

- Under *Extras → Display Variant*, activate the global data definitions and the processing times.
 - Under Global Data, create a counter for the items:

```
DATA: Pos_counter LIKE E1FISEG-BUZEI VALUE 0.
```
 - At the event `__BEGIN_OF_RECORD__` of the header record, reset the counter to zero.

```
Pos_counter = 0.
```
 - At the event `__BEGIN_OF_RECORD__` of the item record, increase the counter by one.

```
Pos_counter = Pos_counter + 1.
```
 - Assign the field `Pos_counter` to the field `E1FISEG-BUZEI`.

```
E1FISEG-BUZEI = Pos_counter.
```
- a) Use the “recipe in the folder” as a guide. All relevant points are displayed on a slide. At a later point, you can check whether your work was successful under “Display converted data”.
If the counter is missing from the field `BUZEI`, the system displays red traffic lights during IDoc processing. This is also good for practice purposes. Good luck!

Task 5:

Execute the process step **Specify Files**.

1. Execute the process step **Specify Files**.
Maintain the path and file name: Source file:
`BC420_DOC_2_HEAD_POS.LEG`
a) Proceed in the usual manner.
2. Execute the process step **Assign Files**:
Document: `BC420_DOC_2_HEAD_POS.LEG`
Items: `BC420_DOC_2_HEAD_POS.LEG`
a) Proceed in the usual manner.
3. Execute the process step **Import Data** and view the results.
a) Proceed in the usual manner.

Continued on next page

4. Execute the process step **Convert Data** and view the results.
 - a) Proceed as usual and make samples of the fields to be converted:
Does the cross-system company code match? Is the fiscal year filled correctly? Is the posting item for each document segment incremented? Do you agree with the conversion of the different date fields?
5. Execute the process step **Start IDoc Generation**.
 - a) Proceed as usual and execute this step by double clicking it or selecting it and choosing F8. The system starts the IDoc generation. If you choose “Trigger Immediately”, you can ignore the step “Start IDoc Processing”. You can check the result in LSMW or, if outside the LSMW, you can check the result using transaction WE05.
6. If required: Execute the process step **Start IDoc Processing**.
Make a note of the IDoc numbers:

- a) You can check the resulting IDocs within the LSMW by choosing “Create IDoc Overview”; you can also do this from outside the LSMW (WE05).
7. In the IDoc display (double click) you view the status confirmation messages. Select an IDoc. What is the last status that was created for one of your IDocs?

- a) Proceed as described on the course slides. Tip: To check the details of the generated IDocs, completely expand all elements of the tree structure on the left. In a small control window at the bottom right edge of the IDoc list overview, the system displays the field contents of the IDoc fields.
8. If the IDocs contain an error, correct the error in the LSMW rules and execute the relevant steps in the LSMW again.
 - a) For information about correcting the error, see the slides in the folder. It helps if you double click the white sheet before the incorrect IDoc segments. On the following screen, switch to change mode using the menu. You can correct incorrect data there.
9. Make a note of the first five document numbers.

- a) After the corrections, the system transfers the incorrect documents to the application again. You must make a note of the numbers of your IDocs.

Continued on next page

10. You can use transaction FB03 to view the documents.
 - a) Start transaction FB03 and analyze the generated document numbers if the IDoc processing was successfully.



Lesson Summary

You should now be able to:

- Describe data exchange using IDocs
- Transfer data in LSMW using IDoc technology
- Perform the required administrative steps for IDoc transfer



Unit Summary

You should now be able to:

- Describe data exchange using IDocs
- Transfer data in LSMW using IDoc technology
- Perform the required administrative steps for IDoc transfer

Unit 9

BAPI

Unit Overview

You can also use BAPIs to transfer data to the SAP system. BAPIs that are used in the data transfer procedure use the IDoc interface.



Unit Objectives

After completing this unit, you will be able to:

- Describe the transfer of external data using BAPIs.
- Use BAPIs to transfer asset master data to the SAP system.

Unit Contents

Lesson: BAPI.....	284
Exercise 15: Transfer of Asset Master Data Using BAPIs	305
Exercise 16: Transfer of Asset Master Data Using BAPIs	309

Lesson: BAPI

Lesson Overview

Contents:



- BAPI Basics
- Asset Master Basics
- Transfer of Assets Using BAPIs



Lesson Objectives

After completing this lesson, you will be able to:

- Describe the transfer of external data using BAPIs.
- Use BAPIs to transfer asset master data to the SAP system.

Business Example

A company wants to transfer external data in an application that provides a BAPI for the transfer.

Data Transfer Using BAPI



Figure 230: BAPI

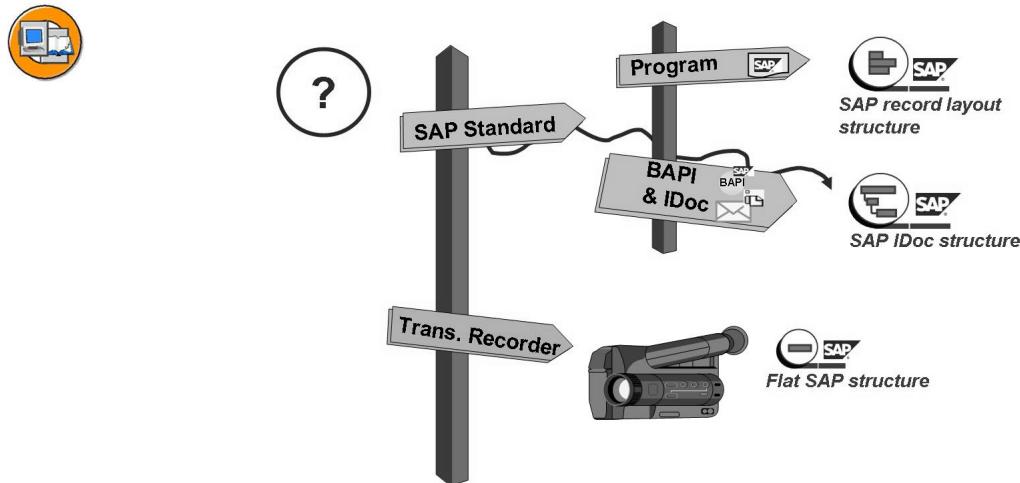


Figure 231: Overview: Standard Transfer Using BAPI

This unit covers the transfer of external data using BAPIs.

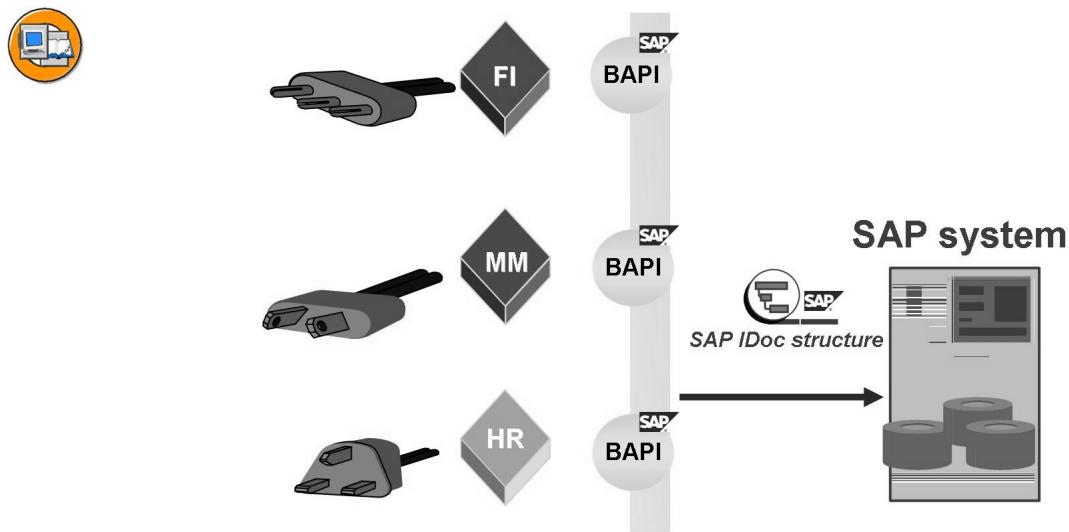


Figure 232: Interfaces for Standard Transfer Using BAPI

To call BAPIs from DX-WB, you require the external data in IDoc format. Therefore, you can use a mapping tool (for example LSMW) to convert the external data to IDoc format.

→ **Note:** The following is a short introduction to the concept of BAPIs. Keep in mind that there is a specific course for this topic (BC417).

Business Application Programming Interface



- Business Application Programming Interface
- A BAPI is a well defined **interface** for the processes and data of a **business** application system, implemented as the **methods of an object** in the **Business Object Repository (BOR)**.

Each BAPI that creates an instance of an object or changes the data of an object leaves a consistent database status. All database changes are performed completely or not at all.

The BAPI itself cannot execute the “COMMIT WORK” command to write the changes to the database. Instead, the calling program must execute this command.

BAPIs are methods without dialog that **do not conduct any screen dialog** with the user.

Characteristics of BAPIs



- **Object-oriented**
 - BAPIs are implemented as object methods from the BOR.
- **Stable interface**
 - The BAPI interface is “frozen”.
- **There is no COMMIT WORK in the BAPI.**
 - There is a performance advantage for update BAPIs.
- **Do not have a presentation layer**
 - The caller itself visualizes results.
- **Confirmation of errors or success by using export parameter RETURN**
 - Structure or internal table (depending on the BAPI)

You can access BAPIs from external clients like methods access objects, which are themselves an instance in the client.

SAP guarantees that no incompatible changes will be made to the interfaces of an SAP BAPI during a release upgrade.

You can also use BAPIs within the R/3 System to access business data. This makes it easier for customers and partners to enhance the system functionality.

Displaying data that is transmitted to a BAPI or that is returned from a BAPI must be done from the calling program.

The **return parameter RETURN** contains success or error messages for the BAPI, and, depending on the release, has the structure of the dictionary structure: BAPIRETURN, BAPIRETURN1, BAPIRET1, BAPIRET2 or BAPIRET2_FIX. The (common) relevant fields of these structures are:

TYPE (Message type: S(uccess), E(rror), W(arning), I(nformation)); ID (message class); NUMBER (message number); MESSAGE (message text); MESSAGE_V1, MESSAGE_V2, MESSAGE_V3, MESSAGE_V4 (message variables)

“Write” BAPIs that make a database change work with update techniques and, as of R/3 4.0, do not execute the required Commit Work for the database update themselves. As a result, several BAPI calls can be bundled in one LUW (logical unit of work) to ensure high performance.

→ **Note:** As an example of a transfer using BAPI, transfer asset data (“assets”). You can illustrate the business object BUS1022 using the BO data model. Brief documentation about “Asset” is also available there. Path: *Tools → ABAP Workbench → Development → Business Object Builder*.

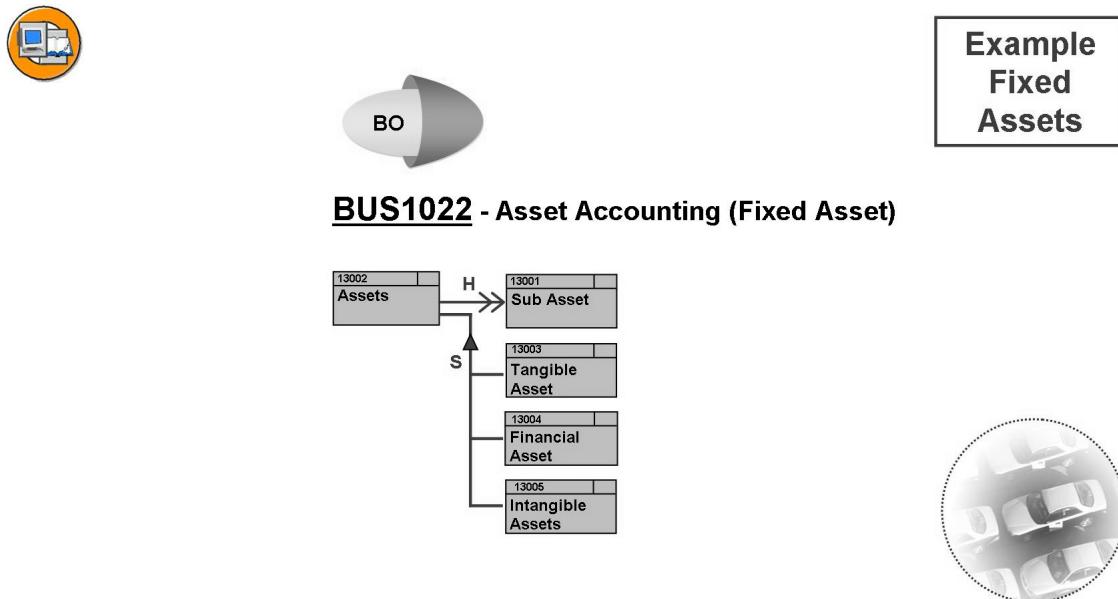


Figure 233: The Business Object “Asset”

Display BUS1022, choose BO Data Model, then choose Utilities → Graphic.

The business object “Fixed asset” is an object, a right or another item of the enterprise's assets which is intended for long-term use, and can be individually identified on the balance sheet.

The development of the values of fixed assets can be viewed as a whole, or broken down into the component parts (sub-numbers) of assets. For example: Fixed asset “printing press” with fixed asset components “rollers” and “replacement spindle”.

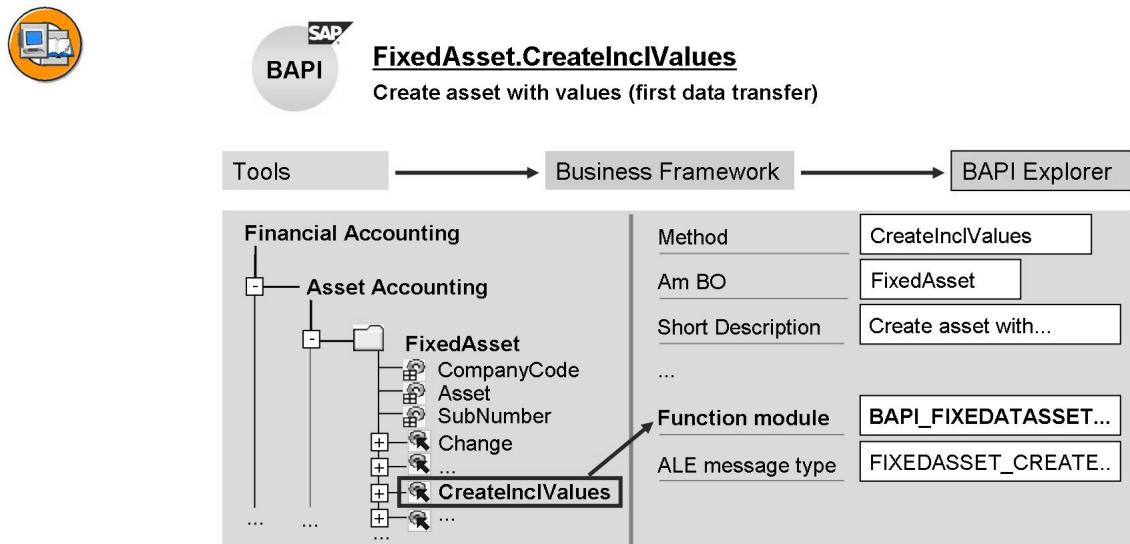


Figure 234: The BAPI "CreateInclValues"

One of the methods of the business object BUS1022 is the method **CreateInclValues**.

It is used to transfer assets to the R/3 system.

You can use the **BAPI Explorer** or the **Business Object Builder** described above to display the function module BAPI_FIXEDASSET_OVRTAKE_CREATE that converts the BAPI.

Path: *Tools → Business Framework → BAPI Explorer; Financial Accounting → Asset Accounting → FixedAsset; Select FixedAsset.*

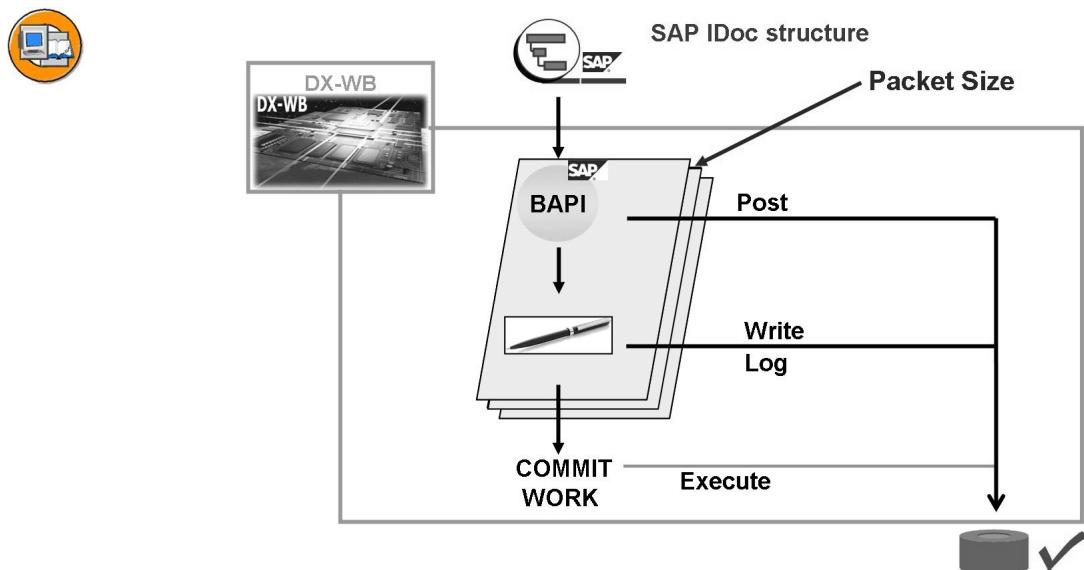


Figure 235: BAPI Processing in DX-WB

If you want to use a BAPI to transfer external data, you can do this by using the DX-WB.

The DX-WB expects the external data in a file in IDoc format. The individual steps within DX-WB are as follows:

- The system reads the external data package by package.
- The package is transferred to ALE inbound processing.
 - In ALE inbound processing, the data is transferred to the BAPI, record by record.
 - The return parameters returned by the BAPI are returned to DX-WB in bundles.
- DX-WB evaluates the returned status record package.
- The system writes a log.
- If there are any errors, either an error file is created or IDocs are generated in the system directly. (This can be set when creating the task in DX-WB).
- Finally, DX-WB executes a COMMIT WORK to complete the whole process consistently.

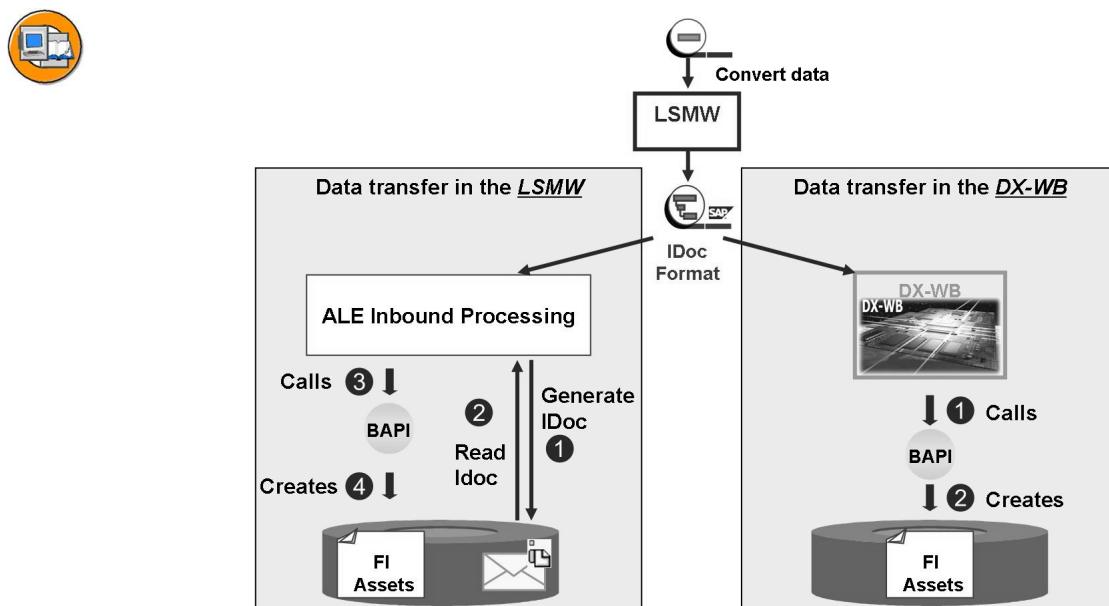
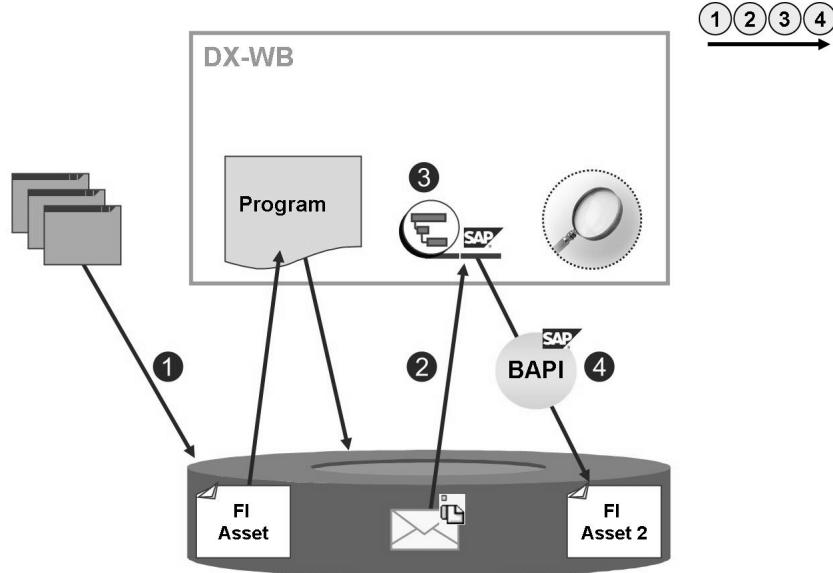


Figure 236: BAPI Technology: LSMW or DXWB?

The file in IDoc format can be processed both in LSMW and in DX-WB.

As the screen clarifies, in the case on the left side of the screen (LSMW), an IDoc is created in the database (DB), it is then read and transferred to the BAPI.

In BX-WB, the file is directly transferred to the BAPI in IDoc format. In this case, no IDoc is generated in the database, which accelerates the transfer.

**BAPI Basics****Test Data for the Asset Master Record****Transfer Assets Using BAPIs****Figure 237: BAPI****Figure 238: Generate and Process Test Data**

For certain objects, you can create test data for BAPIs using DX-WB. The file contains the data in IDoc format.

For this, use the online transaction to create a data record in the SAP system (1).

Then create a file with test data. Use the function *From Test Program* in the tools of DX-WB. When you do this you call an application program by specifying the data record you created previously (for example, with company code and document number).

This program reads the relevant data record from the database (2) and generates an outbound IDoc in the database (3). The DX-WB triggers the ALE outbound processing. This generates a file in IDoc format for outbound processing according to the partner profile (4). BX-WB copies this file (5) and changes the sender and receiver so that this file can be used for ALE inbound processing.

This file can now be used for testing. It simultaneously serves as a template of how the data from the external system should be formed.



Hint: Important: So that the file can be created, DX-WB requires a logical system. As of Release 4.6C, DX-WB creates this automatically. In Release 4.6A/B, users themselves must create this logical system and the relevant partner profiles in the SAP system.

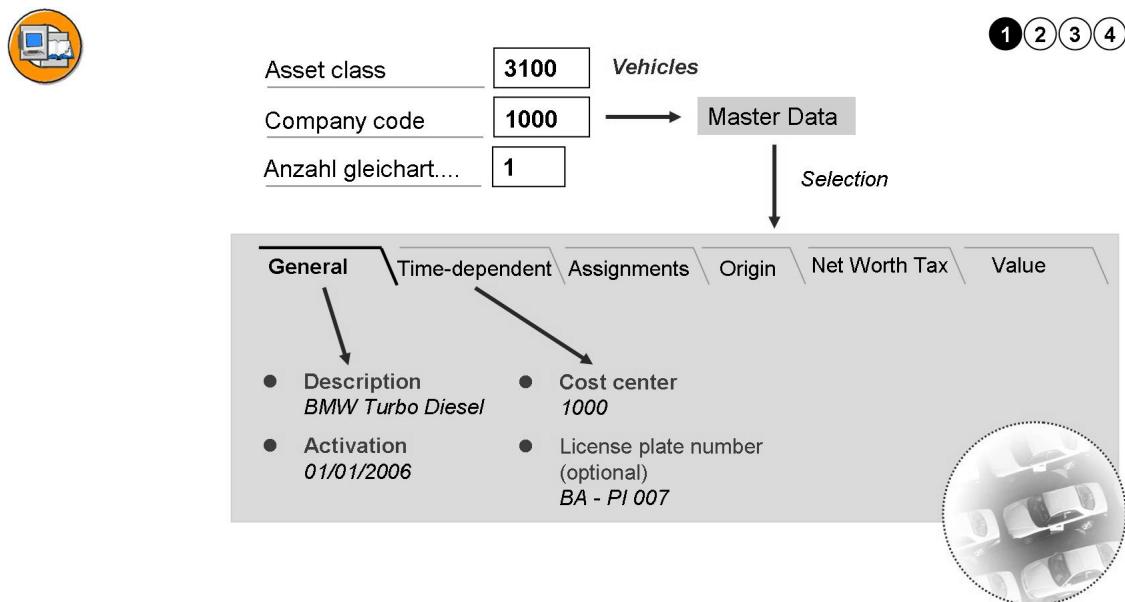


Figure 239: Online: Creating a Vehicle (Vehicles)

To go to the asset master data, choose *Accounting* → *Financial Accounting* → *Fixed Assets* → *Asset*.

The slide demonstrates the creation of an asset (vehicle) in the asset class 3100 (vehicles). The parameters *Description* and *Cost Center* are mandatory.

Description: Description of the asset.

Activation: In this field, the system saves the asset value date of the first posting that resulted in the asset activation. The capitalization date is the value date of an asset. You can also specify it manually when you create an asset master record. However, the asset is not activated; instead is suggested only as an asset value date when the first acquisition posting is made. (This is required for the data transfer.)



Cost center: The system uses the cost center assignment in the asset master record to determine the relevant cost center for the subsequent postings assignable to accounts on the cost center.



Hint: Important: After you save the asset, the **asset number** appears in the status line. To generate a test data record for assets from DX-WB, we recommend that you make a note of this number and use it again at a later stage.

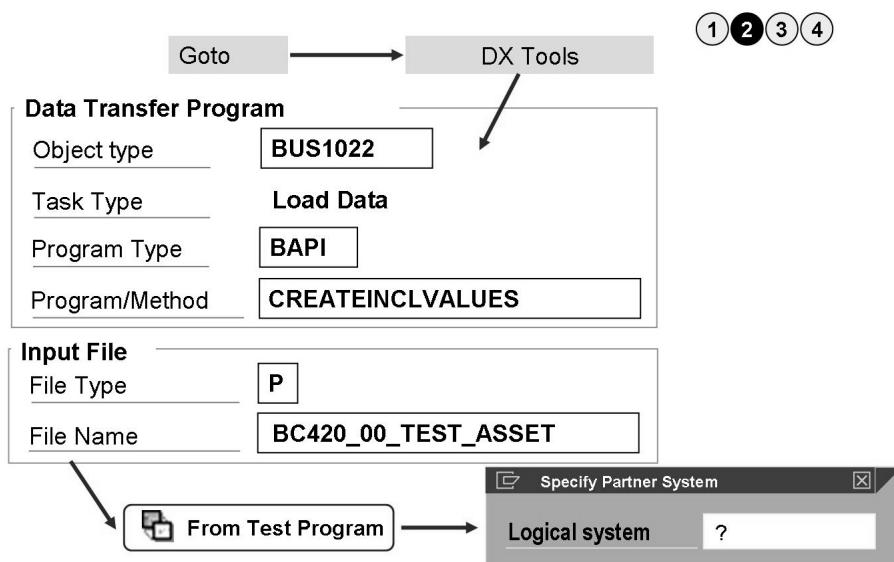


Figure 240: Generate Test Data in IDoc Format

In DX-WB, you can choose *Goto → DX Tools* to generate test data in IDoc format. After you choose *From Test Program*, the system prompts you to enter a logical system.

As of R/3 4.6C, DX-WB itself creates the logical system and the required partner profiles.

Before R/3 4.6C, this must be done manually.

Definition of a “Logical System”:

The distribution of data between systems makes it necessary to uniquely identify each system within a network. The “logical system” fulfills this requirement. A logical system is an application system within which the applications work together on a shared data basis. In the ALE environment, a client is assigned to a logical system.



→ Display File

① ② ③ ④

Structure		Contents
EDI_DC40	Ktr1.satz	EDT_DC40 40000000146A 24 FIXEDASSET..
EDI_DD40	Datensatz	E2FIXEDASSET_CREATEINCLVALU000400..
EDI_DD40	Datensatz	E2BP1022_KEY000 400 000002000001021000
EDI_DD40	Datensatz	E2BP1022_FEGLG001X000 400 0000030000010200003100BMW
EDI_DD40	Datensatz	E2BP1022_FEGLG001X000 400 00000400000102 XX XXXXX
EDI_DD40	Datensatz	E2BP1022_FEGLG011X000 400 0000050000010200000000
EDI_DD40	Datensatz	E2BP1022_FEGLG011X000 400 00000600000102XXX
EDI_DD40	Datensatz	E2BP1022_FEGLG020000 400 000007000001022006050100...
EDI_DD40	Datensatz	E2BP1022_FEGLG002X000 400 00000800000102XX X
EDI_DD40	Datensatz	E2BP1022_FEGLG0030000 400 0000090000010219000101999..
EDI_DD40	Datensatz	E2BP1022_FEGLG003X000 400 00001000000102 XXXXXXX...
...		

The first part of the record contains the segment information.

The application files start here!

→ Double-click record 4 →

Figure 241: Display Test Data

The slide displays a file in IDoc format with a sample asset from the asset class 3100 (vehicles).



Field Name Short Description Field Contents

① ② ③ ④

SEGNAM	Segment
MANDT	Client
DOCUUM	Number of the IDoc
SEGNUM	Segment number
PSGNUM	No. of the parent segment
HLLEVEL	02
SDATA	Application data

Double-click this icon to display the application file in more detail:

ASSETCLASS	3100
DESCRIPT	BMW-Turbo Diesel
DESCRIPT2	...

Figure 242: Detailed Display of the Test Data

The application data of the fourth segment contains, for example, the asset description. The segment contents can be examined in detail if you choose the magnifying glass icon.



Object type BUS1022
Method CREATEINCLVALUES
Packet Size 1
Errors

Number of objects that are transferred in an LUW to the BAPI

1 2 3 4

Parallel Loading in Server Group
 Sequential Loading
 Delete Input Files After Processing
 Log Error Messages Only

Objects That Could Not Be Transferred
 Write to Error Files
 Store as IDoc in R/3

Test file created

Type	Name
P	BC420_00_TEST_ASSET

Input Files

Type	Name
P	ASSETS_ERROR_00

Error Files

Incorrect data records can be stored individually or in files

Figure 243: Maintain Run Attributes

The run attributes for data transfer using BAPIs are:

Packet Size: This determines the number of objects that a BAPI transfers per LUW.

Input file: Transfer file specification with input data.

Objects that could not be transferred: Select whether you want to store incorrect objects in a file or as an IDoc in the SAP system directly.

The other parameters are explained in the next section of this unit.



BAPI Basics

Test Data for the Asset Master Record



Transfer Assets Using BAPIs

Figure 244: BAPI

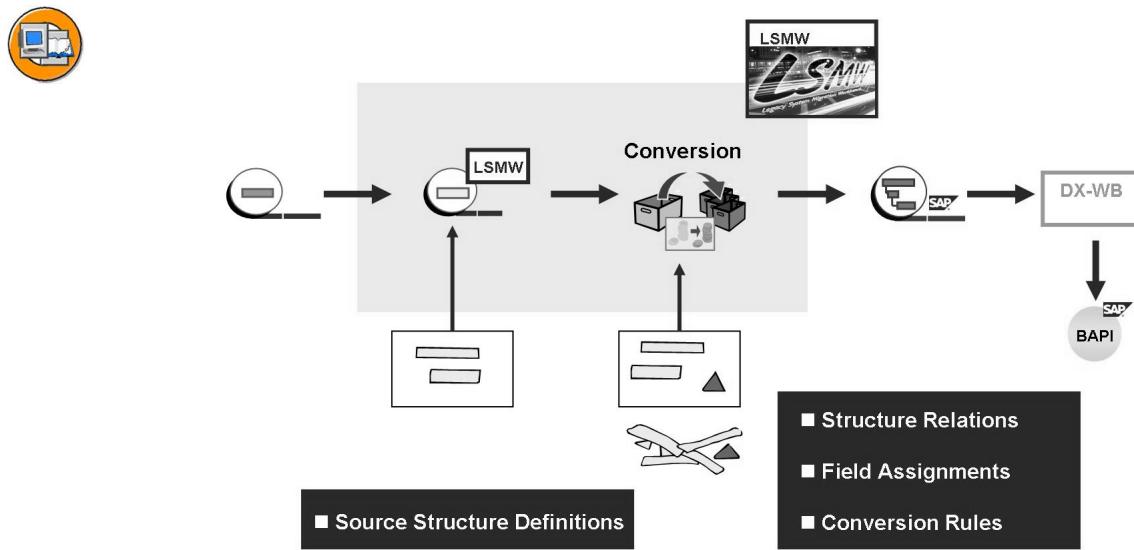


Figure 245: Mapping Program (LSMW)

When transferring data using BAPIs, the LSMW should be used as a mapping tool only. The file created contains the data in the required IDoc format. The DX-WB can then transfer this data to the SAP system by calling the BAPI.

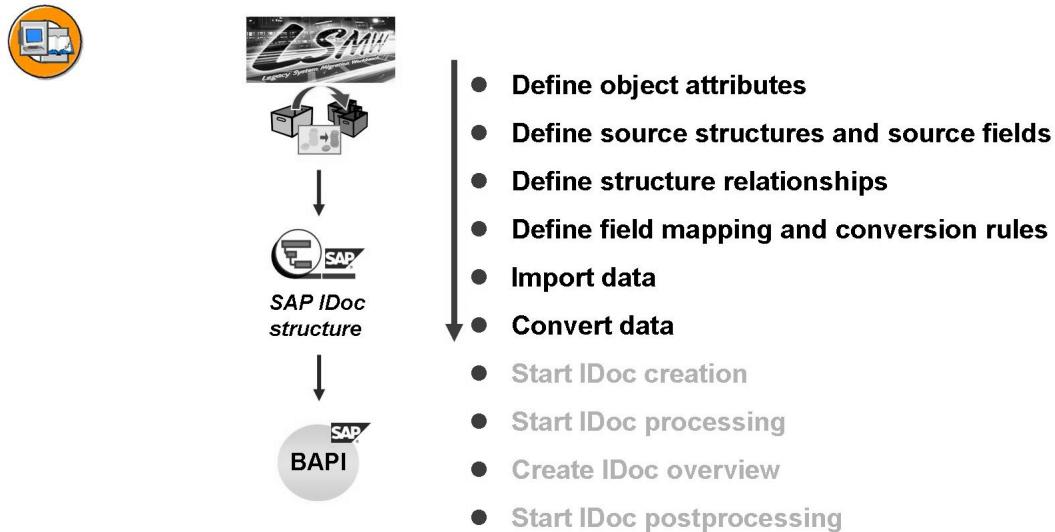


Figure 246: Process Steps for Asset Transfer

The work steps for IDoc transfer as the same here as for the transfer using BAPIs.

In LSMW, the data is now converted to IDoc format. The result is a file that is transferred from a BAIP using DX-WB.

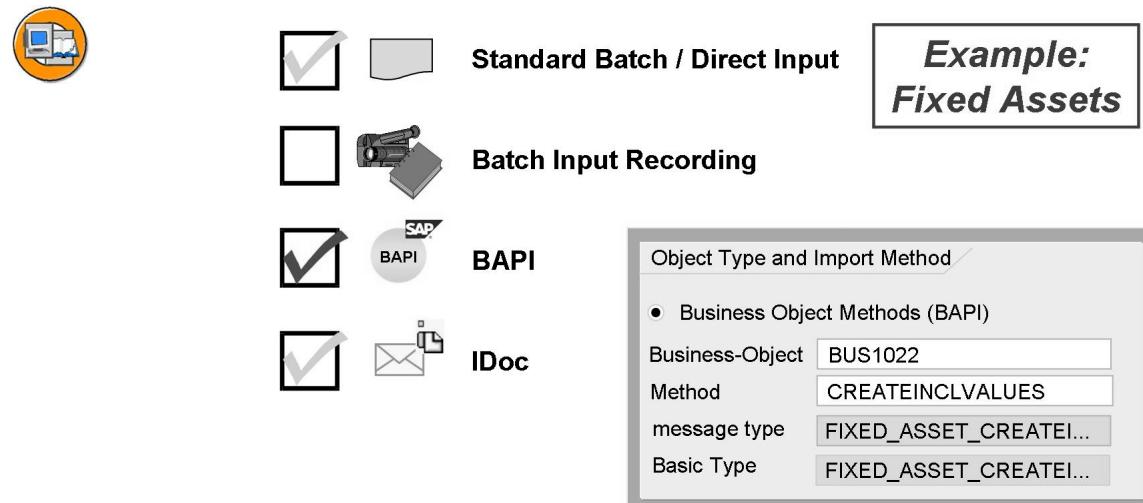


Figure 247: Object Type and Import Method BAPI

When the business object and the BAPI are selected, the message type (and basis type) are determined automatically by the LSMW.

A partner profile is also required for creating the data. If a partner profile for the project has already been maintained in the IDoc settings, the appropriate message type is also entered in the IDoc settings.

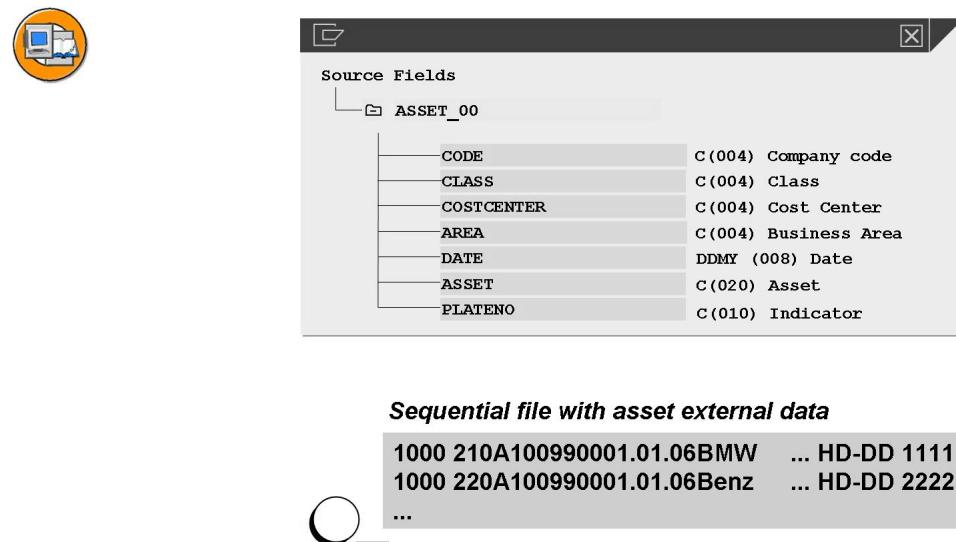


Figure 248: Example: Structure of External Data

The the asset external data should have the structure shown in the example above.
Sample External Data Records:

Company code: 1000
 Asset class: 210 or 220
 Cost Centers: A100
 Business area: 9900
 Date: 01.01.2006
 License plate number: HD-DD 1111 ...



- **Asset class** (*Conversion, leading zeros*)
210 → 00003100
220 → 00003100
- **Cost center** (*Conversion*)
A100 → 1000
- **Company code** (*Transfer*)
- **Capitalization date** (*Transfer*)
- **Business Area** (*Transfer*)
- **Fill X Structures** (*Constant = 'X'*)
these fields indicate the change of the fields.

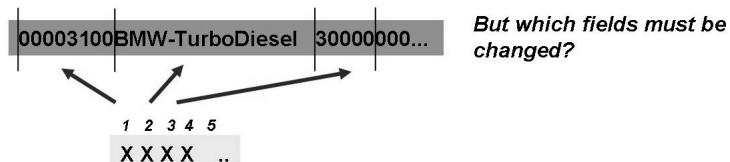


Figure 249: Special Features and Mapping Rules for External Fixed Asset Master Data

For the asset class field and the cost center field, you must define conversions.

For the fields *Company Code*, *Capitalization Date*, *Business Area*, you must set only transfers (MOVE).

Special Fields:

After each segment with application data, there follows a segment with a change indicator for this application data. If, for example, you want to change the third field of an application segment, the relevant third field of the subsequent X segments must contain an “X”.

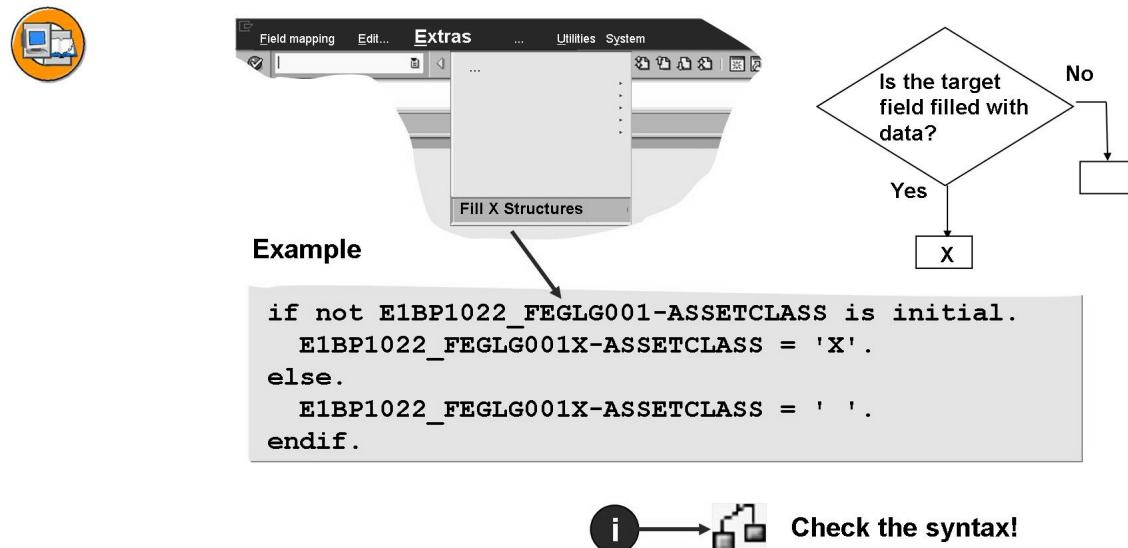


Figure 250: LSMW: Fill X Structures

Some BAPIs have X structures (X segments). The X field of this X structure must be filled if the relevant application data target field is filled and you want to use it to transfer data.

LSMW supports the filling of this structure. If you use the function *Complete X Structures*, the system generates the ABAP source code displayed above in each field of an X structure.

If a rule is already maintained in a field of an X structure, no ABAP source code is generated for this field.

X structure are filled automatically for the field names of the relevant X structure. However, if this is not identical to the name of the relevant data structure, the generated source code is not syntactically correct. In this case, LSMW issues a message.

You can check this by using the function *Syntax Check* in the process step *Maintain Field Mapping and Conversion Rules*. The system then creates and checks the conversion program. If there is an error, the system displays the conversion program. Execute the syntax check again to display the exact point where the error occurred.

→ **Note:** Correct the error in the field mapping, not in the generated program.

→ **Note:** If you work with Unicode character sets, see the following SAP Note: 568291

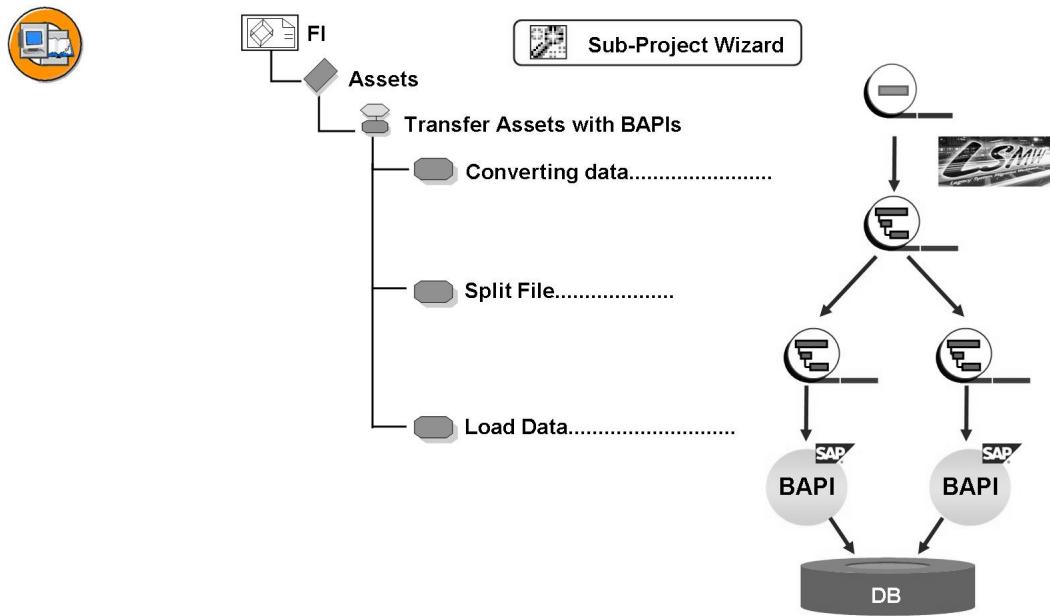


Figure 251: Creating Runs with BAPIs

DX-WB with the program type BAPI, a BAPI can be used for the data transfer. Special attributes need to be defined (see next few diagrams).

A complete run consists of converting the external data, splitting the IDoc file created for the transfer (this is an option but is not compulsory), and processing the external IDoc files.

Subproject - wizard:

If the subproject wizard is used, some attributes are copied from the previous task steps.

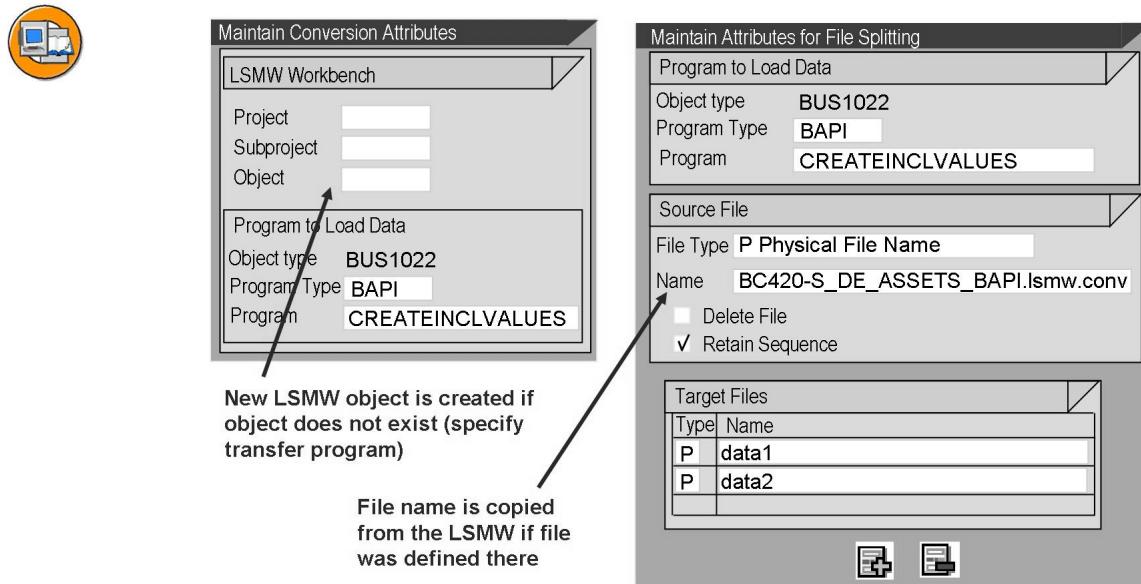


Figure 252: Converting and Splitting the File

The IDoc file created in the conversion can be split up, and these files can be processed in parallel. Primarily, the names of the target files must be specified.

If the subproject wizard is used, with the first step (convert data) in particular, an appropriate LSMW object that has the correct attribute object settings, can be *created* when you select the subsequent transfer method. All further work steps of the LSMW object must be defined as normal. This procedure assumes that the work steps were defined in the DX Workbench and then the conversion was defined in the LSMW.

If an existing object is used, and no new object created, the name of the external file is used in the next work step (split file).

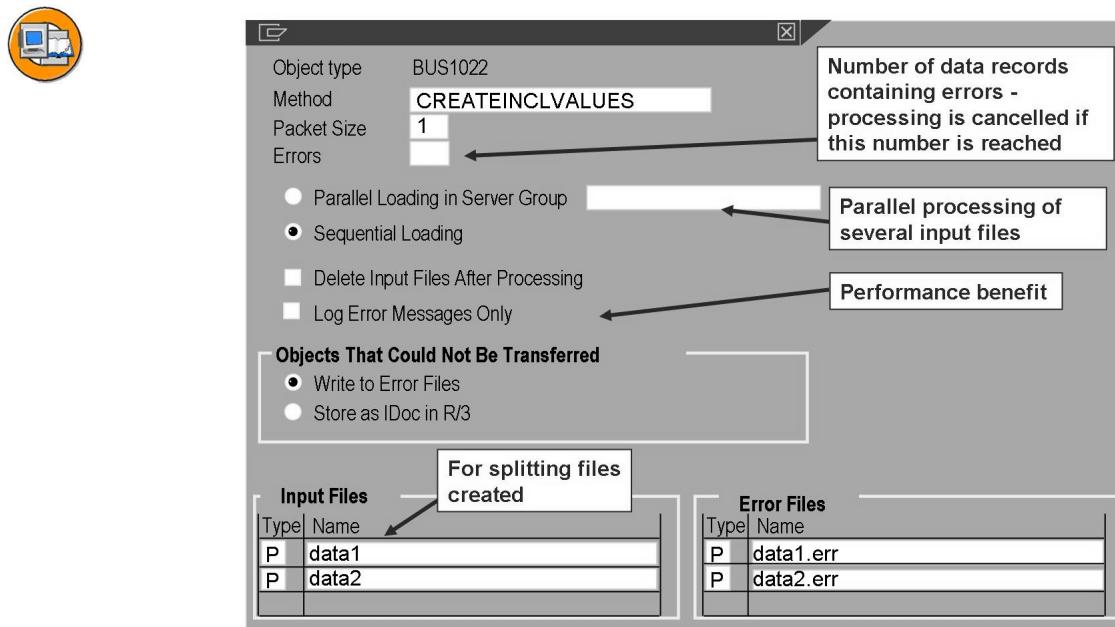


Figure 253: Maintain Run Attributes

The run attributes for data transfer using BAPIs are:

Errors Before Transfer is Cancelled: If when the task is executed more objects are posted than specified by the BAPI, processing is cancelled. If the initial value is left in the field or a zero is entered in it, the task is not cancelled, regardless of how many objects from the file have not been posted.

Delete File: Once the data has been transferred from this file, the file is deleted.

Parallel Loading: If you want to import more than one input file, you can import them in parallel. To do this you must specify a server group that has already been defined. Groups are defined in the RFC server group maintenance of the CCMS (transaction RZ12).

Log Error Messages Only: If you suppress success messages, system performance is improved.

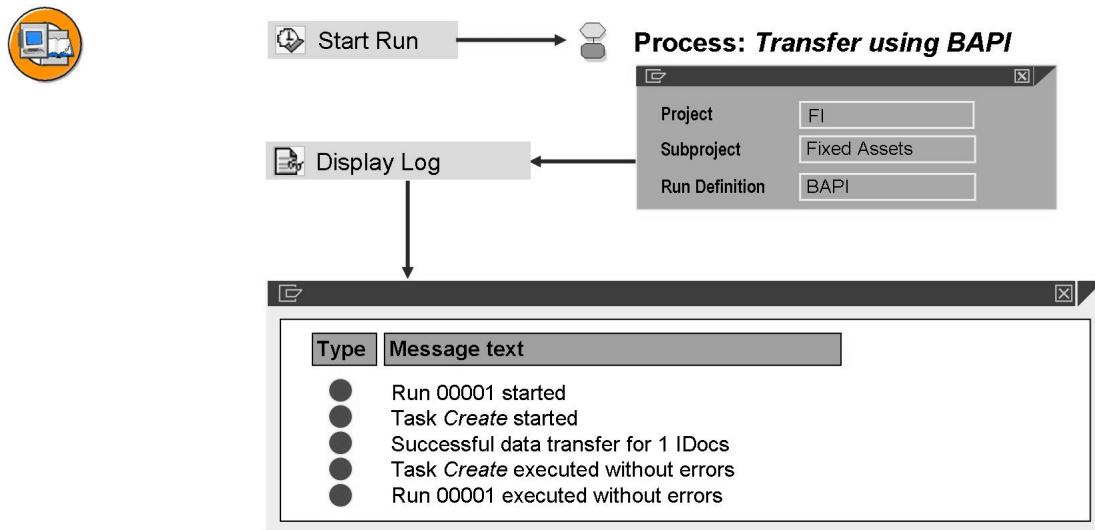


Figure 254: Transferring the Fixed Asset Master Data

After the run, a log is available.

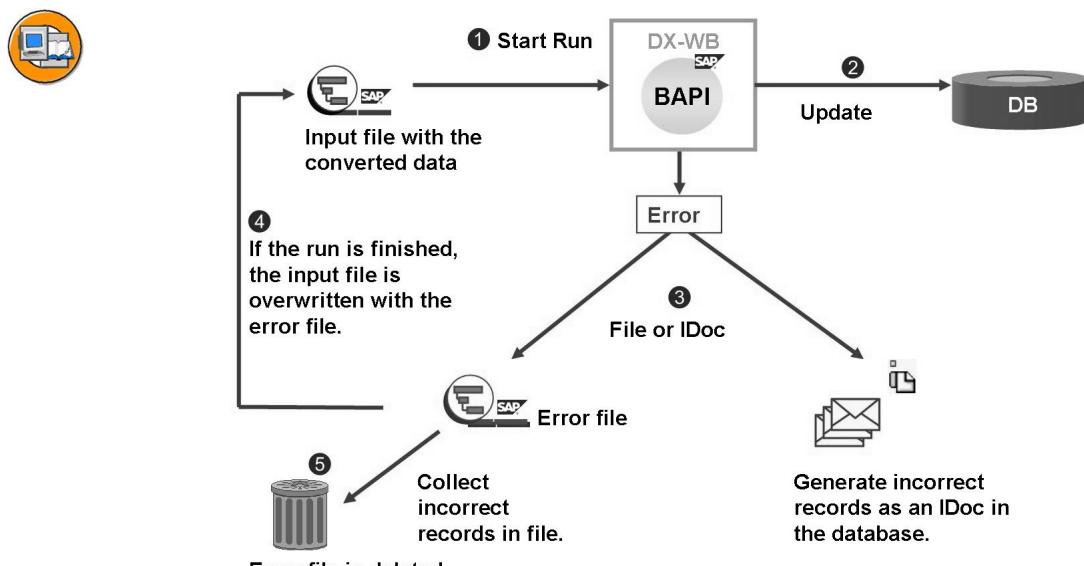


Figure 255: BAPI Error Handling

When transferring data using BAPIs, DX-WB offers two alternative methods of handling errors.

Error handling using an error file.

- For this purpose, you must select the option *Write IDoc to File* in DX-WB.
- During the run, the system reads the source file with the converted data (1). The BAPI updates all records without errors on the database (2). All records with errors are collected in the error file (3). If the run is completely finished (all records were read from the source file or an incorrect run was terminated manually) the system copies the error file to the input file (4). To be exact, the system first deletes the input file and then copies the error file. After successfully copying the file, the system deletes the error file (5).

Error Handling by Creating IDocs:

- For this purpose, you must select the option *Store IDoc in R/3* in DX-WB.
- During the run, the system reads the source file with the converted data (1). The BAPI updates all records without errors on the database (2). All incorrect records are created as IDocs in the database (3). When the run is completely finished, the system reads the source file completely and must use the ALE services to process the IDocs further.
- In this case, you can select the function *Delete File* because all data records from the input file were processed if the run was completed.

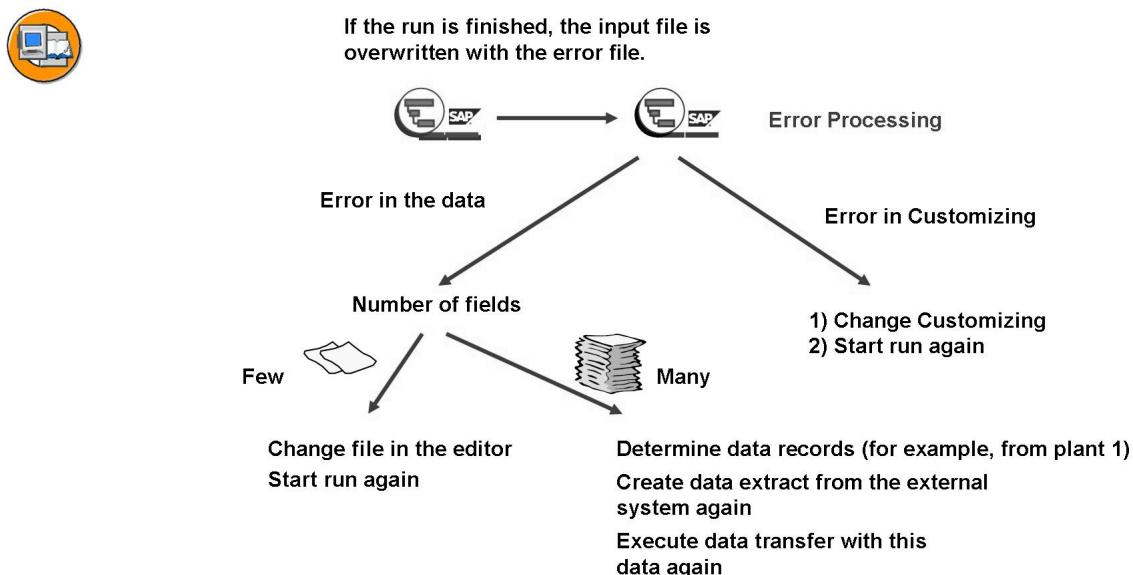


Figure 256: Error Handling with Error File

If errors occurred during the data transfer and the incorrect records were collected in the error file, the procedure depends on the type of error.

If there is an error in Customizing, correct this and restart the run by choosing *Continue Run*. The input file contains only the incorrect records that are not processed and the system processes only these records.

If there is an error in the data, the procedure depends on the number of errors.

- If a small number of data records have an error, you can change the data using the DX-WB editor. (The input file contains only the incorrect records that are not processed.) Then restart the run by choosing *Continue Run*.
- If there are many incorrect data records, it is not useful to work with the editor because this is very time-consuming. If there are systematic errors, for example all records from plant 1 or company code 1000, it is better to execute a new run with these records. For this purpose, you must recreate the extract for the incorrect data only. Then convert the data and correct the systematic error. Then restart the run using this file.

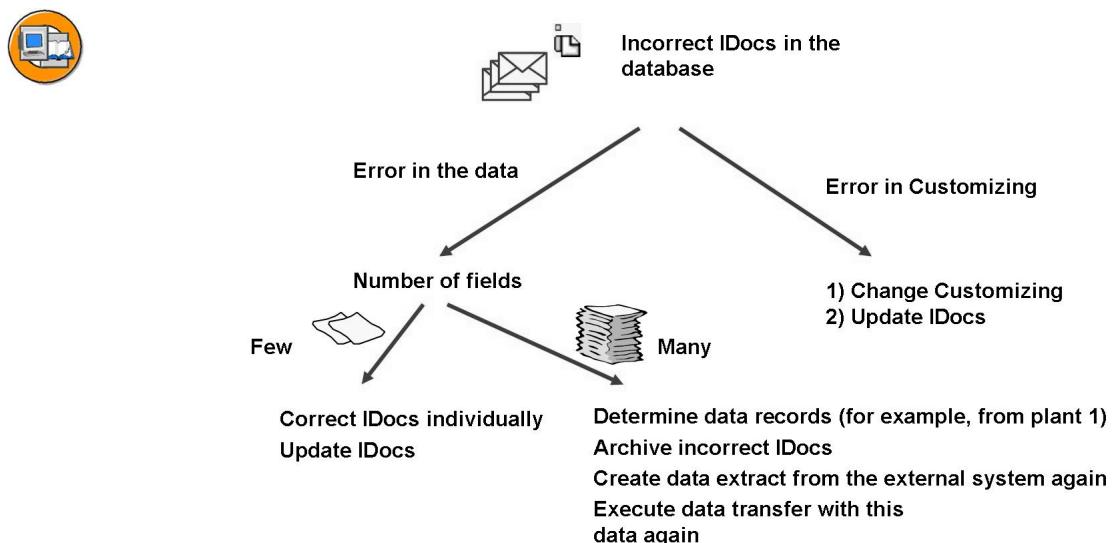


Figure 257: Error Handling Using IDocs

If errors occurred during the data transfer and the incorrect records were stored as IDocs in the database, the procedure depends on the type of error.

If there is a Customizing error, correct this error and update the IDocs.

If there is an error in the data, the procedure depends on the number of error.

- If a small number of data records have an error, you can change the IDocs using the IDoc editor. Then update the IDocs.
- If there are many incorrect data records, it is not useful to change the IDocs using the editor because this is very time-consuming. If there are systematic errors, for example all records from plant 1 or company code 1000, it is better to execute a new run with these records. For this purpose, you must recreate the extract for the incorrect data only. Then convert the data and correct the systematic error. Then restart the run using this file.

Exercise 15: Transfer of Asset Master Data Using BAPIs

Exercise Objectives

After completing this exercise, you will be able to:

- Basics of BAPI technology
- Generate installation test data
- Data transfer of installation data

Business Example

Transfer the external FI installation data to the SAP system using a BAPI. First you must familiarize yourself with BAPI transfer technology.



Hint: Tip: Refer to the course diagrams while you are working through this exercise.

Task 1:

Create asset master data online (transaction: AS91).

1. Create asset master data online (transaction: AS01). For data transfer using batch input, AS91 is actually used for Fixed Asset Master Data. To analyze the previous case of a transfer using BAPI, it is enough to use AS01. (If you have the required authorization for AS91, you can also use this transaction.)

Asset class: 3100

Company code: 1000

Name: Racing car

Capitalization date: 01.01.2006

Cost center 1000

License plate number: BA-PI 007

Write down the asset number: _____

Task 2:

Open the DX-WB, go to the DX Tools (menu *Goto*) and create the asset test data using the relevant button in the menu bar.

1. Object Type: BUS1022

Continued on next page

Program type: BAPI

Program: CREATEINCLVALUES

File type: P (Physical File Name)

File name: **BC420_##_TEST_ASSET.SAP**

(Button “From Test Program”)

Task 3:

In a new subproject in the DX-WB project **FI-##**, create a new run that will process the test file you created using a BAPI.

1. In the existing DX-WB project **FI-##**, create the subproject **ASSET-TEST-##** (description “Asset Test”) with the object type **BUS1022**.
2. Create the run definition **BAPI_TEST-##** (description “Test Assets Using BAPI”).
3. Create the task **LOAD-##** (“Load Data”).

Task type LOA

Program type: BAPI

Method: CREATEINCLVALUES

Pack.size: 1

X	Sequential Loading
X	Delete Input Files After Processing
	Log Error Messages Only
	Objects that were not transferred
X	Write to Error Files

Input file: BC420_##_TEST_ASSET.SAP

Error File: BC420_##_ASSET_ERROR.SAP

(To enter file name choose the icon with the green cross).

4. Start the run and analyze the log.

Solution 15: Transfer of Asset Master Data Using BAPIs

Task 1:

Create asset master data online (transaction: AS91).

1. Create asset master data online (transaction: AS01). For data transfer using batch input, AS91 is actually used for Fixed Asset Master Data. To analyze the previous case of a transfer using BAPI, it is enough to use AS01. (If you have the required authorization for AS91, you can also use this transaction.)

Asset class: 3100

Company code: 1000

Name: Racing car

Capitalization date: 01.01.2006

Cost center 1000

License plate number: BA-PI 007

Write down the asset number: _____

- a) Proceed as described on the example slide.

Task 2:

Open the DX-WB, go to the DX Tools (menu *Goto*) and create the asset test data using the relevant button in the menu bar.

1. Object Type: BUS1022

Program type: BAPI

Program: CREATEINCLVALUES

File type: P (Physical File Name)

File name: **BC420##_TEST_ASSET.SAP**

(Button “From Test Program”)

- a) To create the test data, use the logical system **DXWB##**.

To generate an IDoc data record, you must specify the following:
Company code 1000, asset class 3100, asset number (see the number you made note of), no subnumber.

Go back to the DX-WB.

Display the file you created and search for your asset description.

Continued on next page

Task 3:

In a new subproject in the DX-WB project **FI-##**, create a new run that will process the test file you created using a BAPI.

1. In the existing DX-WB project **FI-##**, create the subproject **ASSET-TEST-##** (description “Asset Test”) with the object type **BUS1022**.
 - a) Follow the usual procedure to create a new run in the DX-WB. Keep in mind that no customers or documents are used as the procedure; instead, Fixed Asset Master Data is used. Therefore label the individual steps to be filled in the DX-WB project environment sensibly.
2. Create the run definition **BAPI_TEST-##** (description “Test Assets Using BAPI”).
 - a) Proceed in the usual manner.
3. Create the task **LOAD-##** (“Load Data”).

Task type LOA

Program type: BAPI

Method: CREATEINCLVALUES

Pack.size: 1

X	Sequential Loading
X	Delete Input Files After Processing
	Log Error Messages Only
	Objects that were not transferred
X	Write to Error Files

Input file: BC420_##_TEST_ASSET.SAP

Error File: BC420_##_ASSET_ERROR.SAP

(To enter file name choose the icon with the green cross).

- a) Use the example slide in the course folder and the instructor's solution as a guide.
4. Start the run and analyze the log.
 - a) Proceed in the usual manner. Keep in mind that, in contrast to the standard procedures, the BAPI procedures write the resulting document numbers directly in the DX-WB log.

Exercise 16: Transfer of Asset Master Data Using BAPIs

Exercise Objectives

After completing this exercise, you will be able to:

- Basics of BAPI technology
- Transfer of asset data

Business Example

Transfer the external FI installation data to the SAP system using a BAPI. Use the tools LSMW and DX-WB.

File with assets in external format:

BC420##_ASSET.LEG



Hint: Refer to the course diagrams while you are working through this exercise.

Task 1:

You will find a mapping plan for the transfer of external asset data in the appendix.

1. You want to transfer file *BC420##_ASSET.LEG* with asset data from the external system. The file has the following structure:

Field name	Field Type	Length	Description
CODE	C	4	Company code
CLASS	C	4	Asset class
COSTCENTER	C	4	Cost Center
AREA	C	4	Business area
DATE	DDMY	8	Date created
ASSET	C	20	Asset name
PLATENO	C	10	License plate number:

Display the file in the file monitor (transaction AL11).

Continued on next page

Task 2:

In the LSMW: Assign the asset data to the IDoc structure.

1. In the LSMW project BC420-##, create the subproject **ASSET-##** with the name “Assets”.

In this project create the object **BAPI-##** with the name “Assets with BAPI”.

2. Maintain the object attributes for BAPI:

Business object: BUS1022

Method: CREATEINCLVALUES

3. Create the source structure **ASSET_##** and maintain the relevant fields (see table above).

4. Assign the following structures to the source structure:

E1FIXEDASSET_CREATEINCLVALU

E1BP1022_KEY

E1BP1022_FEGLG001

E1BP1022_FEGLG001X

E1BP1022_FEGLG002

E1BP1022_FEGLG002X

E1BP1022_FEGLG003

E1BP1022_FEGLG003X

5. Field Mapping

Perform the mapping according to your mapping plan. Consider the special features of the asset transfer using BAPIs (see the training course material)

To fill the X structures, the LSMW enables them to be filled automatically for a complete structure (menu Extras -> Complete X Structures). This works automatically for the field name of the corresponding X structure. For fields whose names are not identical, the syntax of the generated source code is not correct and you will receive an information message. Note down the name of the field and correct the code for the field in the X structure. The field name must be changed in the IF condition (if not <field name> is initial.) How many field names were not identical? _____

6. “Specify files”: Enter the file name **BC420_##_ASSET.LEG**

(A source structure/no separators/end of record indicators/codepage 1100).

7. “Assign Files”: Confirm **ASSET_## <-> BC420_##_ASSET.LEG**

8. Import the external data and check the result.

Continued on next page

9. Convert the data and check the result.

Task 3:

In the Data Transfer Workbench **DX-WB**:

In the existing project **FI-##**, use the subproject wizard to create a subproject for the BOR object BUS1022, containing a run definition with the tasks Convert, Split Data, and Transfer Data.

1. Select your FI-## project and start the subproject wizard.
2. Choose *Continue* to go to the step “Subproject”, via the steps “Start” and “Project”. Select the business object Asset (object type BUS1022). (“Continue” →)
3. In the “Tasks” step, enter a description for the run definition 1 (“Assets Using BAPI”). Leave the tasks “Convert Data”, “Split Data”, and “Load Data” and choose “Check Data”. Choose *Attributes* for the three tasks selected to define the attributes as follows:

Data conversion attributes: Enter the LSMW object BAPI-## for the conversion, which you created in the first part of the task:

LSMW project: BC420-##

LSMW Subproject: ASSET-##

LSMW object: BAPI-##

Next, enter the program to load the data

Program type: BAPI

Program: CREATEINCLVALUES

Data splitting attributes: You must specify only the file names into which the converted file is to be split. The other details were copied over from information specified earlier. Use the icon with the green cross to define two file names **data1-##** and **data2-##** (both are file type P).

Data loading attributes: For the task LOA select the program type BAPI with the program name CREATEINCLVALUES. On the next screen, enter:

Method: CREATEINCLVALUES

Pack.size: 1

Continued on next page

X	Sequential Loading
X	Delete Input Files After Processing
	Log Error Messages Only
	Objects that were not transferred
X	Write to Error Files (The names for the error files are set automatically)

4. If you have already worked with the wizard: Exit the wizard (*Continue*, then *Complete*).
5. Start the run.
6. Analyze the log after you have started the test data run.
7. Make a note of the numbers of the assets created:

Solution 16: Transfer of Asset Master Data Using BAPIs

Task 1:

You will find a mapping plan for the transfer of external asset data in the appendix.

1. You want to transfer file *BC420_##_ASSET.LEG* with asset data from the external system. The file has the following structure:

Field name	Field Type	Length	Description
CODE	C	4	Company code
CLASS	C	4	Asset class
COSTCENTER	C	4	Cost Center
AREA	C	4	Business area
DATE	DDMY	8	Date created
ASSET	C	20	Asset name
PLATENO	C	10	License plate number:

Display the file in the file monitor (transaction AL11).

- a) Call transaction AL11. The work directory contains your personal external data with Fixed Asset Master Data. To simplify the search, search for *##_ASSET*, for example.

Task 2:

In the LSMW: Assign the asset data to the IDoc structure.

1. In the LSMW project BC420-##, create the subproject **ASSET-##** with the name “Assets”.

In this project create the object **BAPI-##** with the name “Assets with BAPI”.

- a) Observe the CREATE button in the toolbar. To save the project with a subproject and object in the LSMW, you must use this button.

2. Maintain the object attributes for BAPI:

Business object: BUS1022

Continued on next page

Method: CREATEINCLVALUES

- a) Maintain the object characteristics in the LSMW as usual. At this point in the course, you are using this function for the third time. (BI/CT/DI and IDoc procedures were already dealt with.)
3. Create the source structure **ASSET_##** and maintain the relevant fields (see table above).
 - a) Proceed in the usual manner.
4. Assign the following structures to the source structure:

E1FIXEDASSET_CREATEINCLVALU

E1BP1022_KEY

E1BP1022_FEGLG001

E1BP1022_FEGLG001X

E1BP1022_FEGLG002

E1BP1022_FEGLG002X

E1BP1022_FEGLG003

E1BP1022_FEGLG003X

- a) Proceed in the usual manner. In this example, it is sufficient to double-click to assign the source structures (there is only one source structure).
5. Field Mapping

Perform the mapping according to your mapping plan. Consider the special features of the asset transfer using BAPIs (see the training course material)

To fill the X structures, the LSMW enables them to be filled automatically for a complete structure (menu Extras -> Complete X Structures). This works automatically for the field name of the corresponding X structure. For fields whose names are not identical, the syntax of the generated source code is not correct and you will receive an information message. Note down the name of the field and correct the code for the field in the X structure. The field name must be changed in the IF condition (if not <field name> is initial.) How many field names were not identical? _____

- a) See the mapping plan or the mapping slide in the folder. Alternatively, you can use the instructor solution as a guide.
6. “Specify files”: Enter the file name **BC420_##_ASSET.LEG**

Continued on next page

(A source structure/no separators/end of record indicators/codepage 1100).

- a) Define the file name as usual. Check once more whether the source files are on the presentation server or the application server.

Tip for a frequent error: Is it a file with several source structures?

7. “Assign Files”: Confirm **ASSET_## <-> BC420_##_ASSET.LEG**
 - a) The LSMW fills the relevant values automatically because the source structure can result from the one source file only.
8. Import the external data and check the result.
 - a) Check whether the source data was imported correctly. Tip: Check whether the date was imported correctly.
9. Convert the data and check the result.
 - a) Proceed in the usual manner. Check the filled X segment fields. For all relevant mapping fields, the following X segment fields must be filled: Otherwise, the BAPI transfer procedure will ignore these fields at a later stage.

Task 3:

In the Data Transfer Workbench **DX-WB**:

In the existing project **FI-##**, use the subproject wizard to create a subproject for the BOR object BUS1022, containing a run definition with the tasks Convert, Split Data, and Transfer Data.

1. Select your FI-## project and start the subproject wizard.
 - a) Proceed in the same way as you have practiced in the DX-WB.
2. Choose *Continue* to go to the step “Subproject”, via the steps “Start” and “Project”. Select the business object Asset (object type BUS1022). (“Continue” →)
 - a) Proceed in the usual manner.
3. In the “Tasks” step, enter a description for the run definition 1 (“Assets Using BAPI”). Leave the tasks “Convert Data”, “Split Data”, and “Load Data” and choose “Check Data”. Choose *Attributes* for the three tasks selected to define the attributes as follows:

Data conversion attributes: Enter the LSMW object BAPI-## for the conversion, which you created in the first part of the task:

LSMW project: BC420-##

LSMW Subproject: ASSET-##

Continued on next page

LSMW object: BAPI-##

Next, enter the program to load the data

Program type: BAPI

Program: CREATEINCLVALUES

Data splitting attributes: You must specify only the file names into which the converted file is to be split. The other details were copied over from information specified earlier. Use the icon with the green cross to define two file names **data1-##** and **data2-##** (both are file type P).

Data loading attributes: For the task LOA select the program type BAPI with the program name CREATEINCLVALUES. On the next screen, enter:

Method: CREATEINCLVALUES

Pack.size: 1

X	Sequential Loading
X	Delete Input Files After Processing
	Log Error Messages Only
	Objects that were not transferred
X	Write to Error Files (The names for the error files are set automatically)

- a) Observe the sequence of the transport requests: 1) Convert, that is to say, enter “MAP” as the task type in the DX-WB including the LSMW as a mapping step or conversion step. Both tools are linked here.
The next step is the loading step. Perform this step in the usual manner. Manually enter the file that was previously converted in the LSMW. This does not complete the DX-WB automatically.
4. If you have already worked with the wizard: Exit the wizard (*Continue*, then *Complete*).
 - a) Proceed in the usual manner.
 5. Start the run.
 - a) You can always start the run at process level in the DX-WB. This is the third hierarchy step from the top.

Continued on next page

6. Analyze the log after you have started the test data run.
 - a) Double click the log line to display the details of the log, especially the generated fixed asset master data. The log display requires at least two sessions. Ensure that you always have enough external sessions free as a reserve.
7. Make a note of the numbers of the assets created:

- a) To check whether the fixed asset master data was updated, call transaction AS03.



Lesson Summary

You should now be able to:

- Describe the transfer of external data using BAPIs.
- Use BAPIs to transfer asset master data to the SAP system.



Unit Summary

You should now be able to:

- Describe the transfer of external data using BAPIs.
- Use BAPIs to transfer asset master data to the SAP system.

Unit 10

TA Recorder

Unit Overview

In addition to using the specified SAP procedure for the data transfer, a customer development of data transfer programs is also possible. You can even use the LSMW for this.



Unit Objectives

After completing this unit, you will be able to:

- Use the recording function in the LSMW to generate a batch input session from the data of the external system
- Use the Transaction Recorder to generate some data transfer programs
- Analyze programs that work with batch input or a call transaction
- Include the programs of the Transaction Recorder in the DX-WB

Unit Contents

Lesson: TA Recorder	322
Exercise 17: Recording Function in the LSMW	365
Exercise 18: Recording with the TA Recorder.....	369

Lesson: TA Recorder

Lesson Overview

Contents



- TA Recorder basics
- Recording function in the LSMW
- Using the Transaction Recorder
- Batch input and call transaction basics
- Integrating the Transaction Records program into the DX-WB



Lesson Objectives

After completing this lesson, you will be able to:

- Use the recording function in the LSMW to generate a batch input session from the data of the external system
- Use the Transaction Recorder to generate some data transfer programs
- Analyze programs that work with batch input or a call transaction
- Include the programs of the Transaction Recorder in the DX-WB

Business Example

You want to transfer external data for which there is no suitable SAP procedure. In this case, you can generate and develop a transfer program using the Transaction Recorder.

Using the Transaction Recorder

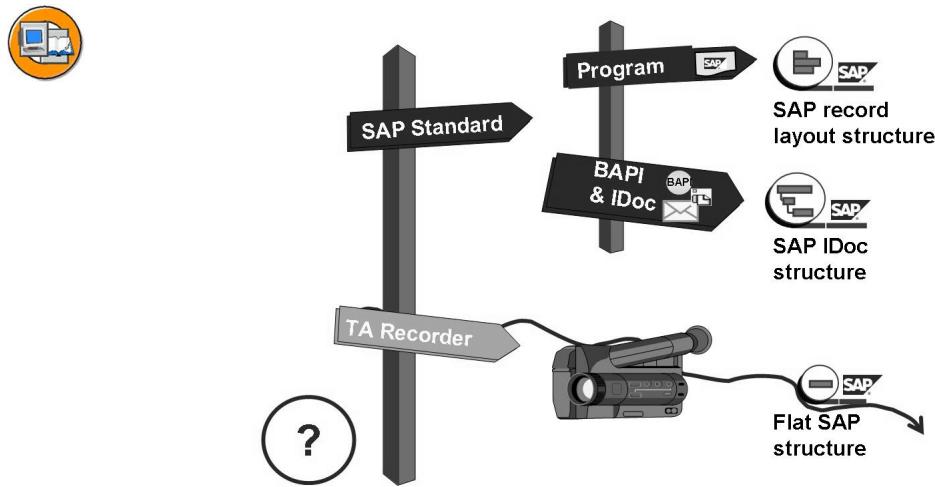


Figure 258: TA Recorder Overview

The standard technology and the IDoc and BAPI technology were already discussed. The following unit discusses customer developments using batch input programming and call transaction programming.



Recording Basics

Recording with the LSMW

Applying the Transaction Recorder

Principles of BI and CT

Integrating the TA Recorder Program into the DX-WB

Figure 259: The TA Recorder

The first part of this lesson introduces the basics of recording in general. This lesson then deals with applying the recorder in the LSMW and the generation and development of a separate batch input or call transaction program as a loading step of a data transfer procedure.

If the SAP standard system does not provide the function that you require to transfer data, you can use the Transaction Recorder (referred to in the following unit as TA Recorder).

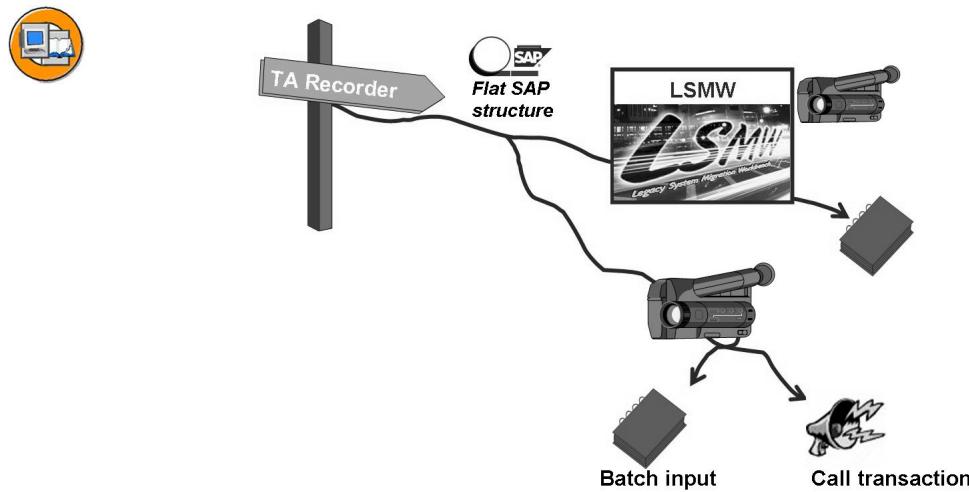


Figure 260: Area of Operation of the Transaction Recorder

The TA Recorder enables you to transfer data to the SAP system using the batch input method or call transaction method.

The TA Recorder is used in the LSMW (among other places) and enables you to generate transaction recordings. It is possible to generate batch input sessions from this recording.

The TA Recorder itself can generate a data transfer program from the recording, this program works internally with batch input or call transaction to execute a data transfer. However, it can work with a flat structure only. You cannot use files with different record structures (for example, header structure and items). This would be possible only through a separate further development of the transfer program generated from the recording.

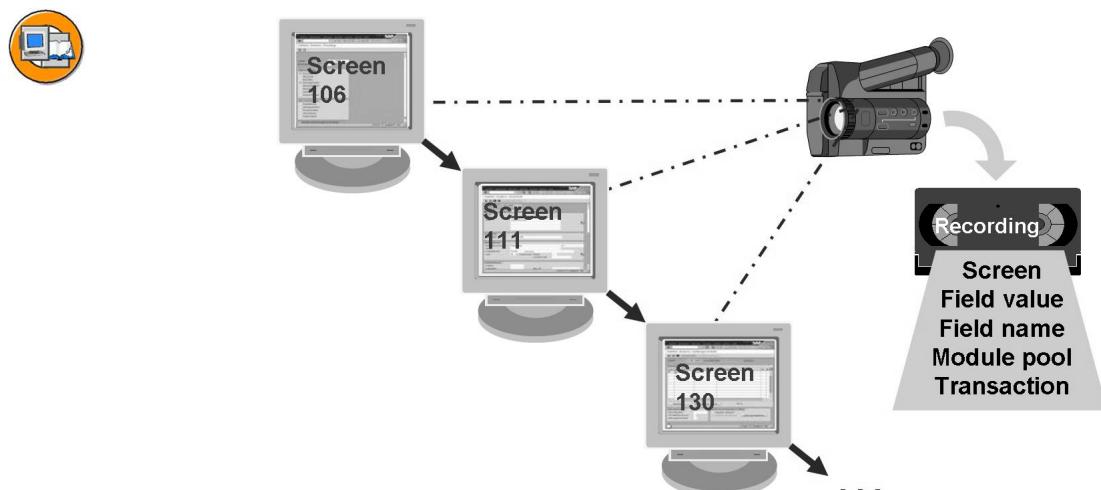


Figure 261: The Recording

The TA Recorder provides the option of recording a transaction. To record a transaction, you must go to the Recorder. Then process the transaction screen by screen as if you are processing it online.

It is useful to test the transaction in advance to ensure that no problems occur when you execute the transaction that would lead to unwanted results.

After the end of the transaction, you receive a recording that includes all screen numbers, field names and field values as well as the relevant module pool and the transaction code.

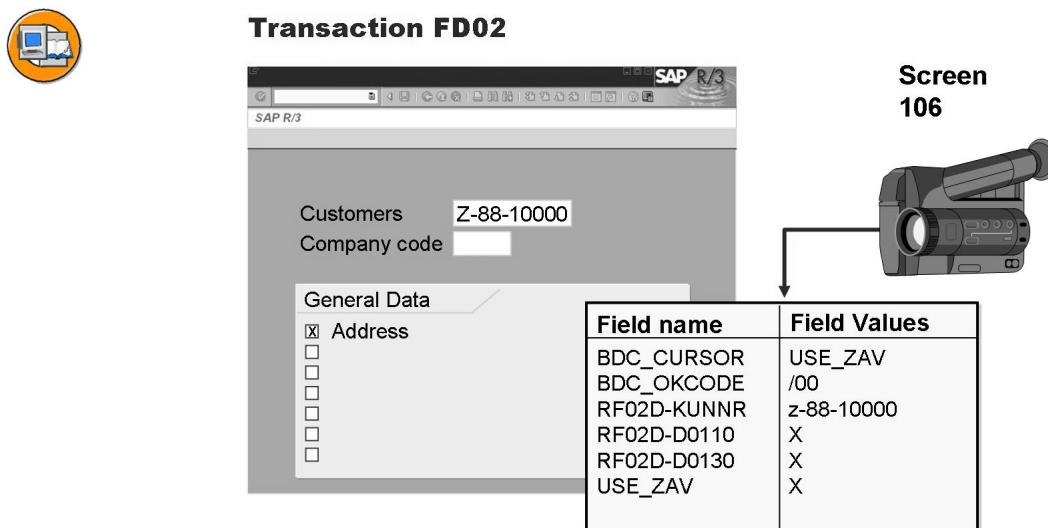


Figure 262: Change Customer: Initial Screen

In the above example, the transaction *Change Customer* FD02 is recorded.

In addition to the fields of the screen, the system also automatically records the fields *BDC_CURSOR* and *BDC_OKCODE*.



Hint:

- **The system records all fields that are filled with values.** Therefore, the system does not only copy the fields that were changed during the recording. Before Release 4.6C, only fields that the user explicitly changed were recorded.
- **In online mode, other screen sequences may appear than in the recording.** For a more detailed analysis of this behavior, observe the system field *SY-BINPT* as a developer. More details are provided later in the course.



SAPMF02D 0106	RF02D-KUNNR /00 z-88-10000 X X
SAPMF02D 0111	=UPDA SAPLSZA1 0300ADDRESS SAPLSZA1 0301COUNTRY_SCREEN SZA1_D0100-TEL_NUMBER Fliederweg 11 20346 Hamburg 040/1726334

Figure 263: Result of the Recording

The result of the recording is a list of all fields and field values that were changed during the recording. You can edit the recording and test it by running it again.

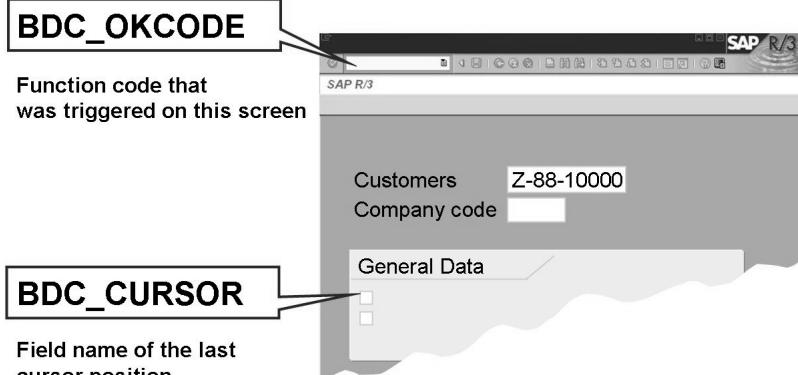


Figure 264: Special Fields of the Recording

During the recording, the system automatically records the following fields:

- The field *BDC_OKCODE* includes the function code that was triggered on the screen.
- The field *BDC_CURSOR* contains the field name of the field in which the cursor was positioned before exiting the screen.

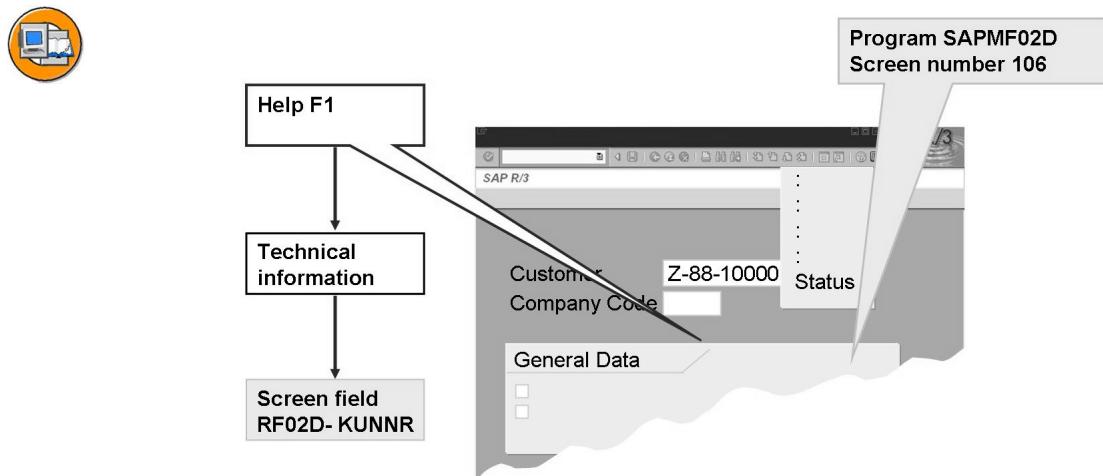


Figure 265: Determining the Field Name

The information recorded in the TA Recorder can also be displayed online for each transaction.

For the current module pool name and the screen number, choose the menu path *System -> Status*

To determine the field name, place the cursor on the relevant screen and display the field help (F1). The name of the screen field is available under “Technical Information”. Keep in mind that some transactions behave differently online than when they are processed by batch input; they may use other screens or fields. We will go into this in more detail later (variable SY-BINPT).



Special screens can be recorded:

- SUBSCREEN areas
 - ◆ Subscreen name, for example BDC_SUBSCR
 - ◆ Field name, for example: ADDR1_DATA-NAME1
- Tabstrip element
 - ◆ Subscreen name, for example BDC_SUBSCR
 - ◆ Field name, for example ADDR1_DATA-NAME1
- Step loops and table control elements
 - ◆ Field names with index, for example KNBK-BANKN(2)

Bank details		
Country	Bank key	Bank account
DE	123456789	08154711
DE	546344536	00708155

Figure 266: Special Screens

The TA Recorder automatically records the special screen areas named above.

SUBSCREEN areas

- The required subscreen names are recorded automatically.

Tabstrip areas

- The required subscreen names are recorded automatically.

Step loops and table control elements are the same for the recording.

- The fields in the rows displayed have the same name.
- Bank details of the customer are displayed for each row. The fields must be addressed through an index (the index is specified in brackets after the field name, for example, KNBK-BANKN(2)). This index must correspond to the row number.
- Only those fields that are visible on the screen can be filled with data. If fields can only be displayed by scrolling down the screen, the scrolling function must also be recorded. All the steps in the online transaction must also be carried out for the recording.

Therefore, for example, “Scroll Down” can be performed using the function code /23. (Scroll icons)

Scroll to the end of a list: /24.

Scroll down one page: /22.

Scroll back to first page: /21.



- **Test the transaction before recording.**
- **Start the recording.**
- **All fields containing data are recorded.**
- **You must exit the transaction properly - SAVE!**



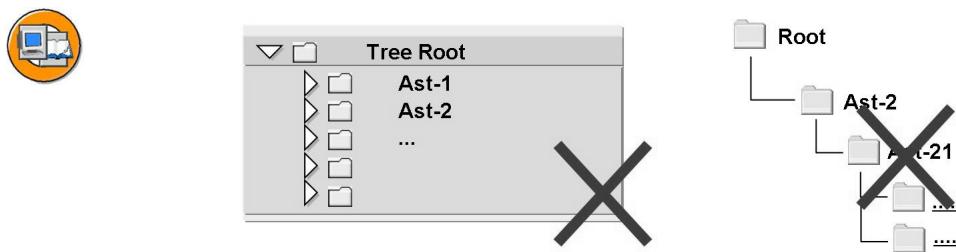
- **Scrollbar use cannot be recorded.**
Use the scroll function of the screen.
- **F1 Help is not recorded.**
- **F4 Help is not recorded.**

Figure 267: Recording Rules

Special features with recording:

- F1, F4 and self-programmed F1 help, F4 help (PROCESS ON HELP-REQUEST, PROCESS ON VALUE-REQUEST) are not recorded. The same applies for all commands under **System** and **Help**. The system does not take standard variant (transaction variants) values into account.
- Error dialogs and warning dialogs are not recorded. That is, the system records only the OK code and the field contents that resulted in the successful further processing in the current screen.

 **Note:** During the recording, the entered data is written to the database.



These and other controls do not support the BI method!

- SAP Picture Control
- SAP HTML Viewer Control
- SAP Text Edit Control
- ALV Grid Control
- SAP Tree Control
- ...

Figure 268: Enjoy SAP Controls and Data Transfer

The new “EnjoySAP Controls” developed for Release 4.6 are **not** batch input enabled or call transaction enabled. If you use the SAP transaction with these controls, use the standard procedure/IDocs/BAPIs provided by SAP in this type of application case.

However, as already shown, the table control element and the tabstrips are batch input enabled.



Recording Basics



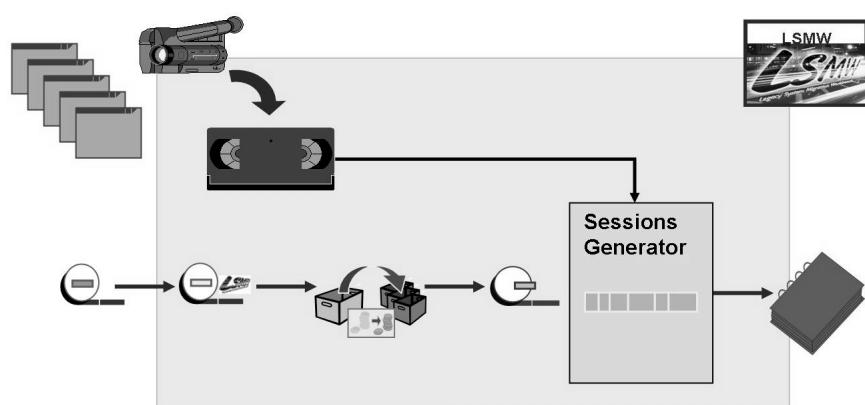
Recording with the LSMW

Applying the Transaction Recorder

Principles of BI and CT

Integrating the TA Recorder Program into the DX-WB

Figure 269: The Transaction Recorder



- Only BI sessions can be generated from the recording.

Figure 270: Recording with the LSMW

The LSMW provides the option of creating your own recording.

The LSMW uses the transaction recorder of the standard system: However, it evaluates the recording itself. The information from the recording is used to generate batch input sessions.

The generated batch input session can be run using the batch input monitor as usual.



Hint: Keep in mind that the functions of the recorder are slightly different in the LSMW than in outside the LSMW.

Most importantly, if you created recordings outside the LSMW you cannot reuse them within the LSMW. Therefore, if you want to use a recording in the LSMW you must record it within the LSMW.

The next figure shows how you record within the LSMW.



- **Creating a Recording:**

- ◆ Choose an LSMW project.
- ◆ Enter the name of the recording and a description.
- ◆ Enter the name of the transaction.

Figure 271: Creating a Recording in the LSMW

Start the recording:

- Go to the LSMW and execute the function *Recordings* under *Goto → Recordings*. The system displays an overview of all recordings created for your project. To start a new recording, choose *Create*.
- For a recording, you must enter the name, the description and the owner of the recording.
- Enter the transaction name of the transaction that you want to record.
- Execute the transaction and exit it correctly.

 **Hint:** Recordings are assigned to one project. Therefore, keep in mind which term you entered for the project.

Recording REC_1 Demo_1	
	
FD02 Change Customer (accounting)	
SAPMF02D 0106	
BDC_CURSOR	RF02D-KUNNR
BDC_OKCODE	/00
RF02D-KUNNR	z-88-10000
RF02D-D0110	X
USE_ZAV	X
SAPMF02D 0111	
BDC_OKCODE	=UPDA
BDC_SUBSCR	SAPLSZA1
BDC_SUBSCR	SAPLSZA1
BDC_CURSOR	SZA1_D0100-TEL_NUMBE
ADDR1_DATA-STREET	Fliederweg 11
ADDR1_DATA-POST_CODE	20346
ADDR1_DATA-CITY1	Hamburg
SZA1_D0100-TEL_NUMBE	040/1726334

Figure 272: The Result: Recording Without the LSMW



The figure shows the result of a recording without the LSMW. After the end of the recording, you have the option of editing the recording. You can delete fields or add new fields. You can therefore edit the resulting tree.

As a comparison, the following figure shows a recording within the LSMW.

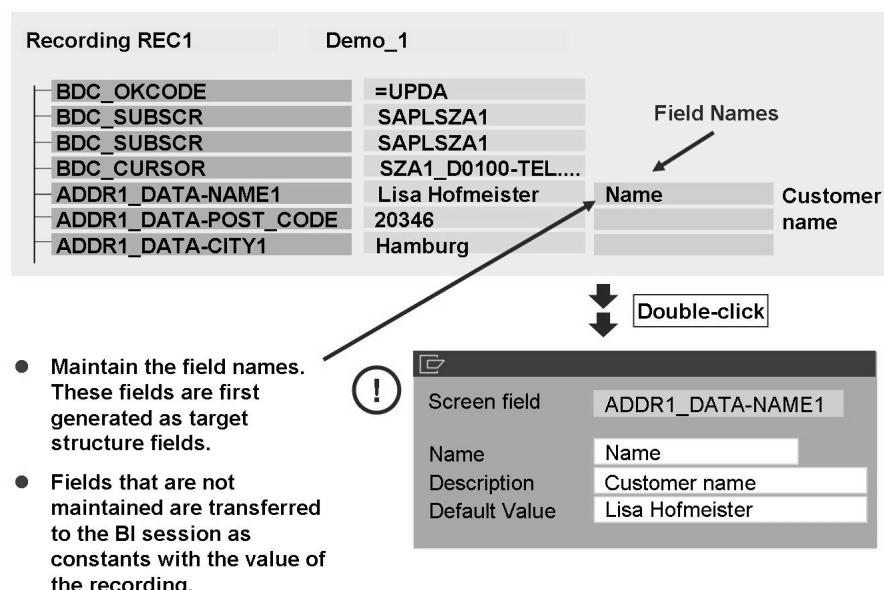


Figure 273: Assign Field Names During Recording with the LSMW

The right hand side of the figure shows the important difference between a recording within and outside of the LSMW. Within the LSMW there are orange, darker shaded areas right of the recorded screen fields. These contain the later description for the target fields of the rules (assignment/mapping). **Orange areas:** All orange areas must be filled with field names. Either enhance the names by choosing *Default* or assign a name manually. If you choose *Default*, the system automatically enters the SAP names for the fields. Keep in mind that these names represent the names of the target fields for the subsequent mapping.

The special fields (BDC_OKCODE, BDC_CUSOR and BDC_SUBSCREEN) are excluded from editing.

Functions:

- Default: Assigns the field names of the underlying target field and its field description.
- Reset: Deletes field names and field descriptions.
- Double-Click: Editing field names, field description and default value.

→ **Note:** You can use field names several times. However, the field name is the field mapping is available only once. Therefore, this is not useful.

For all those fields in which you do not specify any field names, the system uses the specified default value when creating the batch input session. This default values can be seen as constants. This is especially useful for checkboxes (for example MM01, Selection of Views).

You must delete any fields that you do not want to use for the data transfer from the structure.



- Add/remove screen fields



Selection of all available screen fields

- Repeat recording

- ◆ For identical field names, the name and the recording are automatically assigned.

Figure 274: Edit Recording (Only Limited in LSMW)

You can also subsequently add new screen fields if you require further fields. However, you must proceed with caution here. You should add fields only where they are available only the screen; otherwise errors will occur when you run the batch input session.

After you save the recording the status line displays the message: “Data Successfully Saved” The recording is now available with the attributes for the object.

If you repeat the recording you can record a new screen sequence with new fields. If the field names are identical (for example KNA1-NAME1), the names and recordings are automatically assigned again.

As already mentioned, you must keep in mind that the recorder in the LSMW has different functions. Some fields that can be changed in the recorder outside the LSMW cannot be changed here. You must therefore take the utmost care when recording transactions.

The LSMW supports the function of **Auto-Field Mapping**. You can automatically perform field assignments between source fields and target fields. Fields are assigned based on name equality. It is therefore useful to name the field names of the source structure the same as or similar to the field names of the target structure (if you want to use auto field mapping).

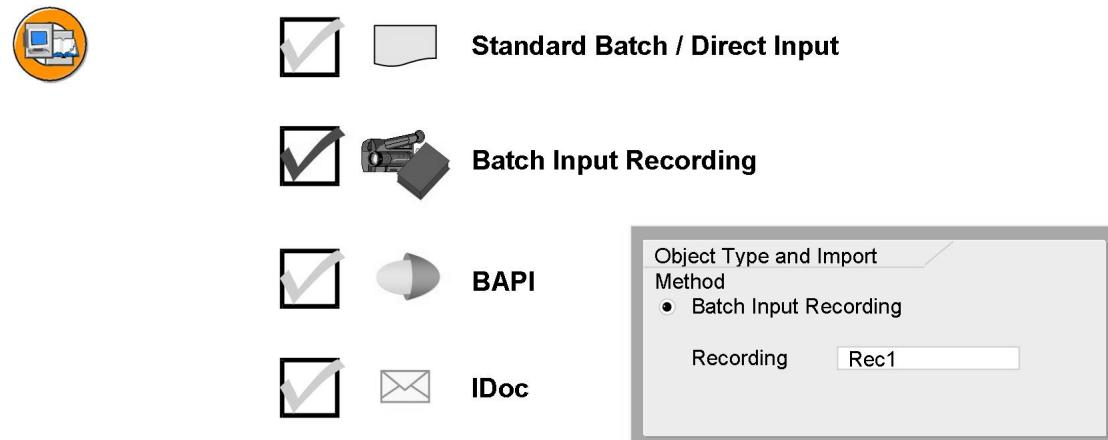


Figure 275: Object Type and Import Method: Recording

Further procedure after recording in the LSMW:

If the recording was created in the LSMW, you can perform all work steps as usual. In the first work step **Maintain Object Attributes**, use the “Batch Input Recording” and enter the name of the recording you just created in the specified field.

All other steps for the source structures and source fields occur as usual.

For the work step **Field Mapping and Conversion Rules**, you must now introduce an additional LSMW function. Keep in mind that this LSMW function can be used for all the methods and procedures shown previously. It was first introduced at the end of the course for didactic reasons. The use of the **Auto-Field Mapping** would also have been possible and useful for all previous demonstrations.

In this way, you can automatically perform field assignments between source fields and target fields. Fields are assigned based on name equality. For the auto field mapping, it is useful to choose source structure field names that are the same as or similar to the field names of the target structure (orange areas).

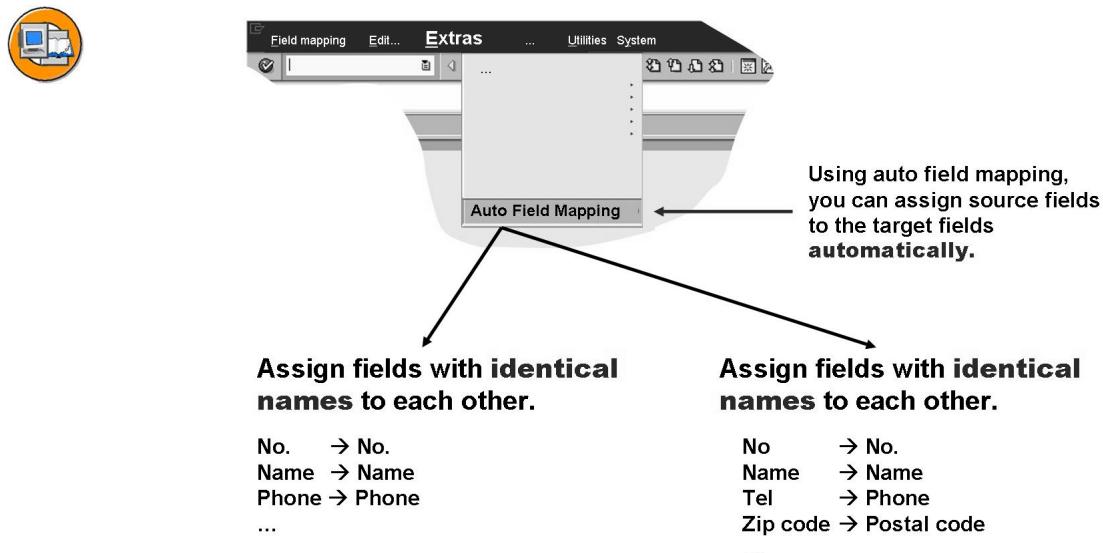


Figure 276: Auto Field Mapping of the LSMW

If the source field names were previously named identical to the SAP target fields, you can assign all fields to each other very quickly. The LSMW would then set the rule MOVE for all assigned pairs. If the source fields do not match the target fields 100%, you can still use the option “Match Fields with Similar Names” to assign fields with similar names. This saves effort and time every time.

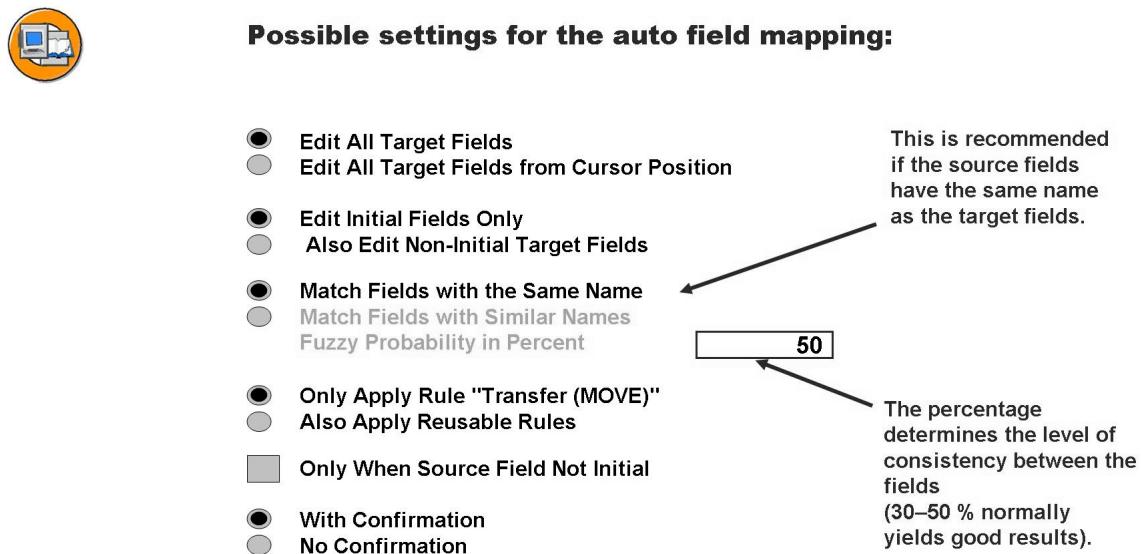


Figure 277: Auto Field Mapping: Similar Fields

The function *Also Apply Reusable Rules* is also practical. Therefore, for example, the system proposes the company code for which a rule was already defined in another LSMW object and reuses its rule “Fixed Value”.

The LSMW therefore automatically proposes already defined conversion rules during auto field mapping.

We recommend that you use the option “With Confirmation” to gain experience when you first try to work with the functions of auto field mapping. The LSMW will issue another confirmation window before each assignment. The user ultimately decides about the application of the assignment of each field.

Keep in mind that the LSMW cannot perform any automatic formatting for new, unknown fields. As before, you must define a rule set (M, V, F, M+C) for this.

You can execute the data transfer in this example as before.

- Maintain the source fields and structures.
- Maintain the field mapping (with auto field mapping).
- Read and convert data.
- Create and run a batch input session.



Recording Basics

Recording with the LSMW



Applying the Transaction Recorder

Principles of BI and CT

Integrating the TA Recorder Program into the DX-WB

Figure 278: The Transaction Recorder

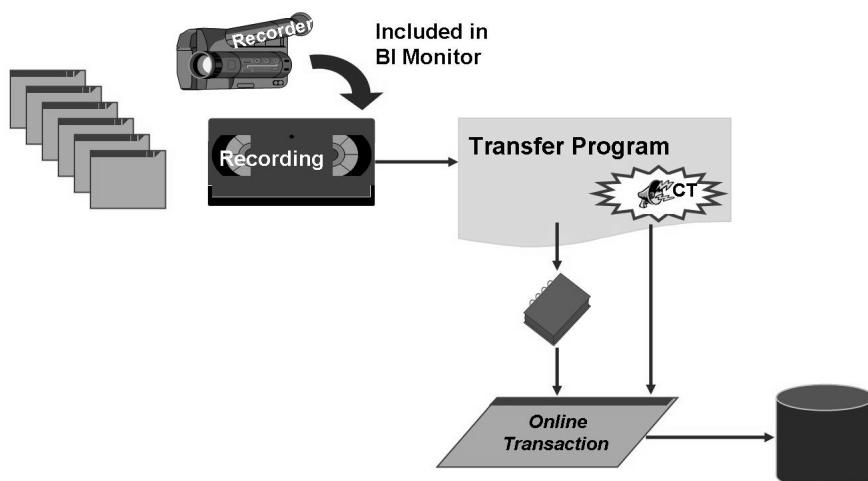


Figure 279: Transaction Recorder

The Transaction Recorder can record transactions. From the recording you can generate a transfer program. This program provides you the option of transferring data using batch input or call transaction.

The Transaction Recorder is available in the batch input monitor (transaction: SM35) or under *System → Services → Batch Input*.

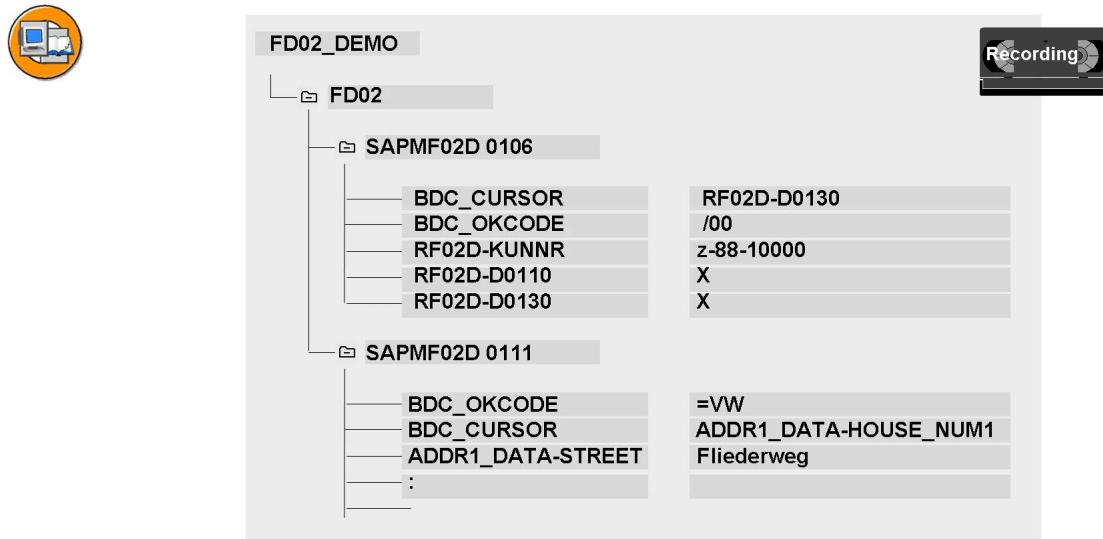


Figure 280: Tree Display of the Recording

When you use the TA Recorder, you also receive a list in the form of a tree display. However, you cannot assign any separate field names as in the LSMW.

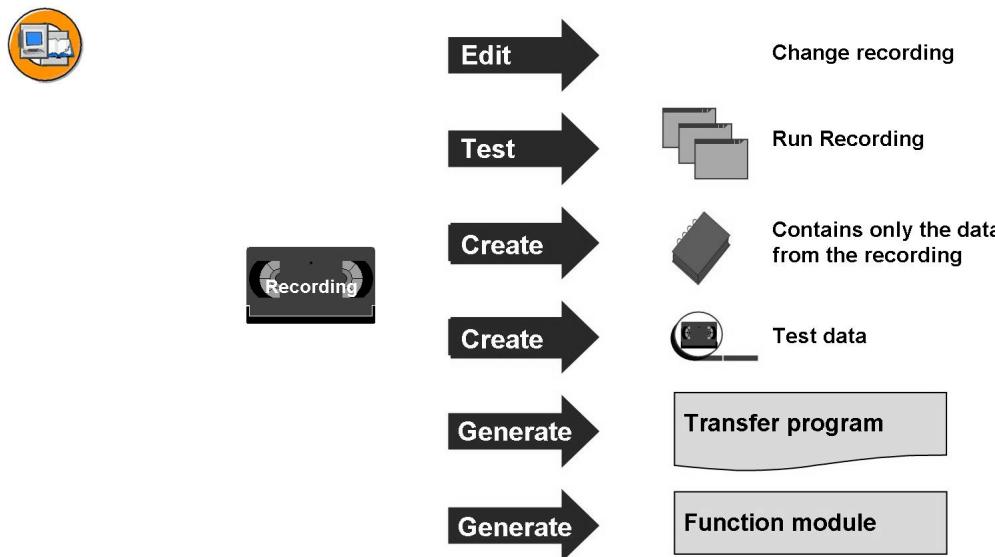


Figure 281: Recording Overview

You can change the recording.

You can test or execute the recording again. In this case, the screen will run again in the same way as in the recording.

You can generate and run a batch input session with the recorded data.

You have the option of generating a field from the recording. This file contains all fields from the recording. In this case, the fields are stored in the file with the length of their definition.

- The sequence matches the sequence in the recording overview. The technical fields (BDC_OKCODE, BDC_CURSOR and so on) are taken into account.
- The data can be stored on the application server only. If you do not enter a path, the file is stored in the “work” directory of the application server.
- If the specified test file already exists, the system attaches the new data record to it.

From the recording, you can create the following:

- A data transfer program that can read a file and processes this data using catch input or call transaction
- A function module that can process an individual data record (for example customers) using batch input or call transaction. See the online documentation for more information.
- Function module interface:
 - The import interface consists of a technical part and of the fields from the recording (one parameter is stored for each filled input field of the recorded transactions).
 - The export interface consists simply of the parameter SUBRC. If the function module is working with call transaction, it contains the actual return value (SY-SUBRC), after it has called call transaction. If a session is created, you will receive a message about whether the data record could be created in the session. This return value is set to zero, if the data record has been processed correctly. (This applies to call transaction only.)
 - The table interface is used only if the function module works with call transaction. The table contains all the system messages.



- Insert/delete rows
- Format check
- Import/export



	Program	Screen	St.	Field name	Field value
1		T		FD02	BS
2	SAPMF02D	106	X	BDC_CURSOR	USE_ZAV
3				BDC_OKCODE	/00
4				RF02D-KUNNR	z-88-10000
5				RF02D-D0110	X
6				RF02D-D0130	X
7				USE_ZAV	X
9	SAPMF02D	111	X	BDC_OKCODE	=VW
10				BDC_CURSOR	ADDR1_DATA-HOUSE_NUM1
11				ADDR1_DATA-STREET	Fliederweg
12					

Figure 282: Editor for Recordings

You can edit a recording.

In the editor, you can add and delete individual lines. In addition, you can change the contents of individual lines.

You can use the function *Check* to determine whether the edited version of the recording is still syntactically correct.

After editing the recording, save your changes.

Import/ Export

- If the editor functions are not sufficient for you, you can use the *Export* function to load the recording to your presentation server and use a PC editor there to edit the recording.
- You can use the *Import* function to import this file to the SAP system again. Ensure that this file still has the right format.



- Directly after the first recording
- In display mode of the recording

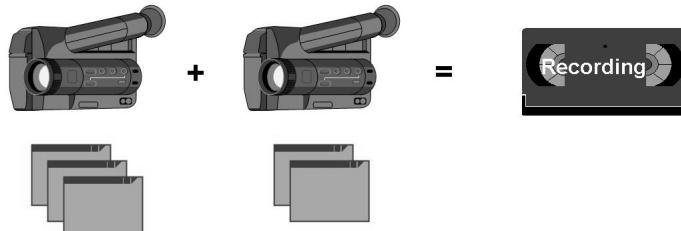


Figure 283: Recording Several Transaction Runs

Here, you have the option of recording other transactions and including more than one transaction run in a recording.

This option is possible directly after a recording or in the editor of a recording.

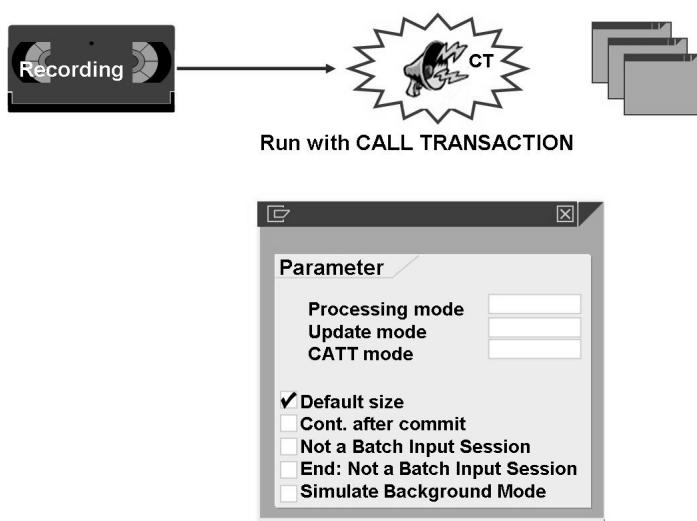


Figure 284: Run Recording

You can execute the recording directly. For this, the system uses the data from the record and runs the transaction using call transaction.

The call transaction method offers a variety of options for calling a transaction.



- **Processing mode**
 - ◆ Display all screens
 - ◆ Display errors
 - ◆ Background processing
 - ◆ Background-enabled and debug-enabled
- **Update mode**
 - ◆ Local
 - ◆ Synchronous
 - ◆ Asynchronous
- **CATT mode**
 - ◆ No CATT
 - ◆ CATT without single screen control
 - ◆ CATT with single screen control

Figure 285: Run Recording Using Call Transaction 1

Processing mode:

You can determine the run mode here. It is identical to that of batch input.

Update mode:

You can determine the update mode. More information is available in this unit under Call Transaction.

CATT Mode:

You can make special settings if you want to test the recording under CATT conditions (required for CATT procedures only). The system calls special function modules in the CATT tool. The call occurs at the end of the PBO and at the start of PAI for each screen.



- **Default size**
 - The screen default size is used for processing.
- **Cont. after commit**
 - Commit Work does not terminate processing.
- **Not a Batch Input Session**
 - No batch input session is active. The transaction runs in the same way as it runs online.
- **End: Not a Batch Input Session**
 - Batch input session has been set. After the end of the data in the BDC table, the BI session is inactive.
- **Simulate Background Mode**
 - SY batch set – runs as in the background.

Figure 286: Run Recording Using Call Transaction 2

→ **Note:** Keep in mind that, in most cases, you do not require these special settings and options (except for the default size). They are used in special cases only and you must extensively test their effects before you use them.

Default size:

If you select this, the screens in the foreground (process in the foreground or display errors only) are run at the default size. You can set the default size manually by choosing “Customize Local Layout”.



Hint: In the background, the system always uses the default size.

Cont. after commit:

The first COMMIT WORK in a transaction does not result in the termination of the recording run.



Hint: A batch input transaction is terminated at the first COMMIT WORK. If you use this special checkbox, the system can continue to process a call transaction (this is not the case for batch input).

Not a Batch Input Session:

The system field SY-BINPT is not set (SY-BINPT = ‘ ’). That is, transactions are run in the same way as they are run online.

End: Not a Batch Input Session

This is useful only if you have chosen run mode E (Display Errors Only). The system then runs the transaction in the batch input session for background processing, and if there is an error it runs the screen in the foreground in online mode.

Simulate Background Mode

The system processes the transaction in that same way as in batch input, that is, in the background. Some transactions behave differently than during processing using dialog work processes (see the variable SY-BATCH).



- When executed in a batch input session or in a background session, some transaction behave differently than when executed online.
- Batch input sessions sets system field SY-BINPT = 'X'
- Background mode sets system field SY-BATCH = 'X'

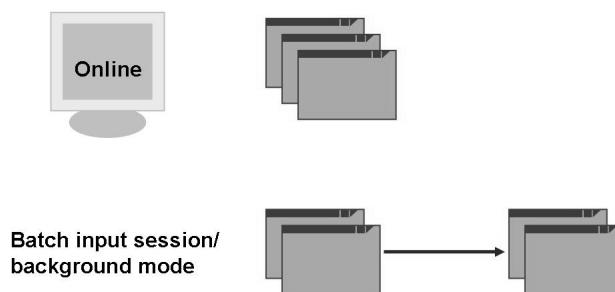


Figure 287: Batch Input Sessions and Background Mode

Not every transaction can run identically online and in a batch input session. That is, the transaction itself must decide whether it is called online or in a batch input session.

The system sets the system field *SY-BINPT = 'X'* if the batch input session is active. (This is also valid for call transaction). The transaction can use this system field to decide whether to run online or in batch input mode. In addition, the transaction decides whether the process is run in the background or in dialog mode (field *SY-BATCH*). Therefore, if you manually run a session in the batch input monitor, the transaction would run the following selection: *SY-BINPT='X'* and *SY-BATCH=' '*. If the same session was scheduled in the background, the transaction would run through *SY-BINPT = 'X'* and *SY-BATCH='X'*. Therefore, it would depend on whether the program even queries this indicator.

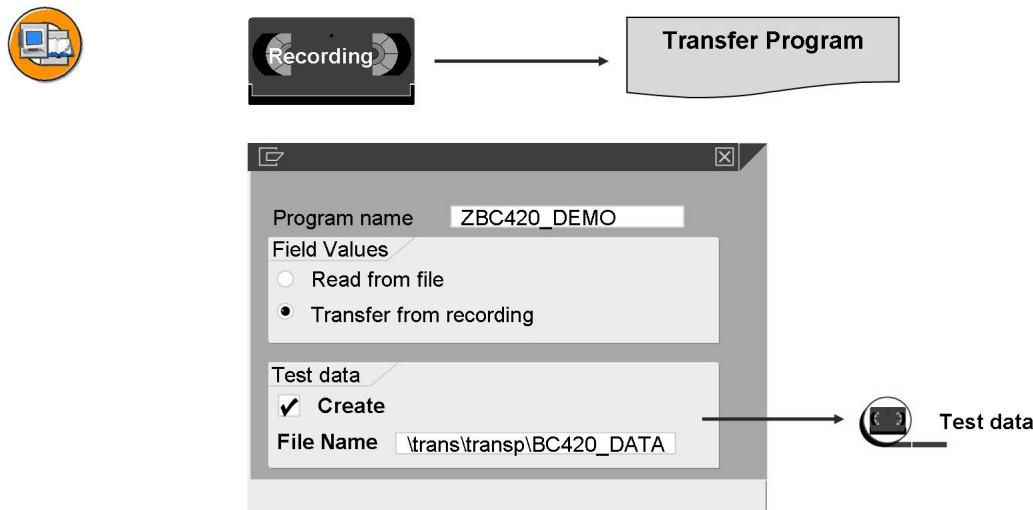


Figure 288: Generating the Program

This function is the strength of the TA Recorder.

You can generate a **complete data transfer program** from the recording. The result would be suited to forming the basis for a complex development because it contains a number of modules and components that are necessary for batch input programming (an example of this type of application is the dynamic reaction to errors when running a transaction; you can construct chains of transaction calls using the means from the recording).???

On the subsequent dialog screen, enter a program name. In addition, you can choose the way in which you want the maintained field contents of the recorded screens to be filled.

- Transferring the values that were used during the recording: If you require a flexible data transfer, you must also modify this program at a later time.
- Parameterizing the input values to be maintained and importing these values from a file: For the parameterization, a data structure is created and import the data records from an external file into this data structure. The program also assumes that the external file was adjusted to this data structure.

If you have decided to parameterize the input values to be maintained, it is useful to create a test file at the same time as the generation. For this purpose, select the checkbox and enter a name for the test file.

You reach the maintenance screen of the attributes of the ABAP editor.. Maintain the attributes and save the program.

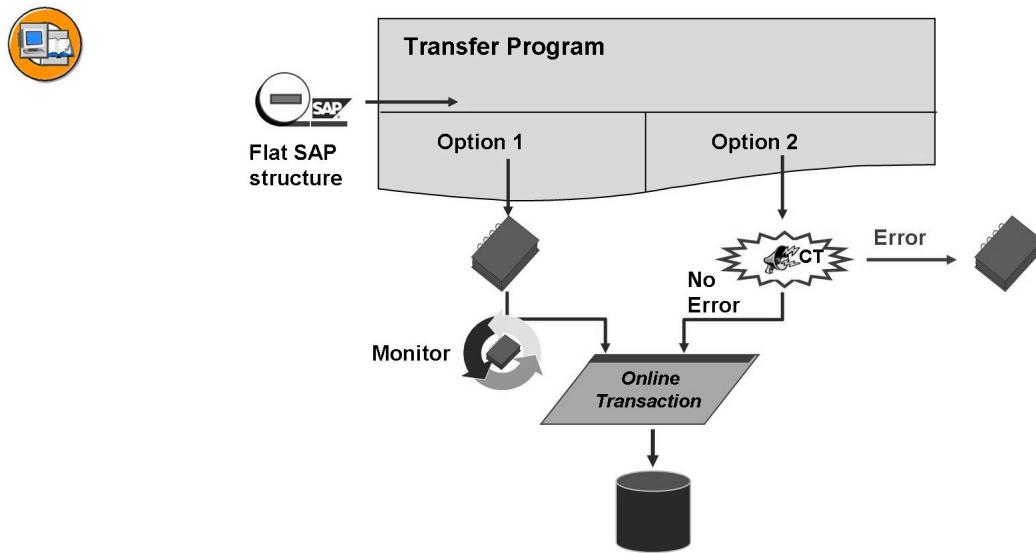


Figure 289: Program Functions

You have now generated a data transfer program that you can use to import data into the SAP system. The program can execute the data transfer using batch input or call transaction.

If you choose the second option, the data record will be put in a batch input session if there are any errors.

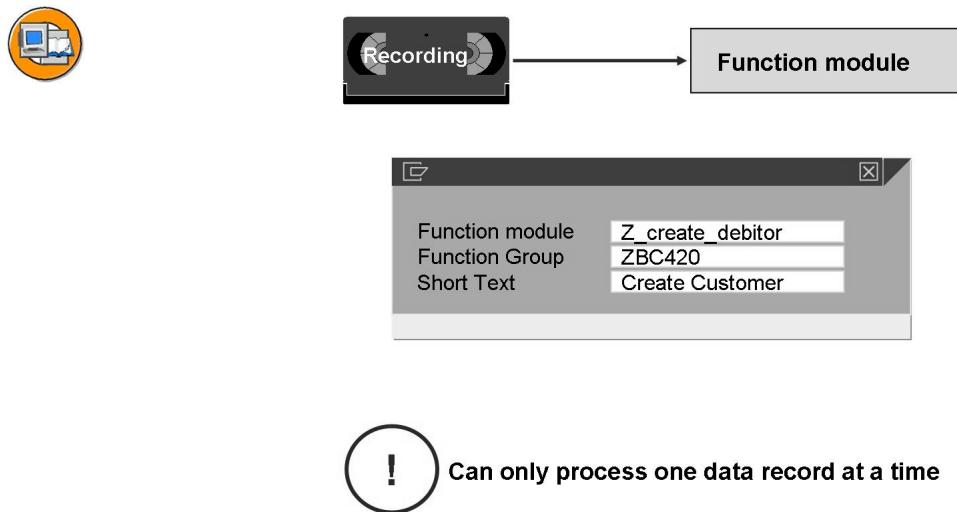


Figure 290: Create Function Module

You can generate a function module from the recording.

In the next dialog box enter a name, a function group, and a short text for the function module. The function module is automatically created in the function group.

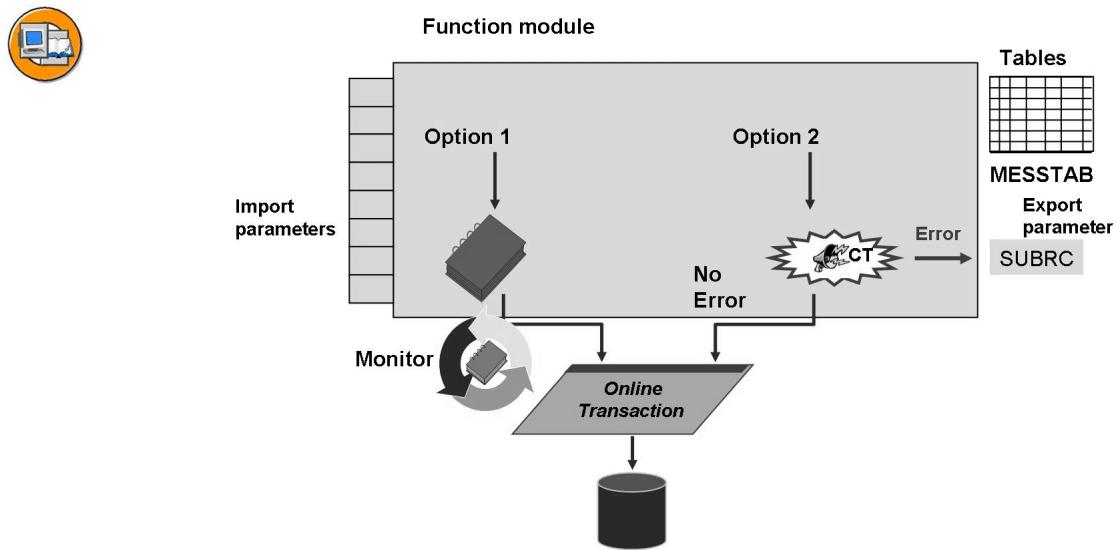


Figure 291: The Role of the Function Module

Import Interface:

The import interface consists of a technical part and of the fields from the recording (one parameter is stored for each filled input field of the recorded transactions).

The export interface:

Consists simply of the parameter SUBRC. If the function module is working with call transaction, it contains the actual return value (SY-SUBRC), after it has called call transaction. If a session is created, you will receive a message about whether the data record could be created in the session.

This return value is set to zero, if the data record has been processed correctly. (This applies to call transaction only.)

The table interface:

Used only if the function module is working with call transaction. The table contains all the system messages. The table contains all the system messages.

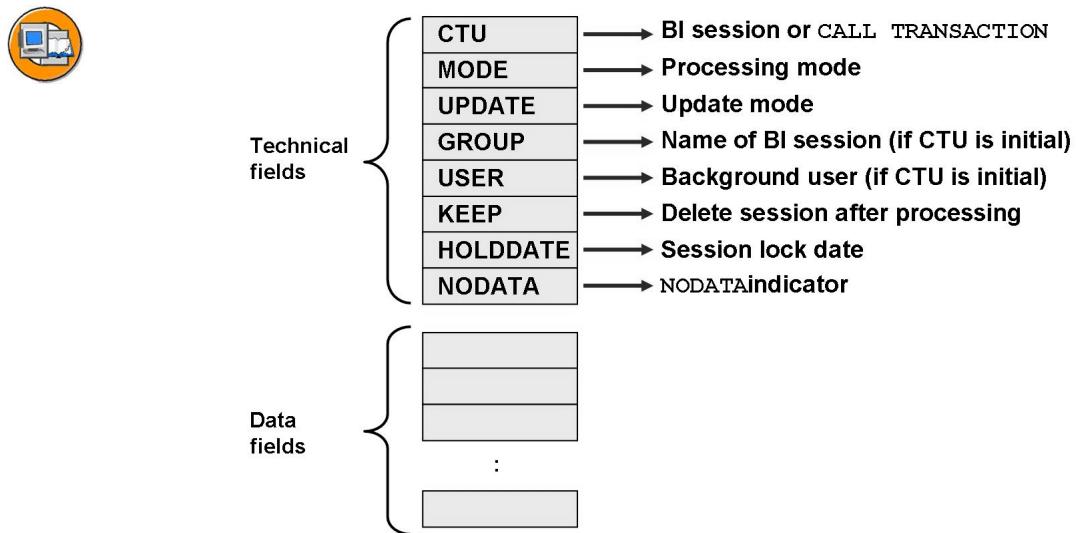


Figure 292: Import Interface of the Function Module

In the parameter CTU you can specify whether the function module creates a batch input session (CTU = ‘ ‘), or updates the data record directly using call transaction (CTU = ‘X’).

The other parameters have to be filled the same as they are in the session header.



Hint: Each time the function module is called, it creates a batch input session.

If you want the session to contain several transactions, you have to:

Make the transaction available several times. This can be done by recording the transaction again. Consequently, the interface of the function module consists of more than one transaction. When the function module is called several, transactions can be created for each session.



Recording Basics

Recording with the LSMW

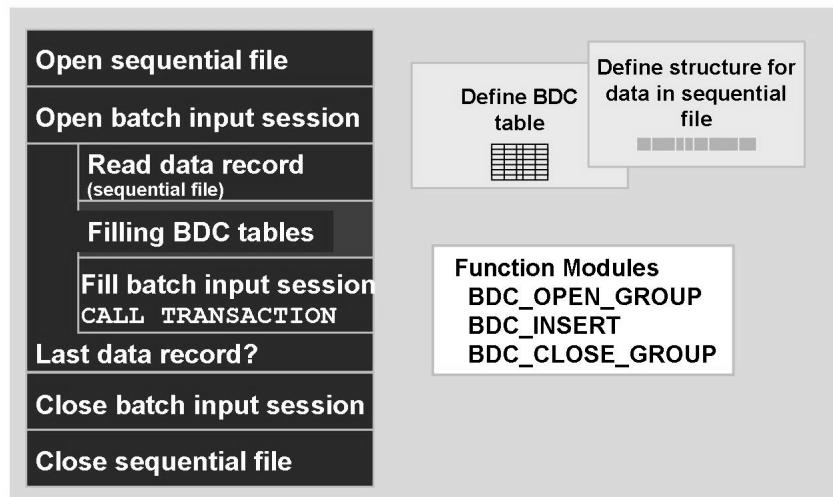
Applying the Transaction Recorder



Principles of BI and CT

Integrating the TA Recorder Program into the DX-WB

Figure 293: The Transaction Recorder

**ABAP Program****Figure 294: Recorder: The Generated Program**

The generated program has the structure displayed above. It uses teh standard function modules to generate the batch input sessions. In addition, it is also able to process call transaction. The user can select this using a selection screen.

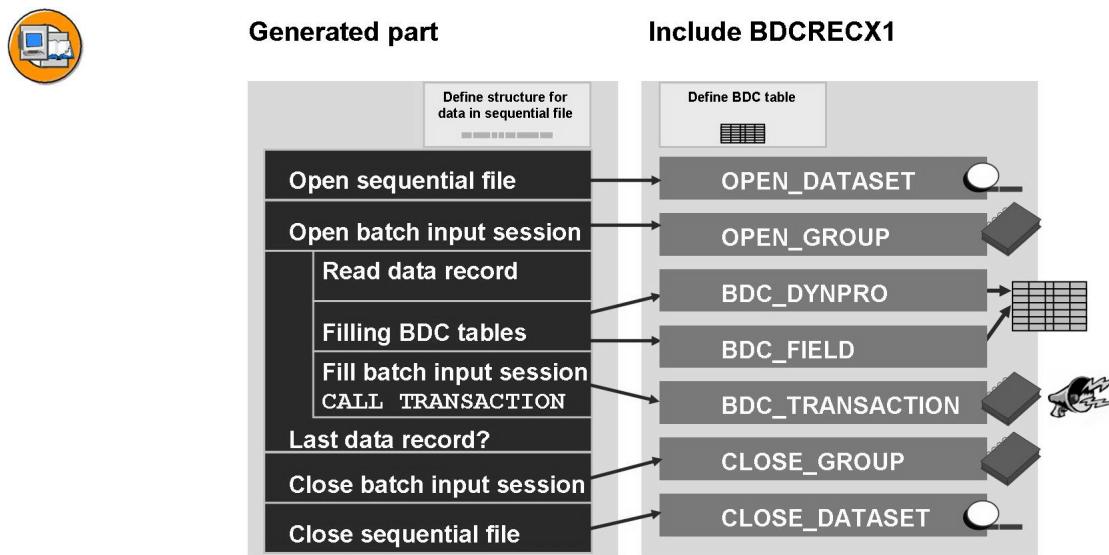


Figure 295: Structure of the Generated Program

→ **Note:** The following slide displays all the relevant parts of the program generated by the recorder. Keep in mind that in releases before the recorder was introduced the whole batch input program had to be programmed manually. In the meantime, all important sections of code are specified by the recorder. Only in the case of further development of the generated program is it necessary to understand all the sections in detail.

The generated program consists of two parts. These are a generated part and the include **BDCRECX1**. The include is contained in the standard delivery.

The include contains the definition of the required BDC table and seven other subroutines. These subroutines perform the following functions:

- Opening and closing files
- Opening and closing sessions
- Filling BDC tables
- Filling sessions

The generated part forms the framework program and calls the subroutines accordingly. In the generated part, a structure is also generated to enable the system to import the file with the external data.



PROGRAM	DYNPRO	DYNBEGIN	FNAM	FVAL
SAPMF02D	0105	X		
			RF02D-KUNNR	Z-100000
			RF02D-BUKRS	0001
			RF02D-KTOKD	KUNA
SAPMF02D	0110	X		
			KNA1-NAME1	Smith
			KNA1-PSTLZ	69000
			KNA1-LAND1	DE
			KNA1-SPRAS	
SAPMF02D	0120	X		

ABAP Dictionary

BDCDATA			
Field Name	Type	Length	Short Text
PROGRAM	CHAR	40	BDC module pool
DYNPRO	NUMC	4	BDC screen no.
DYNBEGIN	CHAR	1	BDC begin a screen
FNAM	CHAR	132	BDC field name
FVAL	CHAR	132	BDC field contents

Figure 296: Structure of the BDC Table

The BDE table has the same structure as for a standard transfer.

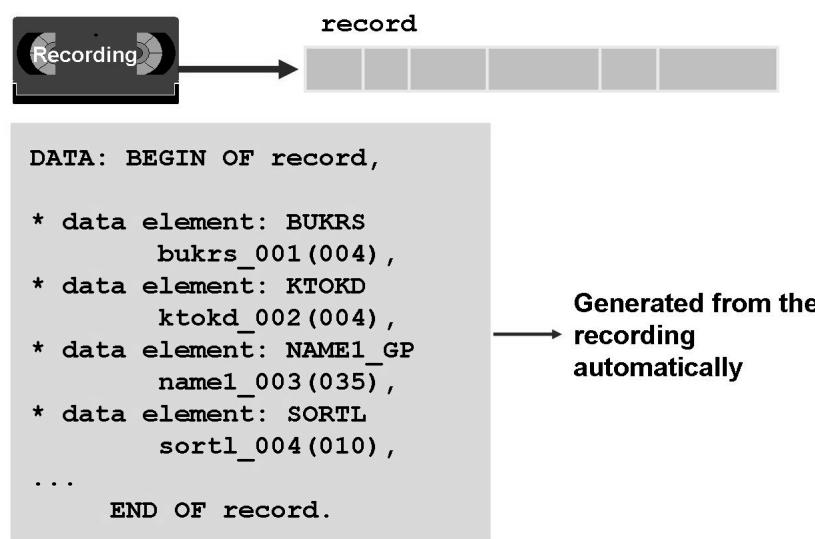


Figure 297: Read Structure

The structure displayed in the figure is automatically generated and always has the name RECORD. Therefore, the recorder imports the external data to this structure; the structure results from a recording of an SAP transaction.

The generated structure always has the same structure. Each field is defined by two generated lines. The first line is a comment line. The line consists of the text “*data element.” and the actual data element of the screen field from the recording. In the next line, the field is defined itself. It receives the name of the data element

followed by a triple-digit item number. The item number indicates the position in the structure. The length of the field is defined by the length of the data element. If a field is not created using a data element, the length of the screen field is used.



Hint: The recorder assumes that the structure fields are simultaneously the fields with converted external data. (It cannot know the actual structure of the external file.) However, because this is not the case, you must perform postprocessing. The easiest way is to ensure that the external data is already formatted when it is delivered. Then users can use the structure as in the course example, that is, without many changes except technical fields such as the checkbox (for the address) and the company code. "X" must be set manually for the checkbox, and "0001" must be set for the company code.

If the data is delivered without being converted, the program would have to be enhanced by importing the external data, formatting the data and moving it to the target structure RECORD. Therefore, the structure of the external file and the import of the external file must be programmed specially.

As a result of the programming work, the internal table BDCDATA is always filled correctly. This table contains the data for a transaction. To check this, always trace your modified program in the debugger and check whether the internal table BDCDATA is filled correctly.



OPEN_DATASET

```
FORM open_dataset USING p_dataset.
OPEN DATASET p_dataset IN TEXT MODE.
  IF SY-SUBRC <> 0.
    WRITE: / text-e00, SY-SUBRC.
    STOP.
  ENDIF.
ENDFORM.
```

Include
BDCRECX1

CLOSE_DATASET

```
FORM close_dataset USING p_dataset.
CLOSE DATASET p_dataset.
ENDFORM.
```

Figure 298: Opening and closing files

The subroutine OPEN_DATASET opens the sequential file.

The sub routine CLOSE_DATASET closes the sequential file.

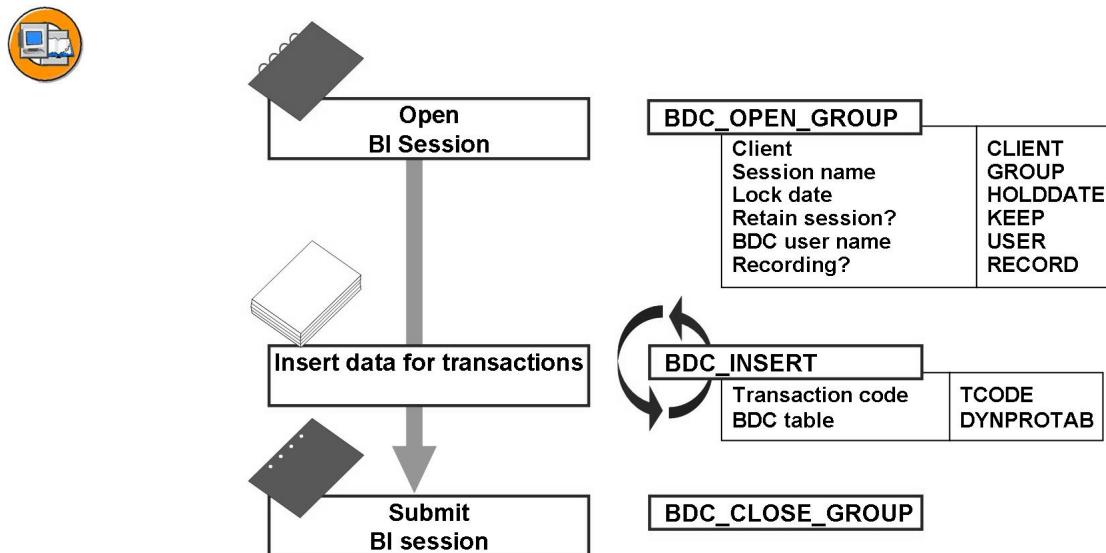


Figure 299: Function Modules for a Batch Input Program

To create batch input sessions, the following function modules are available (function group SBDC):

- **BDC_OPEN_GROUP**
Creates a new session and contains general data about the entire session.
- **BDC_INSERT**
Inserts all data for a transaction into the session. The internal table with the structure BDCDATA must contain all data that is required to process the transaction completely.
- **BDC_CLOSE_GROUP**
Closes the session after all data has been inserted. You can process the session as soon as it has been closed.

The internal table with the structure BDCDATA can contain batch input data for a single transaction flow only. The individual transactions in a loop are therefore inserted into the batch input session using several BDC_INSERT calls.

**OPEN_GROUP**
**Include
BDCRECX1**

```

FORM open_group.
  IF session = 'X'.
  :
  CALL FUNCTION 'BDC_OPEN_GROUP'
    EXPORTING CLIENT = sy-mandt
    GROUP = group
    USER = user
    KEEP = keep
    HOLDDATE = holddate.
  :
  ENDIF.
ENDFORM.

```

Figure 300: Subroutine OPEN_GROUP

The subroutine OPEN_GROUP opens the batch input session.

**BDC_TRANSACTION**
**Include
BDCRECX1**

```

FORM BDC_TRANSACTION USING tcode.

  IF session = 'X'.
    CALL FUNCTION 'BDC_INSERT'
      EXPORTING TCODE = tcode
      TABLES DYNPROTAB = bdcdata.

  ELSE.
    → 
  ENDIF.
ENDFORM.

```

Figure 301: Subroutine BDC_TRANSACTION

The subprogram BDC_TRANSACTION takes the filled BDC table and either places this in a batch input session or uses call transaction to process the data from the BDC table to the data. On the selection screen, you have the option of determining whether you want to create a session or if you want to import the data directly using call transaction. Depending on your choice, the field SESSION is set.



CLOSE_GROUP

Include
BDCRECX1

```

FORM close_group.
  IF session = 'X'.
    "close batch input group
    CALL FUNCTION 'BDC_CLOSE_GROUP'.
    :
  ELSE.
    IF e_group_opened = 'X'.
      CALL FUNCTION 'BDC_CLOSE_GROUP'.
      :
    ENDIF.
  ENDIF.
ENDFORM.
```

Close session

Close error session

Figure 302: Subroutine CLOSE_GROUP

The subroutine CLOSE_GROUP closes the batch input session. In this way, the system decided whether it is a "normal" batch input session (SESSION = 'X') or if it is a session that was opened when an error occurred in call transaction (E_GROUP_OPENED = 'X').



BDC_DYNPRO

Include
BDCRECX1

```

FORM bdc_dynpro USING program dynpro.
  CLEAR bdcdatas.
  Bdcdatas-program = program.
  Bdcdatas-dynpro = dynpro.
  Bdcdatas-dynbegin = 'X'.
  APPEND bdcdatas.
ENDFORM.
```

BDC_FIELD

```

FORM bdc_field USING fnam fval.
  IF fval <> nodata.
    CLEAR bdcdatas.
    bdcdatas-fnam = fnam.
    bdcdatas-fval = fval.
    APPEND bdcdatas.
  ENDIF.
ENDFORM.
```

Figure 303: Subroutine BDC_DYNPRO, BDC_FIELD

The include BDCRECX contains two subroutines BDC_DYNPRO and BDC_FIELD.

Subroutine BDC_DYNPRO fills the BDC table with the program name and screen number.

Subroutine BDC_FIELD fills the BDC table with the field name and field value.

Each subroutine generates a new entry in the BDC_table.



Automatically generated from the recording

```

PERFORM bdc_dynpro      USING 'SAPMF02D' '0105'.

PERFORM bdc_field      USING 'BDC_CURSOR'  'RF02D-BUKRS'.
PERFORM bdc_field      USING 'BDC_OKCODE'   '/00'.
PERFORM bdc_field      USING 'RF02D-BUKRS' record-BUKRS_001.
PERFORM bdc_field      USING 'RF02D-KTOKD' record-KTOKD_002.

PERFORM bdc_dynpro      USING 'SAPMF02D' '0110'.

PERFORM bdc_field      USING 'BDC_CURSOR'  'KNA1-SPRAS'.
PERFORM bdc_field      USING 'BDC_OKCODE'   '/00'.
PERFORM bdc_field      USING 'KNA1-NAME1' record-NAME1_003.
PERFORM bdc_field      USING 'KNA1-SORTL' record-SORTL_004.

...

```

Figure 304: Filling the BDC Table

The system generates the filling of the BDC table with data from the recording. The recorder uses the two subroutines BDC_DYNPRO and BDC_FIELD. The subroutines fill the internal table BDCDATA with the data formatted for the SAP transaction. In the course example, the data is already formatted when it is delivered.



Hint: Keep in mind that the generated structure RECORD results from the recording of an SAP transaction.

As described already, the recorder assumes that the structure fields are simultaneously the fields with converted external data. (It cannot know the actual structure of the external file.) However, because this is not the case, you must perform postprocessing. The easiest way is to ensure that the external data is already formatted when it is delivered. Then users can use the structure as in the course example, that is, without many changes except technical fields such as the checkbox (for the address) and the company code. "X" must be set manually for the checkbox, and "0001" must be set for the company code.

As a result of the programming work, the internal table BDCDATA is always filled correctly. This table contains the data for a transaction. Trace your modified program in the debugger and check whether the internal table BDCDATA is filled correctly.

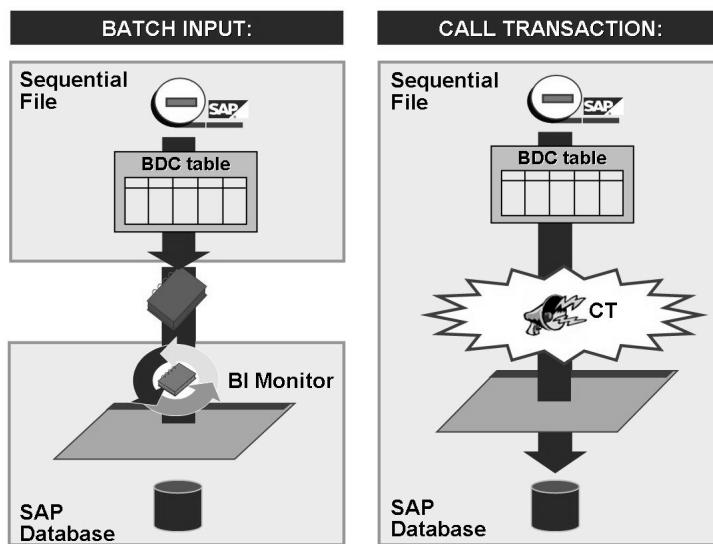


Figure 305: Differences Between Batch Input and Call Transaction

In contrast to the batch input procedure, the call transaction statement allows you to pass data directly to the dialog interface (without using a batch input session). To store screen data temporarily, you use another internal table (a BDC table that has the same structure as with batch input). Then call the relevant transaction in your program.

→ **Note:** The following slide displays all the relevant aspects of the call transaction programming. Keep in mind that in releases before the recorder was introduced the whole call transaction program had to be programmed manually. In the meantime, all the important sections of code are specified by the recorder; Users must understand all the concepts in detail only if they want to implement customer developments using the call transaction method.

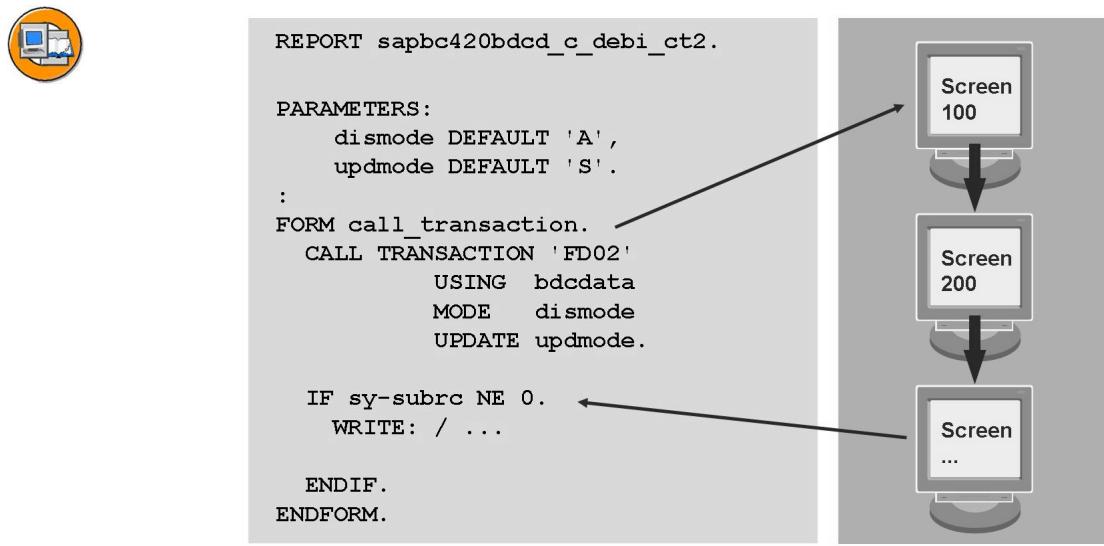


Figure 306: Call Transaction Program: Example

An additional system roll area is created for the called transaction. There, the system processes the individual screens of this transaction. As a result, the data from the table is merged. After the transaction has been processed, the second roll area is released and processing continues in the first roll area.



Hint: Unlike the batch input method, there is no error logging (error sessions). The processing of the calling transaction depends on the authorizations of the user who called it. The BDC table can include data for a single transaction flow only. Before calling an additional transaction, you must perform a REFRESH for the BDC table and refill it with data.



```
CALL TRANSACTION <transactioncode>
    [ USING    <bdc-table> ]
    [ MODE     { A | E | N } ].
```

```
REPORT xxx.
:
CALL TRANSACTION 'FD02'
    USING  bdcdata
    MODE   'E'.
:
```

Default for MODE: A

Figure 307: CALL TRANSACTION (1)

Using the parameter MODE, you can select a particular display mode for processing the transaction. You can use one of the following three modes:

A (Display all)

When you execute the program, the system displays all the screens as well as all of the data contained. This is the default value for MODE in CALL TRANSACTION.

N (Display nothing)

The system does not display the screens. It does not matter whether they contain errors or not. As soon as the transaction has been processed, the system returns control to your program. (The value of the parameter UPDATE determines whether or not you can update the database.)

E (Display errors only)

As soon as an error occurs on a screen, the system switches to display mode so that you can correct it.

These display modes are identical to those of the batch input session processing.



```
CALL TRANSACTION <transactioncode>
  [ USING    <bdc-table> ]
  [ MODE     { A | E | N } ]
  [ UPDATE   { A | L | S } ] .
```

```
REPORT xxx.
:
CALL TRANSACTION 'FD02'
  USING bdcdata
  UPDATE 'L'.
:
```

Default for UPDATE: A

Figure 308: CALL TRANSACTION (2)

You use the parameter UPDATE to control how the database updates caused by a transaction are to be executed. You can use one of the following three modes:

Asynchronous Update

The called transaction does not wait until the database updates have been executed, but passes the update to the SAP update task. This generally means that the call transaction program is executed more quickly. This is NOT recommended for large datasets because the transaction that was called does not receive a closing message from the update module. The batch input program that called the transaction cannot establish whether or not the transaction ended with a successful database update. Use the function for updating records (transaction SM13) to establish whether or not the update was terminated prematurely. The functions for analyzing and correcting errors are not as easy to use as those for synchronous update.

Synchronous Update

In the case of synchronous update, the called transaction waits until all updates have been completed. Processing is therefore slower than with asynchronous update. However, the calling transaction may report any update errors to the program. This simplifies the process of analyzing and correcting errors.

Local Update

In the case of local update, the database is not updated in a separate process, but in the calling program process. (See the online documentation about the ABAP keyword SET UPDATE TASK LOCAL).



```
CALL TRANSACTION <transactioncode>
  [ USING    <bdc-table> ]
  [ MODE     { A | E | N } ]
  [ UPDATE   { A | L | S } ]
  [ MESSAGES INTO <messtab> ] .
```

```
REPORT xxx.
DATA: m_tab TYPE TABLE OF bdcmsgcoll.
:
CALL TRANSACTION 'FD02'
  USING bdcdata
  MESSAGES INTO m_tab.
:
```

Figure 309: CALL TRANSACTION (3)

MESSAGES specifies that all system messages that are output during the CALL TRANSACTION are written to the internal table <MESTAB>. The internal table must have the structure BDCMSGCOLL.

Unlike classic batch input processing using sessions, CALL TRANSACTION does not offer any particular processing procedure for incorrect transactions. There are no restart functions for transactions that contain errors or that cause database updates to terminate.



```
CALL TRANSACTION <transactioncode>
  [ USING    <bdc-table> ]
  [ MODE     { A | E | N } ]
  [ UPDATE   { A | L | S } ]
  [ MESSAGES INTO <messtab> ]
  [ OPTIONS FROM opt ] .
```

```
REPORT xxx.
DATA: my_opt TYPE ctu_params.
CALL TRANSACTION 'FD02'
  USING bdcdata
  OPTIONS FROM my_opt.
:
```

Figure 310: CALL TRANSACTION (4)

The OPTIONS addition enables you to control the call transaction statement using a structure. The structure must have the dictionary type CTU_PARAMS.

The OPTIONS FROM addition must not be combined with the MODE addition or with UPDATE. More details are provided on the next slide.



Structure CTU_PARAMS

- DISMODE Processing mode (the same as the MODE addition)
- UPDMODE Update mode (the same as the UPDATE addition)
- CATTMODE CATT mode (control as in CATT mode)
- DEFSIZE Processing with the screen default size
- RACOMMIT No termination with COMMIT WORK
- NOBINPT No batch input session; that is, SY-BINPT = SPACE
- NOBIEND No batch input session after the end of the BDC data

The individual components have the following meanings:

The components DEFSIZE , RACOMMIT, NOBINPT, NOBIEND can expect the following values: 'X' = yes, ' ' = no

If the OPTIONS FROM addition is not specified, the following settings are valid for the control parameter.

- DISMODE of the MODE addition
- UPDMODE of the addition
- UPDMODE of the UPDATE addition
- CATTMODE no CATT active
- DEFSIZE Processing occurs in the screen default size
- RACOMMIT Successful end with COMMIT WORK
- NOBINPT Batch input session active, that is SY-BINPT
- NOBIEND Batch input session active, even after end of the BDC data



Return value:

Value	Description
0	successful
< 1000	Errors in transaction
> 1000	Errors during call transaction processing, see the specified system fields

System fields:

Field name	Description
SY-MSGID	Message ID
SY-MSGTY	Message - Type (E,I,W,S,A)
SY-MSGNO	Message - Number
SY-MSGV1	Message - Variable 1
SY-MSGV2	Message - Variable 2
SY-MSGV3	Message - Variable 3
SY-MSGV4	Message - Variable 4

Figure 311: Return Code and System Fields

To find out whether the processing of the transaction was successful or not, you can read the return code value. In addition, other system fields are filled: Number, identification, type and variables of the dialog message issued by the transaction called. Using this information, you can issue the message text.



```

REFRESH MESSTAB.
CALL TRANSACTION tcode USING bdcdata
      MODE ctumode
      UPDATE cupdate
      MESSAGES INTO messtab.

l_subrc = sy-subrc.
LOOP AT messtab.
  :
ENDLOOP.           → Output of the message table
IF l_subrc <> 0 AND e_group <> SPACE.
  IF e_group_openend = ' '.
    CALL FUNCTION 'BDC_OPEN_GROUP'
      EXPORTING ..... .
  E_group_opened = 'X'.
  ENDIF.
  CALL FUNCTION 'BDC_INSERT'
    EXPORTING TCODE      = tcode
    TABLES     DYNPROTAB = bdcdata.
ENDIF.
:
REFRESH bdcdata.

```

Open error session

Place data record with
error in error session

Figure 312: The Generated Program of the TA Recorder

If you chose call transaction on the selection screen, the generated program runs as follows:

- The online transaction is called using call transaction and the BDC table is transferred. All messages that occur during the call transaction are available in the internal table MESSTAB.
- The MESSTAB is issued as a list.
- If you specified an error session on the selection screen (E_GROUP \diamond SPACE) and an error occurs in the call transaction (L_SUBRC \diamond 0), the system opens the error session. However, the error session is opened once only. The system then places the data record in the session.

All other data records for which an error occurs in call transaction are placed in the same session.

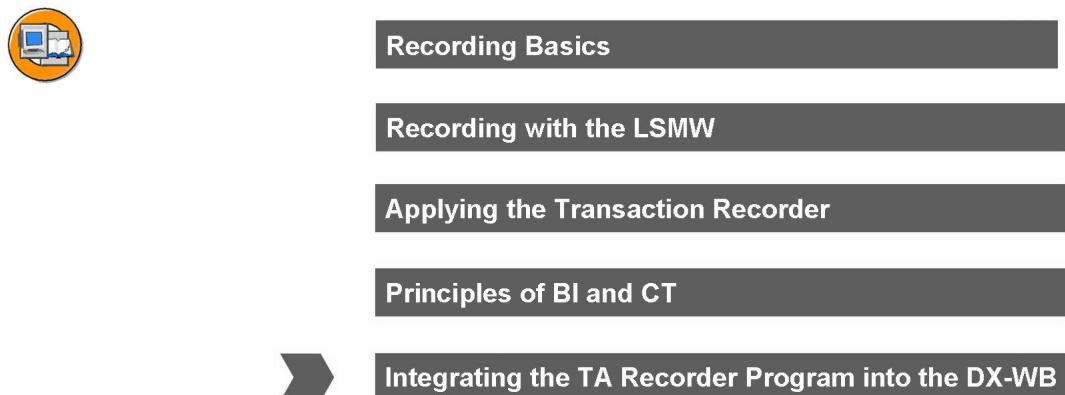


Figure 313: The Transaction Recorder



- Registering the generated program
- All functions of standard objects are supported
- Exception: The function
Create file with test program

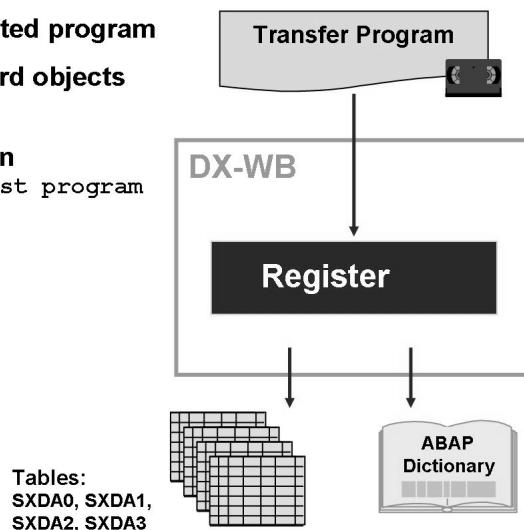


Figure 314: Registering in the DX-WB

→ **Note:** It is not necessary to register in the DX-WB the program that was generated using the recorder and then adjusted. However, if you want to use the data as a complete loading step or batch input step you must register it.

You can integrate the generated program into the DX-WB. All the functions in the DX-WB and LSMW are available for the generated program. The only exception is that test data cannot be generated in the DX-WB from a test program. The structure of the file must be created in the ABAP Dictionary, and the tables SXDA0, SXDA1, SXDA2 and SXDA3 must be filled with the structure information.

The program generated by the transaction recorder is registered in the DX-Workbench library. All required entries are automatically made in the ABAP Dictionary and in the SXDA tables. To do this, select a business object (for example, KNA1, for a customer). Register the program under the **task type LOA** and **program type BINP**.

Create a name for the generated program and assign a structure name to it. This structure is then created in DDIC. Make sure you do not use any hyphens.

Exercise 17: Recording Function in the LSMW

Exercise Objectives

After completing this exercise, you will be able to:

- Create a recording using the LSMW.
- Generate a batch input session.

Business Example

Change the address data of customers in the R/3 system.

Name of the recording: DEBI_##

Name of the file: BC420-##-BIFILE

Task 1:

Use the LSMW to create the recording DEBI_##. Record the transaction FD02 (change customer).

1. Before you create the recording, test the transaction online once.

Use customer Z-##-10001 for the recording. On the FD02 initial screen, only maintain the following fields:

Customer: Z-##-10001

Company code: 0001

Address data: Select



Hint: Important: When recording, the data record must be saved.

2. Maintain the name and the recording for the following fields: *Name, street, city, postal code, and telephone number*. It is useful to use the same technical names as displayed in the table below (automatic field mapping). Delete all other fields of screen 110 from the recording (for example SORTL, LAND1, SPRAS).

Continued on next page

Task 2:

In your project (BC420-##) create the subproject DEBI-## and the new object TA_##.

Enter the object attributes, choose Batch Input Recording and choose your recording DEBI_##.

Create the source structure address. Create the source fields according to the table specified below or copy this from the project BC420-S_DE, subproject DEBI-00, object TA-00.

1. File Structure: The data from the external systems is displayed in the following table:

Field name	Field Type	Field Length	Description
NO	C	16	Customer number
NAME	C	35	Name
STREET	C	35	Street
CITY	C	35	City
ZIP	C	10	Postal code
TEL	C	16	Telephone number

2. Maintain the field mapping. Use the automatic field mapping (choose the menu option: *Extras*)
3. Specify the files. Maintain the file name of the source file only.

Name of the file BC420-##-BIFILE

This file includes the customer Z_##_5xxxx.

Code page: 1100

4. Import and convert the data. Display the data individually.
5. Create the batch input session.
6. Run the session.

Solution 17: Recording Function in the LSMW

Task 1:

Use the LSMW to create the recording DEBI_##. Record the transaction FD02 (change customer).

1. Before you create the recording, test the transaction online once.

Use customer Z-##-10001 for the recording. On the FD02 initial screen, only maintain the following fields:

Customer: Z-##-10001

Company code: 0001

Address data: Select



Hint: Important: When recording, the data record must be saved.

- a) There is no solution.
2. Maintain the name and the recording for the following fields: *Name, street, city, postal code, and telephone number*. It is useful to use the same technical names as displayed in the table below (automatic field mapping). Delete all other fields of screen 110 from the recording (for example SORTL, LAND1, SPRAS).
 - a) There is no solution.

Task 2:

In your project (BC420-##) create the subproject DEBI-## and the new object TA_##.

Enter the object attributes, choose Batch Input Recording and choose your recording DEBI_##.

Create the source structure address. Create the source fields according to the table specified below or copy this from the project BC420-S_DE, subproject DEBI-00, object TA-00.

1. File Structure: The data from the external systems is displayed in the following table:

Continued on next page

Field name	Field Type	Field Length	Description
NO	C	16	Customer number
NAME	C	35	Name
STREET	C	35	Street
CITY	C	35	City
ZIP	C	10	Postal code
TEL	C	16	Telephone number

- a) There is no solution.
2. Maintain the field mapping. Use the automatic field mapping (choose the menu option: *Extras*)
 - a) There is no solution.
3. Specify the files. Maintain the file name of the source file only.
Name of the file BC420-##-BIFILE
This file includes the customer Z_##_5xxxx.
Code page: 1100
 - a) There is no solution.
4. Import and convert the data. Display the data individually.
 - a) There is no solution.
5. Create the batch input session.
 - a) There is no solution.
6. Run the session.
 - a) There is no solution.

Exercise 18: Recording with the TA Recorder

Exercise Objectives

After completing this exercise, you will be able to:

- Create a recording using the TA recorder
- Test the recording.

Business Example

Change the address data of customers in the SAP R/3 system.

Name of the recording: TA_DEBI_##

Name of the file: BC420_##_BIFILE

Task 1:

Use the transaction recorder to create the recording TA_DEBI_##. Record the transaction FD02 (change customer).

1. Use customer Z-##-10001 for the recording. On the FD02 initial screen, only maintain the following fields:

Customer: Customer Z-##-10001

Company code: 0001 Address data: Select

Record the following fields:

Name, street, city, postal code, and telephone number.



Hint: Important: When recording, the data record must be saved.

Delete all fields that do not exist in the source file BC420_##_BIFILE from screen 110 in the recording (for example KNA1-SORTL, KNA1-LAND1, KNA1-SPRAS).



Hint: Do not delete fields RF02D-BUKRS and RF02D-D0110.

Save the recording.

Continued on next page

Task 2:

Switch to the recordings overview.

1. Test the recording using processing mode “A” (process in the foreground).
2. Test the recording with the processing mode “N” (background processing).
3. If the recording has been tested and no error has occurred, generate a batch input session and run it using the batch input monitor. If an error should occur, generate the recording again.

Task 3:

From the recording, generate the program Z##DEBI_REC in your package ZBC420##. Select the following:

- Read field contents from file
 - Create test data in the file BC420##DEBI_TEST.txt
1. Save this program and execute it. Generate and process a batch input session in each case. Carry out the same steps again using the call transaction.

Task 4:

The generated program should now be changed so that file BC420##BIFILE can be imported and processed. The file contains the customer with the number Z##-5xxxx, which was created in the previous chapter.

1. Change the default value of the file name (parameter dataset) in BC420##BIFILE.
2. Change the structure definition. Delete the company code field and the address data checkbox from the structure.
3. Change the sub-routine call perform bdc_field for the company code ('RF02D-BUKRS') so that the value 0001 is transferred.
4. Change the sub-routine call perform bdc_field for the address data checkbox ('RF02D-D0110') so that the value X is transferred.
5. Start the program again and read the data from the file BC420##BIFILE with customer data. Use this to generate and process a batch input session.

Solution 18: Recording with the TA Recorder

Task 1:

Use the transaction recorder to create the recording TA_DEBI_#. Record the transaction FD02 (change customer).

1. Use customer Z-##-10001 for the recording. On the FD02 initial screen, only maintain the following fields:
Customer: Customer Z-##-10001
Company code: 0001 Address data: Select
Record the following fields:
Name, street, city, postal code, and telephone number.



Hint: Important: When recording, the data record must be saved.

Delete all fields that do not exist in the source file BC420_##_BIFILE from screen 110 in the recording (for example KNA1-SORTL, KNA1-LAND1, KNA1-SPRAS).



Hint: Do not delete fields RF02D-BUKRS and RF02D-D0110.

Save the recording.

- a) There is no solution.

Task 2:

Switch to the recordings overview.

1. Test the recording using processing mode “A” (process in the foreground).
 - a) There is no solution.
2. Test the recording with the processing mode “N” (background processing).
 - a) There is no solution.
3. If the recording has been tested and no error has occurred, generate a batch input session and run it using the batch input monitor. If an error should occur, generate the recording again.
 - a) There is no solution.

Continued on next page

Task 3:

From the recording, generate the program Z_##_DEBI_REC in your package ZBC420_##. Select the following:

- Read field contents from file
 - Create test data in the file BC420_##_DEBI_TEST.txt
1. Save this program and execute it. Generate and process a batch input session in each case. Carry out the same steps again using the call transaction.
 - a) There is no solution.

Task 4:

The generated program should now be changed so that file BC420_##_BIFILE can be imported and processed. The file contains the customer with the number Z-##-5xxxx, which was created in the previous chapter.

1. Change the default value of the file name (parameter dataset) in BC420_##_BIFILE.
 - a) There is no solution.
2. Change the structure definition. Delete the company code field and the address data checkbox from the structure.
 - a) There is no solution.
3. Change the sub-routine call perform bdc_field for the company code ('RF02D-BUKRS') so that the value 0001 is transferred.
 - a) There is no solution.
4. Change the sub-routine call perform bdc_field for the address data checkbox ('RF02D-D0110') so that the value X is transferred.
 - a) There is no solution.
5. Start the program again and read the data from the file BC420_##_BIFILE with customer data. Use this to generate and process a batch input session.
 - a) There is no solution.



Lesson Summary

You should now be able to:

- Use the recording function in the LSMW to generate a batch input session from the data of the external system
- Use the Transaction Recorder to generate some data transfer programs
- Analyze programs that work with batch input or a call transaction
- Include the programs of the Transaction Recorder in the DX-WB



Unit Summary

You should now be able to:

- Use the recording function in the LSMW to generate a batch input session from the data of the external system
- Use the Transaction Recorder to generate some data transfer programs
- Analyze programs that work with batch input or a call transaction
- Include the programs of the Transaction Recorder in the DX-WB

Unit 11

Background Processing and Special Techniques

Unit Overview

After you have tested the data transfer procedure online, you can also use the procedure in the background (batch). This reduces the load on the limited online work processes and also offers additional scheduling options. Under certain conditions, it is also possible to schedule the transfers periodically. The interval rate is the decisive factor here.



Unit Objectives

After completing this unit, you will be able to:

- Execute the data transfer in the background using the LSMW and DX-WB tools
- Describe and evaluate how the LSMW and DX-WB can be used for periodic data transfer

Unit Contents

Lesson: Background Processing	376
Exercise 19: Scheduling Batch Input in the Background.....	393

Lesson: Background Processing

Lesson Overview

Contents



- Background Processing
- Periodic Data Transfer



Lesson Objectives

After completing this lesson, you will be able to:

- Execute the data transfer in the background using the LSMW and DX-WB tools
- Describe and evaluate how the LSMW and DX-WB can be used for periodic data transfer

Business Example

In the background, you want to periodically schedule a data transfer that was tested in online operation. DX-WB and LSMW offer different options for this.



Background Processing

Periodic Data Transfer

Figure 315: Contents – Background Processing (1)

Scheduling a data transfer procedure in the background



- **Advantages**
 - Runtimes can be longer than in dialog mode.
 - The load can be distributed (RSBDCSUB, background server)
 - Logged processing (jobs)
- **Prerequisites “Go life” (data transfer)**
 - Data transfer procedure has been thoroughly tested
 - “Go life” Data from the external system system is available

The DX-WB and the LSMW can transfer the data in the background.

Some data transfer techniques can be carried out in the background without the LSMW and the DX-WB. The prerequisite for this is that the data transfer file is already in the correct format.

The advantages of using background processing over online processing are:

- Dedicated servers
- Exclusive use of background work processes
- Runtimes may be longer than in dialog mode (rdisp/max_wprun_time).

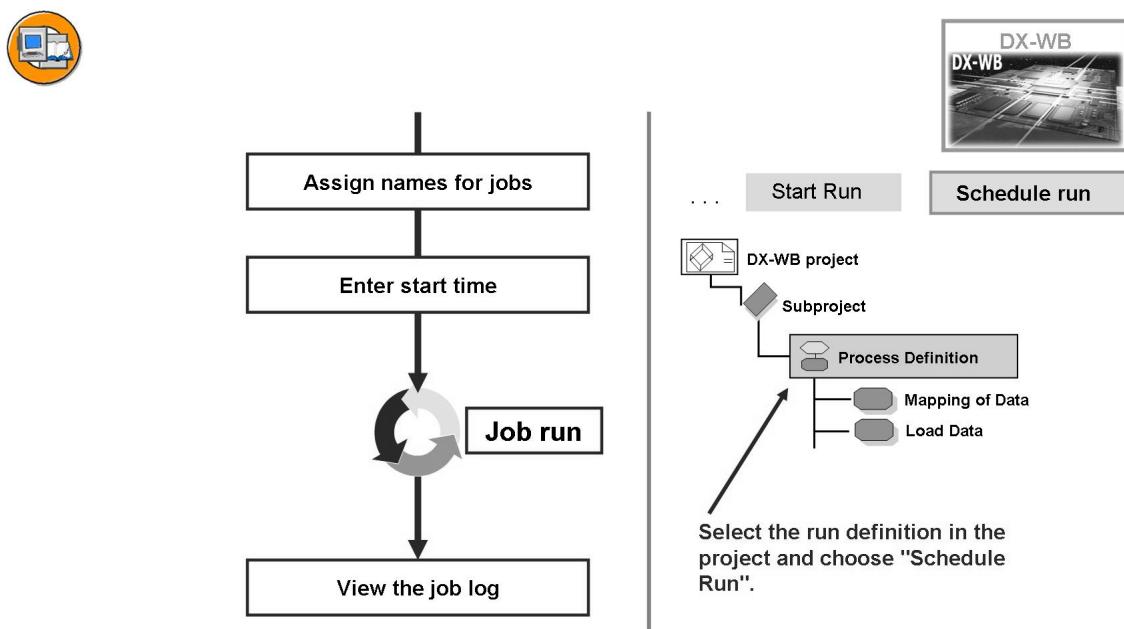


Figure 316: Schedule the DX-WB Run in the Background

In the DX-WB, you can choose “Schedule Run” to schedule a run in the background.

Choose the name and the start time of the job. It is not necessary to select a dedicated background server.

Background processing is useful only if all tasks within the job can run in the background.

→ **Note:** After the background processing, the results of the DX-WB run can be viewed in the log of the DX-WB. Only formal information about the run is available under the job log in the job overview (SM37); There, the system logs whether or not the run was successful. Therefore, you must check the details in the DX-WB. Similarly, for example, the generated document master number or asset master number is not displayed in the job log.

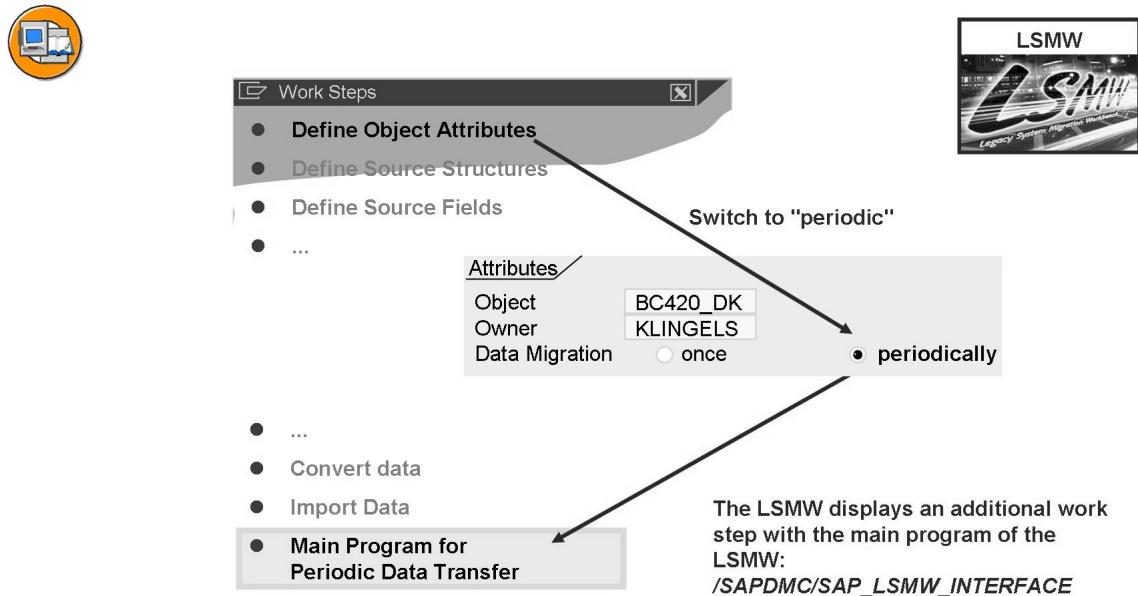


Figure 317: The LSMW Main Program

Data can also be transferred without the DX-WB, it can be transferred with the framework program of the LSMW and its various selection parameters only.

In the object attributes of an LSMW object, you can define whether the data transfer should occur once or periodically.

For periodic data transfer, there is an additional work step "**Frame Program for Periodic Data Transfer**".

The framework program **/SAPDMC/SAP_LSMW_INTERFACE** makes it possible to start the LSMW automatically.

It is possible to enter a number of parameters of this framework program using a selection screen.

To do this, you must create another variant for this LSMW framework program for the background run and specify the job.

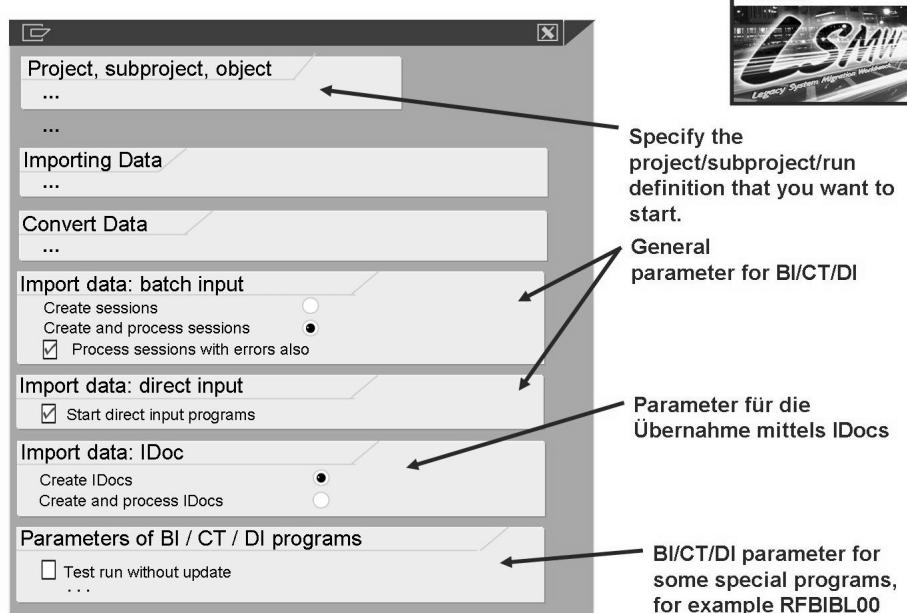


Figure 318: The Selection Screen of the LSMW Framework Program

The selection screen of the LSMW framework program offers many selection parameters for transferring data using the methods of batch input, call transaction, direct input and IDoc.

The lower section of the selection screen contains specific parameters for some specific batch input, call transaction or direct input programs, for example RFBIBL00. A list of these special programs and their parameters is available in the LSMW documentation under “Periodic Data Transfer”.

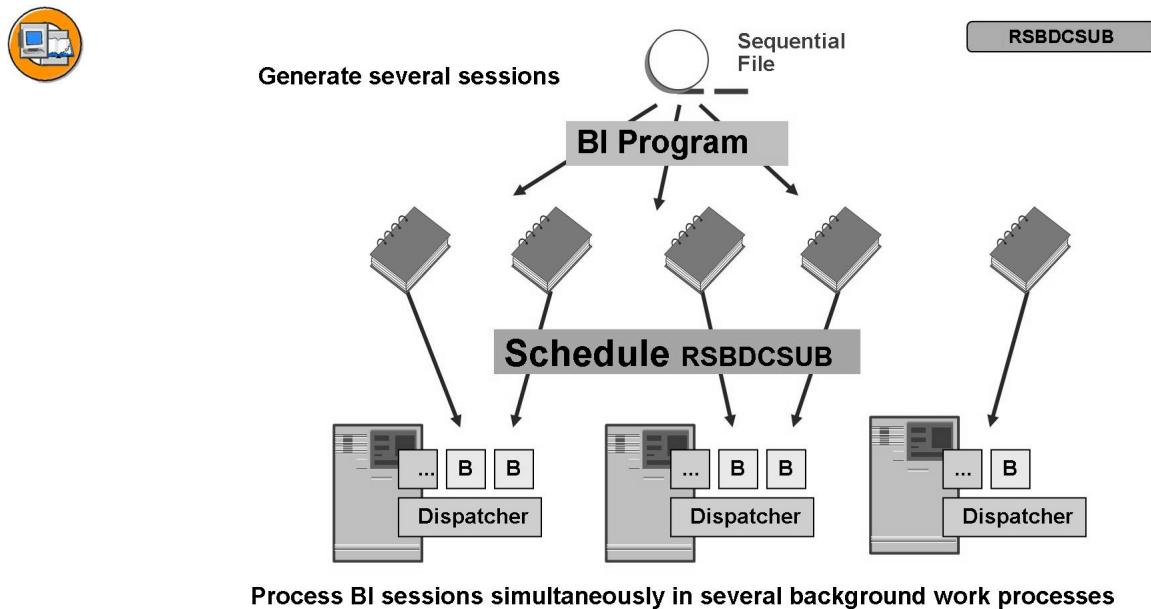


Figure 319: Parallel Batch Input Processing Using RSBDCSUB

Batch input programs can also be used without the DX-WB or LSMW. The file you want to import must already have the same format as the target file. In this case, the transfer procedure creates batch input sessions, which can then be processed automatically.

Batch input sessions can be processed in parallel in the background using the program **RSBDCSUB**. If several batch input sessions have been created from the external data, RSBDCSUB can, for each batch input session, schedule a job that processes the session.

The names of the batch input sessions can be maintained on the selection screen of RSBDCSUB (also generically “bc420*”).

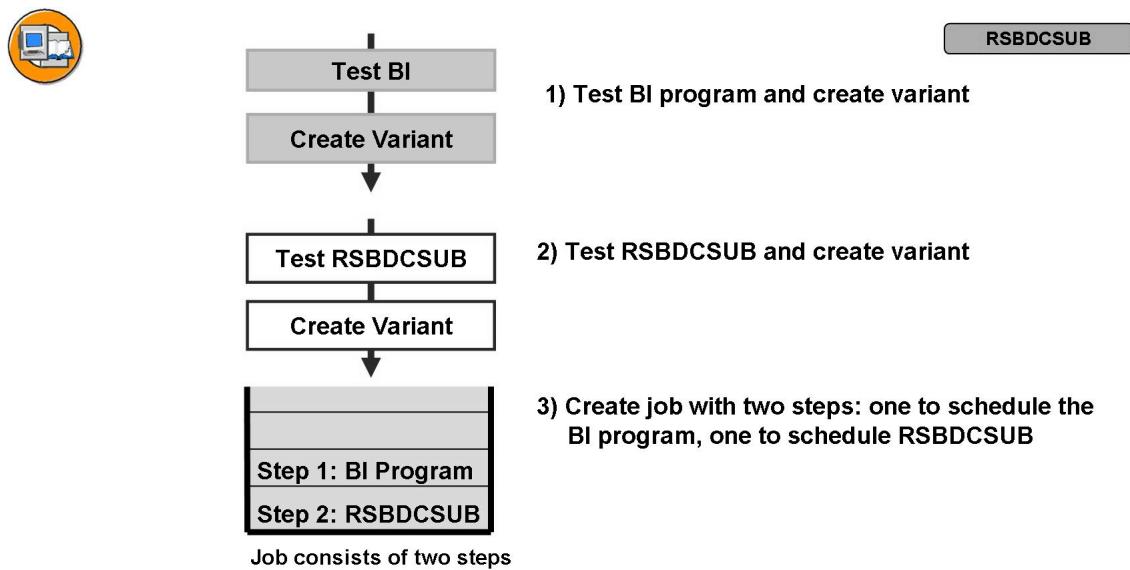


Figure 320: Scheduling Batch Input Using RSBDCSUB

- You can use program RSBDCSUB to schedule the sessions to be processed in the background.
 - To do this, proceed as follows:
 1. Test the batch input program or the batch input data transfer and create a variant for the batch input program. The variant should contain the required parameters, such as the file name.
 2. Call the program RSBDCSUB and test it so that you become familiar with the input parameters on the selection screen and the functions of the program.
- Create a variant for program RSBDCSUB. The selection criteria are:
- Session name (you can also enter a generic name, for example “bc420*”)
 - Date created
 - session status
 - The target host is not normally specified (load distribution occurs automatically)
 - Extended log
3. Create a background job for the batch input program and the program RSBDCSUB with one of the variants you defined.
 - Step 1 consists of calling the actual BI program that creates one or more BI sessions.
 - Step 2 consists of calling RSBDCSUB, which automatically transfers the BI sessions created earlier to background processing.



Background Processing



Periodic Data Transfer

Figure 321: Contents – Background Processing (2)



- Which is the best technique and which is the best workbench to use for periodic data transfer?

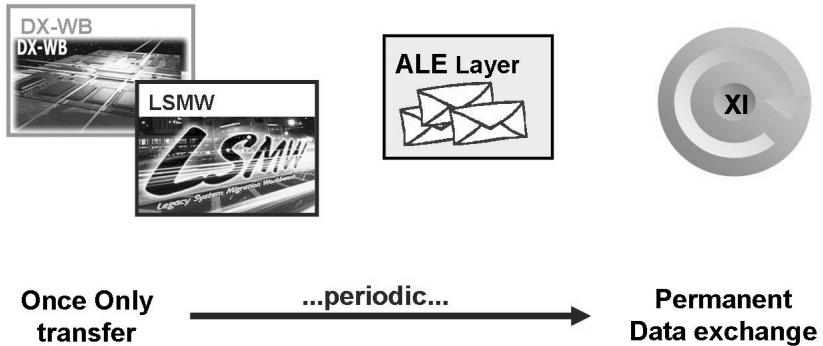


Figure 322: Periodic Transfer: Overview

The main tasks of the DX-WB and LSMW are initial data transfer.

Both tools have limitations when used for transferring data periodically from a legacy system.

The following factors play an important role:

- Data quantity
- Data quality
- Time interval of periodic data transfer

Depending on these factors, other transfer techniques could be more appropriate for periodic data transfer. For example, ALE technology or SAP Exchange Infrastructure (XI).

Periodic Data Transfer: Requirements



- Important requirements for periodic transfer:
 - Restart capability
 - Creating logs
 - Workflow support
 - Distributed data exchange
 - Monitoring of process chain

There are additional requirements for periodic data transfer. These requirements must be met to ensure that periodic transfer works smoothly over a long period of time.

- Restart capability is necessary to continue incomplete transfers at the correct point. Incomplete transfers must be completed before the next periodic transfer is due.
- A log must be created for each periodic transfer to evaluate the transfer results.
- Since periodic transfers run automatically, a message must be output if an error arises.
- Depending on the data it may be necessary to distribute it to more than one receiving system or to process it on a different system.
- The periodic transfer process must be available on a daily basis and it must be monitored.

Application Areas of DX-WB and LSMW



- **What can DX-WB and LSMW be used for?**
 - One-off data transfers from one or more source systems to the SAP system, SAP CRM, SAP APO System
 - Periodic data transfer with a maximum of one transfer per day (may vary in special cases)
- **What can DX-WB and LSMW NOT be used for?**
 - Small datasets with a frequent time interval (once to several times daily)
 - Data distribution - the content of the data has to be distributed to different systems
 - XML-based technology
 - Source system provides XML or IDocs

The DX-WB and LSMW support data transfer in the following target systems:

- SAP R/3 (4.0, 4.5, 4.6, Enterprise, and so on)
- SAP CRM (3.0, 3.1, 4.0, and so on)
- SAP APO (2.0, 3.0, 3.1, and so on)

Both tools have limitations when used for periodic transfer.

Neither tool should be used for data transfer with frequent time intervals (several transfers each day).

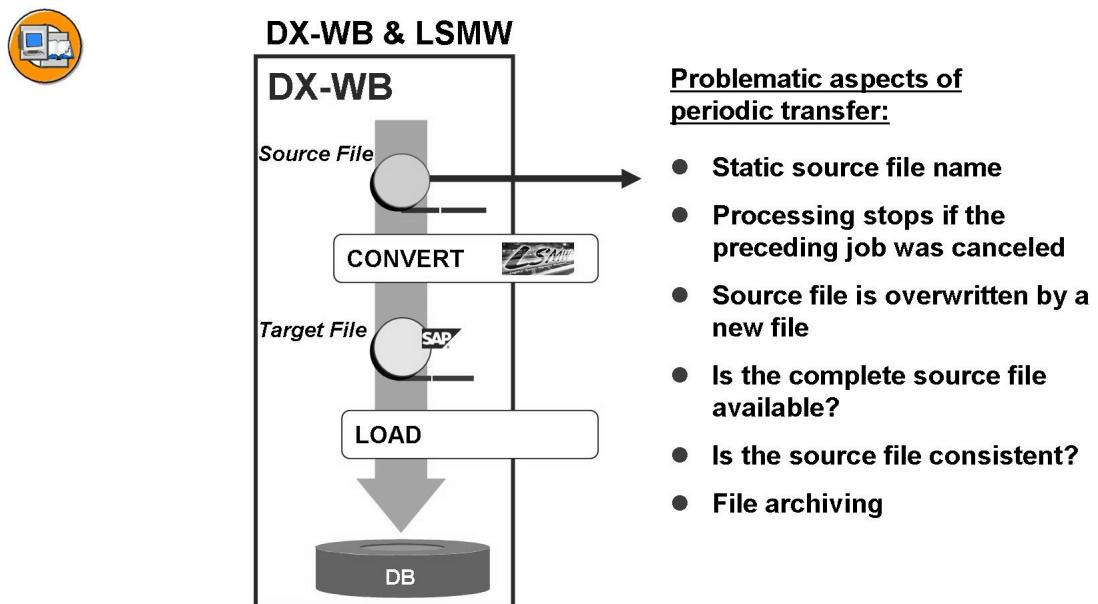
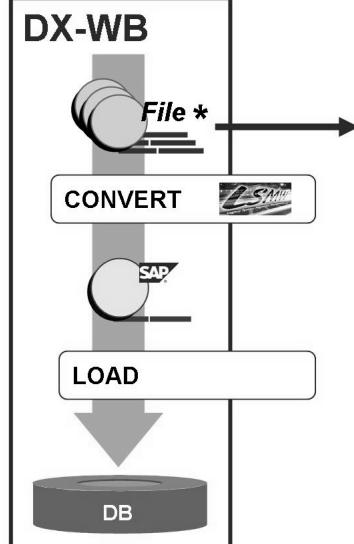


Figure 323: Periodic Data Transfer: Critical Points

The following limitations play an important role when the DX-WB or the LSMW is used for periodic data transfer:

- The source file name must normally be assigned statically in the LSMW: That is, the same file is always read. Therefore, you must ensure that this file contains new data before each periodic run is started. Dynamic file names are also possible in the LSMW. We will go into this in detail at a later stage.
- If a run is cancelled in the DX-WB, any subsequent periodic run that uses the same run definition cannot be started.
- You must ensure at the start of each periodic run that the source file exists and is complete. The cause of an error could be that the source file had not yet been fully written or that the creating or copying process was canceled.
- You must ensure at the start of each periodic run that the source file exists and is complete. The cause of an error could be that the source file had not yet been fully written or it contains incorrect data.
- For periodic runs, a source file must always be freshly created in advance. It is good practice to archive each source file after it has been transferred.
- The following slides show how these problems can be avoided.

The importance of these factors depends on the data transfer project.

**DX-WB & LSMW****Enhancing the Standard Functions**

- (1) Dynamic file names using wildcards (*) in LSMW
- (2) File names determined in DX-WB using separate function modules
- (3) Enhancing the DX-WB log
- (4) Flag file in LSMW used for handshaking

Figure 324: DX-WB / LSMW: Enhancing the Functions

As mentioned earlier the DX-WB and the LSMW can be used with restrictions for periodic data transfer.

The following utilities are provided for this:

- “Wildcards” allow different source file names to be used in the LSMW.
- Users can use a separate function module in the DX-WB to determine the file name themselves.
- The log can be extended in the DX-WB to display additional information, error situations, and so on.
- Flag file in LSMW is used as the “handshake” method. This improves the reconciliation or synchronization with the external system supplying the source file.

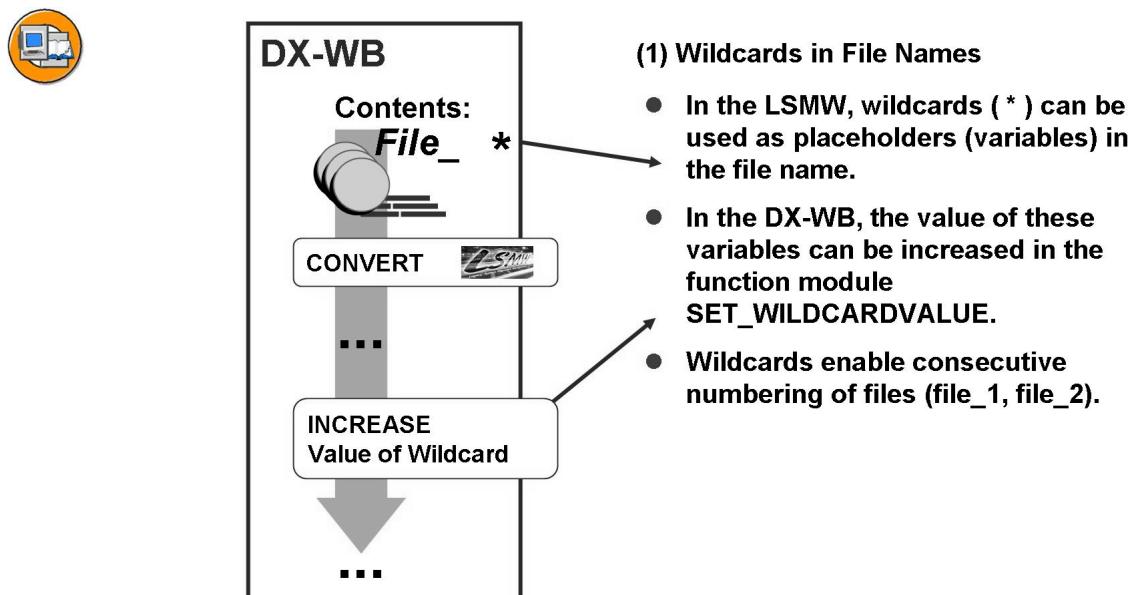


Figure 325: Dynamic File Names: Wildcards in the LSMW

Procedure when **using wildcards**:

The external system provides a file with the data to be imported (datei_1).

The DX-WB starts the first task (convert). In the LSMW, the name of the source file is specified with an asterisk (wildcard) (file_*). The value applying to the wildcard is created in *Wildcard Values*. The value is manually set to 1 for the first transfer.

The LSMW reads the file, "file_1" and executes the conversion. The converted data is saved in a fixed file name (for example, file.lsmw.conv).

The DX-WB starts the second task (load data). This task may also be performed later.

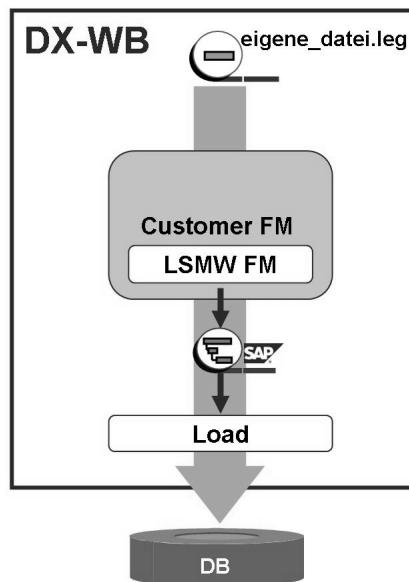
If the "Load" task was successfully completed, the third task is started in the DX-WB ("INCREASE" → Increase Wildcard / Rename Files).

The function module /SAPDMC/SET_WILDCARDVALUE is used for this step. This function module renames the source file - file_1 becomes file_2.

Renaming ensures that the same file is not used again.

The wildcard value is then increased by one and the external system can provide the next file (datei_2).

Consecutive numbering of source file names ensures that the source file can never be imported more than once.



(2) Determining the file name yourself

- **Procedure**

- Customer creates his/her own function module
- The customer function module determines the current file name at runtime (for example, the table containing the file name and status flag).
- The customer FM calls function module /SAPDMC/LSM_RUN_FROM_DXWB, and transfers the relevant file details

- **Advantage**

- Name of source file can be determined anew for each run

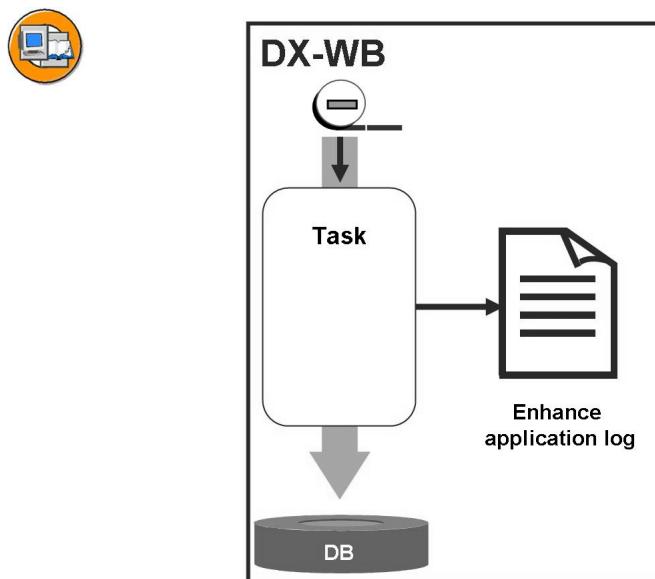
Figure 326: Dynamic File Names: Customer FM Calls LSMW Using File Names

Further ways to dynamically modify the file name:

The customer develops a function module (FM). This function module is registered in the DX-WB. It can determine the required file names for the source files in different ways (one or more ways are possible). The file name can be determined from a table (select ...) or from a file (read dataset). As a result, the user would have the option of working with time stamps in file names, for example. The external system could deliver the file names with time stamps and store these file names in a special table or file. The customer function module then reads the file names from the table or file and calls the LSMW function module /SAPDMC/LSM_RUN_FROM_DXWB to call up the LSMW with the dynamic file.

The function module has the following interface:

Import parameters:	PROJECT	LSMW project name
	SUBPROJ	LSMW Subproject name
	OBJECT	LSMW object name
	FILE_READ	Name of the read file
	FILE_CONV	Name of the converted file
Table	T_SRCSTRFILE	Structure of the source file and source structure
Export:	RETURN	Return Code



(3) Enhance the DX-WB log

- The log created by the DW-WB can be extended.
- To do this, the following function modules must be used:
 - 'BAL_LOG_CREATE'
 - 'BAL_LOG_MSG_ADD'
 - 'BAL_DB_SAVE'

Figure 327: DX-WB: Enhance Log (Application Log)

Log: Create with header data

- You can open a log in the application log using the function module **BAL_LOG_CREATE**.
- When you call **BAL_LOG_CREATE** you receive the log handle (**LOG_HANDLE**, CHAR22).

The **LOG_HANDLE** is a **GUID** (globally unique identifier), which uniquely identifies a log. You can use this handle to access the log.

Message: Insert

- If you call function module **BAL_LOG_MSG_ADD**, messages can be added to the log using the IMPORTING parameter **I_S_MSG**.

Message: Save

- Logs in the main memory can be saved using function module AL_DB_SAVE. The log from the main memory can be stored in the database using the importing parameter **I_T_LOG_HANDLE**.



Hint: Function modules developed by customers should have the field return with the structure /SAPDMC/LSSDXRETURN as their return value. This structure contains the LOG_HANDLE field. This field must be set with the value of the **LOG_HANDLE**. The function module log will then appear as a lower-level log in the DW-WB log hierarchy.

Structure /SAPDMC/LSSDXRETURN is used also when integrating the LSMW as a task in the DX-WB. This means the log can be extended in the LSMW as described above.

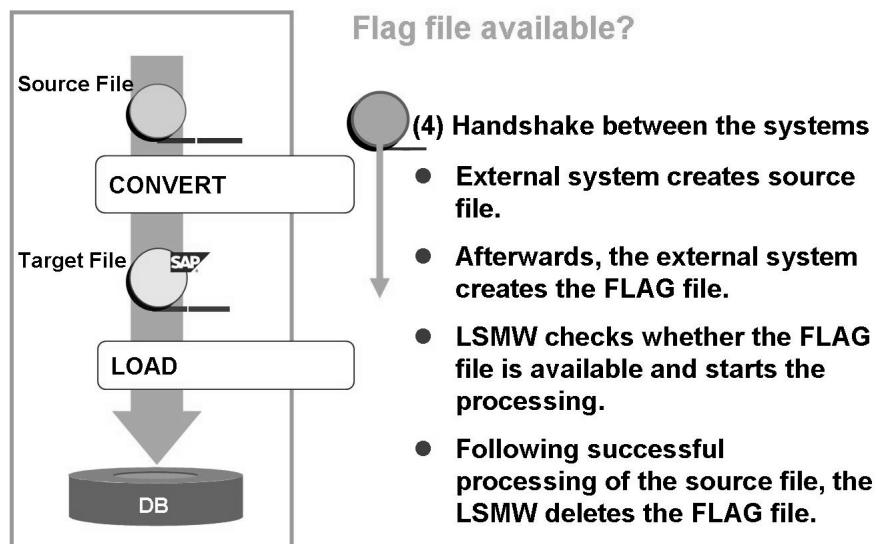


Figure 328: LSMW: Flag File for Periodic Transfer



Caution: The following procedure can be used only when using the LSMW, it cannot be combined with the DX-WB.

The flag file in the LSMW is used to establish a “handshake” with the application (external system) that provides the input file(s):

- The main program for periodic transfer will execute its tasks, only if the specified flag file exists.
- Once the data transfer is completed, the main program for periodic data transfer deletes the flag file.
- The “delivering” application should perform complementary tasks:
 - Before new files are created, it checks if the flag file exists.
 - If it does, the program stops.
 - Otherwise the source files and the flag file are created.

Disadvantage: This procedure works without the DX-WB.

Exercise 19: Scheduling Batch Input in the Background

Exercise Objectives

After completing this exercise, you will be able to:

- Schedule batch input sessions in the background
- Perform parallel processing

Business Example

You want to automate the process of transferring external data. To do this, schedule a special ABAP program (RSBDCSUB) using background processing in the R/3 System. Several scheduling options exist.



Hint: Job: BC420_##_Job

Task 1: Optional

You want to start a job that consists of one step using the date/time.

1. Define a variant for the program SAPBC420_SPTD_LIST_CHECKBOXES.
2. Create the job BC420JOB##-1, that is started using the date/time. As a start date, you can enter *Start Immediately* or, for example, the system time plus three minutes. The job should contain the ABAP report as the only step.
3. Trace the job process using transaction SM37 and analyze the job log.
Trace the job process using transaction SM37 and analyze the job log.

Task 2:

Define and schedule a job to generate and process batch input sessions. You want to define **one job BC420_##_Job** that execute the following tasks in **two steps**:

- a) Process the program to generate one or several batch input sessions. First you must have created a variant for the batch input program to be scheduled.
- b) Call program RSBDCSUB to schedule the batch input session that was generated in the previous step. First you must have created a variant for the program RSBDCSUB.

1. Process Steps:

Create the variant VAR_## for program RFBIBL00. Maintain the following data:

Continued on next page

- File names = (For the file name, see the outbound file or the converted data in the LSMW object CT-##).
 - The type of data transfer is batch input.
2. Create a variant for program RSBDCSUB. This variant should process the batch input sessions CT-##. (This session name comes from the LSMW outbound file and corresponds to the object name.)
 3. Schedule a job that consists of two steps: The first step contains your batch input program RFBIBL00; the second contains program RSBDCSUB. For both steps, you require the variants that were defined already.
 4. Trace the job process and check the job log of all jobs that were generated by RSBDCSUB to process batch input session.

Task 3: Optional

Generate several batch input sessions by changing an LSMW setting and scheduling the LSMW framework program.

1. Display all objects for your project; select the object CT-## and copy it to the new object BI-##.
2. In the event END_OF_RECORD, change the structure BGR00 as follows. The field g_cnt_transactions_group determines after how many records the next batch input session is generated. Change this value to 2.

```
at_first_transfer_record.
if g_cnt_transactions_group = 2.
g_cnt_transactions_group = 0.
transfer_record.
endif.
```

3. In the object attributes, set the data transfer to *Periodic*. When you have done this, you receive the framework program /SAPDMC/SAP_LSMW_INTERFACE to automatically execute the work steps of the LSMW.
4. Create the variant Var-## for this framework program.

Settings:

Project:	BC420-##
Subproject:	DOCU-##
Object:	BI-##

Continued on next page

Set the parameters of some batch input programs or direct input programs
Batch Input, Direct Input, Call Transaction to B.

5. Schedule a job that consists of one step and contains the framework program with the variant VAR-##. Start the job.
6. Trace the job process and check the job log.

Solution 19: Scheduling Batch Input in the Background

Task 1: Optional

You want to start a job that consists of one step using the date/time.

1. Define a variant for the program SAPBC420_SPTD_LIST_CHECKBOXES.
 - a)
2. Create the job BC420JOB##-1, that is started using the date/time. As a start date, you can enter *Start Immediately* or, for example, the system time plus three minutes. The job should contain the ABAP report as the only step.
 - a)
3. Trace the job process using transaction SM37 and analyze the job log.
Trace the job process using transaction SM37 and analyze the job log.
 - a)

Task 2:

Define and schedule a job to generate and process batch input sessions. You want to define **one job BC420 ##_Job** that execute the following tasks in **two steps**:

- a) Process the program to generate one or several batch input sessions. First you must have created a variant for the batch input program to be scheduled.
 - b) Call program RSBDCSUB to schedule the batch input session that was generated in the previous step. First you must have created a variant for the program RSBDCSUB.
1. Process Steps:
Create the variant VAR_## for program RFBIBL00. Maintain the following data:
 - File names = (For the file name, see the outbound file or the converted data in the LSMW object CT-##).
 - The type of data transfer is batch input.
 - a)
 2. Create a variant for program RSBDCSUB. This variant should process the batch input sessions CT-##. (This session name comes from the LSMW outbound file and corresponds to the object name.)
 - a)

Continued on next page

3. Schedule a job that consists of two steps: The first step contains your batch input program RFBIBL00; the second contains program RSBDCSUB. For both steps, you require the variants that were defined already.
 - a)
4. Trace the job process and check the job log of all jobs that were generated by RSBDCSUB to process batch input session.
 - a)

Task 3: Optional

Generate several batch input sessions by changing an LSMW setting and scheduling the LSMW framework program.

1. Display all objects for your project; select the object CT-## and copy it to the new object BI-##.
 - a)
2. In the event END_OF_RECORD, change the structure BGR00 as follows. The field g_cnt_transactions_group determines after how many records the next batch input session is generated. Change this value to 2.

```
at_first_transfer_record.  
if g_cnt_transactions_group = 2.  
g_cnt_transactions_group = 0.  
transfer_record.  
endif.
```

- a) To change a technical LSMW settings for the field mapping, choose *Extras → Display Variant* and select the checkboxes *Technical Fields* and *Processing Times*. *Code* and *Global Data Definitions* should also be set.

Below the structure BGR00, the processing time “END OF RECORD” is displayed. Here, a counter determines after how many transactions the system should process a new batch input session.

Change the query “if g_cnt_transactions_group = 5000” to “if g_cnt_transactions_group = 3”. This ensure that the system generates a new batch input session after three external data records have been read.

Continued on next page

3. In the object attributes, set the data transfer to *Periodic*. When you have done this, you receive the framework program */SAPDMC/SAP_LSMW_INTERFACE* to automatically execute the work steps of the LSMW.
 - a)
4. Create the variant Var-## for this framework program.

Settings:

Project:	BC420-##
Subproject:	DOCU-##
Object:	BI-##

Set the parameters of some batch input programs or direct input programs *Batch Input*, *Direct Input*, *Call Transaction to B*.

- a)
5. Schedule a job that consists of one step and contains the framework program with the variant VAR-##. Start the job.
 - a)
6. Trace the job process and check the job log.
 - a)



Lesson Summary

You should now be able to:

- Execute the data transfer in the background using the LSMW and DX-WB tools
- Describe and evaluate how the LSMW and DX-WB can be used for periodic data transfer



Unit Summary

You should now be able to:

- Execute the data transfer in the background using the LSMW and DX-WB tools
- Describe and evaluate how the LSMW and DX-WB can be used for periodic data transfer

Unit 12

Appendix I

Unit Overview



Unit Objectives

After completing this unit, you will be able to:

- Describe the test concepts of data transfer
- Use other special techniques with the LSMW

Unit Contents

Lesson: Appendix I	402
--------------------------	-----

Lesson: Appendix I

Lesson Overview



Lesson Objectives

After completing this lesson, you will be able to:

- Describe the test concepts of data transfer
- Use other special techniques with the LSMW

Business Example

A company wants to use the LSMW to convert external data and intends to import differently structured external data from several files.

Before the data transfer is performed in the productive system, it must be tested in advance in a test environment with representative data.

LSMW – Other Functions



This slide shows the mapping between two different structures in LSMW coming from two different files.

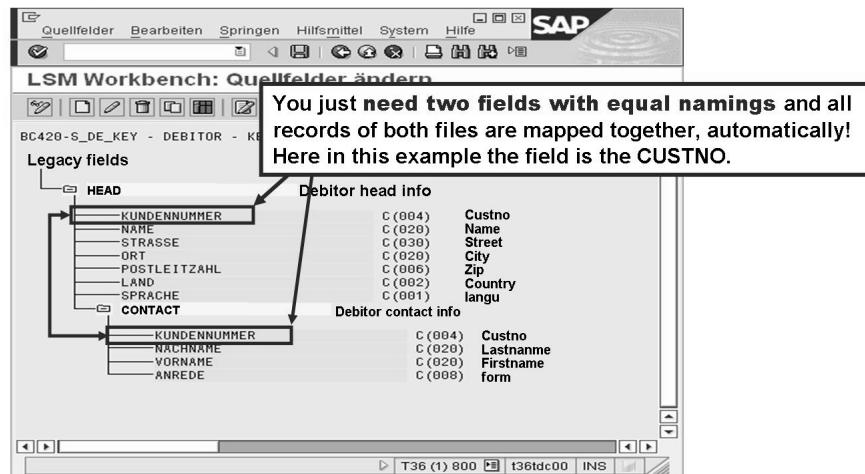


Figure 329: Hierarchical External Structures from Different Data

As the figure above shows, hierarchically structured external data that originates from different structures can be distributed to several files. It would be possible to read header information from one file and document items from another file.

In the example, the customer header and customer contact information are assigned to each other.

The assignment between the records requires a type of “foreign key”. For this, an identically named field must be used in both files. The LSMW would then assign the corresponding records for all records in which the contents of this field matched. For selects on data base tables, this behavior is indicator as “JOIN”.

When you use this method, you must ensure that you do not accidentally assign identical names to several fields in both files. This may have undesired effects. The “key” must be unique and must consist of only one field. It is not necessary to sort the data in advance.

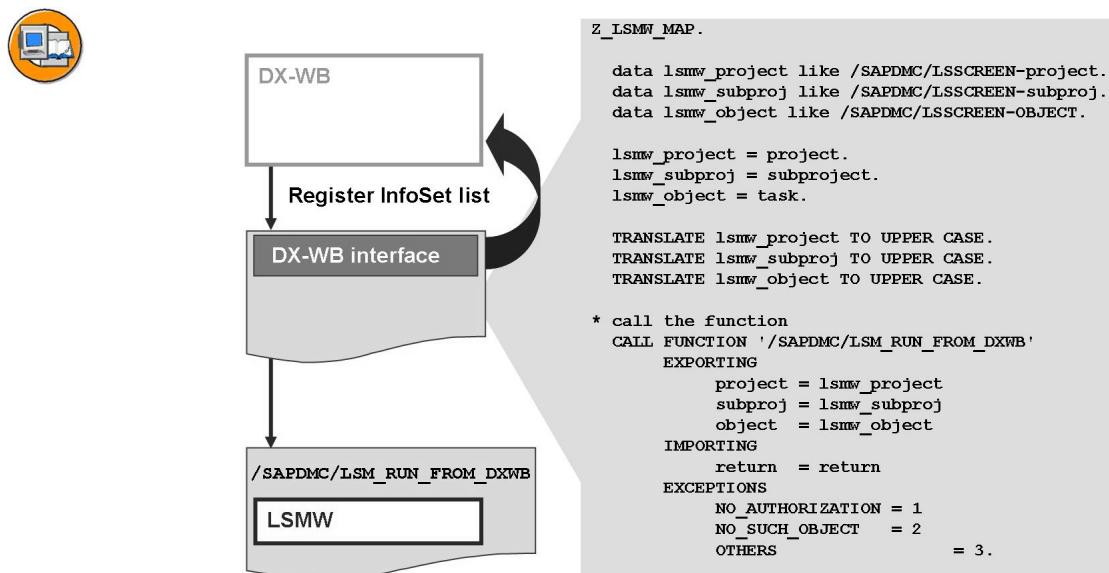


Figure 330: Including the LSMW in the DX-WB as a Mapping Task in Releases Lower Than 4.6B

If you want to call the LSMW as a task in the DX-WB in **Release 4.6A or 4.6B**, you must write a function module and register it.

The mapped function module uses the supported interface of the function modules in the DX-WB (see the online documentation). In this function module, the system calls a project/subproject/object in the LSMW. According to the names of the project, subproject or run in the DX-WB, this function module calls the components with the same names in the LSMW.

The following applies for the matching names:

- LSMW project = DX-WB project
- LSMW subproject = DX-WB subproject
- LSMW object = DX-WB object

Test Principles for Data Transfer

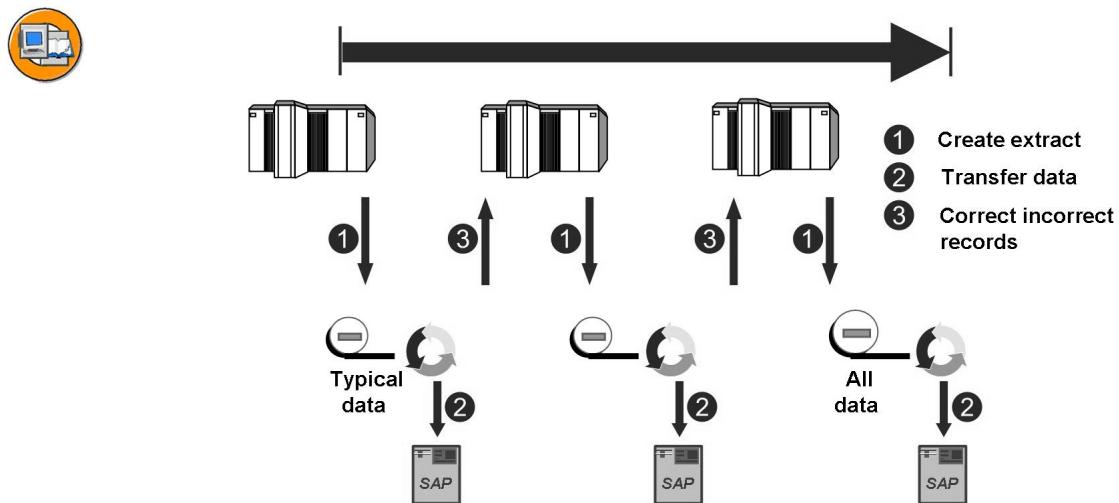


Figure 331: Testing Phase

For the first testing phase, the dataset must not be larger than 10 records. If the entire test process runs correctly, you can increase the dataset gradually. The last test must be execute with the complete dataset.

You must correct any errors that occur in the testing phase.

- If the data record is incorrect (for example postal code), you must correct it in the external system.
- If there is an incorrect Customizing entry you must correct it.
- If the error is in the conversion program, you must adjust the program.

All data must be transferred without errors in a test before you can assume that there will be no errors during the actual transfer.

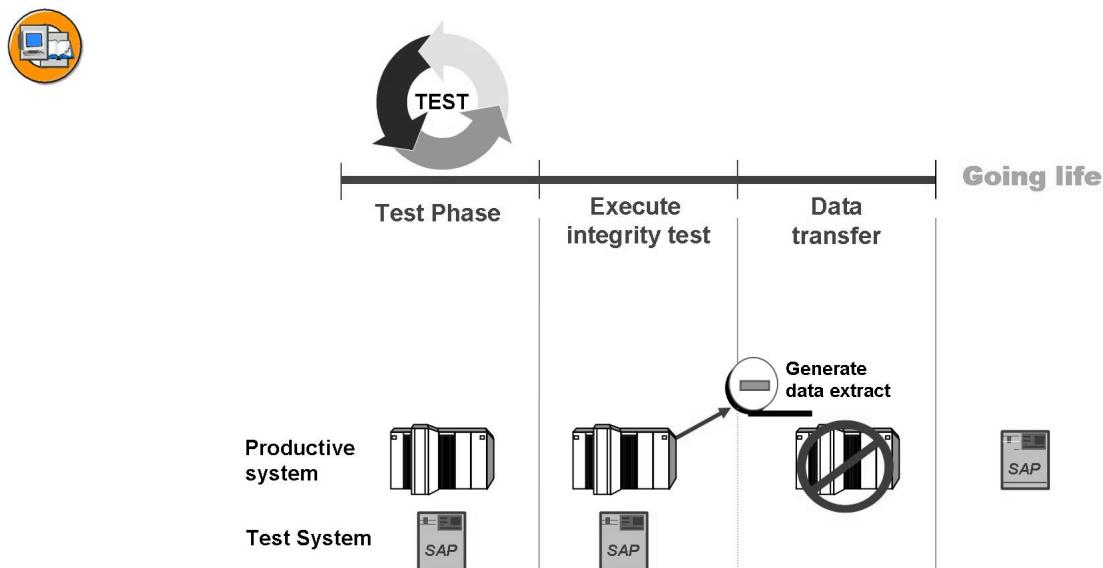


Figure 332: Execute Data Transfer

The actual productive transfer does not occur directly after the testing phase. Many other steps must be performed after the testing phase of the integration testing. Within integration testing, all required business processes must be tested with the data imported up to this point.

If the integration test proceeds positively, you can then execute the actual transfer.

Important: If Customizing is changed during the integration test, errors may occur during the data transfer. This means: Each change made to Customizing required that the transfer is tested again. Skip this test in exceptional cases only.

During the entire testing phase and integration phase, the external system remains the productive system in which productive data is processed. During the actual, productive transfer, no system is available. The SAP system then assumes the role of the productive system.

Determining the Data for the Transfer



- Is all the required data available in the external system (for example, company code)?
Missing data must be created from the context of the data.
- Can all fields of the external system be mapped?
- Are there dependencies with other objects?

Plan the process flow.

For fields that occur in the SAP system, there may be no corresponding field in the external system. If this field is required in the SAP system, you can proceed as follows:

- If this value is the same for all data records, it can be determined and entered as a fixed value.
- If the value has a dependency, you must determine this. The dependence determines the value for each data record.

If there is no corresponding SAP system field for a field in the external system, you can proceed as follows:

- Consider whether or not the field is required in the SAP business process. If it is not required, it can be omitted.
- SAP fields that are not directly required for the business process can be filled with this field.

The business processes in the SAP system always have a dependency. Therefore, transaction data can be transferred only if the relevant master data is already available. **Therefore it is essential to take into account the dependencies or the sequence of the objects.**

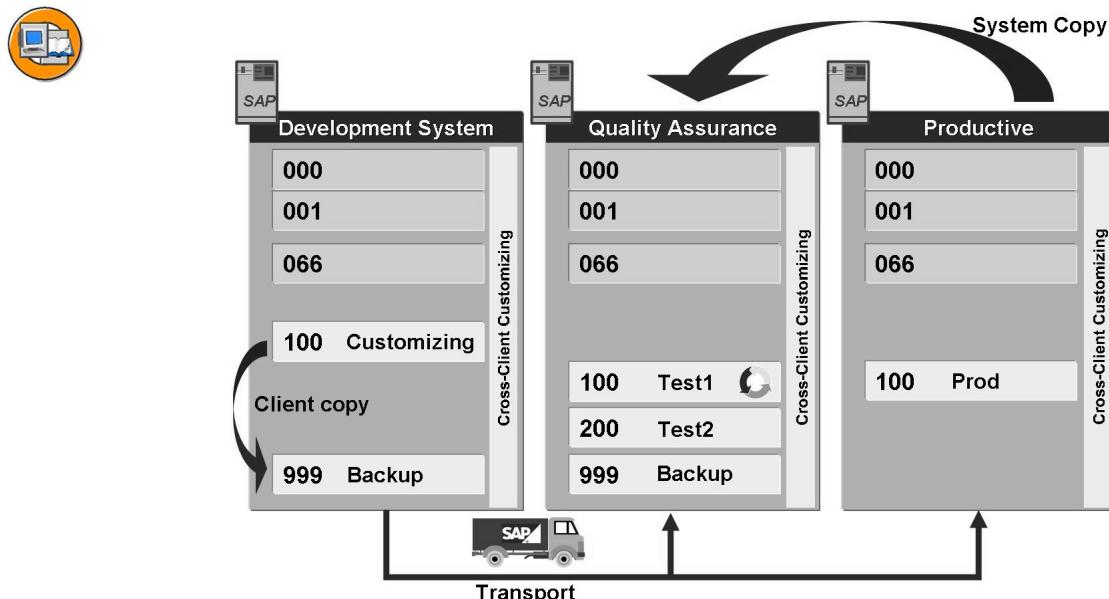


Figure 333: System landscape

For a data transfer, it is essential to distribute client-specific and cross-client Customizing settings to all clients and systems of a system landscape.

If, for example, the data transfer is tested in client 100 of the test system (Quality Assurance), the Customizing settings must first be transported to there from the Customizing client of the development system.



Caution: In practice, the transport of cross-client or client-specific Customizing is often forgotten. As a result, errors occur during the data transfer.



Lesson Summary

You should now be able to:

- Describe the test concepts of data transfer
- Use other special techniques with the LSMW



Unit Summary

You should now be able to:

- Describe the test concepts of data transfer
- Use other special techniques with the LSMW

Unit 13

Appendix II

Unit Overview



Unit Objectives

After completing this unit, you will be able to:

- Use interactive lists for transferring data
- Evaluate the effects of the SAP LUW architecture on the external data transfer
- Execute a data transfer in the background
- Use BI utility programs

Unit Contents

Lesson: Appendix II	412
Exercise 20: Interactive Lists and Batch Input.....	431

Lesson: Appendix II

Lesson Overview

Contents:



- Transferring Data Using Interactive Lists
- Business Address Services (BAS)
- SAP LUW
- Hints and Tips
- Principles of Batch Processing



Lesson Objectives

After completing this lesson, you will be able to:

- Use interactive lists for transferring data
- Evaluate the effects of the SAP LUW architecture on the external data transfer
- Execute a data transfer in the background
- Use BI utility programs

Business Example



Interactive Lists

BAS

SAP LUW

Hints and Tips

Principles of Batch Processing

Figure 334: Special Topics (1)



Demo program: SAPBC420_SPTD_INTERACTIV_LIST

Customer overview	
Customer number	Name
<input type="checkbox"/> Z-00-10002	Fausto Rosini
<input type="checkbox"/> Z-00-10003	Helga Beimer-Schiller
<input type="checkbox"/> Z-00-10004	Anna Ziegler
<input type="checkbox"/> Z-00-10005	Carsten Flöter
<input checked="" type="checkbox"/> Z-00-10006	Else Kling
<input type="checkbox"/> Z-00-10007	Eva-Maria Sperling
<input type="checkbox"/> Z-00-10008	Hajo Scholz
<input type="checkbox"/> Z-00-10009	Erich Schiller
<input type="checkbox"/> Z-00-10010	Isolde Pavarotti
<input type="checkbox"/> Z-00-10011	Klaus Beimer
<input type="checkbox"/> Z-00-11001	Lisa Hofmeister
<input type="checkbox"/> Z-00-12344	Treusch

Figure 335: Batch Input for Interactive Lists

The demo program SAPBC420_SPTD_INTERACTIV_LIST displays a list which provides you with information using checkboxes.



Customer overview	
Customer number	Name
<input type="checkbox"/> Z-00-10002	Fausto Rosini
<input type="checkbox"/> Z-00-10003	Helga Beimer-Schiller
<input type="checkbox"/> Z-00-10004	Anna Ziegler
<input checked="" type="checkbox"/> Z-00-10005	Carsten Flöter
<input checked="" type="checkbox"/> Z-00-10006	Else Kling
<input type="checkbox"/> Z-00-10007	Eva-Maria Sperling
<input type="checkbox"/> Z-00-10008	Hajo Scholz
<input type="checkbox"/> Z-00-10009	Erich Schiller
<input type="checkbox"/> Z-00-10010	Isolde Pavarotti
<input type="checkbox"/> Z-00-10011	Klaus Beimer
<input type="checkbox"/> Z-00-11001	Lisa Hofmeister
<input type="checkbox"/> Z-00-12344	Treusch

Field Name	Field Values
Enhance the fields!	...
BDC_CURSOR	07/02
BDC_OKCODE	=PICK
...	

You must enhance these entries!

Row Column

The "X" for the checkbox is missing, therefore the actual field is not set.

Figure 336: Results of the Transaction Recorder

If this transaction is recorded using the Transaction Recorder, the checkboxes are not selected in the recording list. You must **manually enhance** these important entries.

Specify the position in the list in X/Y coordinates. X represents the row and Y represents the column.

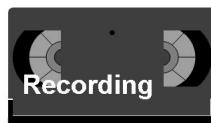


Hint: Important - The first column on the list starts with the value 2.

The selected row does **not always** match the position in a row because the headings must always be taken into account for the number of rows.

The selected fourth checkbox in the example is the sixth row of the list because two rows are used for the heading. The value in the field BDC_CURSOR is set to **07/02** because this was the last item on the list.

The fields to be filled are therefore row 6/column 2 and row 7/column 2.



Changes in the Recording

Program	Screen	St.	Field Name	Field value
		T	BC420	BS
...				
SAPMSSY0	120	X		
			06/02	X
			07/02	X
			BDC_CURSOR	07/02
			BDC_OKCODE	=PICK
SAPMSSY0	120	X		
			BDC_OKCODE	=BACK
...				

Figure 337: Editing the Recording

The figure above clarifies the changes in the editor of the recording.



Interactive Lists



BAS

SAP LUW

Hints and Tips

Principles of Batch Processing

Figure 338: Special Topics (2)



- **More data fields**
- **Standardization of data maintenance**
- **Available to customers, creditors, and contact persons**



Figure 339: Business Address Services (BAS)

As of Release 4.5 address data in various master objects has been moved to Business Address Services (BAS). Through the introduction of BAS the maintenance of addresses was merged and into one standard function and as a result standardized.

In the dialog maintenance the address screen for the company address has been replaced with a new screen, on which the BAS subscreens are integrated. The addresses are stored in separate tables in the database. The tables of the master data object contain pointers to their assigned address data from the BAS.

Only the address data available in tables KNA1, LFA1 and KNVK can be imported by the standard batch input programs for the customer and vendor master (RFBIDE00 and RFBIKR00). As the information stored in these fields is redundant, their length cannot be adjusted to the corresponding fields in the BAS. If you want to transfer new address fields that were not available on the old address screens (for example, a separate field for the house number), or if you want to transfer the data in BAS format for the relevant fields, you must transfer this data in a separate address run.



Transfer addresses to BAS using BAPI Customer.SaveReplica

- All customer data other than address data is transferred using report RFBIDE00.

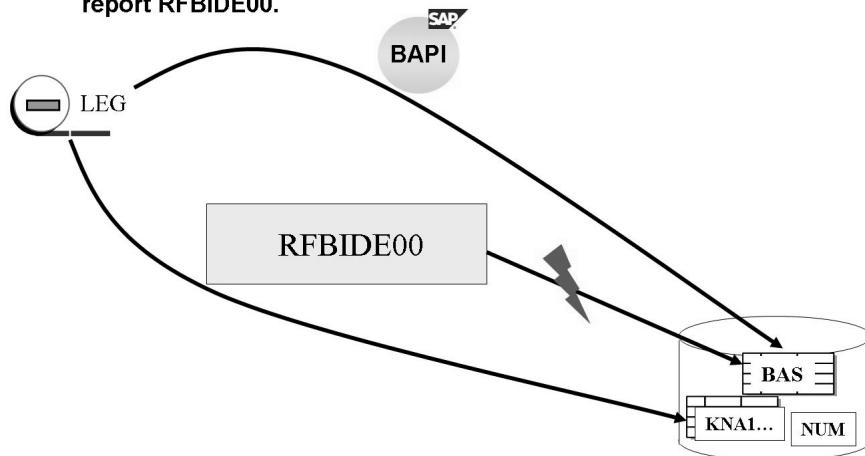


Figure 340: BAS: Customer Data Transfer

Principles: Two DX-WB runs solve the problem (could be implemented as two steps of a normal SM36 job): First the addresses are transferred into the BAS, and then all other customer data is transferred into the customer tables (KNA1, and so on).

First DX-WB run, for business object (BO) BUS4001: Transferring address data to the BAS

Task 1: Fetch internal numbers, and add internal numbers to LEG file (ABAP report). This task is not required for external number assignment. In this case, the customer number is already contained in the LEG file.

Task 2: Convert LEG file into a BAPI file (using the LSMW).

Task 3: Process BAPIs (load data)

Second DX-WB run, for business object KNA1: Transferring all other customer data

Task 1: Convert LEG file (the same one as in task 2 above) into record layout file (using the LSMW).

Task 2: Use the batch input report RFBIDE00 to create a batch input session.

Task 3: Use the report RSBDCSUS to process the session in the background.

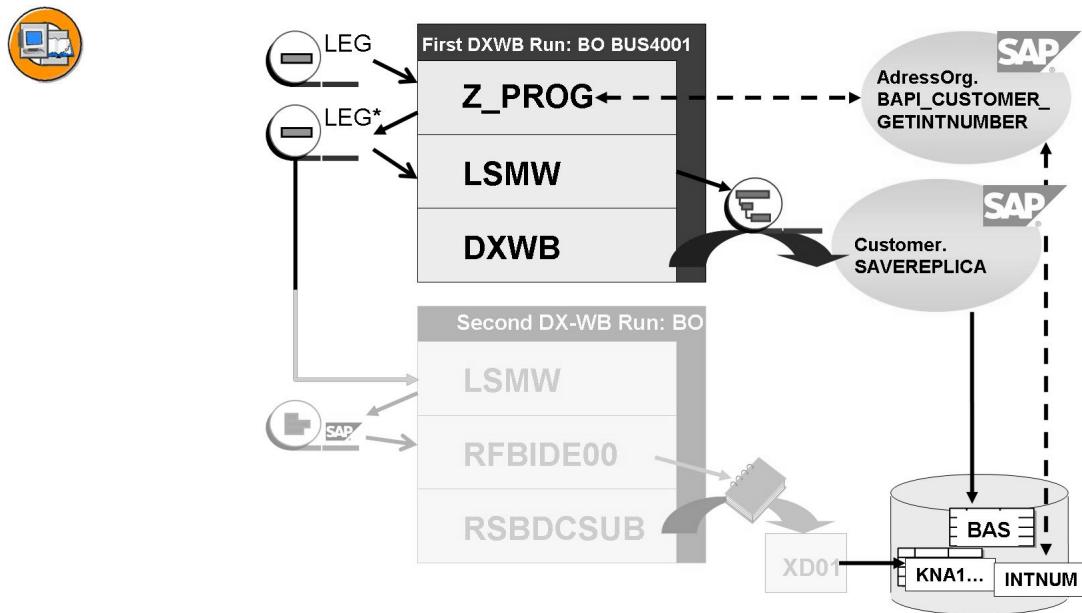


Figure 341: BAS: Customer Data Transfer, First DX-WB Run

First DX-WB run/task 1:

BAPI: BAPI_CUSTOMER_GETINTNUMBER retrieves the internal number. This must be right-aligned and saved with leading zeros.

The legacy file is copied and a new attribute for each record (the internal number) is added to it (using ABAP commands READ DATASET and TRANSFER).

A separate street and house number should already exist in the legacy file. If the programs were technically separate they would be liable to errors (special cases, for example, 2. Rheinstrasse 17, different country formats, such as 2120 South Michigan Avenue), see the sample ABAP report SAPBC420_APPD_GET_INT_NUM.

First DX-WB run/task 1:

LSMW object with import method BAPI, business object BUS4001, method SAVEREPLICA.

It is best to take the correct structure relationships and the minimum field mapping required from a test file created beforehand using the SXDA tools. A few constants in particular have to be assigned values.

With internal number assignment, the inbound file is the modified legacy file from task 1. Otherwise it is the original legacy file. Since this inbound file is required again in the second DX-WB run, it must not be deleted at the end of the DX-WB run.

First DX-WB run/task 1:

Load the data using BAPI_SAVEREPLICA for the business object BUS4001 (packet size 1).

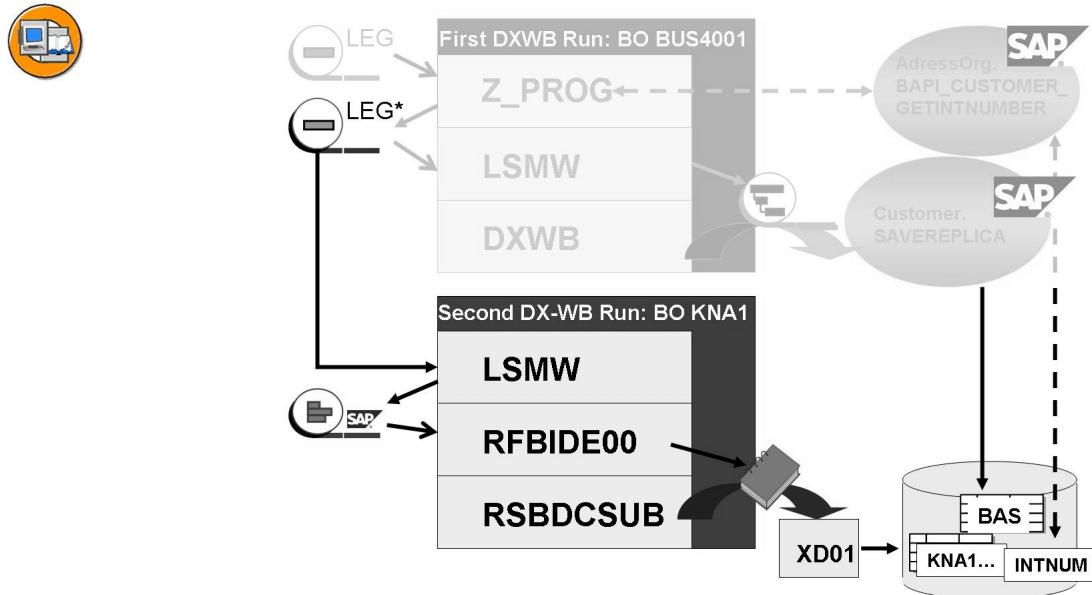


Figure 342: BAS: Customer Data Transfer, Second DX-WB Run

Second DX-WB run/task 1:

LSMW object with import method standard batch/direct input, object type 0050, program name RFBIDE00.

Input file and source structure are identical with those in the first DX-WB run/task 2.

With field mapping, only non-address data can be mapped. If the address data is mapped again here, it would be displayed in transaction XD03 (display customers) instead of the data from the BAS.

For more information, see SAP Notes 384462 and 306257.

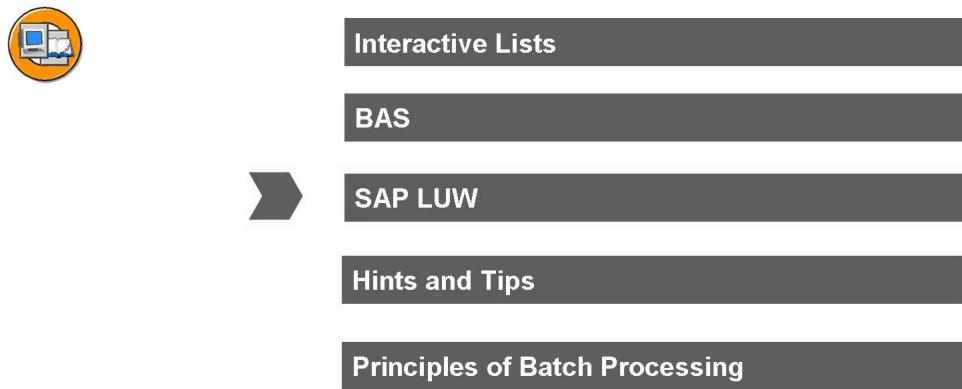
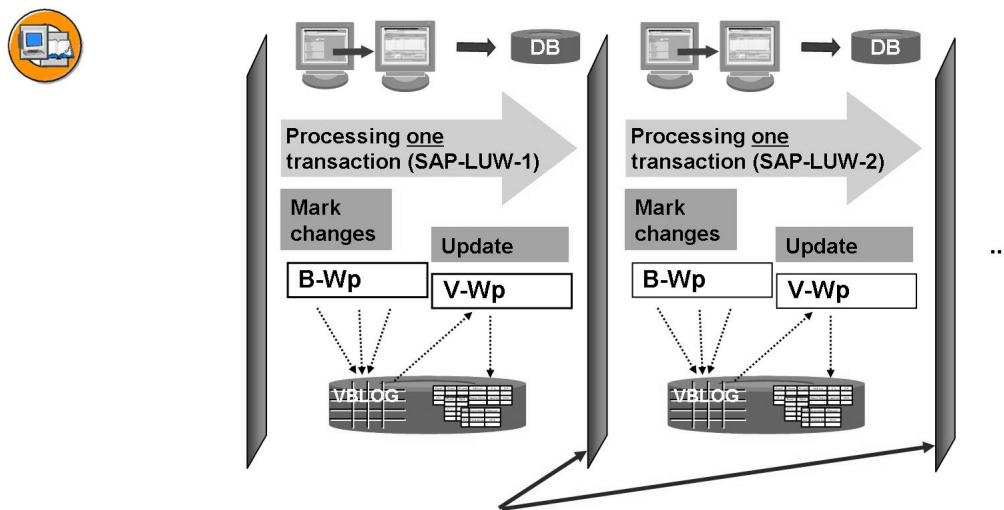


Figure 343: Special Topics (3)



For BI, the next transaction cannot be processed until the end of the SAP-LUW. Therefore, processing is always synchronous

Figure 344: Synchronous Processing

An SAP LUW (Logical Unit of Work) consists of an SAP transaction that a user runs online (first part of the LUW) and the subsequent update (second part of the LUW). Therefore, once a user has saved a transaction, usually once the transaction has been processed, he or she can start to process the next transaction immediately. You can start a subsequent SAP LUW while the system is updating the first SAP LUW.

Batch input sessions are always processed synchronously, that is, SAP LUW 2 must follow the update part of SAP LUW 1.

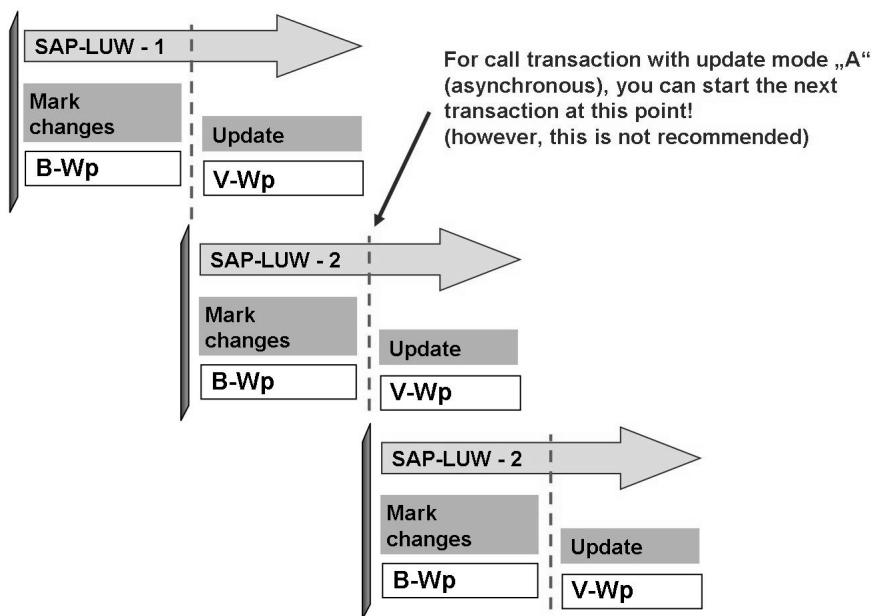


Figure 345: Asynchronous Processing

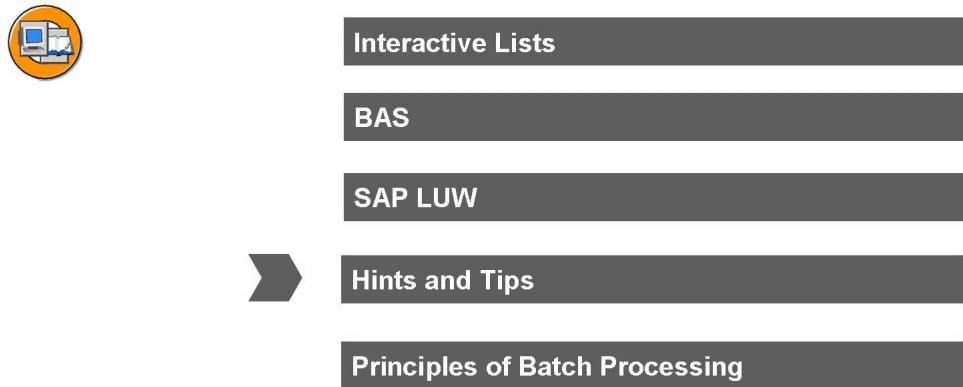
Processing mode “A” for call transaction:

After the first part of an SAP LUW in which planned changes are described, the next SAP LUW can be started directly. External data that is imported using call transaction can theoretically “overlap”, that is they can partially be imported in parallel (if the update takes longer than the calling process for the next record to be processed).

We recommend that you do not use the asynchronous session for call transaction. You must extensively test it beforehand.

SAPNet messages concerning the update process/locks:

- Lock table: 17267, 13907, 97760
- Update/reprocessing: 70085

**Figure 346: Special Topics (4)**


Subject area:	BC-ADM, BC-DB-... BC-KRN, CA-CL, ...	
Notes:		
LSMW	101014, 168644, 158198, 158285	
BI Recorder (pre-Inst.)	78448	
BI logs and reorg.	18307, 39282, 24438, 18319	
All reorg jobs	16083	
BI and Commit Work	26703, 24141	
BI and updating process	33421, 49633	
BI in the foreground	45507, 26171, 49132	
BI and subsequent processing	15999	
BI in the background	33319, 19422	
BI and long texts	159738	
BI and table controls	11788	
Character sets...	42709	
Confirmation dialog box	13882	
Data transfer using CATTs	87162	

Figure 347: SAPNet Notes About Data Transfer

SAPNet enables you to enter problem messages independently in the SAP Support system. This ensures that SAP Hotline can process customer problems as quickly and efficiently as possible.

You can also use the SAP Notes system to help you solve any problems free of charge.



Presentation

Application

Database

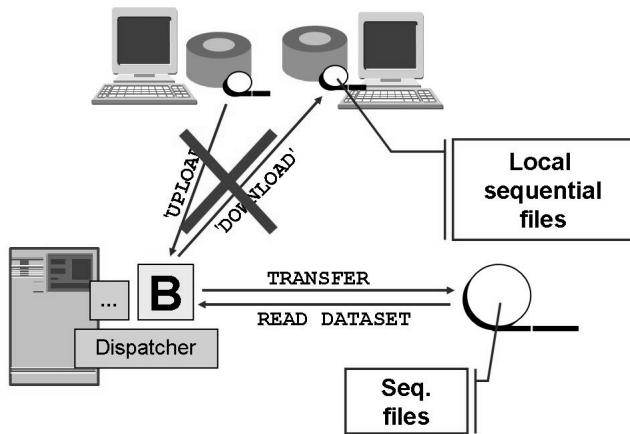


Figure 348: Front End Access in the Background

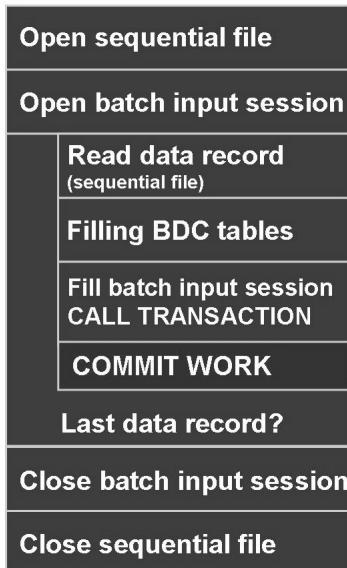
The function modules DOWNLOAD and UPLOAD are designed for online use and **cannot be used** in the background.

Batch Input Utility Programs



- **RSBDCREO**
 - Deletes all sessions (and their logs) that were processed successfully and that are still in the system.
 - Physically deletes any logs for which there are no more sessions.
 - Reorganizes the log file (this function is integrated into the Batch Input Monitor)
- **RSBDCLOG**
 - Generates a list of batch input logs according to session name.
 - Logs can be displayed or deleted; if sessions are available, analysis can be activated (this function is integrated into the Batch Input Monitor)
- **RSBDCDRU**
 - Enables you to selectively print session contents (this function is integrated into the Batch Input Monitor)

Of the three utility programs above, RSBDCREO is the most important. You use this program to reorganize batch input sessions and their logs.



Every 100 to 1,000 loop passes:
Issue a COMMIT WORK to protect the rollback segments from possible overflow.

Figure 349: Batch Input Programming: Credit Rollback Segments

The database rollback segments are buffer areas that hold the “before image” of the database (change information for restarting the database) during a database LUW (a DB-LUW is a database logical unit of work). If an error occurs during this automated database processing step, you can use these segments to return to the initial status.

The statement BDC_INSERT to fill the batch input sessions makes changes to the database that cause the size of the rollback segments to increase. To restrict the growth of these segments, you should trigger a database COMMIT regularly (depending on the size of the data record every 100 or 1,000 loop passes). To do this, use the ABAP command COMMIT WORK. This resets the rollback segments.

Additional Tips and Information



- Batch input logs are stored in DIR_GLOBAL.
- Batch input sessions are stored in the database in the tablespace PSAPSTABD.

Protocols:

Batch input logs can be found in the directory DIR_GLOBAL (.../global) on the application server.

The ABAP program RSBDCREO reorganizes these files.

Tablespace size:

Depending on the volume of external data being transferred to the SAP system, the batch input sessions could potentially run out of database tablespace. In this case, the database administrator must increase the tablespace PSAPSTABD.

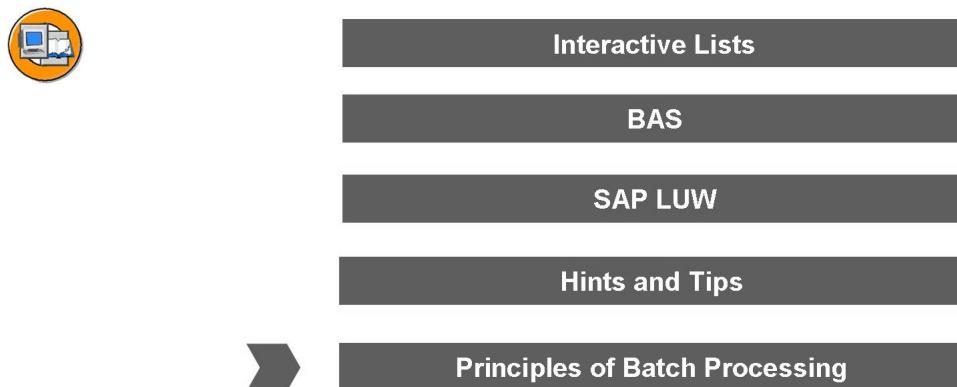


Figure 350: Special Topics (5)

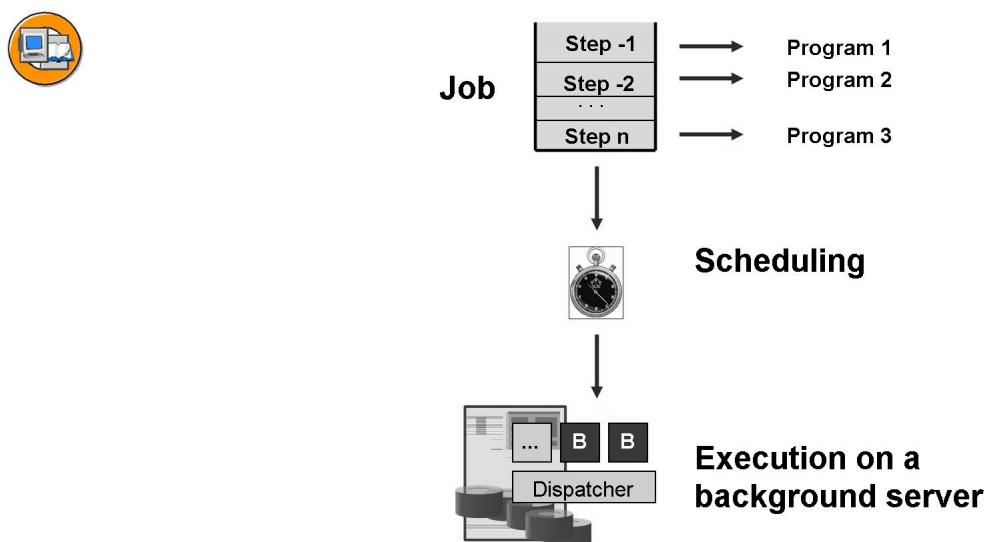


Figure 351: Background Processing Concept

SAP background processing is a method for automating routine tasks and for optimizing the use of your organization's R/3 computing resources. In background processing, you instruct the R/3 System to run programs for you. You or your users tell the system what program to run and when to run it by scheduling background jobs.

The SAP system offers complex support for background processing. You can choose from a variety of methods for scheduling and maintaining jobs. You can run both SAP-internal programs and external programs. And, for easier scheduling and management, you can run related internal or external programs as "job

steps" within a single background processing job. You can use these job steps to define a background job as a complex task consisting of several processing steps. Scheduling the job includes all processing activities required to perform the task.

There are also sophisticated tools, including a graphic monitor, for managing jobs and diagnosing any problems. There is also a powerful and easy-to-use programming interface for developing your own background-processing applications. In addition, there is also a job scheduling wizard that automates the basic definition for background jobs and can walk novice SAP users through the entire process.

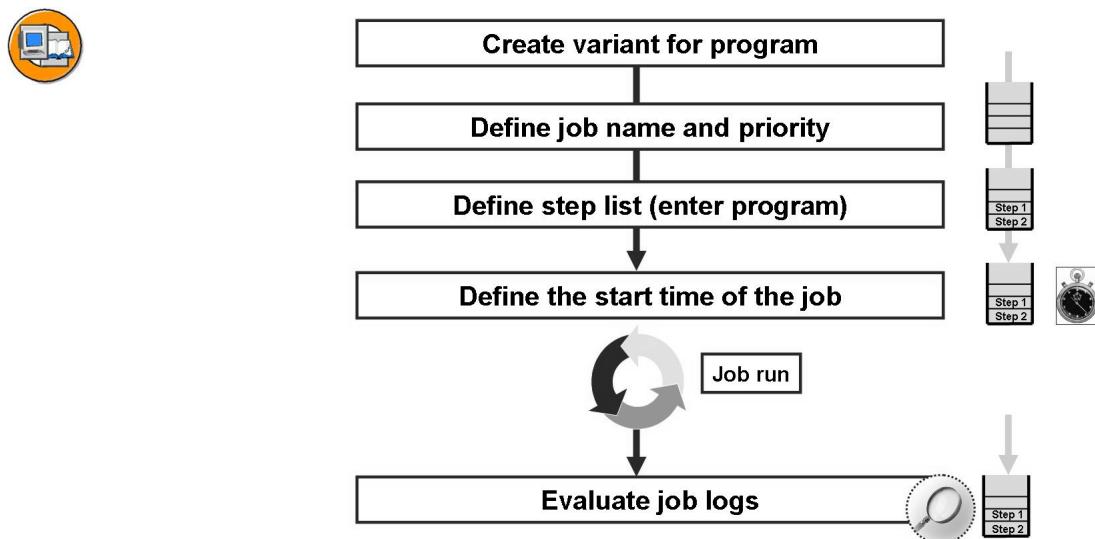


Figure 352: Background Processing – Work Steps

To schedule a job in background processing, you must proceed as follows:

1. If the program to be scheduled has a selection screen, you must create a variant for it.
2. In the job definition, you must define the name and the priority of the job. The target server specification is optional.
3. The step list contains all programs that must be run in sequence within the job.
4. The start time of the job must be defined.
5. At the start time, the system processes the job in a free batch work process.
6. The job logs are then visible.



Transaction "Define Job" (SM36)

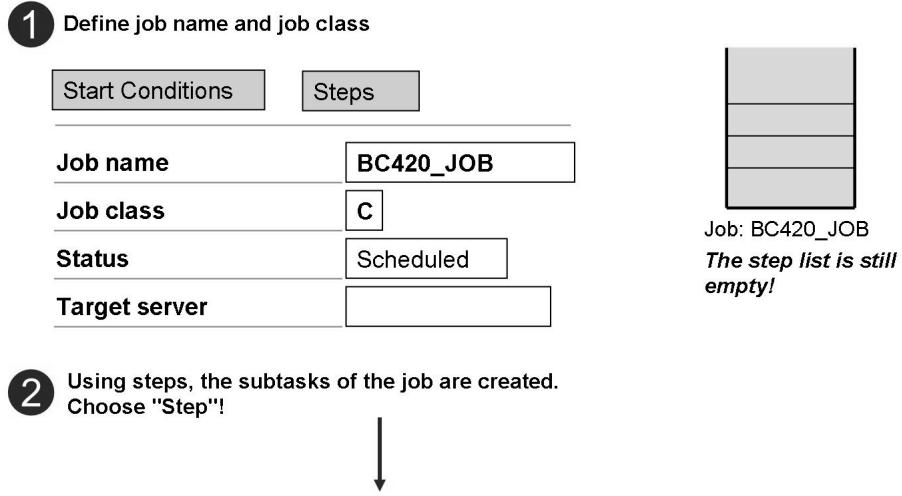


Figure 353: Creating a Job (1)

In the job definition, the job receives a name and a job class (A = highest priority, C = lowest priority).

The individual job steps (individual programs within the job) are defined under “Steps”.

The field *target server* must be left blank to realize an automatic “workload-balancing”. The job is then execute on the server that uses free capacity (batch workprocesses) at the start time.

Scheduling is performed under *Start Conditions*.

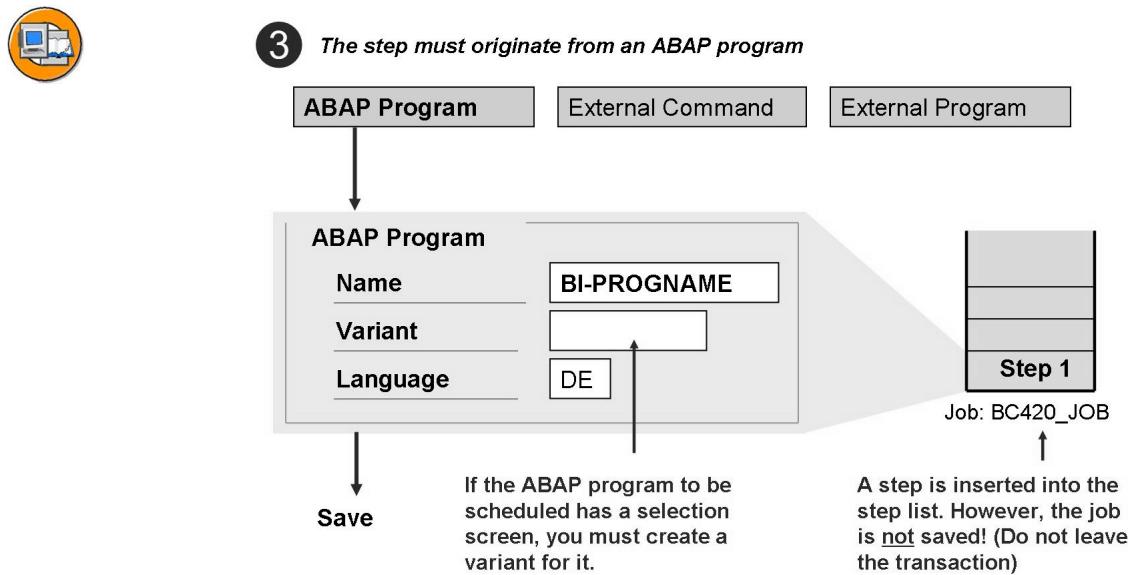


Figure 354: Creating a Job (2)

One step may be an internal ABAP program, for example. If this program has a selection screen, you must enter a variant. A variant is a predefined selection screen for this program.

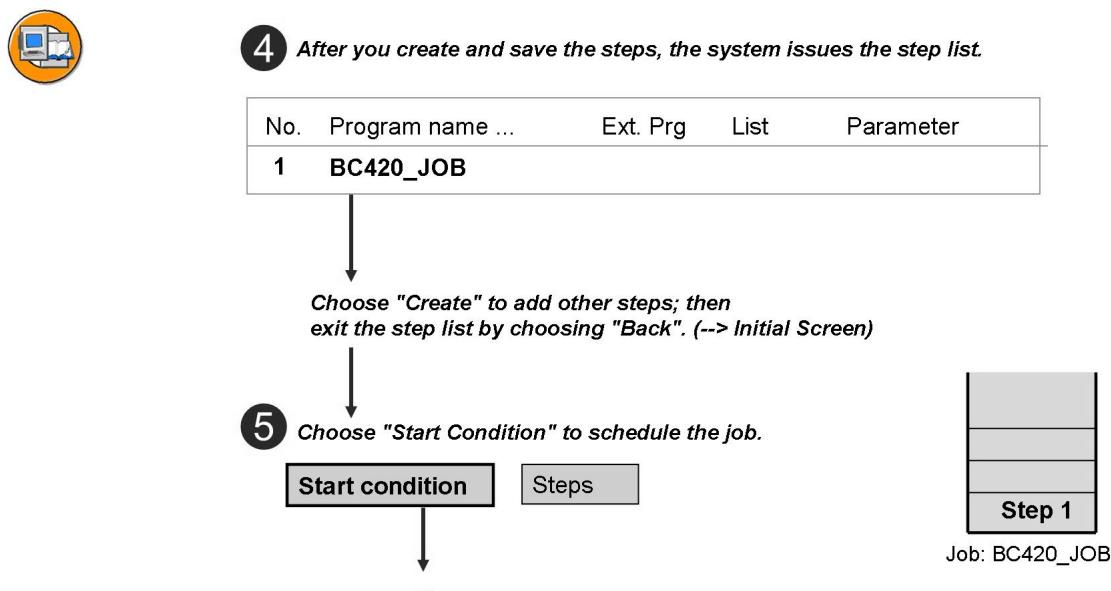


Figure 355: Creating a Job (3)

After you save the steps, the system issues the step list. You can choose *Create* to create additional steps for this job.

After you define and save the step list, you can continue by entering the start time in the initial screen (after choosing Save and Back).

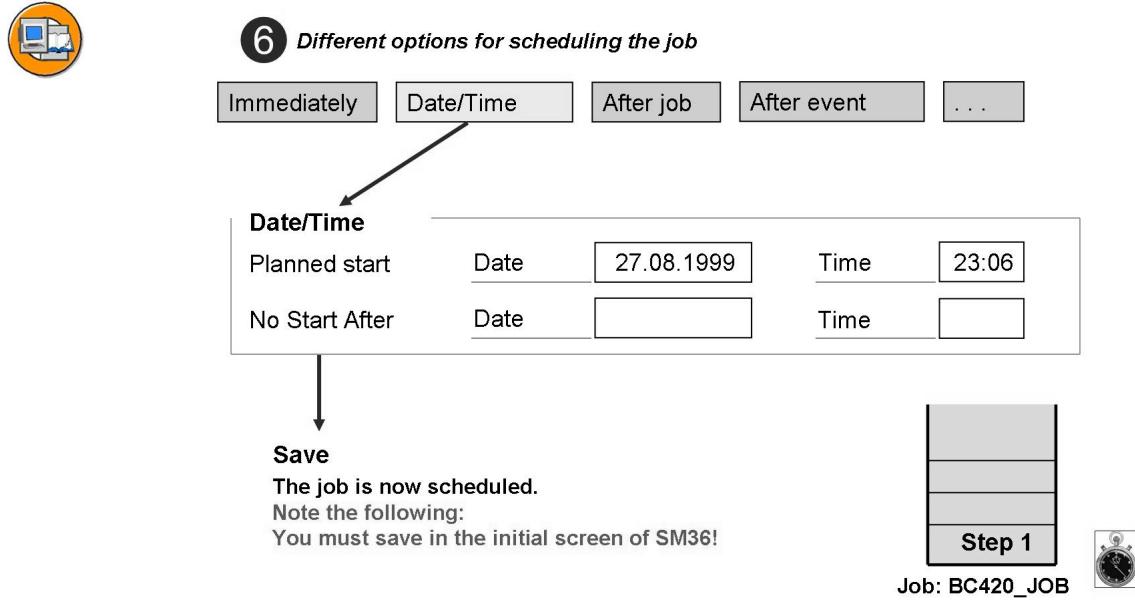


Figure 356: Creating a Job (4)

You can schedule the job using *Date/Time*.T

In addition to the scheduled start time, you can also set a *Expiration Date*.

If a job is not scheduled by this time at the latest (for example, because no batch workprocesses are free), it is no longer executed. It then has the status “Terminated” in the job overview.



Transaction "Job Overview" (SM37)

Job name	BC420*	
User name	*	
Job Status <input type="checkbox"/> Planned <input type="checkbox"/> Released <input checked="" type="checkbox"/> Ready <input type="checkbox"/> Active <input type="checkbox"/> Finished <input type="checkbox"/> Canceled		
Job start condition From <input type="text" value="27.08.1999"/> <small>23:24</small> To <input type="text" value="27.08.1999"/> <small>23:56</small> Or start after event <input type="text" value="*"/>		
		Step 1 Job: BC420_JOB

Figure 357: Job Overview (1)

In the job overview, you can display jobs by selecting them according to certain criteria.

That is, for example, jobs whose start time has already passed but have not yet started ("Ready").

If jobs are displayed, that were scheduled irrespective of events, at least one "*" must be set for *Start after event*.



After you select the selection parameters, the system displays the selected jobs

...

Job overview from: 27.08.1999 ...

Job name	JobCreatedBy	Status	Start date	Start time	Duration [sec]
BC420_JOB	KLINGELS	finished	27.08.1999	5:34:23	12
BC420_JOB1	TREUSCH	finished	27.08.1999	6:23:00	1
BC420_JOB3	TREUSCH	finished	27.08.1999	6:23:34	3
BC420_0815	MAYER	sched.			
....					

Finished: Job was executed Scheduled: Job is defined but not assigned a time

Figure 358: Job Overview (2)

Based on the job overview, you can view the list output of the individual job steps in the spool.

The job log appears under *Job Log*.

Job Status:

- Scheduled: The job is defined, but no start time was assigned yet.
- Released: The job is scheduled.
- Ready: Start conditions are met; the job is ready to be executed.
- Active: The job is running; see transaction SM50.
- Completed: The job was executed without any errors.
- Canceled: The job was canceled; Errors occurred.

Exercise 20: Interactive Lists and Batch Input

Exercise Objectives

After completing this exercise, you will be able to:

- Transferring Data Using Interactive Lists

Business Example

You want to call up the checkboxes of an interactive list using the batch input method or call transaction method. You can use the transaction recorder to record the program and then edit the recording.

Recording: ILIST_##

Recording transaction: BC420

Task:

Recording interactive lists

1. The program SAPBC420_SPTD_LIST_CHECKBOXES displays customer master data using an interactive list. Call this program, select some checkboxes, choose the magnifying glass symbol → detail display. (The detail display graphically presents the customer addresses.)
 2. Record this program (transaction code BC420).
 3. Directly execute the recording to test it. Was the list processed correctly and were the selection fields set correctly?
-
4. Edit and correct the recording in such a way that the checkbox can be set. Use the course folder as a guide.

Solution 20: Interactive Lists and Batch Input

Task:

Recording interactive lists

1. The program SAPBC420_SPTD_LIST_CHECKBOXES displays customer master data using an interactive list. Call this program, select some checkboxes, choose the magnifying glass symbol → detail display. (The detail display graphically presents the customer addresses.)
 - a) There is no solution.
2. Record this program (transaction code BC420).
 - a) There is no solution.
3. Directly execute the recording to test it. Was the list processed correctly and were the selection fields set correctly?

-
- a) The list was not processed correctly. The setting of checkboxes is missing from the recording. For example, if fields seven and eight are selected, the following entries must be included in the recording:

Underneath the module pool line SAPMSSY0, 120, X, add the following:

Field name	Field value
09/01	x and
10/01	x

As a result, checkboxes seven and eight are selected.

Make sure that the two lines above the first checkbox (list header and the underline) are included in the line numbering in the output list. For this reason, the line numbers nine and ten are used in this example to select checkboxes seven and eight. (The “02” is the column number).

4. Edit and correct the recording in such a way that the checkbox can be set. Use the course folder as a guide.
 - a) There is no solution.



Lesson Summary

You should now be able to:

- Use interactive lists for transferring data
- Evaluate the effects of the SAP LUW architecture on the external data transfer
- Execute a data transfer in the background
- Use BI utility programs



Unit Summary

You should now be able to:

- Use interactive lists for transferring data
- Evaluate the effects of the SAP LUW architecture on the external data transfer
- Execute a data transfer in the background
- Use BI utility programs



Course Summary

You should now be able to:

- Use the following techniques to transfer data:
- Use batch Input
- Use call transaction
- Use direct input
- Use IDocs
- Use BAPIs
- Use the Legacy System Migration Workbench (LSMW) as a tool
- Use the data transfer workbench (DX-WB) as a tool

Appendix 1

Tasks

Mapping Plans: Exercises and Solutions for Mapping

Here are the mapping plans required for the exercises. The plans that have not been filled are for practicing. They should be filled in the exercises about customers and documents. The plans that have been filled can be used as a guide for the exercises in the units about IDocs and BAPIs. If Excel is available in the training rooms, the participants can also fill the plans electronically.



Target Fields	Field Description	Type	Length	Source Fields	Length	Conversion Method	Coding/Comment
BGR00 – Batch Input Structures for Mapping Data							
STYPE	Batch input interface record type	CHAR	1				
GROUP	Queue group name	CHAR	12				
MANDT	Client	CLNT	3				
USNAM	Queue user ID/historic	CHAR	12				
START	Queue start date	DATS	10				
XKEEP	Indicator: Keep batch input session after processing?	CHAR	1				
NODATA	No batch input exists for this field	CHAR	1				
BKN00 – Customer Master Record Transaction Data for Batch Input							
STYPE	Batch input interface record type	CHAR	1				
TCODE	Transaction code	CHAR	20				
KUNNR	Customer Number	CHAR	10				
BUKRS	Company code	CHAR	4				
VKORG	Sales organization	CHAR	4				
VTWEG	Distribution channel	CHAR	2				
SPART	Division	CHAR	2				
KTOKD	Acct. Assignment Grp Cust.	CHAR	4				
KKBER	Credit control area	CHAR	4				

Figure 359: Exercises Mapping for customers (1)



BKNA1 – General Customer Master Record Part 1 (Batch Input)			
STYPE	Batch input interface record type	CHAR	1
TBNAM	Table name	CHAR	30
ANRED	Form of address	CHAR	15
NAME1	Name 1	CHAR	35
NAME2	Name 2	CHAR	35
NAME3	Name 3	CHAR	35
NAME4	Name 4	CHAR	35
SORTL	Sort field	CHAR	10
STRAS	Street and house number	CHAR	35
PFACH	Postal code	CHAR	10
ORT01	City	CHAR	35
PSTLZ	Postal code	CHAR	10
ORT02	District	CHAR	35
PSTL2	PO Box postal code	CHAR	10
LAND1	Country key	CHAR	3
REGIO	Region	CHAR	3
SPRAS	Language according to ISO 639	CHAR	2
TELX1	Telex number	CHAR	30
TELF1	1. Telephone number	CHAR	16
TELFX	Fax number	CHAR	31
LIFSD	Central delivery block for the customer	CHAR	2
FAKSD	Central billing block for customer	CHAR	2
STCEG	VAT registration number	CHAR	20
STKZN	Indicator: Business partner natural person?	CHAR	1

Figure 360: Exercises Mapping for customers (2)

BKNB1 – Customer Master Record Company Code Data (Batch Input)			
STYPE	Batch input interface record type	CHAR	1
TBNAM	Table name	CHAR	30
AKONT	Reconciliation account in general ledger	CHAR	10
ZUAWA	Key for sorting according to assignment numbers	CHAR	3
KNRZE	Head office account number (in branch accounts)	CHAR	10
BLNKZ	Subsidy indicator for determining the reduction rates	CHAR	2
XDEZV	Indicator: Local processing?	CHAR	1
LOEVM	Deletion flag for master record (company code level)	CHAR	1
SPERR	Posting block for company code	CHAR	1
ALTKN	Previous master record number	CHAR	10
ZGRUP	Key for payment grouping	CHAR	2

Figure 361: Exercises Mapping for customers (3)

Solution Unit: Tools for DX-WB



Target Fld				Source Fields			
Fld Name	Field Description	Type	Lgth	Field Name	Length	Conversion Method	Coding/Comment
BGR00 – Batch Input Structures for Mapping Data							
STYPE	Batch input Interface record type	CHAR	1			F	0
GROUP	Queue group name	CHAR	12				DEB-##
MANDT	Client	CLNT	3			V	SY-MANDT
USNAM	Queue user ID/historic	CHAR	12			V	SY-UNAME
START	Queue start date	DATS	10				
XKEEP	Indicator: Keep batch input session?	CHAR	1			F	X
NODATA	No batch input exists for this field	CHAR	1			F	/
BKN00 – Customer Master Record Transaction Data for Batch Input							
STYPE	Batch input interface record type	CHAR	1			F	1
TCODE	Transaction code	CHAR	20			F	XD01
KUNNR	Customer number	CHAR	10	C_NO		M + C	Z-##-00088
BUKRS	Company code	CHAR	4			F	0001
VKORG	Sales organization	CHAR	4				
VTWEG	Distribution channel	CHAR	2				
SPART	Division	CHAR	2				
KTOKD	Acct. Assignment Grp Cust.	CHAR	4			F	KUNA
KKBER	Credit control area	CHAR	4				

Figure 362: Solution: Mapping for customers (1)



BKNA1 – General Customer Master Record Part 1 (Batch Input)						
STYPE	Batch input interface record type	CHAR	1			F
TBNAM	Table Name	CHAR	30			F
ANRED	Form of Address	CHAR	15			
NAME1	Name 1	CHAR	35	NAME	M	Olli Klatt
NAME2	Name 2	CHAR	35			
NAME3	Name 3	CHAR	35			
NAME4	Name 4	CHAR	35			
SORTL	Sort Field	CHAR	10	SEARCH	M	Klatt
STRAS	Street and house number	CHAR	35	A_DATA	M	Ringstraße 10
PFACH	Postal code	CHAR	10			
ORT01	City	CHAR	35	B_DATA	M	Berlin
PSTLZ	Postal code	CHAR	10	ZIP	M	13407
ORT02	District	CHAR	35			
PSTL2	PO Box postal code	CHAR	10			
LAND1	Country Key	CHAR	3	CO	M	DE
REGIO	Region	CHAR	3			
SPRAS	Language according to ISO 639	CHAR	2	SPO	M+C	D -> DE
TELX1	Telex number	CHAR	30			
TELF1	1. Telephone number	CHAR	16	TELF	M	0302052345
TELFX	Fax number	CHAR	31			
LIFSD	Central delivery block for the customer	CHAR	2			
FAKSD	Central billing block for customer	CHAR	2			
STCEG	VAT registration number	CHAR	20	VAT	M	DE123456789
STKZN	Indicator: Business partner natural person?	CHAR	1			

Figure 363: Solution: Mapping for customers (2)



BKNB1 – Customer Master Record Company Code Data (Batch Input)						
STYPE	Batch input interface record type	CHAR	1			
TBNAM	Table name	CHAR	30			
AKONT	Reconciliation account in general ledger	CHAR	10		F	120000
ZUAWA	Key for sorting according to assignment numbers	CHAR	3			
KNRZE	Head office account number	CHAR	10			
BLNKZ	Subsidy indicator for determining the reduction rates	CHAR	2			
XDEZV	Indicator: Local processing?	CHAR	1			
LOEVM	Deletion flag for master record	CHAR	1			
SPERR	Posting block for company code	CHAR	1			
ALTKN	Previous master record number	CHAR	10	C_NO	M	4537
ZGRUP	Key for payment grouping	CHAR	2			

Figure 364: Solution: Mapping for customers (3)

Exercise Unit: LSMW



Target Fields				Source Fields			
Field Name	Field Description	Type	Lgth	Field Name	Length	Conversion Method	Coding/Comment
BGR00 – Batch Input Structures for Mapping Data							
STYPE	Batch input interface record type	CHAR	1			F	0
GROUP	Queue group name	CHAR	12				
MANDT	Client	CLNT	3			V	SY-MANDT
USNAM	Queue user ID/historic	CHAR	12			V	SY-UNAME
START	Queue start date	DATS	10				
XKEEP	Indicator: Keep batch input session?	CHAR	1			F	X
NODATA	No batch input exists for this field	CHAR	1			F	/
BBKPF – Document header for accounting document (batch input structure)							
STYPE	Batch input interface record type	CHAR	1			F	1
TCODE	Transaction code	CHAR	20			F	FB01
BLDAT	Date	CHAR	8				
BLART	Document Type	CHAR	2				
BUKRS	Company code	CHAR	4				
BUDAT	Date	CHAR	8				
MONAT	Fiscal period	CHAR	2				
WAERS	Currency Key	CUKY	5				
KURSF	Exchange rate direct quotation	CHAR	10				
BELNR	Accounting document number	CHAR	10				
WWERT	Date	CHAR	8				
XBLNR	Reference document number	CHAR	16				

Figure 365: Exercises Mapping for documents for RFBIBL00 (1)



BBSEG – Accounting Document Segment (Batch Input Structure)						
STYPE	Batch input interface record type	CHAR	1			
TBNAM	Table name	CHAR	30			
NEWBS	Posting Key for the next line item	CHAR	2			
DUMMY	Do not use field any more	CHAR	10			
NEWUM	Special G/L indicator for the next line item	CHAR	1			
NEWBK	Company code for the next line item	CHAR	4			
WRBTR	Amount in document currency	CHAR	16			
DMBTR	Amount in local currency	CHAR	16			
WMVST	Tax amount in document currency	CHAR	16			
MWSTS	Tax amount in local currency	CHAR	16			
MWSKZ	Tax code	CHAR	2			
PERNR	Employee ID	CHAR	8			
BEWAR	Consolidation transaction type	CHAR	3			
SGTXT	Item text	CHAR	50			
BLNKZ	Subsidy indicator for determining the reduction rates	CHAR	2			
SPRAS	Language key	LANG	2			
XINVE	Indicator: Capital goods affected?	CHAR	1			
NEWKO	Account or matchcode for the next line item	CHAR	17			
NEWBW	Asset transaction type	CHAR	3			

Figure 366: Exercises Mapping for documents for RFBIBL00 (2)

Solution Unit: LSMW



Target Fld				Source Fields			
Fld Name	Field Description	Type	Lgth	Field Name	Length	Conversion Method	Coding/Comment
BGR00 – Batch Input Structures for Mapping Data							
STYPE	Batch input interface record type	CHAR	1			F	0
GROUP	Queue group name	CHAR	12				
MANDT	Client	CLNT	3			V	SY-MANDT
USNAM	Queue user ID/historic	CHAR	12			V	SY-UNAME
START	Queue start date	DATS	10				
XKEEP	Indicator: Keep BI session?	CHAR	1			F	X
NODATA	No batch input exists for this field	CHAR	1			F	/
BBKPF – Document header for accounting document (batch input structure)							
STYPE	Batch input interface record type	CHAR	1			F	1
TCODE	Transaction code	CHAR	20			F	FB01
BLDAT	Date	CHAR	8	HEAD-DATE	10	M	Document Date
BLART	Document type	CHAR	2	HEAD-TYPE	4	M+C	07-> SA
BUKRS	Company code	CHAR	4			F	0001
BUDAT	Date	CHAR	8				
MONAT	Fiscal period	CHAR	2				
WAERS	Currency Key	CUKY	5	HEAD-CU	2	M+C	Currency \$-> USD and EU->EUR
KURSF	Exchange rate direct quotation	CHAR	10				
BELNR	Accounting document number	CHAR	10				
VWERT	Date	CHAR	8				
XBLNR	Reference document number	CHAR	16	HEAD-REF	9	M	

Figure 367: Solution: Mapping for documents for RFBIBL00 (1)



BBSEG – Accounting Document Segment (Batch Input Structure)						
STYPE	Batch input Interface record type	CHAR	1			F
TBNAM	Table name	CHAR	30			F
NEWSB	Posting key for the next line item	CHAR	2	POSI-KEY	4	M+C
						Posting key if 0004 ->40 , 0005 -> 50
DUMMY	Do not use field any more	CHAR	10			
NEWUM	Special G/L indicator for the next line item	CHAR	1			
NEWBK	Company code for the next line item	CHAR	4			
WRBTR	Amount in document currency	CHAR	16	POSI-AMOUNT	15	M
DMBTR	Amount in local currency	CHAR	16			
WMWST	Tax amount in document currency	CHAR	16			
MWSTS	Tax amount in local currency	CHAR	16			
MWSKZ	Tax code	CHAR	2			
PERNR	Employee ID	CHAR	8			
BEWAR	Consolidation transaction type	CHAR	3			
SGTXT	Item text	CHAR	50	POSI-TEXT	6	M
BLNKZ	Subsidy indicator for determining the reduction rates	CHAR	2			
SPRAS	Language key	LANG	2			
XINVE	Indicator: Capital goods affected?	CHAR	1			
NEWKO	Account or Matchcode for the next line item	CHAR	17	POSI-ACCOUNT	10	M
NEWBW	Asset transaction type	CHAR	3			Document account

Figure 368: Solution: Mapping for documents for RFBIBL00 (2)

Solution Unit: IDoc



Target field				Source fields			
Field name	Field description	Type	Lgth.	Field name	Lgth.	Conversion method	Coding/Comment
E1FIKPF - FI-IDOC: Document header (complete document)							
BUKRS	Name of global company code	CHAR	6			Constant	E1FIKPF-BUKRS = 'GL0001'.
BELNR	Accounting document number	CHAR	10				
FYEAR	Fiscal year	NUMC	4			Constant	E1FIKPF- GJAHR = '2002'.
BLART	Document type	CHAR	2	HEAD-TYPE	4	M+C	Perform Ismw_translate using 'BLART' HEAD-TYPE changing E1FIKPF- BLART."
BLDAT	Document date in document	DATS	8	HEAD-DOC_DATE	10	MOVE	* Date in internal format (YYYYMMDD) E1FIKPF-BLDAT = HEAD-DOC_DATE.
BUDAT	Posting date in the document	DATS	8	HEAD- POSTING_DATE	10	MOVE	* Date in internal format (YYYYMMDD) E1FIKPF-BUDAT = HEAD_ POSTING_DATE.
MONAT	Fiscal period	NUMC	2				
WWERT	Translation date	DATS	8	HEAD_TRANS_DATE	10	MOVE	E1FIKPF-WWERT = HEAD_TRANS_DATE.
USNAM	User name:	CHAR	12				
TCODE		CHAR	4				
BVORG	Number of cross-company code posting transaction	CHAR	16				

Figure 369: Solution: Mapping for documents for IDoc (1)



XBLNR	Reference document number	CHAR	16	HEAD-REF_NO	9	MOVE	E1FIKPF-XBLNR = HEAD-REF_NO.
BKTXT	Document header text	CHAR	25				
WAERS	Currency key	CUKY	5	HEAD-CURR_KEY	2	M+C	perform lsmy_translate using 'WAERS' changing E1FIKPF-WAERS.
KURSF	Exchange rate	DEC	11				
GLVOR	Business transaction	CHAR	4			Constant	E1FIKPF-GLVOR = 'RFBU'.
AWTYP	Reference procedure	CHAR	5				
AWREF	Reference document number	CHAR	10				
AWORG	Reference organizational units	CHAR	10				
FIKRS	Financial management area	CHAR	4				
HWAER	Local currency	CUKY	5			Constant	E1FIKPF-HWAER = 'EUR'.
E1FISEG - FI-DOC: Item data of a complete FI document							
BUZEI	Line item number within the accounting document	NUMC	3		1	ABAP code	E1FISEG-BUZEI = pos_counter.
BUZID	Identification of the line item	CHAR	1				
AUGDT	Clearing date	DATS	8				
AUGCP	Clearing entry date	DATS	8				
AUGBL	Document number of the clearing document	CHAR	10				
KOART	Account type	CHAR	1				
SHKZG	Debit/credit indicator		1			ABAP code	IF e1fiseg-bschl = '50'. e1fiseg-shkzg = 'H'. ELSE. e1fiseg-shkzg = 'S'. ENDIF.

Figure 370: Solution: Mapping for documents for IDoc (2)



GSBER	Globally unique business area	CHAR	4				
PARGB	Globally unique business area	CHAR	4				
MWSKZ	Tax on sales/purchases code	CHAR	2				
DMBTR	Amount in local currency	CURR	15	POSI-LOCAL_AMOUNT	15	MOVE	E1FISEG-DMBTR = POSI-LOCAL_AMOUNT.
DMBE2	Amount in second local currency	CURR	15				
DMBE3	Amount in third local currency	CURR	15				
WRBTR	Amount in document currency	CURR	15	POSI-DOC_AMOUNT	15	MOVE	
							E1FISEG-WRBTR = POSI-DOC_AMOUNT.
KZBTR	Original reduction amount in local currency	CURR	15				
PSWBT	Amount for updating in general ledger	CURR	15				
PSWSL	Update currency for general ledger transaction figures	CUKY	5				
HWBAS	Tax base amount in local currency	CURR	15				
FWBAS	Tax base amount in document currency	CURR	15				
MWART	Tax type	CHAR	1				
KTOSL	Transaction key	CHAR	3				
VALUT	Value date	DATS	8				
ZUONR	Assignment number	CHAR	18				
SGTXT	Item text	CHAR	50	POSI-TEXT	30	MOVE	E1FISEG-SGTXT = POSI-TEXT.
VBUND	Company ID of trading partner	CHAR	6				
BEWAR	Transaction type	CHAR	3				
VORGN	Transaction type for general ledger	CHAR	4			Constant	E1FISEG-VORGN = 'RFBU'.
FDLEV	Planning level	CHAR	2				
SAKNR	G/L account number	CHAR	10	POSI-ACCOUNT	10	ABAP code	* Transfer (MOVE)
							E1FISEG-SAKNR = POSI-ACCOUNT.

Figure 371: Solution: Mapping for documents for IDoc (3)



						CALL FUNCTION 'CONVERSION_EXIT_ALPHA_INPUT'
						EXPORTING
						input = E1FISEG-SAKNR
						IMPORTING
						OUTPUT = E1FISEG-SAKNR.
						.
HKONT	General ledger account	CHAR	10		ABAP code	E1FISEG-HKONT = E1FISEG-SAKNR.

Figure 372: Solution: Mapping for documents for IDoc (4)

Solution Unit: BAPI



Target Fields				Source Fields			
Field Name	Field Description	Type	Length	Field Name	Length	Conversion Method	Coding/Comment
E1FIXEDASSET_CREATEINCLVALU – Header segment							
CREATESUBNUMBERTER	Subnumber assets	CHAR	1				
TESTRUN	Simulation mode	CHAR	1				
E1BP1022_KEY – Key fields for create							
COMPANYCODE	Company code	CHAR	4	ASSET_00-CODE		MOVE	E1BP1022_KEY-COMPANYCODE = ASSET_00-CODE
ASSET	Assets - main number	CHAR	12				
SUBNUMBER	Assets subnumber	CHAR	4				
E1BP1022_REFERENCE – Reference asset for creating							
COMPANYCODE	Company code	CHAR	4		4		
ASSET	Assets - main number	CHAR	12				
SUBNUMBER	Assets subnumber	CHAR	4				
E1BP1022_FEGLG001 - Logical field group 001 – General data							
ASSETCLASS	Asset class	CHAR	8	ASSET_00-CLASS	4	M+C	perform lsmw_translate
							using 'ASSETCLASS'
							ASSET_00-CLASS

Figure 373: Solution: Mapping for customers (1)



						changing E1BP1022_FEGLG001-ASSETCLASS.	
						CALL FUNCTION 'CONVERSION_EXIT_ALPHA_INPUT'	
						EXPORTING	
						input= E1BP1022_FEGLG001-ASSETCLASS	
						IMPORTING	
						OUTPUT= E1BP1022_FEGLG001-ASSETCLASS.	
DESCRIPT	Description of the asset.	CHAR	50	ASSET_00-ASSET	20	MOVE	E1BP1022_FEGLG001-DESCRIPT = ASSET_00-ASSET.
DESCRIPT2	Description of the asset.	CHAR	50				
ACCT_DETRM	Account determination	CHAR	8				
SERIAL_NO	Serial number	CHAR	18				
INVENT_NO	Inventory number	CHAR	25				
QUANTITY	Quantity	QUAN	15				
BASE_UOM	Base Unit of measure	UNIT	3				
BASE_UOM_ISO	Base Unit of measure	CHAR	3				
HISTORY	Managed historically	CHAR	1				
E1BP1022_FEGLG001X – Logical field group 001 – General data							
ASSETCLASS	Change info	CHAR	1		Constant	E1BP1022_FEGLG001X-ASSETCLASS = 'X'.	
DESCRIPT	Change info	CHAR	1		Constant	E1BP1022_FEGLG001X-DESCRIPT = 'X'.	

Figure 374: Solution: Mapping for customers (2)



E1BP1022_FEGLG011 – Logical field group 011 – Inventory						
DATE	Last inventory date	DATS	8			
NOTE	Supplementary inv. specifications	CHAR	15			
INCLUDE_IN_LIST	Inventory indicator	CHAR	1			
E1BP1022_FEGLG011X – Logical field group 011 – Inventory						
DATE	Change info	CHAR	1			
NOTE	Change info	CHAR	1			
INCLUDE_IN_LIST	Change info	CHAR	1			
E1BP1022_FEGLG002 – Logical field group 002 – Posting information						
CAP_DATE	Activation data asset	DATS	8	ASSET_00- DATE	8	MOVE
						E1BP1022_FEGLG002-CAP_DATE = ASSET_00-DATE.
CAP_KEY	Results analysis key	CHAR	6			
E1BP1022_FEGLG002X – Logical field group 002 – Posting information						
CAP_DATE	Change info	CHAR	1		Constant	E1BP1022_FEGLG002X-CAP_DATE = 'X'.
E1BP1022_FEGLG003 – Logical field group 003 – Time dependent data						
FROM_DATE	Date for beginning of validity	DATS	8			
TO_DATE	Date validity ends	DATS	8			

Figure 375: Solution: Mapping for customers (3)



BUS_AREA	Business area	CHAR	4	ASSET_00- AREA	4	MOVE	E1BP1022_FEGLG003-BUS_AREA = ASSET_00-AREA;
COSTCENTER	Cost Center	CHAR	10	ASSET_00- COSTCENTER	4	M+C	perform lsmw_translate using 'KOSTL' ASSET-COSTCENTER changing E1BP1022_FEGLG003- COSTCENTER.
PLATE_NO	License plate number	CHAR	15	ASSET_00- PLATENO	10	MOVE	E1BP1022_FEGLG003-PLATE_NO = ASSET_00-PLATENO.
PERSON_NO	Employee ID	NUMC	8				
SHIFT_FACT	Multiple-shift factor for multiple shift operation	DEC	5				
SHUTDOWN	Asset shutdown	CHAR	1				
E1BP1022_FEGLG003X – Logical field group 003 – Time dependent data							
FROM_DATE	Change info	CHAR	1				
TO_DATE	Change info	CHAR	1				
BUS_AREA	Change info	CHAR	1		Constant	E1BP1022_FEGLG003X-BUS_AREA = 'X'.	
COSTCENTER	Change info	CHAR	1		Constant	E1BP1022_FEGLG003X-COSTCENTER = 'X'.	
LICENSE_PLATE_NO	Change info	CHAR	1		Constant	E1BP1022_FEGLG003X- LICENSE_PLATE_NO = 'X'.	

Figure 376: Solution: Mapping for customers (4)

Feedback

SAP AG has made every effort in the preparation of this course to ensure the accuracy and completeness of the materials. If you have any corrections or suggestions for improvement, please record them in the appropriate place in the course evaluation.