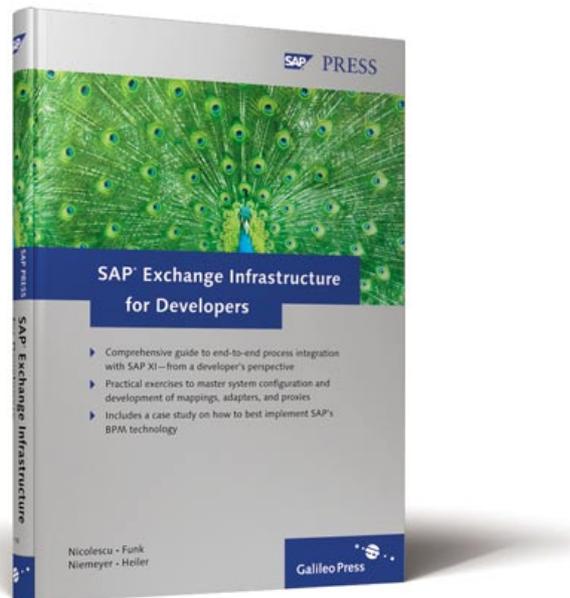


Valentin Nicolescu, Burkhardt Funk, Peter Niemeyer,
Matthias Heiler, Holger Wittges, Thomas Morandell,
Florian Visintin, Benedikt Kleine Stegemann

SAP® Exchange Infrastructure for Developers



Galileo Press 

Bonn • Boston

Contents at a Glance

	Foreword	11
	Preface	13
	PART I BASICS	
1	Integrating Enterprise Information Systems	19
2	SAP NetWeaver Exchange Infrastructure	41
	PART II PRACTICES AND EXERCISES	
3	Basic System Configuration	79
4	Technical Exercises	117
5	SARIDIS Case Study in Sales and Distribution	199
6	Enhancements and Outlook	289
	APPENDIX	
A	Exercise Materials	315
B	Bibliography	329
C	The Authors	331
	Index	335

Contents

Foreword	11
Preface	13

PART I BASICS

1 Integrating Enterprise Information Systems 19

1.1	Basic Principles	19
1.1.1	Historical development	19
1.1.2	Reasons for and goals of integrating IT systems	21
1.1.3	Characteristics of integration	22
1.2	Practical Integration Example	24
1.3	Integration Concepts and Technologies	28
1.3.1	Architectures	28
1.3.2	Integration approaches	30
1.3.3	Technologies	33
1.4	EAI Platforms and Their Significance in Enterprises	36
1.5	Basics of Business Process Management	38

2 SAP NetWeaver Exchange Infrastructure 41

2.1	SAP XI as Part of SAP NetWeaver	41
2.1.1	Challenges of Process Integration	41
2.1.2	SAP NetWeaver	46
2.1.3	IT Practices and IT Scenarios	47
2.2	Functionality	53
2.2.1	Address Example	55
2.2.2	Classification of Messages	55
2.2.3	Implementing a Message Flow	59
2.3	Components	60
2.4	Objects	62
2.4.1	Software Products in the Integration Repository	63
2.4.2	Message Interfaces and Mapping Objects in the Integration Repository	64
2.4.3	Configuration Objects in the Integration Directory	67

Contents

2.5	Advanced Concepts	69
2.5.1	Outside-In vs. Inside-Out	69
2.5.2	Generating ABAP Proxies	70
2.5.3	RFC Adapter	71
2.5.4	Other Adapters	72
2.5.5	Mappings	73
2.5.6	Monitoring Messages and Processes	74
2.5.7	Services and Partners	75

PART II PRACTICES AND EXERCISES

3	Basic System Configuration	79
3.1	Prerequisites	79
3.2	Defining the Connected Systems in the System Landscape Directory	80
3.2.1	Creating the Systems—Technical Systems	81
3.2.2	Creating the Systems in the SLD— Business Systems	85
3.3	Integrating the SAP Systems with the SLD	87
3.3.1	Creating the RFC Connections	87
3.3.2	Configuring the SLD Integration	90
3.4	Configuring the Local Integration Engine	92
3.4.1	Defining the Role of the Business System	93
3.4.2	Defining and Activating Message Queues	95
3.4.3	Activating the XI Service	96
3.4.4	Establishing the Connection to the Integration Builder and the Runtime Workbench	97
3.5	Adapter-Specific System Settings	100
3.5.1	Checking the ABAP-Proxy Integration	100
3.5.2	Settings for the Use of the RFC Adapter	101
3.5.3	Settings for the Use of the IDoc Adapter	102
3.6	Course-Specific Preparations	106
3.6.1	Creating and Assigning the Software Product	107
3.6.2	Importing the Software Product in the Integration Repository and Setting it Up	111

4	Technical Exercises	117
4.1	Exercise 1: RFC-to-File	119
4.1.1	Basic Principles	119
4.1.2	Design	121
4.1.3	Configuration	133
4.1.4	Process and Monitoring	140
4.2	Exercise 2: File-to-IDoc	144
4.2.1	Basics	145
4.2.2	Design	147
4.2.3	Configuration	151
4.2.4	Process and Monitoring	158
4.3	Exercise 3: ABAP-Proxy-to-SOAP	161
4.3.1	Basics	162
4.3.2	Design	163
4.3.3	Configuration	172
4.3.4	Process and Monitoring	175
4.4	Exercise 4: Business Process Management	176
4.4.1	Basics	176
4.4.2	Design	178
4.4.3	Configuration	189
4.4.4	Process and Monitoring	192
4.5	More Adapters	194
4.5.1	JDBC Adapters	195
4.5.2	JMS Adapter	195
4.5.3	SAP Business Connector Adapter	196
4.5.4	Plain HTTP Adapter	197
5	SARIDIS Case Study in Sales and Distribution	199
5.1	Creating the Query	201
5.1.1	Basic Principles	202
5.1.2	Design	204
5.1.3	Configuration	214
5.1.4	Process and Monitoring	218
5.2	Submitting the Quotations	220
5.2.1	Basic Principles	220
5.2.2	Design	223
5.2.3	Configuration	242
5.2.4	Process and Monitoring	251
5.3	Entering a Sales Order	254
5.3.1	Basic Principles	255

Contents

5.3.2	Design	256
5.3.3	Configuration	267
5.3.4	Process and Monitoring	272
5.4	Delivering the Invoice	275
5.4.1	Basic Principles	276
5.4.2	Design	277
5.4.3	Configuration	280
5.4.4	Process and Monitoring	283
6	Enhancements and Outlook	289
6.1	Possible Enhancements of the Case Study	289
6.2	Beer Distribution Game	291
6.2.1	Predefined Software Components	293
6.2.2	Design and Implementation	295
6.2.3	Options for Enhancement	296
6.3	SAP XI and Enterprise Services	296
6.3.1	Composite Applications	297
6.3.2	From Web Services to Enterprise Services	300
6.3.3	SAP XI as a Service Infrastructure	301
6.4	Further Development of SAP XI	302
6.4.1	SAP NetWeaver as a Business Process Platform	302
6.4.2	Enterprise Services Repository	304
6.4.3	Modeling Next Practices from Best Practices	307
6.4.4	Business Activity Monitoring	308
	Appendix	313
A	Exercise Materials	315
B	Bibliography	329
C	The Authors	331
	Index	335

Foreword

The current discussion about *enterprise service-oriented architecture* (enterprise SOA) is firmly rooted in the idea of integrating specialist enterprise services to form flexible and modular solutions. In this context, integration platforms are used to connect and coordinate individual services on the technical level. Integration platforms thus play a central role in the modern enterprise landscape. Analyzing and learning about these platforms is therefore a key requirement in the design of modern software architectures.

Integration platforms come in many shapes and sizes, from comprehensive EAI solutions to lighter-weight offerings that currently tend to be known as *enterprise service bus* (ESB). The heterogeneity of these solutions alone prompts us to look at this topic in greater detail. Also, the sheer quantity of very specific requirements that arise from integrating processes within and across enterprises creates a significant barrier to entry that the user must first overcome.

This book is intended as a “leg-up” over this barrier. It is the result of a process of cross-pollination between theory and practice—theory in terms of its structured approach and didactic presentation of a complex topic; and practice in that it analyzes part of a piece of enterprise software that has become indispensable for most enterprises worldwide. The book invites the reader to explore new concepts by means of comprehensive examples. The focus is on practical issues in the areas of application of *SAP NetWeaver Exchange Infrastructure* (SAP XI), while the relevant underlying theory is explained to the degree that is necessary for the reader to understand the exercises.

Our hope is that this book will help many readers to get to grips with the topic of process and integration design on the basis of the SAP XI.

Foreword

On that note, we wish you interesting and enlightening reading!

Prof. Dr. Helmut Krcmar

(Chair of Information Management,
Technische Universität München)

Dr. Wolfgang Fassnacht

(Senior Product Manager, SAP NetWeaver
Process Integration and Enterprise Services)

The exercises in this chapter show how to use SAP NetWeaver Exchange Infrastructure (XI) components by presenting scenarios that are linked in content but technically independent to prepare you for the case studies in Chapters 5 and 6.



4 Technical Exercises

Use of the available concepts and adapters of the SAP NetWeaver Exchange Infrastructure (XI) is the basis for implementing complex integration scenarios. This chapter shows you how to configure adapters, create mappings, and monitor the course of scenarios. The individual exercises build on each other and get more complex so you gain the knowledge necessary to implement the case studies presented in Chapters 5 and 6. Although the individual exercises depend on each other, you can use the lists of every exercise to track which objects are reused so you can start with a more advanced lesson. Pre-defined objects aren't used in the exercises, so you can reproduce all of the steps for completing the integration scenario at any time.

All exercises are designed in such a way that they can be performed by a group of participants at the same time. Still, some steps can only be carried out once. For steps that must be performed by the instructor, you will be notified either prior to or during the course.

Appropriate for several participants

Although the exercises are appropriate for workgroups, you can also complete them on your own. In this case, perform the steps to be implemented by the instructor as well. Even if you are going to complete the exercises alone, we recommend you to use a user number as this simplifies the comparison of your work with the described procedure. In this case, you should use the instructor's number of 00.

Most exercises are completed as a development consultant, and particularly in the beginning of the exercise block, you will assume the role of the system administrator or the lead developer. In some places, you will have the opportunity to develop your own small applications in ABAP or Java. You hardly need any prerequisites

Possibility of your own developments

because the sample listings can be found in the Appendix of this book as well as in digital form on the web site for this book under <http://www.sap-press.com> and <http://www.sap-press.de/1383>.

Exercises as a preparation for the case study

The exercises deal with selected adapters and aspects of the XI environment. Various elements played a role when selecting the integration scenarios. When an adapter can be reused in a later exercise, its necessary aspects are discussed in that section. In hands-on exercises that are independent of each other, you will get to know the elements of SAP XI.

Even though the individual exercises do not have to be completed in the given order, there is a reason why we chose this order. In this chapter, you will implement integration scenarios that can be regarded as a preparation for the case study in the next chapter. Using XI messages, you will create a material in another system and verify its success. In a final step, the creation of material master data is reported to the person responsible for all material. A business process ensures that these reports are delivered in bundles per agent.

Procedures of the exercises

At first, you will use an ABAP program in System A, which records the data of a material to be created and transfer this data to the XI system using the RFC adapter (see Section 4.1). There, the material master record is converted to a file in XI format via the file adapter. In the next exercise, this file is read via the file adapter and converted into an IDoc which is transferred to and directly processed on System B (see Section 4.2). In the third example, you will check to ensure that the material has been created successfully. Based on an ABAP proxy, you will send a call to the XI system. This request is converted to a web service call provided by System B. The response of this control is returned synchronously to the calling System A (see Section 4.3). The agent uses an ABAP program to report the successful creation of the material master records using a business process (see Section 4.4).

Even though the contents of the individual exercises are based on one another, you can start with any of the exercises by using the appropriate templates.

4.1 Exercise 1: RFC-to-File

In the first exercise you will use your own ABAP program to call a remote-enabled function module that transfers the entered material master data via the RFC adapter to the XI system. Once there, data is converted to the XI XML format and stored as a file. To keep things simple, the file is created directly on the file system of the XI server. Later, you will see what changes are necessary to store the file per FTP on another machine.

Course of the first exercise

Although the file technically remains on the XI system, from a logical viewpoint you will configure the receiving file adapter for System B. The communication in this integration scenario is asynchronous because no business response is returned after sending the material data. The roles of System A and B, as well as the adapters used in this exercise, are schematically illustrated in Figure 4.1.

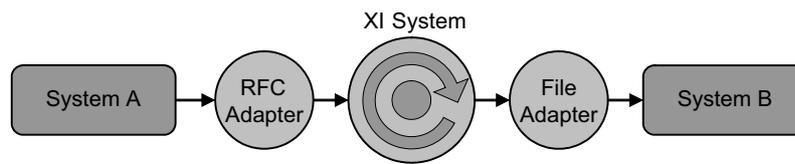


Figure 4.1 Scheme of Exercise 1: RFC-to-File

4.1.1 Basic Principles

Because this book does not focus on development of ABAP programs and remote-enabled function modules, we will only give you a basic explanation of the used program and the function module. You can get an appropriate transport with the function module and the program for 20 participants and 1 instructor from the book's web page and implement it in your System A. For implementing the transport, consult your landscape administrator, if necessary.

New ABAP components

If you want to create the program and the function module yourself, you will find the corresponding sample source code in Appendix A of this book.

First, log on to the client of System A as the user **SYS_A-##**, where **##** is your participant number. There, you can view the remote-enabled function module using the Function Builder in **Transaction SE37**. Select the function module **Z_RFM_MATERIALINPUT**.

Structure of the function module

You will see that from the function module's point of view the parameters are only imported and no value is returned. This is one of the two prerequisites for asynchronous communication from a sending RFC adapter.

Except for the interface definition, the function module does not contain any ABAP code. This means that the function module is used as a kind of dummy that forwards the transferred data to SAP XI and serves as the definition of an interface. The information about where the data transferred to the function module is to be forwarded is contained in the calling program.

Function of the ABAP program

The program `Z_PROG_MATERIALINPUT_###`, which you can find together with the function module in the same transport request and that you can view in **Transaction SE38**, has two functions:

- ▶ First, it accepts the basic material master data that will be used to create a new material in System B.
- ▶ Second, it calls the function module (described earlier) with the parameters listed in Table 4.1.

The calling of these parameters is explained in the second exercise, in Section 4.2.

Transferred Data	Description
MATNR	Material number
MAKTX	Material description
ERSDA	Creation Date (will be added automatically)
ERNAM	User name of creator (will be added automatically)
MTART	Material type
MBRSH	Industry sector
MATKL	Material group
MEINS	Quantity unit
BRGEW	Gross weight
GEWEI	Weight unit
MTPOS_MARA	General item category group

Table 4.1 Data Transferred to the Function Module `Z_RFM_MATERIALINPUT_###`

In this call, two things need to be mentioned, in particular:

- ▶ The remote-enabled function module is called in a specific destination (i.e., in the system behind this RFC connection). In the case of the destination **SystemA_Sender-##**, this is the XI system, so the values transferred to the function module are forwarded to the XI system.
- ▶ The second aspect is the call in the background that causes the communication to be asynchronous.

4.1.2 Design

At first, you need to create the various data and message types as well as the message interfaces with the required mappings in the Integration Repository. In a later phase of the configuration, these elements will be linked to the connected business systems (System A and System B).

Creating the design objects in the Integration Repository

First, call **Transaction SXMB_IFR** from one of the connected systems or the XI system itself. This opens the menu of the XI tools in your web browser which will look familiar to you if you already carried out the preparation of the exercises (see Chapter 3). At the top-left, select the entry to the **Integration Repository**.

First steps in the Integration Repository

After the Java Web Start application has been updated and you have logged into the XI system as the appropriate user, the user interface to the Integration Repository is displayed. Ensure that you do not logon using the initial password; instead, change it during the logon to the SAP GUI.

On the left side, you will find the software components that have already been declared. This also includes the software component **SC_Training_XI_##** with the namespace `http://www.sap-press.com/xi/training/##`, which is where you will store your elements in the Integration Repository.

If you take a look at the individual categories within the namespace, you will notice that there are only two data types that have been generated during the namespace creation. For a better overview, restrict the view to your software component. In the tree structure, click on your software component, and above the tree select the icon **Only display selected subtree**.

The Integration Repository should then be presented as shown in Figure 4.2.

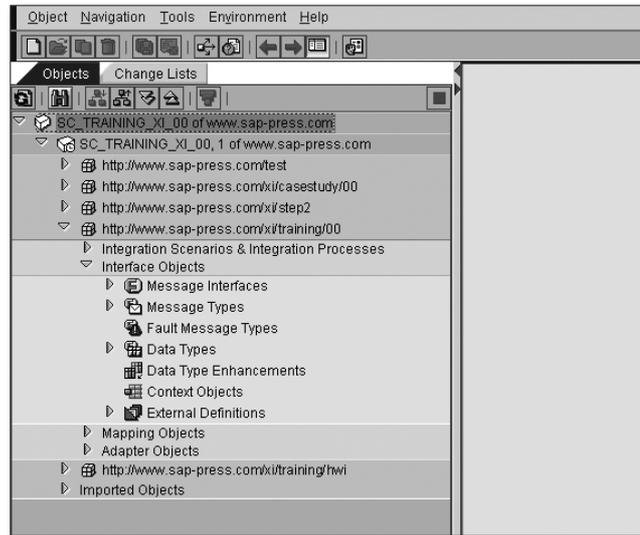


Figure 4.2 Entry to the Integration Repository

An overview of the elements required for this exercise is given in Table 4.2. The roles of individual elements and their connections have already been explained in Chapter 2.

Object Type	Sender Side	Receiver Side
Message interface	Z_RFM_MATERIALINPUT_##	MI_Material_Async_In
Message type		MT_Material
Data type		DT_Material
Interface mapping	IM_Z_RFM_MATERIALINPUT_##_to_MI_Material_Async_Out	
Message mapping	MM_Z_RFM_MATERIALINPUT_##_to_MT_Material	

Table 4.2 Elements in the Integration Repository for the First Exercise

Solutions to interrupted connections

Note
 If the connection to the Integration Repository is interrupted while an object is being edited, you can click on the **Administration** option in the XI tools on the right side and release the locked object for editing in the **Lock Overview** area.

Creating Elements for the Sending System

Due to the use of an RFC adapter, this scenario has a particular aspect: all elements on the sender side are replaced with the interface definition of the RFC module. Instead, the interface is imported from System A (and not created in the Integration Repository) to accelerate work and reduce the error rate.

Design objects on the sender side

To import the RFC interface, expand the bottom directory **Imported Objects** and right-click to open the context menu to find the **Import of SAP Objects** function. In the following window, select the option **According to Software Component Version** in the **Connection Data** area because the system data has already been stored (see Figure 4.3). If this option is not available, enter the host name and the system number of System A. Next, enter your user **SYS_A-##** and the appropriate password before continuing.

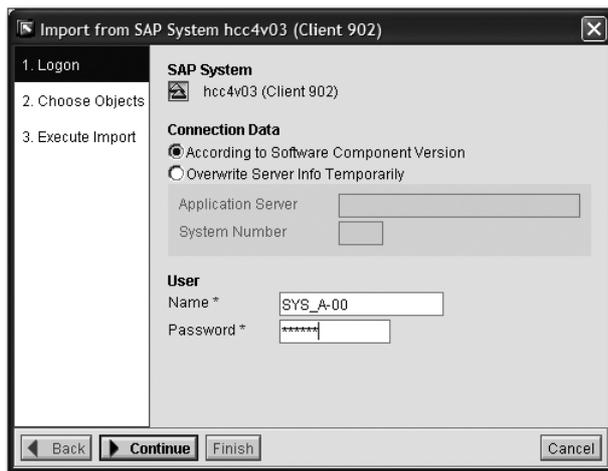


Figure 4.3 Import of RFC Interfaces—Login

The next step lets you choose between RFC and IDoc interfaces. Expand the RFC option. All remote-enabled function modules in System A are determined and displayed. Because this data collection can take a while, it is possible to import all interfaces before starting the exercise when you perform these steps in a group. From the list, select your function module **Z_RFM_MATERIALINPUT_##** (see Figure 4.4) and continue with the import.

Import the RFC interface

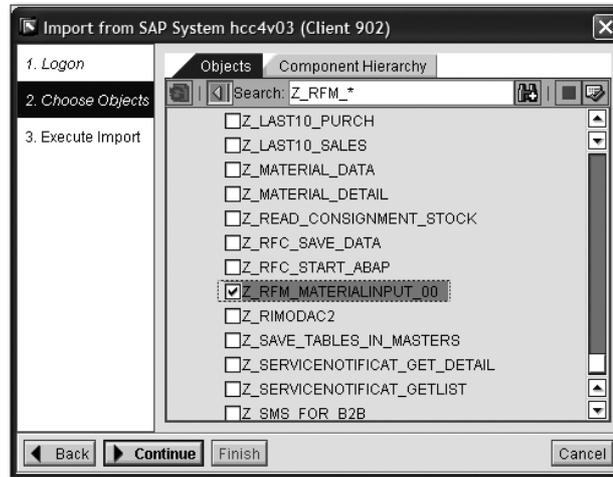


Figure 4.4 Import of RFC Interfaces—Selection

In the final step of the import process, check your selection and finish the import. After the import has completed, you can see your newly imported interface for your software component version in the directory **Imported Objects • RFC**. It is marked with a separate icon that indicates that this element has not yet been activated.

Creating Elements for the Receiving System

Design objects on the receiver side

While all elements are created on the side of the sending system by importing the RFC interface, you will create a data type, a message type, and a message interface for the receiving system. It is recommended to begin with the individual elements (i.e., with those on the lowest hierarchy level), which in this case is the data type.

Creating a data type

Within your namespace, expand the **Data Types** directory and open the creation dialog via the **New** entry of the context menu. In this window, you have the option to enter the name of the new object along with a description (see Figure 4.5).

The namespace and the software component version are automatically completed because you called the dialog in the appropriate context. Also, it is important to note the left area of this screen, which lists the elements that can be created within the Integration Repository. You can change the kind of element you want to create at any

time. You will see a similar structure later when working in the Integration Directory.

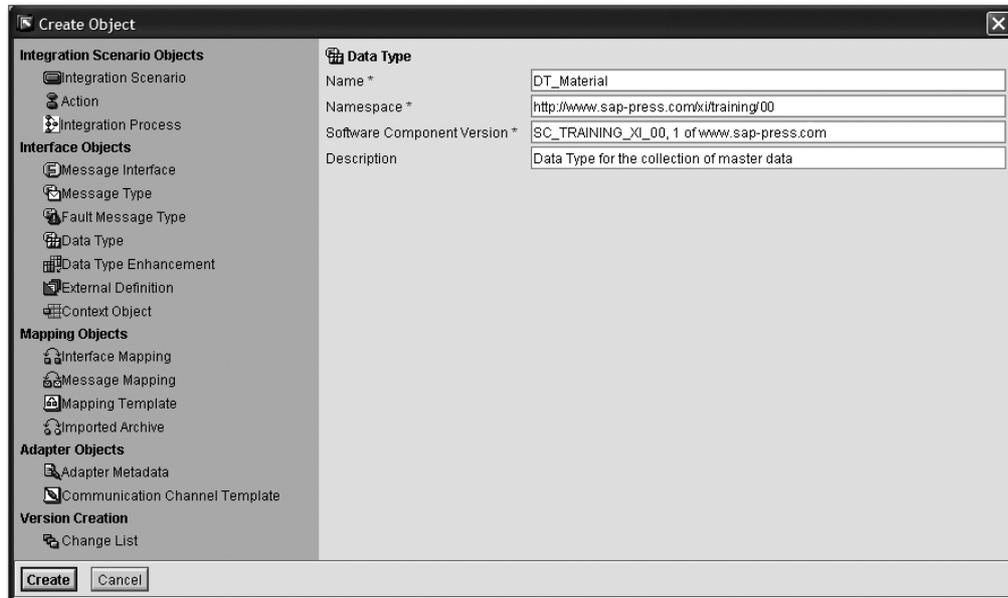


Figure 4.5 Dialog for Creating an Object in the Integration Repository

Name the new data type **DT_Material** and click on **Create**. The details window of a new data type is displayed on the right side. Because the structure of this window is typical of all detail views in the Integration Builder, it is used to explain some functions.

Next to the menu line of the Integration Repository is a separate details menu, the most important functions of which are also displayed as icons to its right. In addition to the icons for switching between the display and changing mode, and for creating a copy, you will also find an icon for the where-used list of this element, for example. The icon group to the right allows you to control the view. For example, you can hide header data or detach the details window as an independent element.

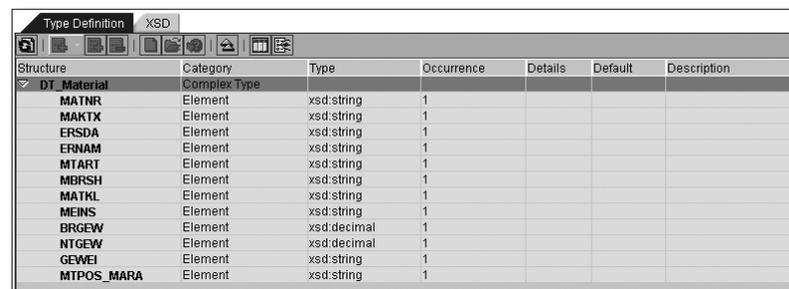
In the case of your data type, the lower area of the details window contains a list of all data type elements. Using the relevant icons you can add new rows to the top of the table and enter the elements from Table 4.1. Please note that this is the type **xsd:string**. Only the

Structure of the data type DT_Material

BRGEW element has the type **xsd:decimal**; you will later perform a calculation using this value (see Figure 4.6). Additionally, add the missing element NTGEW of the type **xsd:decimal**. You will use this element to calculate the net weight of the material (based on the gross weight) in the message mapping. Save the data type after all of the elements have been inserted.

Creating a message type

Since data types in the XI environment are exclusively used for the modularization of data formats, and cannot appear in a mapping or interface themselves, they are embedded in message types. While data types can only be assigned to message types in a 1:1 ratio, data types can be combined in any ratio.



Structure	Category	Type	Occurrence	Details	Default	Description
DT_Material	Complex Type					
MATNR	Element	xsd:string	1			
MAKTX	Element	xsd:string	1			
ERSDA	Element	xsd:string	1			
ERNAM	Element	xsd:string	1			
MTART	Element	xsd:string	1			
MBRSH	Element	xsd:string	1			
MATKL	Element	xsd:string	1			
MEINS	Element	xsd:string	1			
BRGEW	Element	xsd:decimal	1			
NTGEW	Element	xsd:decimal	1			
GEWEI	Element	xsd:string	1			
MTPOS_MARA	Element	xsd:string	1			

Figure 4.6 Editing a Data Type

To create a message type, open the appropriate context menu by right-clicking on the directory **Message Types**. Select the **New** option. The familiar creation dialog box is displayed, this time for a message type. Name the new object **MT_Material** and enter a description. Continue with the detail view by clicking **Create**. Pay attention to the **Data Type Used** area in the middle; this is where you should insert the data type you just created. You have three different possibilities of doing so:

Methods for selecting the data type

- ▶ The most obvious method is typing the name and the namespace, which, however, involves the risk of mistyping.
- ▶ The second option is to select the object in an ABAP-based SAP system, such as in the input help. For this, click on the hand and question mark icon to the right of the namespace field. A window opens, containing all of the data types created in your software component version for selection. The two fields **Name** and **Namespace** are then populated.

- ▶ The third option is to Drag and Drop the selection. This is particularly suitable if your software component version contains a lot of data types, but there are only few in your namespace. You can also pick the data type from the directory structure to the left and drop it on the hand next to the namespace field. Only by dropping it over the hand you can ensure a correct data transfer.

As you can see, all three possibilities work, even without activating the data type.

After selecting the appropriate data type, the lower area of the details window shows the structure of the used data type (see Figure 4.7). Check the structure and save the message type.

Structure	Category	Type	Occurrence	Details	Default	Description
MT_Material	Element	DT_Material				
MATNR	Element	xsd:string	1			
MAKTX	Element	xsd:string	1			
ERSDA	Element	xsd:string	1			
ERNAM	Element	xsd:string	1			
MTART	Element	xsd:string	1			
MBRSH	Element	xsd:string	1			
MATKL	Element	xsd:string	1			
MEINS	Element	xsd:string	1			
BRGEW	Element	xsd:decimal	1			
NTGEW	Element	xsd:decimal	1			
GEWEI	Element	xsd:string	1			
MTPOS	Element	xsd:string	1			

Figure 4.7 Editing a Message Type

The last object on the receiver side is the message interface, which determines if a message can be received or sent and whether the message is sent synchronously or asynchronously.

Creating
a message
interface

To create this message interface, open the context menu of the corresponding directory. Enter the name **MI_Material_Async_In** and an appropriate description, and then click **Create** to get to the details window. You can choose between the options **Inbound**, **Outbound**, and **Abstract** for the interface category; the individual categories have been discussed in Chapter 2. Because we are dealing with the interface on the receiver side, select **Inbound**. The communication mode determines whether a response regarding the contents is expected or not. Because this is a one-way scenario, select the **Asynchronous** mode.

You have probably noticed that the input options for message types change with every time the attributes are modified. You should now see the fields for the **Input Message** and the **Fault Message Type**. You will, however, only use the former (see Figure 4.8). Using one of the three methods discussed earlier, select the message type **MT_Material** as the input message, and then save your message interface.

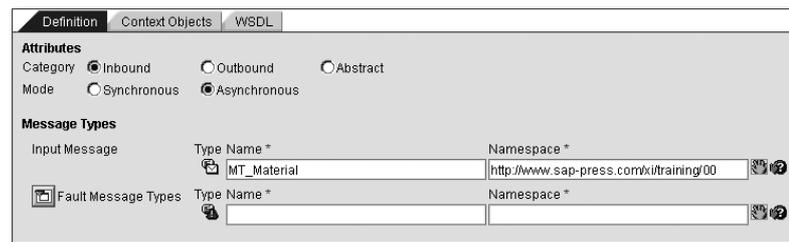


Figure 4.8 Editing a Message Interface

Creating the Mapping Objects

The connection between the elements of the sending and the receiving side is established via mapping. The contents conversion of the data formats in the form of message mapping is embedded in the interface mapping which connects a pair of inbound and outbound interfaces.

Creating the message mapping

At first, the independent message mapping should be created. In your namespace, open the context menu of the directory **Mapping-Objects • Message-Mappings**. In the creation dialog, enter the name **MM_Z_RFM_MATERIALINPUT_##_to_MT_Material**, where ## represents your participant number. Choose a description and create the object.

Selecting the outbound and the target message

The center area of the details window is divided into two parts, allowing you to select the sending message type on the left side and the receiving message type on the right side. First, start with the message type on the sending side: You have the option to either use the input help or to drag the appropriate message type to the label, **Enter a source message**. In this exercise, there is no explicit message type on the sender side, so use the RFC interface.

Regardless of the used selection method, you must choose which RFC message of the interface you would like to use. This is because a

synchronous communication is expected for an RFC interface. Therefore, you can choose between the two messages **Z_RFM_MATERIALINPUT_##** and **Z_RFM_MATERIALINPUT_##.Response**. Select the former because no response is expected.

The left part of the center area now lists the elements of the RFC interface. For the receiving part, select your **MT_Material** message type.

If you look at the elements on the right side, you'll find a red mark next to every entry. This indicates an incomplete mapping for the respective target element. Since we didn't change anything in the **Occurrences** column when creating a data type, the default value of **1..1** is applied. This means that this element is mandatory. If one of the target fields does not receive a value from the mapping, an error occurs. The connection between the elements of the two message types can again be established via three different methods:

Methods for
mapping elements

- ▶ The most obvious possibility is the connection via drag and drop, where it isn't important which side is dragged to the other. The two elements are displayed in the lower screen area and connected automatically.
- ▶ The second option is to double-click on the source and on the target element to move them to the lower screen area where they are displayed as rectangles. There you can connect the two rectangles by dragging the white area of the sending element to the corresponding area of the receiving element. In particular, this method should be used if the mapping is to be extended by predefined functions.
- ▶ The third method is suitable for connecting a large number of elements of the same name. For this, parent elements must be selected on both sides. In this mapping, these are **Z_RFM_MATERIALINPUT_##** on the sender side, and **MT_Material** on the receiver side. Then, above the sending message type, select the icon **Map selected Fields and Substructures if Names are Identical**. An alert dialog box appears, asking you to confirm the action. After dismissing the dialog, all of the elements on the sender side are connected to those on the receiver side. It's important to note that mapping of element names is case-sensitive.

Graphical function
for calculating the
net weight

Perform a mapping using the third method and have the result displayed in the overview by clicking on the **Dependencies** icon. The two message types then move apart and give way to the display of connection lines. You will also notice that the marks next to the receiver elements have now turned green. Only the **NTGEW** element is still red because it was not automatically provided with a value.

For demonstrating the integrated mapping functions, we assume that the net weight of the material is 90 percent of the gross weight. To map this, first select the **NTGEW** element on the receiver side and then the **BRGEW** element on the sender side by double-clicking on these items so both are displayed in the bottom area. To make this calculation, you first need a multiplication function that calculates the net weight from the gross weight with a constant of 0.9.

In the toolbar at the bottom of the screen, select the functions for **Constants** and click on the **Constant** function to the right, which is then displayed as a rectangle in the work area of the mapping. The cog wheel means that you can maintain parameters within this function. Double-click on the constant rectangle and change its value to **0.9** for the 90 percent of the net weight.

Now change to the **Arithmetic** area in the toolbar to insert the **mul** (short for *multiply*) function in the work area. Connect the **BRGEW** and **Constant 0.9** elements to the **mul** function by dragging the white subareas. In fact, these functions would be sufficient for calculating the correct net weight. However, the three decimal places permitted for the **xsd:decimal** type might be exceeded. If this message mapping were tested, it would result in an error.

Before the result of the calculation can be mapped to the **NTGEW** element, it must be formatted using the **FormatNum** function from the **Arithmetic** functional area. Configure the internal parameter **Number Format** of the function so the result matches the scheme **000.000**. Insert the **FormatNum** function between the **mul** function and the target element **NTGEW** (see Figure 4.9). All rectangles and the mark next to the **NTGEW** target element should now be displayed green. Save the message mapping.

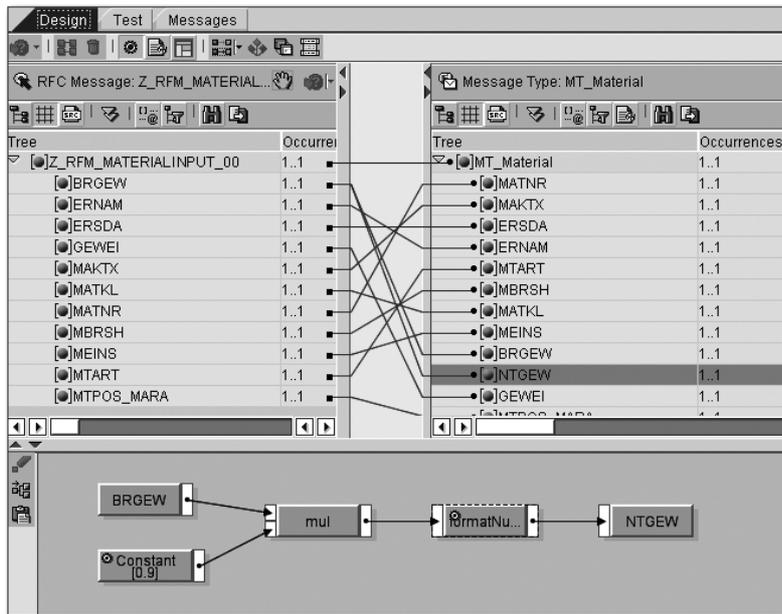


Figure 4.9 Message Mapping of the RFC-to-File Exercise

To ensure that the new mapping works, a test function is added to the Integration Repository, which you can select via the **Test** tab in the top area of the details window. On the left side of the test area, there is the structure of the sending message type whose elements are populated with test values. Be sure to use a decimal point as the decimal character for the **BRGEW** element.

Testing the mapping

The test itself is started with the **Start Transformation** icon (indicated by a vise) at the bottom-left of the test area. If the test program does not find any errors, the structure of the receiving message type with its respective values is displayed on the right. In particular, you should verify whether the **NTGEW** element has been populated correctly.

The last object of the integration scenario you are creating in the Integration Repository is the interface mapping. Start the creation dialog by opening the context menu of the directory **Mapping-Objects • Interface Mappings** in your namespace. Name the interface mapping **IM_Z_RFM_MATERIALINPUT_##_to_MI_Material_Async_In**, and then enter a description for the object and create it by clicking the **Create** button.

Creating the interface mapping

This object's detailed view is divided into the upper interface area and the lower mapping area. In the interface area, first select the sender interface; that is the RFC interface **Z_RFM_MATERIALINPUT_##**. Note that the RFC interface is not stored in your namespace; instead, you will find it in the directory **Imported Objects • RFC**. Perform the same steps for the target interface, **MI_Material_Async_In**.

By selecting the two interfaces, you have now specified which interfaces will communicate with each other as well as which message types are used. However, you still need to determine how the two data formats are converted to each other, since there might be different message mappings for the same message pair.

In the lower mapping area, click the **Read Interfaces** button to display the message types of the used interfaces (see Figure 4.10). After the **Source** and **Target Message** fields have been filled, click in the **Name** field located in the **Mapping Program** section area (located between the the source and target message fields) and select the input help that appears. A list is displayed, which contains all message mappings existing between the interfaces in this scenario of sender and receiver; you should only see mappings of the scheme **MM_Z_RFM_MATERIALINPUT_##_to_MT_Material**. Select the mapping with your participant number.

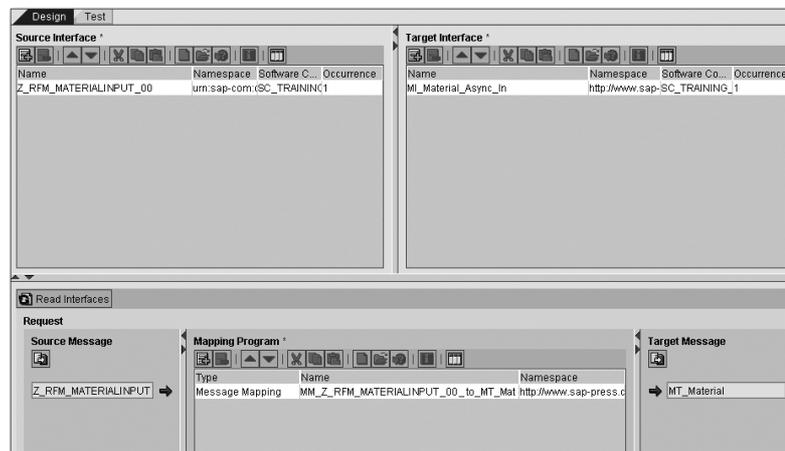


Abbildung 4.10 Interface Mapping of the RFC-to-File Exercise

If you have a closer look at the **Mapping Program** area (see Figure 4.10), you will notice that the tabular structure allows you to select

several mappings. All selected message mappings are processed sequentially according to their order in the table. When creating the message mapping, for example, you can use the **Test** tab to perform a test which, in addition to the message mapping, also checks interface compatibility. Save your interface mapping after it has been successfully tested.

As you can see, all newly created objects were usable throughout the entire Integration Repository although they were not activated. However, you will not be able to access all of these objects in the Integration Directory in this state, so the next step is to activate your change.

For this, go to the directory structure on the left and select the **Change Lists** tab. The tree structure is hidden and your software component version is displayed, which you should fully expand. Beneath that list, you will find a **Standard Change List** containing all newly created objects. Verify that all elements presented in Table 4.2 are included in the change list, and then select the **Activate** option from the change list's context menu. A window containing all objects of the list is displayed. You have the option to exclude specific objects from the activation. Activate the entire list and return to the **Objects** tab. Notice that the icons indicating that the new objects are not yet activated have disappeared.



Activating the new design objects

4.1.3 Configuration

Based on the objects created in the Integration Repository, you can now set up the communication between systems A and B in the Integration Directory. The Integration Directory can be called either via **Transaction SXMB_IFR** (via the direct link in the web browser), or by selecting **Environment • Integration Builder (Configuration)** in the Integration Repository menu.

As with the Integration Repository, the interface is divided into two parts; however, the objects are no longer arranged according to software component versions. Instead, they are arranged according to scenarios and object types. Above the directory structure you'll see three tabs: Change Lists, Objects, and Scenarios. The **Change Lists** tab serves the same function as in the Repository. The **Objects** tab lists all objects of the Directory by their type. Because the number of these objects will significantly increase over time you have the option to

First steps in the Integration Directory

arrange them in configuration scenarios on the **Scenarios** tab. Except for the scenario, you will create for all of your objects in the Integration Directory; this exercise uses all elements listed in Table 4.3.

Object Type	Sender Side: System A	Receiver Side: System B
Communication channel 1	RFC_SenderChannel_##	File_ReceiverChannel_##
Sender agreement	SystemA Z_RFM_MATERIALINPUT_##	
Receiver agreement		SystemA SystemB MI_Material_Async_In
Receiver determination	SystemA Z_RFM_MATERIALINPUT_##	
Interface determination	SystemA Z_RFM_MATERIALINPUT_## SystemB	

Table 4.3 Elements in the Integration Directory for the RFC-to-File Exercise

Setting Up the Business Systems and Their Communication Channels

Creating a Configuration Scenario

To create the **XI_Training_##** scenario, use the context menu of an existing scenario or click the **Create Object** icon to the bottom left of the menu bar. Save the object so the scenario is displayed in the listing on the left side, select the new scenario, and then restrict the view by clicking on the icon **Only Display Selected Subtree** above the list.

Change to the **Objects** tab and open the directory **Service Without Party • Business System**. Below the branch you will see at least the two business systems **SystemA** and **SystemB** which were declared in the System Landscape Directory when you prepared in Chapter 3 for the exercises. Click the **Add to Scenario...** option in the context menu of **SystemA** and select the scenario you just created.

Due to this mapping, this business system and its communication channels are displayed in your scenario as well. Repeat this step for the business system **SystemB** and return to the **Scenarios** tab.

The adapters available for the appropriate business systems are mapped in the Repository as communication channels (see Figure 4.11). Therefore, you need to configure a sending RFC adapter for System A and a receiving file adapter for System B.

Using the context menu from the path **Service Without Party • Business System • SystemA • Communication Channel**, open the creation dialog and enter the name **RFC_SenderChannel_##** along with an appropriate description. In the details window, use the input help to set the adapter type to **RFC**. Select the direction **Sender** for this adapter. For the fields **Transport Protocol**, **Message Protocol**, and **Adapter Engine** in the upper area, just use the default values.

Creating an RFC sender channel

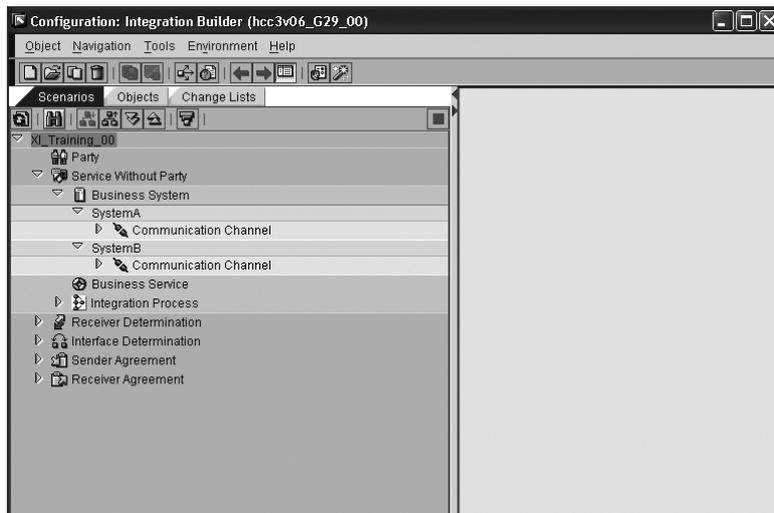


Figure 4.11 Integration Directory

The **RFC Server Parameter** area establishes a TCP/IP connection to the RFC destination on the side of System A. During the preparation of the exercises (in Section 3.5.2), you created an RFC connection named **SystemA_Sender-##**. This RFC connection is registered on the gateway server of the XI system and waits for a corresponding counterpart.

Connection to the existing RFC connection

In the **Application Server** field, enter the host name of the XI system, and in the **Application Server Service** field, enter the gateway service of the XI system according to the scheme **sapgwXX**, where **XX** represents the instance number. The **Program ID** follows the scheme **SystemA_Sender-##** and, like the two values mentioned earlier, exactly matches with the values entered in the corresponding RFC connection in System A. The **SNC** option specifies whether communication over RFC connection takes place via Secure Network Connection. The **Unicode** checkbox must be enabled if System A is a Unicode system.

Access to RFC metadata

The **RFC Metadata Repository Parameter** section is used to identify and log onto the system, and provides metadata about the used RFC interfaces. This integration is required because the metadata are checked by the XI system when calling the sending RFC adapter. In this example, the RFC interface is imported from System A during the design phase. Enter the host name and the instance number of System A, as well as your user **SYS_A-##**, your password, and the corresponding client, before saving the communication channel. If you enabled the communication channel at a later stage, the connection test for the destination **SystemA_Sender-##** is carried out successfully from System A.

Figure 4.12 provides an overview of all the settings of this communication channel.

Adapter Type *	RFC	http://sap.com/xi/XI/System	SAP BASIS 6.40
Adapter Status	Active		
Adapter Engine *	Integration Server		
Transport Protocol *	RFC		
Message Protocol *	RFC (RFC XML)		
Application Server (Gateway) *	<xihost>		
Application Server Service (Gateway) *	sapgw<XX>		
Program ID *	SystemA_Sender-00		
Initial Connections	1		
Maximum Connections	1		
Logon User *	SYS_A-00		
Logon Password *	*****	*****	
Logon Language *	EN		
Logon Client *	904		

Figure 4.12 Setting up the RFC Sender Channel for System A

The receiving communication channel for System B is created with the context menu from the path **Service Without Party • Business System • SystemB • Communication Channel**. The name of the new channel should be **File_ReceiverChannel_##**. In the details window, select the **File** adapter type using the input help and specify the direction **Receiver**. Set the parameter **Transport Protocol** to **File System (NFS)**, which means that the XI system can use its own local file system to access the directory in which the file is to be created.

Creating a file receiver channel



An alternative to the *Network File System (NFS)* is the *File Transfer Protocol (FTP)*, which allows access to file systems of remote computers. If you select FTP, you can specify the server and user data to log on to a remote FTP server. The **Message Protocol** field should be set to the **File** value which causes the written file to be stored in the XI format. The **File Content Conversion** characteristic, however, allows you to write the file as a list containing several entries.

The **File Access Parameters** determine the directory to which the file is written, and the scheme according to which its name is structured. After consulting your landscape administrator, it is recommended to use */tmp* for Unix installations or *C:\temp* for Windows.

Setting the source file

The **File Name Scheme** can be freely chosen. However, you should select the name *xi_output_##.dat* for this exercise, where **##** represents your participant number. During the course of this scenario, we will refer to this file so you need to take this into account in Section 4.1.4 if you select a different name.

The **Processing Parameters** specify how to create the file, that is if the name scheme specified earlier is used as-is or if, for example, a time stamp, a counter value, or the message ID is to be included in the file name. Select the **Add Time Stamp** option as well as the write mode **Directly** and the file type **Binary** (see Figure 4.13).

The write mode, **Directly**, causes data to be written out when no temporary file is used. The selected file type, **Binary**, makes it so not only text can be output.

In addition to the basic settings, you also have the option of specifying the path of the file storage dynamically using variable replacement or to trigger an operating system command before or after the message processing. Save the receiver channel.

Parameters		Identifiers	Module
Adapter Type *	File	http://sap.com/xi/XI/System	SAP BASIS 6.40
Sender	<input type="radio"/>	Receiver	<input checked="" type="radio"/>
Transport Protocol *	File System (NFS)		
Message Protocol *	File		
Adapter Engine *	Integration Server		
File Access Parameters			
Target Directory *	/tmp		
File Name Scheme *	xi_output_00.dat		
Processing Parameters			
File Construction Mode *	Add Time Stamp		
Write Mode	Directly		
File Type *	Binary		

Figure 4.13 Setting up the File Receiver Channel for System B

Creating the Connection Elements

Connection elements between the sender and the receiver side

Based on the basics we just created, and the objects in the Integration Repository, the integration scenario can be completed using some connection elements. The first two missing elements you need to create are the sender and the receiver agreement. This determines how a message is converted from or to the interface of a specific business system so the XI system or the receiving system can further process the message. In the case of the incoming RFC communication channel, for example, the message must be converted from the RFC adapter format to the XI XML format.

Creating a sender agreement

Let's start with the sender agreement, which you can create using the context menu of the **Sender Agreement** directory: In the creation dialog, select business system A as a service. The sending interface is the RFC interface **Z_RFM_MATERIALINPUT_##**, which you imported to the Integration Repository. In the details window of the new object, you can specify the communication channel of the sender by opening the input help and selecting the sender channel **RFC_SenderChannel_##**. Save the sender agreement.

Creating a receiver agreement

As you did for the sender agreement, create a receiver agreement for business system B and the receiving interface **MI_Material_Async_In**. Note that you also need to specify the sending business system A. In the details window, select the channel **File_ReceiverChannel_##** as the communication channel of the receiver and save the agreement.

For logically routing, messages in the XI system first need a receiver determination, which specifies available receiver services for a pair of business system and interface. Create a new receiver determination with the corresponding context menu for the sending business system A and the interface **Z_RFM_MATERIALINPUT_##**.

Creating a receiver determination

In the details window, the **Configured Receivers** area allows you to specify various possible receivers. If the review result of the relevant condition is true, the message is delivered to this system. This means that the message is delivered to several systems. For example, the condition can check elements of a message for a specific content.

If none of the configured systems is specified as the receiver, you can specify a default receiver below the receiver table. In the **Service** column of the existing row, select business system B as a potential receiver. Since the message in this exercise should be delivered to this receiver, you don't have to set a condition.

Save the receiver determination and then look at the lower area **Configuration Overview for Receiver Determination**, which now includes the **SystemB** entry. Expand the entry: As you can see in the entries, no matching interface determination and no appropriate interface mapping could be determined. Above this listing, click on the **New** icon to create a new interface determination.

Creating the interface determination

By calling the creation dialog from this context, all mandatory fields can be populated so you only need to enter a description. In the details window of the **Configured Inbound Interfaces** area, use the input help to select your message interface **MI_Material_Async_In** from the namespace. To the right of it, specify the only interface mapping available for the combination of the sending and receiving interfaces (see Figure 4.14). Save and close the interface determination and return to the receiver determination.

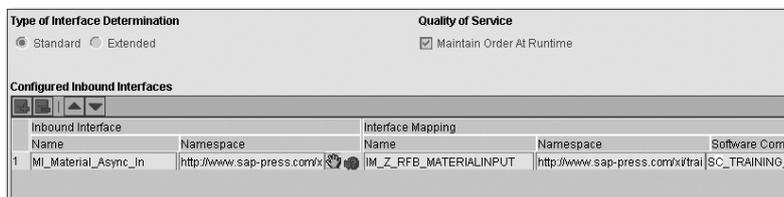


Figure 4.14 Editing the Interface Determination for the RFC-to-File Exercise

Activating the new configuration objects

In the lower area, click the **Refresh** icon so the receiver agreement for the receiving System B is also displayed with the target interface and the matching interface mapping (see Figure 4.15).

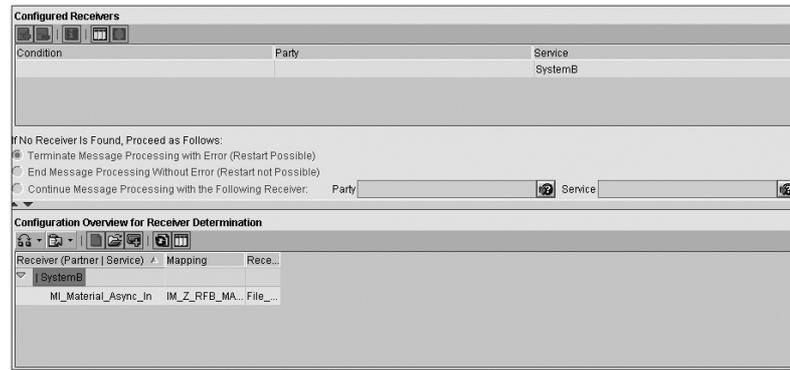


Figure 4.15 Editing the Receiver Determination for the RFC-to-File Exercise

Save the receiver determination and activate all newly created objects using the **Standard Change List** in the **Change Lists** tab. You have now created and activated all objects for this integration scenario.

4.1.4 Process and Monitoring

Now that you have created all design and configuration objects, you've completed the task of preparing the integration scenario for the course. Next, you will monitor the process and examine any possible errors.

Course of the Scenario

Calling the ABAP program in System A

The configured integration scenario is started by calling the program `Z_PROG_MATERIALINPUT_###`. Log in to the client of System A using your user `SYS_A-##` and call **Transaction SA38**; type the name of the program and execute it.

An input mask for basic material master data is displayed. You now want to enter the data for creating this XI developer manual as a material master record in System B. This is just a test; you won't use it any further to create a production or sales order, for example.

The used data correspond to the mandatory fields of the two views Basic Data 1 and 2 from the materials management in SAP R/3 or SAP

ECC, respectively. Because this data is used in the second exercise to actually create a material using an IDoc, we recommend you use the data from Table 4.4.

Field	Recommended Value
Material	XI_BOOK-##
Material description	arbitrary (for example, SAP XI developer book ##)
Material type	FERT (Finished product)
Industry sector	1 (Retail)
Material group	030 (Documentation)
Quantity unit	ST (Piece)
Gross weight	arbitrary (for example, 1.2)
Weight unit	KGM (kilogram)
General item category group	NORM (Normal item)

Table 4.4 Recommended Values for Creating a Test Material

This data works in an IDES R/3 or ECC system without further adaptation. For the second exercise (see Section 4.2), you can later use the appropriate template files.

Entering the recommended values

Enter the data in the individual fields and note that the input help displayed for some fields only returns values of the sending system that might not exist in the receiving system (see Figure 4.16).

The screenshot shows a SAP program window titled "Programm Z_PROG_MATERIALINPUT_00". The window contains a data entry form with the following fields and values:

Material	XI_BOOK-00
Material description	SAP XI Developerbook 00
Material type	FERT
Industry sector	1
Material group	030
Base Unit of Measure	ST
Gross weight	1,200
Weight unit	KGM
Gen. item cat. grp	NORM

Figure 4.16 Calling the Program Z_PROG_MATERIALINPUT_##

After you have clicked on the menu option **Program • Execute** or the corresponding **Execute** icon you will receive a success message. This message only notifies you that the function module belonging to the program has been called successfully. However, it does not confirm that the message has been successfully delivered.

Monitoring the process

Correct delivery and processing of the message can be verified in the XI system, for example. Log on to the appropriate client and call **Transaction SXMB_MONI**. Double-click on the path entry **Integration Engine • Monitoring • Monitor for Processed XML Messages**. This opens a selection mask that allows you to select all processed messages.

If the XI system is used only for training or testing purposes a restriction is hardly necessary. Otherwise, you could restrict the selection to messages with the sending server **SystemA**, for example.

Execute the message query via the **Program • Execute** menu option or the corresponding **Execute** icon. If your message has been successfully delivered and processed you should see an entry showing a black-and-white checkered flag in the status column (see Figure 4.17).

Status	Ack. Status	Executed From	Start Time	End Time	Sender Serv.	Sender Namespace	Sender Interface	Receiver Service
		25.03.2006	21.04.52	21.04.53	SystemA	urn:sap-com:document:sap:rfc:functions	Z_RFM_MATERIALINPUT_00	SystemB

Figure 4.17 Display of the First Message in Transaction SXMB_MONI

A green flag means that the message is currently being processed. Most other icons represent an error in our case. You can display the legend of all possible icons via the menu option **Goto • Legend** or the corresponding **Legend** icon.

Viewing the created file

To get an ultimate proof that the message was successfully processed, look at the created file. For example, this is possible by using **Transaction AL11** in the XI system, by clicking on the row of the directory alias **DIR_TEMP**. In the file list, search for a file matching the scheme *xi_output_##.dat*; it should include the time stamp of your message. Double-click on the file to open it. Since the display is limited to a specific width and the lines are not wrapped automatically, we recommend you use the file tools to import the transport request from the web page of this book to locally view the file.

Troubleshooting in Monitoring

To find the cause of an error in the message display of **Transaction SXMB_MONI**, double-click in any field of the corresponding row. This brings you to the **Display XML Message Versions** view. In the case of an asynchronous message, you will see the different statuses of the message on its way through the central integration engine. In the directory structure to the left, navigate to the place that has an error icon. In the windows on the right side, look for an error message indicating the cause. In most cases, the error was caused by a mapping or an object that was inadvertently selected from the input help.

Process analysis

In some cases, however, the error was caused by incorrectly configured communication channels or adapters. To check this, start **Transaction SXMB_IFR** which is used for calling the XI tools. On the bottom right, select the **Runtime Workbench** and logon using your user **XI-##**.

Checking the adapters

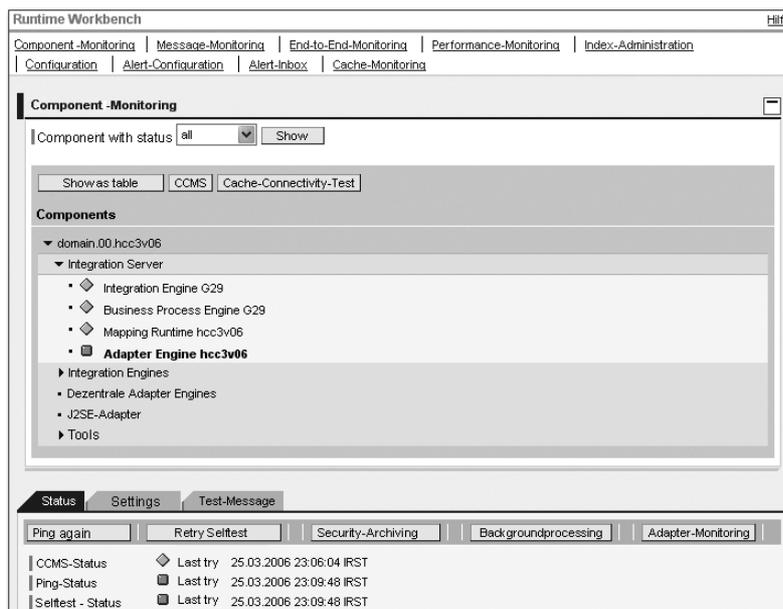


Figure 4.18 Entry Point to Component Monitoring of the Runtime Workbench

You will see the options for the Runtime Workbench, most of which you will become familiar with while working on the following exercises and the case study. Even though you already know a different way to display a message overview using **Transaction SXMB_MONI**, you can use the **Message Monitoring** menu option to view the mes-

sage status in the XI system. First select **Component Monitoring**, and display the components with every possible status. In the directory structure of the components, select the path **domain.XX.<XIhost-name> • Integration Server • Adapter Engine**. A status view opens beneath the directory structure providing you with information about the general status of the adapter engine. On the top-right, click the **Adapter-Monitoring** button (see Figure 4.18).

Selecting the adapter type

After expanding the namespace `http://sap.com/xi/XI/System`, a new browser window opens and displays the selection of all available adapters. A gray diamond next to an adapter type indicates that a communication channel for this type hasn't been created. A green square indicates that all communication channels of this type have been correctly configured and that no error has occurred during processing. A red circle, however, indicates that at least one communication channel of this type is faulty.

You'll need to see if an error occurred for the adapter types **RFC** or **File**. For a closer analysis, you can click on the relevant type to list all communication channels. If the communication channel displays an error, you are presented with a detailed error description to the right which you can use to correct the error.

4.2 Exercise 2: File-to-IDoc

Course of the second exercise

The file containing the material master data that you created in the first exercise now needs to be integrated in the business system B to become a material. Although the file has already been transferred to System B from a logical point of view, it technically still resides on the file system of the XI system, in the `/tmp` or `C:\temp` directory, respectively. This allows us to keep the arrangement that System A is the sender and System B the receiver because System A can also access the file system of the XI system to read the file. System A reads the file using the file adapter and transfers it to the XI system from where the record will be sent as an IDoc to System B.

A diagram of the used adapters and their use are shown in Figure 4.19.

Index

A

ABAP class 170
ABAP proxy 92, 161
ABAP proxy generation 70
Abstract 225
Adapter 37, 56
Adapter engine 75, 135
Adapter monitoring 144, 155, 286
Adapter type 55, 135, 144, 157
Add-on 87
Address 55
Administrator guide 53
ALE 84
ALE communication 145, 271
ALE messages 160
Alternative functions 184
Apache 294
Append 241
APPINT 87
Application log 161
Application server 83, 135
Application system 86
Archive
 Imported 279
Archive file 279
Archiving 153
Argument 167
ARIS 303, 305
ARIS for SAP NetWeaver 307
Arithmetic 130
Asnyc-sync-bridge 255
Asynchronous 127
Authorization object 106
Authorization role 100

B

B2B integration 42
BAM listener 309
BAPI 31
Base64 282
Basic data 158
Basic type 254
Batches 212

Beer distribution game 291, 295
Best practices 307
Best-of-breed approach 20
Billing document 276
Bill-to party 276
Boolean 227
BPEL 35, 177
BPEL process 292, 293, 296
BPM editor 183
Business activity monitoring 308
Business intelligence 44
Business landscape 85
Business process 253
Business process engine 176
Business process management 176
Business Process Modelling 37
Business process platform 302
Business service 67, 75, 243, 281
Business system 75, 85, 134
Business workflow 177

C

Callback 27
Case study 199
Category 65
Change lists 113, 133, 140
Checklist 113
Class Builder 172
Classes 168
Classification 55, 202
Client proxy definition 272
Clients 83
CollectiveNumber 207, 210, 218
Communication
 Internal 245
Communication category 56
Communication channel 62, 137, 152, 157
Communication mode 57
Communication type 155
Component monitoring 62, 144, 155, 159
Components 60
Composite Application Framework 46

Index

Composite applications 297, 298
concat 260
Condition 246, 283
Condition editor 187
Condition overview 248
Configuration 59
Configuration objects 151, 214
Configuration scenario 67, 189
Configuration time 67
Configuration wizard 151, 155, 172,
190, 216, 245
Connection method 129
Connection parameters 173
Constants 130
Container 183
Container elements 184, 240, 265
 Multiline 184
Container objects
 Multiline 235
Container operation 188
Context 225, 231
Context change 210, 228, 231
Conversion 271
CORBA 33
Core services 306
Corporate group 200
Correlation 185
Correlation editor 185
Correlation list 185
Cost quotation management 304

D

Data collection 91
Data converter 54
Data dictionary 223
Data type 65, 125, 205
Date format 262
DateTrans 263
Declaration 242, 267
Definition
 External 164
 Graphical 186
Definitions wizard 189
Dependencies 130
Deployable proxy 272
Description conflicts 26
Design 59

Design objects 122, 147, 163
Design phase 178
Details menu 125
Details window 125
Developer guide 53
Development consultant 117
Display
 Graphical 194
Distribution channel 208
Distributor 291
Drag&Drop 127, 183, 187
Drivers 195, 196
Duplicate subtree 209

E

End-to-end monitoring 62
End-user integration 42
Enhancement concepts 289
Enqueue 83
Enterprise Application Integration 20,
42
Enterprise Central Component 81
Enterprise Service Bus 29
Enterprise services 300
Enterprise Services Repository 303, 305
Enterprise SOA 297, 300
equalsS 228
EXECUTE_SYNCHRONOUS 172
Extensions 289
External definition 226

F

Factory 291
Fault message type 65, 128
FI area 199
Field
 Disabling 150
Field data 159
File access parameters 137
File adapter 119
File name scheme 137
File sender channel 215
File Transfer Protocol 137
FormatNum 130
Function
 User-defined 167

Function module 119, 222
RFC-enabled 294
 Further data 252, 286

G

Gateway server 88, 135
 Gateway service 89
 Goods availability check 296
 Goods issue 285
 Guided procedures 50, 310

H

Header data 208
 Heterogeneity 19
 Hierarchy level 212
 Hitech AG 200
 HTTP post 197
 Hub-and-spoke 29

I

ICF 96
 IDES 199
 IDES system 88
 IDoc 145, 157, 201, 214, 242
 IDoc adapter 102
 IDoc metadata 104
 IDoc structure 149
 IDoc types 150
 IfWithoutElse 227
 IMAP 280
 IMAP4 276
 Implementation Guide 53
 Import 111, 123, 205
 Inbound 127, 165
 Inbound options 146
 Inbound parameters 146, 202, 255
 Incoming invoice 275
 InfoCube 44
 Input help 126, 128, 132
 Input message 128, 166
 Input message type 65
 Inquiry 205, 219
 Inquiry item 205
 Inside-out 69
 Instance 227

Integration
 Data-oriented 30
 Process-oriented 31
 Integration direction 22
 Horizontal integration 23
 Vertical integration 22
 Integration directory 62, 133
 Integration engine
 Local 92
 Integration objects 22
 Integration process 76, 177, 183, 223
 Checking 189
 Integration repository 61, 121
 Integration scope 23
 INTEGRATION_DIRECTORY_HMI 97
 Interface
 Abstract 178, 184
 Direction-related 178, 192
 Read 132
 Interface definition 154
 Interface determination 58, 69, 139, 157
 Interface mapping 66, 131, 150, 169,
 191
 Interface variable 185
 Internet Communication Framework 96
 IT department
 New role 299
 IT practices 47
 IT scenarios 47
 Item data 210

J

J2EE HTTP port 80
 JAR 279
 Java 201
 Java application 290
 Java Connector 71
 Java Connector Architecture 34
 Java Database Connectivity 195
 Java mapping 168
 Java program 272
 Java proxy 256
 Java standalone proxy 254
 Java Web Start 80, 121
 JDBC adapter 195, 291
 JMS adapter 196, 291
 JMS payload 196

Index

L

Label
 Fictitious 200
LCRSAPRFC 88
Legacy applications 47
Legacy system 201
Load test instance 210
Location transparency 26
Lock mechanism 246
Logging service 293
Logical system 83, 103, 202, 271
Loop 187

M

Mail adapter 286
Mail package 282
Mail provider 280
Mail server 276
Mandatory field 129
Mapper 36
Mapping 73
Mapping function 130
Mapping programs 74
Mapping templates 212
Mass generation 100
Material description 203
Material master data 119, 144
Material price 204
MATMAS 146
Maximum occurrence 207
Message
 Nested 205
 Synchronous 256
Message control 221
Message distribution 54
Message ID 137
Message interface 64, 127, 147, 165
Message mapping 66, 128, 148, 166,
 181
Message monitoring 62, 159, 253, 286
Message path 59
Message protocol 135
Message queue 95
Message server 83
Message type 65, 126, 207, 221, 276
Metadata 136, 148
Milestone monitoring 309

Mode 65
Modeling
 Graphical 186
Monitoring 37, 75, 142, 218, 253

N

Name collision 171
Namespace 63, 112, 121, 204
Nested structure 180
Network File System 137
Next practices 307
Node functions 210, 229, 230

O

Objects 133
 Imported 123
Occurrence 129, 180, 181, 206, 238,
 239
Operation 188, 241
Operational data store 44
Order number 284
Outbound 165
Outbound delivery 284
Outbound message 191
Outbound status 193
Output 251
Output message 165
Output message type 65
Outside-in 69
Outside-in model 301

P

Package 170
Partner agreement 145, 255, 276
Partner number 202
Partner profile 202, 221
Partner status 202
Partner type 146
Persistence layer 175
Picking 284
Plain HTTP adapter 197
Point-to-point integration 28
Poll interval 215
Port 104
Power User Guide 53
Prerequisites 117

Process 220
 Process code 221
 Process execution language 177
 Process selection 193
 Process structure 225
 Processing log 183
 Processing mode 215
 Processing parameters 137
 Profile 100, 107
 Program ID 88
 Protocols 54
 Proxy
 Activating 172
 Proxy class 163, 172

Q

Qualifier 208, 248
 Queue 210, 228
 Display 213
 Quotation 251
 Creation 251
 Quotation list 224

R

Receive 188
 Receiver agreement 68, 138, 157, 271
 Receiver channel 153, 157, 173, 214
 Receiver determination 58, 68, 139,
 157, 246
 Receiver parameters 157
 Recipient determination 27
 removeContexts 229
 Repository 37
 REQOTE 203
 Request 170, 224, 232
 Requested date 206
 Response 170, 233
 Retail 204
 Retailer 291
 Return code 175, 256
 RFC 31, 135
 transactional 105
 RFC adapter 71, 119, 176
 RFC call 177, 189
 Transactional 160
 RFC connection 135
 RFC function module 179, 292

RFC interface 124, 128
 RFC message 128
 RFC server parameter 135
 RFC-KOMM 102
 RMI 33
 Router 36
 Routing
 Logical 139
 Runtime 60
 Runtime Environment 37
 Runtime Workbench 62, 143, 155

S

S/A bridge 266
 Sales order 274
 Sales organization 208
 Sales process 276
 SAP Business Connector 196
 SAP Easy Access Menu 219
 SAP Education 199
 SAP NetWeaver 302
 SAP NetWeaver Application Server ABAP
 81
 SAP NetWeaver Application Server Java
 81
 SAP NetWeaver Developer Studio 272
 SAP NetWeaver Exchange Infrastruc-
 ture 41, 174, 297, 301
 SAP NetWeaver Visual Composer 303
 SAP Service Marketplace 277
 SAP Solution Manager
 Lifecycle Management 303
 SAP Web Service Repository 162, 164
 SAPSLDAPAPI 90
 SARIDIS 199
 Scenarios 134
 SD area 199
 SDK 80
 Search criteria 155
 Secure Network Connection 135
 Security 290
 Security Guide 53
 Security protocol 273
 Segments 149, 208, 227
 Selection method 126
 Sender 135
 Sender agreement 68, 138
 Sender channel 135, 152, 157, 242

Index

Sender parameters 155
Server program
 Registered 88, 101
Service 173
Service enabling 301
Service level agreement 310
Service orchestration 301
Shared collaboration knowledge 54
SID 81
SLD 81
SLD bridge 91
SLDAPICUST 90
SLDCHECK 91
SMTP 276, 280
SOAP 34, 35, 173, 243
SOAP communication channel 173
SOAP receiver channel 243
Software catalog 107
Software component 63, 84, 121, 204
Software component version 111
Software Deployment Archive 195
Software Deployment Manager 195
Software product 60, 84, 107
Sold-to party 206
SonicMQ 195
Source text view 210
Special cases 290
SplitByValue 210, 264
SPROXY 100, 170
SQL query 195
SQL update 195
Standalone proxy project 272
Standard change list 133, 140
Standard order 284
Start process 240
Status record 160
Step type 186
Stock volumes 293
Stored procedures 195
Substructure 226
Sunny Electronics 200
Supply chain 291, 292
Supply chain levels 293
Support package 79
SXI_MONITOR 74, 219
SXMB_MONI 142, 218
SXMB_MONI_BPE 193, 253, 274
Sync-async-bridge 254, 259, 266
Synchronous 165, 233, 259

System
 Logical 145
 Technical 81
System ID 81
System landscape 42, 60
System Landscape Directory 60

T

Table 223
Target structure 213
Technical configuration 59
Technical help 159
Technical landscape 81, 109
Test 131, 133, 150, 210
Test mode 218
Test program 131
Time format 262
Time stamp 137
Tomcat 294
Trace file 194
Tracking the message path 253
Transfer order 284
Transformation 58, 131, 188, 213
Translation 56, 57
Transport protocol 135
Transports 119
Troubleshooting guide 113

U

Unbounded 207, 238
University 199
User interface 298

V

VA21 253
Validity area 184
Variable 184
Variable replacement 137
Views 158

W

Web service 30, 161, 221, 222, 298, 300
 Definition 268
WebSphereMQ 195
Where-used list 125

Whiplash effect 292
Wholesaler 291
Work area 187
Work item 194, 254
Workflow 27
Workgroup 117
Write mode 137
WS-BPEL 310
WSDL 34, 35, 162
WSDL interface 163

X

xApps 298
xCQM 304
XI format 55
XI service 96

XI tools 107
XIALL 281
XIISUSER 98
XIPAYLOAD 281
XIRWBUSER 99
XML document 232
XML file 232
XML message 143
XSD document 56
XSD file 278
XSLT file 279
XSLT mapping 74, 277

Z

ZIP 279
Zip code area 55