Sabine Maisel

Practical Guide to IDoc Development for SAP^{*}





Contents at a Glance

1	Introduction	15
2	Generating IDocs	29
3	Test Tools	51
4	Changes to IDocs	67
5	Confirmations	187
6	Serializing IDocs	193
7	Administration	211
8	IDocs in Conjunction with SAP NetWeaver Process Integration	243

Contents

Pre	face		11
1	Intro	oduction	15
	1.1 1.2 1.3 1.4	What Is ALE?BAPIs and IDocs — An Introduction1.2.1BAPIs1.2.2IDocsDifferentiation of ALE and EDISummary	15 18 19 25 27
2	Gen	erating IDocs	29
	2.1 2.2 2.3	Standard Methods for the IDoc Generation2.1.1Shared Master Data Tool2.1.2Message Control2.1.3Special FunctionsUse of Logical Systems in the Message ControlSummary	29 29 43 46 49 50
3	Test	Tools	51
	3.13.23.33.4	Individual IDocsTesting Processing of Multiple IDocs3.2.1Message Control3.2.2Sending Ready-for-Dispatch IDocs3.2.3FilesProcessing Status FilesSummary	52 57 58 59 61 66
4	Cha	nges to IDocs	67
	4.1	Customizing 4.1.1 Filtering Using Filter Objects	67 67

		4.1.2	Custom Filter Objects	73
		4.1.3	Filtering Segments	76
		4.1.4	Reducing IDocs Through Views	77
		4.1.5	Rules	80
		4.1.6	Version Conversion	84
		4.1.7	Special Conversions in SAP ERP Financials	85
	4.2	Adaptir	ng Existing IDoc Types	89
		4.2.1	Different Exit Types on the Basis of the	
			Material Master Example	90
		4.2.2	General Exits	119
		4.2.3	Custom Segments	122
		4.2.4	Special Requirements for Master Data	134
	4.3	Custom	I Function Modules for File Generation	
		in File I	Ports	138
	4.4	Custom	IDocs	141
		4.4.1	Creating Custom IDoc Types and	
			Message Types	143
		4.4.2	Generating an IDoc	153
		4.4.3	Updating an IDoc	161
		4.4.4	Generating IDoc Function Modules	172
		4.4.5	Error Workflow for Custom IDocs	175
		4.4.6	Useful Function Modules	184
	4.5	Summa	ıry	186
5	Conf	firmatic	ons	187
	5.1	"ALEAU	JD" IDocs	187
	5.2	"STATU	S" IDocs	190
	5.3	Summa	.ry	192
6	Seria	alizing	IDocs	193
		0		
	6.1	Serializa	ation Using Groups	193
	6.2	Serializa	ation Using Timestamps	198
	6.3	Serializa	ation Using Business Objects	200
	6.4	Serializa	ation Using qRFC	205
	6.5	Summa		209

7	Adm	inistration	211
	7.1	IDoc Links	211
	7.2	Regular Jobs	216
	7.3	Transaction Code Overview	217
	7.4	Archiving	224
	7.5	Status Conversion	230
	7.6	Summary	242
8	IDoo	s in Conjunction with SAP NetWeaver	
0			
	Proc	ess Integration	243
	Proc	ess Integration	243
	Proc 8.1	ess Integration Conversion of Logical Systems to Business Systems	243 243
	Proc 8.1 8.2	ess Integration Conversion of Logical Systems to Business Systems Conversion of IDoc Partner Roles	243 243 244
	Proc 8.1 8.2 8.3	ess Integration Conversion of Logical Systems to Business Systems Conversion of IDoc Partner Roles Header Mapping	243 243 244 245
	Proc 8.1 8.2 8.3 8.4	ess Integration Conversion of Logical Systems to Business Systems Conversion of IDoc Partner Roles Header Mapping Handling the Control Record in SAP NetWeaver PI	243 243 244 245 246
	8.1 8.2 8.3 8.4 8.5	ess Integration Conversion of Logical Systems to Business Systems Conversion of IDoc Partner Roles Header Mapping Handling the Control Record in SAP NetWeaver PI Updating IDocs Directly in SAP NetWeaver PI	243 244 245 246 248
	Proc 8.1 8.2 8.3 8.4 8.5 8.6	ess Integration Conversion of Logical Systems to Business Systems Conversion of IDoc Partner Roles Header Mapping Handling the Control Record in SAP NetWeaver PI Updating IDocs Directly in SAP NetWeaver PI Summary	243 244 245 246 248 248
	8.1 8.2 8.3 8.4 8.5 8.6	ess Integration Conversion of Logical Systems to Business Systems Conversion of IDoc Partner Roles Header Mapping Handling the Control Record in SAP NetWeaver PI Updating IDocs Directly in SAP NetWeaver PI Summary	243 244 245 246 248 248
Th	8.1 8.2 8.3 8.4 8.5 8.6 e Auth	ess Integration Conversion of Logical Systems to Business Systems Conversion of IDoc Partner Roles Header Mapping Handling the Control Record in SAP NetWeaver PI Updating IDocs Directly in SAP NetWeaver PI Summary	243 244 245 246 248 248 248

In the sending system, it's necessary to generate IDocs so that they can be sent to the receiving system. How this is done depends on the type of data and the application. This chapter describes the different generation options and their use.

2 Generating IDocs

SAP has provided tools for the generation of IDocs in all locations where they are used in ALE scenarios or classic EDI, and they can usually be activated using the Customizing settings. However, there are different methods of IDoc generation depending on the type of data and the location where the IDoc will be generated. This chapter presents the most common methods of IDoc generation. Usually the IDoc administrator not the developer — implements the settings required for creating the partner profiles, for instance, so they are outlined only briefly, and more importance is attached to the functional process flow.

2.1 Standard Methods for the IDoc Generation

Initially, you must distinguish between the generation of master data and the generation of transaction data because there are different requirements on the generation process or the generation frequency depending on the type of data. A special tool is available to generate master data, called the *Shared Master Data Tool* (SMD). Transaction data IDocs are generated via the already-existing message control. There are also some special functions for IDocs that are directly generated in a process.

2.1.1 Shared Master Data Tool

The SMD is a special tool for sending master data via IDoc. Master data is characterized by a relatively long retention period in the system during in the IDoc which the data is changed rarely. Master data usually consists of multiple views that are not used all the time. You can omit views, including those that contain mandatory fields, because the check of whether all mandatory fields are populated is only carried out if the view is actually used. This enables you to select from the wealth of information that is offered for a specific object and use exactly the data that is actually required within your enterprise.

To distribute data using IDocs, an automated process is desirable that responds to the creation and modification of master data without requiring further user interventions. Also, empty views are not supposed to be transferred.

Automation and The SMD takes these requirements into account. The technical implecontrol via views mentation of automation and control via views entails that already-existing procedures can be used for both functions. For automation, you revert to recording changes, which is implemented by default; for control via views, you use the option (which originated from batch input processing) to control irrelevant fields using a NO_DATA character. Then, the IDocs are regularly generated via background jobs.

Additionally, for almost all objects, you also have the option to explicitly generate IDocs or to request IDocs. You can use this option if waiting for periodically scheduled jobs is impossible.

Recording Changes

- Change pointer For consistency reasons, changes to the master data are updated independently of the use of ALE in SAP systems. For each individual data element of the tables concerned, SAP has stipulated whether a change should be logged or not. Figure 2.1 shows the CHANGE DOCUMENT FLAG, which is activated in this case as an example of BISMT (*Old Material Number*) from the MARA TABLE.
- Writing the history in the application For updating the changes, you always call the CHANGEDOCUMENT_OPEN function module, which prepares the writing of the change history. Then, all changes to be written are collected, and the process is concluded using the CHANGEDOCUMENT_CLOSE function module. Because IDocs are

supposed to be generated wherever changes are updated by default, the CHANGEDOCUMENT_CLOSE function module has an ALE share in addition to its standard function. This enables you to generate *change pointers* for ALE for the desired message types. In all Unicode-enabled releases, this is done using the CHANGE_POINTERS_CREATE_LONG function module; in old releases, this is done using the CHANGE_POINTERS_CREATE function module.

Dictionary: Display Data Element						
◓◓।ፇਃਞ।₽₭ݝਫ਼੶₽ਫ਼ੵਸ਼ਫ਼ੵੑਸ਼ੑੑੑੑੑੑੑੵੑੑੑ						
Data element	BISMT	Active				
Short Description	Old material number					
Attributes Dat	a Type / Further Characteristics	Field Label				
Search Help						
Name						
Parameters						
Parameter ID						
Default Component N	ame OLD_MAT_NO					
Change document						

Figure 2.1 Characteristics of Data Elements

You also have the CHANGE_POINTERS_CREATE_DIRECT function module as a second option to generate change pointers. This function module is called by applications that are not connected to the previously described change management for documents.

Generating change pointers directly

In both cases, change pointers are only written if you use the SMD for performance reasons. In the ALE Customizing, you can specify whether this is the case and for which master data you require change pointers. There is a separate Transaction code SALE for the ALE Customizing that takes you directly to the correct position in the menu tree. Figure 2.2 shows the menu path in Customizing in which you make the necessary settings.

Structure	
▽ 🛃	IDoc Interface / Application Link Enabling (ALE)
D	Basic Settings
D	Communication
	Modelling and Implementing Business Processes
Þ	🛃 Global Organizational Units
	🔜 🕀 Maintain Distribution Model and Distribute Views
Þ	🗟 Configure Predefined ALE Business Processes
	😹 Master Data Distribution
	🗢 📴 🛛 Replication of Modified Data
	🛃 🕀 Activate Change Pointers - Generally
	🛃 🚯 Activate Change Pointers for Message Types

Figure 2.2 Activating Change Pointers for the SMD

Activating change
pointersInitially, you activate the generation of change pointers generally. As
a result, the share of the CHANGEDOCUMENT_CLOSE function module that
has not been used up to now and that is responsible for the SMD is run
through. This must be set once only for all master data. Figure 2.3 shows
the appropriate functionality.

Activate Change Pointers Generally
Activation status
Change pointers activated - generally

Figure 2.3 Activate Change Pointers Generally

Activating change pointers for object

If the change pointers are activated generally, you can specify in a second step for which message type you require the generation of change pointers. This is done in the second menu subitem (Activate Change Pointers for Message Types) and has been implemented for the message types, MATMAS and MATMAS_WMS (see Figure 2.4). No change pointers are written for MATCOR and MATMAS_GDS, which have not been activated.

In this context, it's important that a change pointer isn't written for each changed field because there are fields whose values are not significant for the downstream system. By means of Transaction BD52, SAP provides fields for each message type connected to the SMD, which are relevant for changes from SAP's point of view. For the material master, the DMAKT-SPRAS field is important, for example. In the transaction, you enter the fields that are used within your enterprise. If you've made changes to the corresponding master data tables via the SAP enhancement concept and use customer-specific fields, you can also set the customer-specific fields here. Refer to Chapter 4, Changes to IDocs, to learn how to provide these fields.

0	Change View "Activate Change pointers for Message Type":					
🎾 New Entries 🗈 🗔 🗊 🗐						
	Activate Change pointers for Message Type					
	Messg.Type	active	0			
	MATMAS					
	MATMAS_GDS		T			
	MATMAS_WMS					
	MATQM					

Figure 2.4 Activate Change Pointers for Message Type

Change View "Change document items for message type": Overview				
🦅 New Entri	es 🗈 🖬 🖍 🖪 🖪]		
Message Type	MATMAS			
Change doo	cument items for message typ	e		
Object	Table Name	Field Name		
MATERIAL	DGESV	KEY		
MATERIAL	DGESV	KOVBW	-	
MATERIAL	DGESV	VBWRT		
MATERIAL	DMAKT	KEY		
MATERIAL	DMAKT	MAKTX		
MATERIAL	DMAKT	SPRAS		
MATERIAL	DMARM	BREIT		
MATERIAL	DMARM	BRGEW		
MATERIAL	DMARM	EAN11		
MATERIAL	DMARM	GEWEI		
MATERIAL	DMARM	GTIN_VARIANT		
MATERIAL	DMARM	HOEHE		
MATERIAL	DMARM	KEY		

Figure 2.5 Change-Relevant Fields in Transaction BD52

Figure 2.5 shows a section of the fields for the MATMAS message type, which are provided by SAP as change-relevant. The message type is directly referenced to fields and tables of the material master except for the KEY field. This field isn't part of the respective table but assumes a very important, additional control role: It ensures that the creation of a

Change-relevant fields table entry can be sent via IDoc. If the KEY field is specified in Transaction BD52, a change pointer is written during the creation of the corresponding object, for example, during the initial creation of the material for the MARA-KEY dummy field or during the creation of a text in a new language for the MAKT-KEY dummy field. Imagine that the key value of the table concerned is changed from "empty" to the new value. As a result, all fields of this table are transferred.

Assigning change-
relevant fieldsAdditionally, for each of the change-relevant fields, you must specify to
which field in which segment of the IDoc type it belongs. This is done
in Transaction BD66, which is shown in Figure 2.6. The DMAKT-SPRAS
sample field from Transaction BD52 belongs to the E1MAKTM IDoc seg-
ment and to the field that is also called "SPRAS." For your own fields,
you must specify this using the New Entries button.

Change View "Segment Field - Change Document Field": Overview						
💖 New Entries 🗎						
Message Type MA	TMAS					
Segment Field - Chang	je Document Field					
Segment type	Field Name	Object	Table Name	Field Name		
E1MAKTM		MATERIAL	DMAKT	KEY		
E1MAKTM	MAKTX	MATERIAL	DMAKT	MAKTX	-	
E1MAKTM	SPRAS	MATERIAL	DMAKT	SPRAS		
E1MARAM		MATERIAL	MARA	KEY		
E1MARAM	AENAM	MATERIAL	MARA	AENAM		
E1MARAM	AESZN	MATERIAL	MARA	AESZN		
E1MARAM	BEGRU	MATERIAL	MARA	BEGRU		
E1MARAM	BEHVO	MATERIAL	MARA	BEHVO		
E1MARAM	BISMT	MATERIAL	MARA	BISMT		
E1MARAM	BLANZ	MATERIAL	MARA	BLANZ		
E1MARAM	BLATT	MATERIAL	MARA	BLATT		
E1MARAM	BMATN	MATERIAL	MARA	BMATN		

Figure 2.6 Assignment of IDoc Fields to Change-Relevant Fields

Evaluating change
pointersThe change pointers are then evaluated. It just depends on the object
concerned which function module is used here. When you call Transac-
tion BD60, you can view these function modules and replace them with
your own function modules if you want to make so many changes to the

standard functionality that you don't want to enhance or modify the SAP original. Figure 2.7 shows another example of the material master data. There, the function module that uses change pointers to generate IDocs is called MASTERIDOC_CREATE_SMD_MATMAS.

C	Change View "Additional Data for Message Type": Overview						
60	🦅 🕄 New Entries 🗈 🖶 🐼 🖶 🖪						
	Additional Data for Message Type						
	Messg.Type	Ref.msg.	Funct.Mod.	Table	111		
	MATMAS	MATMAS	MASTERIDOC_CREATE_SMD_MATMAS	MARA			
	MATMAS_GDS	MATMAS_GDS		MARA	Ŧ		
	MATMAS_WMS	MATMAS	MASTERIDOC_CREATE_SMD_MATMAS	MARA			

Figure 2.7 Function Modules to Evaluate Change Pointers

The RBDMIDOC report, which must be scheduled at regular intervals, must then use these function modules to generate IDocs from the change pointers and update which change pointers have been processed. As a transfer value, you can specify for which message type you want to implement the evaluation. For this purpose, you specify the appropriate type (here: MATMAS) in the Message Type field in the initial screen of the RBDMIDOC report. This is illustrated in Figure 2.8.

Creating IDoc Type	from Change Pointers	0
•••••••••••••••••••••••••••••••••••••••		
Message type	MATMAS	ð

Figure 2.8 Initial Screen of the "RBDMIDOC" Report

For sending IDocs using the SMD and for change pointers, the application and the ALE communication layer interact closely. Figure 2.9 schematically shows the process flow of IDoc generation. The entire process of writing change pointers takes place in the application; the evaluation of the change pointer and the generation of IDocs are carried out by the ALE communication layer.

"RBDMIDOC" background job



Figure 2.9 IDoc Generation Using the SMD

Note that when you send IDocs via change pointers, you only send those views in which changes have actually been made. If a new view has been created, all its fields are transferred; if a view has been changed, only the changed fields are transferred to increase performance. Chapter 4, Changes to IDocs, uses the example of the material master to describe how you can change this default behavior using a minor modification.

You can find the change pointers in the BDCP table and the corresponding status records in the BDCPS table. As of SAP NetWeaver Application Server 6.20 (SAP NetWeaver AS), processing with higher performance is possible via a shared table called BDCP2. However, this new procedure isn't supported for all message types. Whether it's applicable for your message type is indicated in the detail view of Transaction BD60. As of Release 7.1, only the new processing using Table BDCP2 is available for all message types.

Distribution lock For the distribution via the SMD, you must consider some specifics for some master data. You can set a distribution lock for the material master to generally prevent the sending of a material. For this purpose, you must make a short detour. In the design data of the material in the MARA table, there is a cross-plant material status (MARA-MSTAE field). This status refers to an existing entry in the T141 table. Here, you can assign additional properties for each status value. If the DLOCK field provided here is selected, the distribution lock is set.

For contracts (BLAORD message type), only released contracts are transferred using the SMD.

Reducing Messages

The second requirement on the distribution of master data arises from the distribution of all master data to individual views and the option to define, in Customizing, which fields of a view are actually supposed to be used. This is scalable for the IDoc transfer by using the reduced message types.

A reduced message type always relates to an existing message type but transfers less data. The reduction isn't possible for all message types; so the developer of the message type must explicitly define it as reducible. All views and fields that must be transferred as a minimum are predefined here; all other views and fields can be selected additionally if required. Transaction BD60 in the detail view for a message type is the transaction for defining a message type as reducible. By calling Transaction BD65, you define the mandatory fields.

For each delivered message type, SAP specifies whether it's reducible. Because this involves functions in the generation and update module for the corresponding IDoc, customers can't simply set the message type to reducible retroactively. The mandatory fields that SAP provides with Transaction BD65 correspond to the Customizing that SAP provides for transactions that are used to maintain master data, for instance, Transaction MAT1 for maintaining the material master. If you make changes to the mandatory fields in the application's Customizing, you also adapt them in Transaction BD65 so that custom-developed mandatory fields are also mandatory for reducing in the IDoc. Figure 2.10 shows a section of the data for the MATMAS message type. Reduced message type



Figure 2.10 Basic Maintenance for Reducible Message Types

Reduction in
CustomizingYou then create your own reduced message types in Customizing of
Transaction SALE. Under the Create Reduced Message Type menu item
or via Transaction BD53 you can find the initial screen as shown in Fig-
ure 2.11.

Structure	
▽ 🛃	Doc Interface / Application Link Enabling (ALE)
Þ	Basic Settings
Þ	Communication
▽ 🕞	Modelling and Implementing Business Processes
Þ	🔒 Global Organizational Units
	🗟 🕀 Maintain Distribution Model and Distribute Views
Þ	Configure Predefined ALE Business Processes
\bigtriangledown	🗟 Master Data Distribution
	Replication of Modified Data
	Serialization for Sending and Receiving Data
	🗢 🗟 Scope of Data for Distribution
	🗢 🛃 Message Reduction
	🗟 🕒 Create Reduced Message Type
	🗟 🕒 Generate Transport Request for Message Type

Figure 2.11 Create Reduced Message Type

When you specify the name of the new reduced message type (see Figure Reduction steps 2.12), you must consider the naming rules for your own objects (the name must start with Y or Z or your own namespace).



Figure 2.12 Creating a Reduced Message Type

Segments and fields that are green in the SAP system and displayed with an * behind the name are mandatory and can't be reduced. Fields and segments that are red or marked with – are optional and not selected; segments or fields that are white or marked with + are optional and selected in the respective reduced message type. You now specify which segments you want to have in addition to the mandatory segments by selecting the segment and clicking Select. As soon as you've activated a segment, you can select the fields within the segment you want to have in addition to the mandatory fields, and then click Select again.

Change pointers for reduced message types If you want to generate change pointers for custom reduced message types, you can activate the generation in Transaction BD53 using the Activate Change Pointers button. Of course, you can also activate the change pointers for the reduced message type in Customizing of Transaction SALE. Here, you must make sure that you not only set the flags for generating change pointers but also copy all standard field assignments and mandatory field assignments, which is automatically done in Transaction BD53. In Transaction BD53, you can also disable the writing of change pointers using the Deactivate Change Pointers button (see Figure 2.13).

IDoc Reduction Maintenance: Initial Screen		
🗋 🖉 🚱 🛅 Activate change pointers Deactivate change pointers		
Reduced message type	ŻSM1	

Figure 2.13 Activate the Change Pointer for Reduced Message Types

Directly Generating or Requesting Master Data

If you want to publish the creation or change of material master data without waiting for IDocs that are generated from change pointers, you can use Transaction BD10. In Table 7.5 of Chapter 7, Section 7.3, Transaction Code Overview, you can also find the transactions that belong to other master data.

Sending Because master data usually offers the option of reduction, these transacmaster data because master data usually offers the option of reduction, these transactions "expect" you to enter the message type you want to use for sending and the logical target systems you want to send to. Additionally, you can select the objects you want to generate IDocs for. However, this is only possible via the material numbers or the class membership of the object (see Figure 2.14).

Send Material			
⊕ 🔁 🖬			
Material		ð	₽
Class		to	_=>
Message Type (Standard)	MATMAS		
Logical system			
Send material in full			
Parallel processing			
Server group			
Number of materials per proces	20		

Figure 2.14 Targeted Sending of Material Master IDocs

If you activated the Send Material in Full flag and also set the distribution of classification IDocs to the same partner, the system generates the classification IDoc that belongs to the material additionally to the material IDoc itself. The specifications for the parallel processing help you increase the performance if you send a high quantity of data, for instance, for initial data load. If you leave the Logical System field empty, the data is sent to all partners that are available in Transaction BD64. If a selection is made, Transaction BD64 checks whether the selected logical system is permitted as a receiver of the material master IDocs. If there is a positive result, the system sends the IDoc.

If you are the receiver of master data IDocs and know that the sending system has changed or has newly created data, you can request a corresponding master data IDoc. The name of the appropriate message type starts like the master data IDoc but uses another abbreviation at the end, that is, FET (for "fetch") instead of MAS. For example, the name is MATMAS for the message type of the material master IDocs and MATFET for the fetch IDoc.

These *fetch IDocs* must be maintained as usual in the distribution model Fetch IDoc (see Figure 1.4 in Chapter 1, Section 1.3, Differentiation of ALE and EDI) — just in the other direction; here, the partner that receives the master data IDoc sends the fetch IDoc. Fetch IDocs always transfer the

Sending the material object key for which master data IDocs are requested and the message type that is supposed to be used to send the data. You can use Transaction BD11 to "get" material masters (see Figure 2.15).

Get Material			
Ð			
Material		to	\$
Class		to	÷
Message type	ZSM1	6)

Figure 2.15 Requesting the Material Master IDoc

"ALEREQ01" IDoc The same ALEREQ01 IDoc type is assigned to all fetch message types. It contains the segments shown in Figure 2.16.

ALEREQ01	■ General request - Basis IDoc type Basic type Released □ Released from Release 30A Display for Release 46C Version 3 of IDoc record types IDoc record types for SAP Release 4.0
ETALER1	 ALE request IDoc header segment Segment definition E2ALER1001 Released from Release 40A mandatory min./max. 0000000001/0000009999
- Segment fields	
MESTYP	■ o 0 Logical message type Internal length 000006 External length 000006 Offset 000063 Data type CHAR Data element FDI MESTP3
MESTYP40	 o o Hessage type Internal length 000030 External length 000030 Offset 000069 Data type CHAR Data element EDI_HESTYP
EIALEQ1	 ALE Request IDoc - Item Segment Segment definition E2ALEQ1 Released from Release 30A mandatory min./max. 0000000001/00000009999
OBJVALUE	• 0 0 Object value (with old length of 40) Internal length 000040 External length 000040 Offset 000063 Data type CMMP Data element FOR WOLD
SIGN	 o o ABAP: ID: I/E (include/exclude values) Internal length 000001 External length 000001 Offset 000103 Data type CHAR Data element TVARPY_SIGN
OPTION	 • • • ABAP: Selection option (EQ/BT/CP/) Internal length 000002 External length 000002 Offset 000104 Data type CHAR Data element TWARP.OPTI
LOW	 • • • Character field of length 40 Internal length 800040 External length 800040 Offset 800186 Data type CHAR Data element CHAR40
LHICH	■ o ¢ Character field of length 40 Internal length 000040 External length 000040 Offset 000146 Data type CHAR Data element CHAR40

Figure 2.16 "ALEREQ01" IDoc Type

Example of "MATFET" IDoc

The actually sent fetch message contains information on the message type that is expected as the response, the short and the long name (befo to be the 7 is "N

re or after Release 4.0), and the keys of elements that are supposed
sent as a response. Figure 2.17 shows a MATFET IDoc that request
SM1 material. For the MATMAS IDoc, both the long and the short name
ATMAS" because this is a very old message type.

MESTYP	Logical message type	MATMAS
MESTYP40	Message Type	MATMAS
SEGNUM	Segment Number	000002
SEGNAM	Segment Name	E1ALEQ1
OBJVALUE	Object value (with old length	MATNR
SIGN	ABAP: ID: I/E (include/exclude	1
OPTION	ABAP: Selection option (EQ/BT/	EQ
LOW	Character field of length 40	ZSM1

Figure 2.17 MATFET IDoc

Because master data is usually exchanged between different systems of the same enterprise, logical systems are used as partners like they are used in ALE.

Message Control 2.1.2

The message control is a standard function of SAP, which triggers a data Message control transfer for all transaction data that is supposed to be received by other enterprises, too. This can be done using a printer, fax, or an IDoc. For processing using IDocs, you can use transmission medium 6 for processing using partner functions, and you can use transmission medium A for processing using logical systems. All the required settings for the message control are in Transaction NACE.

You use the condition technique to specify when you generate which messages in which way. The key for the conditions is composed of the application you're in (e.g., "EF" for purchasing), the output type you want to generate (e.g., "NEU" for a purchase order), and the partner role everything will be sent to (e.g., partner "vendor" in its role as goods vendor).

43

"EDI_ PROCESSING" function In message determination, the actual message is generated using the RSNAST00 report. The EDI_PROCESSING function that is used here transfers the message as an EDI message using the IDoc; it belongs to transmission medium 6. Depending on the system setting, you can call the RSNAST00 report directly upon saving the document or at regular intervals as a batch job.

In *message control*, an EDI communication is usually assumed so that you work with partners and not with logical systems.

- Outbound partner profile The information on how an IDoc is sent to the receiver (e.g., via RFC or file) and whether an EDI subsystem is supposed to be used if necessary, is set in the *outbound partner profile* in Transaction WE20 both for the communication with partners and for the communication with logical systems.
 - **Process code** Here, you also specify which IDoc type is supposed to be used. If you work with the message control, in the outbound partner profile under the Message Control tab, you specify which process code (and which underlying function module) will be used to populate the IDoc data. You can find the valid process codes for the respective output type in Transaction WE41 (Figure 2.18).

Dialog Structure	Process code	ME10	ORDERS: Purchase order
Cal Logical message	Assignment to logical me	ancoord	
	Message type	ORDERS	Purchase order / order
	O All types		
	Message code		
	O All codes		
	Message function All functions		

Figure 2.18 Assignment of the Process Code to the Message Type

Message codes and message functions which you know from the partner profiles, to be able to use different process codes for updating the IDocs. Message codes and functions are freely selectable, and you don't need to adhere to naming rules. However, this also means that no input help is provided in Transaction WE20, so you must ensure the correct spelling of the names yourself. In the details of the sample process code for the ORDERS generation, which is shown in Figure 2.19, you can view the link to the associated function module.

Dialog Structure		
Logical message	Process code	ME10
	Description	ORDERS: Purchase order
	Function module	IDOC OUTPUT ORDERS
	Option ALE	-Service/inb. procg
	Process	ing with ALE service
	O Process	ing w/o ALE service
	O Process	ing w. trigger (inbound)
	Version of f	unction module
	Process	ing with function module version 3.0
	O Process	ing with function module version 2.2

Figure 2.19 Assignment of the Process Code to the Function Module

- Under Option ALE-Service/Inb. Procg, you can select whether the ALE ALE services services are supposed to be used or not. The ALE services represent options of IDoc manipulation using filters and rules. Chapter 4, Section 4.1, Customizing, discusses the ALE services topic in more detail.
- Because you often require a lot of partner profiles especially if you Default values work with partners you have the option to create templates to save some time. The partner profile from Transaction WE20 can be created from this template, so you don't need to create it manually. Transaction WE24 is the transaction for the template in the outbound processing (see Figure 2.20). Transaction WE27 is the corresponding transaction for the inbound processing.

Change View "F	Proposal for outbound pa	rtner profiles": Details
🦅 New Entries 🗈		
Partn.Type	LI Vendor	
Direction	1	
Message Type	ORDCHG	
Partner Role		
Message code		
Message function		
Collect IDocs		
Start subsystem		
Receiver port	SUBSYSTEM Port für ein EDI-Subsyst	em via Dateischnittstelle
Basic type	DELFOR01	Delivery schedule/JIT schedule
Extension		
View		
Recipient type	0 Organizational unit	
Recipient	50010120 EDI Department	
Message control paramet	er	
Application	EF Change message	9
Output type	NEU	
Process code (out)	ME11	ORDCHG: Purchase order change

Figure 2.20 Default Values for Outbound Partner Profiles

2.1.3 Special Functions

Warehouse management example

In some cases, a specific business process can be implemented completely locally on an SAP system or distributed across multiple SAP or non-SAP systems. IDocs are only generated when processes are distributed across multiple systems, and the generation can be activated via the Customizing functions of the respective application. Warehouse management is an example of this direct IDoc generation. The default setting of warehouse management assumes that your warehouse is managed by your SAP system. If this isn't the case, you can activate the connection of your warehouse system via ALE in Customizing. This connection causes a WMTORD IDoc to be generated directly when you create a warehouse transport request to notify the external warehouse of what is supposed to be transported. However, these special cases can't be described in general but instead must be named and set up in cooperation with the individual end-user. Because all transactions that are required for such a special case are module-specific, they are not discussed further here, but just be aware that such special cases exist.

Within the scope of ALE scenarios, there can also be cases in which sample postings are implemented synchronously via BAPI, and the actual postings are implemented asynchronously via IDoc. Then, the BAPI is created on the development side, and the appropriate IDoc is generated using Transaction BDBG. You can also use this transaction if SAP doesn't provide an IDoc for a BAPI and you still require it for a distribution scenario that isn't provided for by SAP. Again in this case, you must observe the naming rules for customer objects.

Figure 2.21 shows an IDoc generated by SAP in the SAP namespace. In the IDoc Interface tab, you can view the names for the message type and the IDoc type; in the ALE Outbound Processing tab, you can view the function group in which the IDoc modules are located as well as the name of the module that generates the IDoc; and in the ALE Inbound Processing tab, you can view the name of the module that extracts the IDoc in the receiving system and triggers the update.

The process of generating an IDoc from a BAPI is as follows:

- 1. The sending system wants to call the BAPI and checks whether this is supposed to be carried out locally or remotely.
- 2. If the call is remote and is supposed to be carried out transactionally, the function module that is generated in Transaction BDBG is called in the sending system. This function module transfers the transfer parameters of the BAPI to the IDoc format.
- 3. After you've made the settings in the customer distribution model and in Transaction WE20, this generated IDoc is transferred to the receiving system.
- 4. In the receiving system, the BAPI_IDOC_INPUT1 function module is called using the BAPI process code or BAPI_IDOC_INPUTP using the BAPP process code; this depends on whether one or more data records are received simultaneously. These function modules call the inbound function module that is generated in Transaction BDBG, which extracts

IDoc – BAPI interface

BAPI processing

the IDoc and uses the transferred data to call the original BAPI that carries out the actual update.

🗋 🖉 🛅 🚰 Check inte	erface 🕜 Display interface	🔗 Display available interfaces
Object/Interface Type	BUS6006	
Method	POST	
Category		
Object Type Er Ent	er Interface Parameters	
O Interface	nterface /	
Mess	age Type	ACC_EMPLOYEE_PAY
IDoc	type	ACC_EMPLOYEE_PAY02
Pack	age	ACID_IF
ALE C	Outbound Processing	
Func	tion Module	ALE_ACC_EMPLOYEE_PAY_POST
Func	tion group	ACAS
		Data Filtering Allowed
ALE II	nbound Processing	
Func	tion Module	IDOC_INPUT_ACC_EMPLOYEE_PAY
Func	tion group	ACC4
		Call in Update Task
		Packet Processing Allowed

Figure 2.21 IDoc Interface to a BAPI

Asynchronous BAPIS Because both cases represent an ALE scenario, you maintain the customer distribution model for BAPIs and IDocs. For the BAPIs, you enter the methods that must be processed both synchronously and asynchronously via IDocs. Additionally, you require partner profiles for the transactional case as usual. If you have these partner profiles generated from the distribution model, the system automatically knows for which BAPIs there are message types and thus for which it requires a partner profile.

Determining the target system for synchronous BAPIs Use Transaction BD97 to maintain the destination for the synchronous BAPI call in a remote system. This can be done both generally for all method calls and only for special BAPIs and for dialog calls. You need the dialog calls if you want to work with the IDoc tracing in Transaction BD87. The distinction with regard to dialogs is made for security reasons. Because a dialog user must be used in the RFC destination, this user should have only a few authorizations.

Figure 2.22 shows an example for each of the three cases.

Assign RFC Destinations for Synchronous Method Calls
🖉 🗃 🕏 🚖 🗋 Standard BAPI destination 📄 Standard dialog destination 📄 Special method destination
TBOCLINTOGO XD0 client 800 AHRCLNT000 AHR (HR system) client 000 AHRCLNT003 AHR (HR system) client 003 AHRCLNT006 AHR CATTS client 026 AHRCLNT026 AHR CATTS client 026 AHRCLNT026 AHR CATTS client 026 AIL 00 800 AII System client 800 AIN1 Auto ID Node 1 AIN2 Auto ID Node 2 AIN8 000 Auto ID Node 2.1 client 800 AIN 800 NB1 Auto ID Node 2.1 client 800 AIR CLIT006 AIR client 000 () ALRCLNT006 AIR client 006 ARCLNT006 AIR Mandt 006 ARDCLNT8@0 APOCLNT800
Standard RFC destination for BAPI calls
Standard RFC destination for dialog calls
ALEMANU Test ALE communication with Manugistics
ERFC destinations for special method calls
AcctngEmplyeePaybles.Check Accounting: Check Vendor Acct Assignment for HR Posting (OAG:LOAD PAYABLE)

Figure 2.22 Customizing for Synchronous BAPI Calls in ALE Scenarios

The standard destination for BAPI calls for the APOCLNT800 logical system is APOCLNT800, the destination for the dialog calls is called ALEMANU, and the BACKEND destination is used for the AcctngEmplyeePaybles.Check method only.

2.2 Use of Logical Systems in the Message Control

Transaction data may be exchanged within ALE scenarios. For example, this could be the scenario of central sales/decentralized shipping. Here, purchase orders, deliveries, and invoices are exchanged between plants of the same enterprise. For this case, in message control, you simply use the customer or vendor as the partner. Instead of "6" for the EDI pro-

"ALE_ PROCESSING" subroutine cessing, however, you enter "A" for the ALE processing as the transmission medium for the found message. As a result, the system no longer uses the EDI_PROCESSING subroutine and instead uses the ALE_PROCESS-ING subroutine.

- Message control and ALE You then maintain a customer distribution model as usual and use a logical system as the sender and a logical system as the receiver of the message. After the message control within the application, the transmission medium A reads the customer distribution model and replaces the partner found in the message control with the logical system found in the model. For this purpose, you need the assignment of output types to message types. They are available in the process codes in the outbound processing in Transaction WE41 and in the settings that you made for the message control in the outbound partner profile in Transaction WE20.
- Uniqueness of the assignment The data from Transaction WE20 is evaluated in the ALE_PROCESSING subroutine. Because the subsequent receiver isn't yet determined in this first step, the system searches using the "Logical System" partner type and the output type only. If more than one entry is found in Transaction WE20, for example, NEU – ORDERS, which would be the standard, and NEU – ZSMORD for a self-programmed message type, ALE_PROCESSING cancels with an error. You establish the required uniqueness by using your own output types for your own message types.

The search via the "Logical System" partner type and the partner type results in the benefit of a reduced number of partner profiles. Then you can just have a partner profile for the logical system instead of a partner profile for each vendor or customer.

2.3 Summary

In this chapter, we've taken a closer look at the creation of IDocs. You've learned which methods for creation exist and how they are related to the different needs of different processes or data types. The decision as to which one should be used is made by SAP.

Now let's take it a step further regarding our work with IDocs. The next chapter will teach you how to test the creation and the posting of IDocs.

Index

<u>A</u>

ABAP List Viewer \rightarrow ALV, 134 Acknowledgment, 187 Agency, 244 Agent assignment, 184 ALE, 15 business process, 15, 18 communication layer, 35 Customizing \rightarrow SALE, 31 scenario, 15, 18 service, 45, 73, 84, 134 ALE00001, 119 ALEAUD, 187, 230 ALE interface for BAPI, 173 for function modules, 174 ALE_MODEL_DETERMINE_IF_ TO_ SEND, 155 ALE_PROCESSING, 50 ALEREQ01, 42 ALE_SERIAL_KEY2CHANNEL, 200 ALV, 231 ANSI ASC X12, 27 API (Application Programming Interface), 18 Application, 43 Application acknowledgment, 187 Application Link Enabling \rightarrow ALE, 15 Application record, 21 Archiving object, 225 Asynchronous interface, 17 a_t_complex_matnr, 135

B

BAdI, 91 classic, 103 implementing class, 113 interface, 91, 113 new, 110 BAPI, 18, 47 release, 19 BAPI_IDOC_INPUT1, 47 BAPI_IDOC_INPUTP, 47 BAPIRETM, 174 BAPIRETS, 174 BAPI_TRANSACTION_COMMIT, 19 BAPI_TRANSACTION_ROLLBACK, 19 BAPP, 47 Basic type, 126 BD10, 40 BD40, 195 BD41, 196 BD42, 197 BD43, 197 BD44, 194 BD51, 168 BD52, 32 BD53, 38, 40 BD54, 25 BD55,84 BD56,77 BD57, 198, 199 BD58, 89 BD59,74 BD60, 34, 36, 69, 70, 135, 151 BD64, 26, 41, 67 BD65, 37 BD66, 34 BD68, 72 BD79,81 $BD87 \rightarrow IDoc-Monitor, 24$ BD95, 73, 198 BD97, 48 BD100;BD97a, 202, 204 BD101;BD97b, 202 BD102;BD97c, 204 BD103;BD97d, 205 BD104;BD97e, 200 BD105;BD97f, 200 BDA4, 176 BDBG, 24, 47, 173, 174

BDCP, 36 BDCP2, 36 BDCP_BEFORE_WRITE, 121 BDCPS, 36 BDFG, 174 **BDSER**, 198 BF01, 100 BF05, 100 Binding, 181 BOMMAT, 200 BOR, 18, 168 Breakpoint, 110 BTE. 90 BUS1001006, 168 Business Add-In \rightarrow BAdI, 91 **Business Application Programming** Interface \rightarrow BAPI, 18 Business Object Repository \rightarrow BOR, 18 Business object type, 18 Business-related error, 20 Business system, 243 Business transaction event \rightarrow BTE, 90 Business workflow, 18

С

CALL CUSTOMER-FUNCTION, 93 C file, 23 Change document, 30 CHANGEDOCUMENT_CLOSE, 30 CHANGEDOCUMENT_OPEN, 30 Change pointer, 31 activate generally, 32 for message type, 32 for reduced message types, 40 CHANGE_POINTERS_CREATE, 31 CHANGE_POINTERS_CREATE_LONG, 31 Channel number, 200 CIMTYP, 93 CL02, 72 Class, 71 Classic BAdI, 103 Classification, 69, 72 CL_EXITHANDLER, 103

CMOD, 94 COMMIT, 92 Communication channel, 245 Company code, cross-system, 85 CONDENSE, 153, 157 Condition technique, 43 Conversion exit, 83 CONVERSION_EXIT_ALPHA_ OUTPUT, 83 Conversion routine, 83 Cross-system business area, 85 Cross-system company code, 85 Customer distribution model, 26, 67 management object, 101 project, 95 segment, 122 Customer exit, 90, 93

D

Data channel registry, 204 Data Dictionary \rightarrow DDIC, 124 Data record, 21 Data Source, 228 Date field, 153 DDIC, 124 Debugging, 55 Decimal number, 153 Description *semantic, 16 technical, 16* Dialog call, 48 Distribution class type, 70 Document number, 198 DTD (Document Type Description), 23

Ε

EDI, 26 EDIFACT, 22, 27, 144 EDI_PROCESSING, 44, 50 EDIS, 65, 175 Electronic Data Interchange \rightarrow EDI, 26 Enhancement, 91, 95, 107 check, 129 dynamic, 118 explicit, 107 implement, 108 implementation, 111 *implicit*, 115 partner profile, 131 point, 107 project, 90 release, 128 section, 107, 108 spot, 91, 107, 108, 110 static, 118 technology, 67 type, 125 EOIO (Equally Once In Order), 205 Error business-related, 20 status, 167 technical, 20 Error workflow, 175, 176 connection, 167 Exit, 90 EXIT_SAPLBD11_001, 119 Extensible Markup Language \rightarrow XML, 23 Extension, 126

F

Fetch IDoc, 41 FIBF, 101 FIDCMT, 89 File, 23 FILE, 140 File port, 56, 138 Filter group, 68 Filter object, 67, 68, 73 Floating point number, 153 Function module *evaluation of change pointers, 34 file name generation, 138 for queue names, 206 test, 55*

G

GDPdU, 224 General method call, 48 General rule, 83 General task, 181, 184 Generate control record, 158 German Principles on Data Access and the Examination of Digital Documents, 224 GET*BADI, 110 GMP (Good Manufacturing Practice), 224

Η

Header mapping, 245 Header record, 21 Header table, 141 HTML (Hypertext Markup Language), 23 HTTP (Hypertext Transfer Protocol), 23

I

IDoc, 18 adapter, 243 administrator, 184 monitor, 24, 160, 168 request, 41 status file, 62 suppress, 68, 76 tracking, 24 type, 22 IDOCAPPL, 176 IDOCBOMMAT, 200 IDOC_DATE_TIME_GET, 24 IDOC_INBOUND_ASYNCHRONOUS, 23 IDOC_INBOUND_IN_QUEUE, 205 IDOC_INPUT_MATMAS01, 133 IDOC_REDUCTION_FIELD_REDUCE, 156 IDOCREL, 211, 212 IDOC_SERIALIZATION_CHECK, 199

IDOC_SERIAL_POST, 199 IDX_DELETE_IDOCTYP_WITHOUT_IS, 248 IDXIDOCINB, 248 IDX_SELECT_IDOCTYP_WITHOUT_IS, 248 Importing parameter unique, 165 Inbound interface, 15 Info structure, 227 Initial, 153 INPUTERROROCCURRED, 179 inputFinished, 178, 181 **INPUTFOREGROUND**, 179 Interface asynchronous, 17 synchronous, 16 Intermediate Document \rightarrow IDoc, 18 ISO code, 144, 153 Item table, 141

Κ

KEY, 33

Ē

Link, 211 Logical message, 151 Logical system (LS), 25

Μ

Mandatory field, 37 Mandatory segment, 127, 148 Master data, 29 MASTERIDOC_CREATE_MATMAS, 93, 107 MASTERIDOC_CREATE_SMD_ MATMAS, 135 MASTER_IDOC_DISTRIBUTE, 159 MATFET, 26, 41 MATMAS, 22, 26 mbdconwf, 162 Message code, 44 control, 29, 43 determination, 44 function, 44 Message type, 22, 151 reduced, 37 Meta information, 22 Method call, 48 MGV00001, 95 MGV00100, 100 Modification, 90 MOVE, 81

Ν

NACE, 43 Naming rule, 148 *customer object, 47 own object, 39* New BAdI, 110 NO_DATA character, 30

0

01CL. 70 Object channel serialization, 200, 201 Object Key, 214 Object services, 214 OBJTYPE, 212 OPEN_FI_PERFORM_ XXXXXXXX_E';OPEN_FI_PERFORM_ XXXXXXXX_E, 99 ORDCHG, 22, 198 ORDERS, 22 Organizational management, 184 Outbound interface, 15 Outbound partner profile, 44 Output type, 43 OYEA, 184

Ρ

Partner, 26 inbound profile, 24 outbound profile, 24 profile, 45, 79 role, 43, 145 Party, 244 PFTC_INS, 178 $PI \rightarrow SAP$ NetWeaver PI, 187 Port management, 138 Process code, 168 BAPI, 174 BAPP, 174 Processing layer, 63 Process Integration \rightarrow SAP NetWeaver PI, 187 Process interface, 91 Proposal for rule, 81 Proxy interface, 16 P/S interface, 91 Publish&Subscribe \rightarrow P/S interface, 91

Q

qRFC, 205 Qualifier, 144, 145 Quality of Service, 205 Queue, 209 Queued RFC \rightarrow qRFC, 205

R

RBDMIDOC, 35 RBDSER01, 194, 195 RBDSER02, 196 RBDSER03, 197 RBDSER04, 197 RBDSRCLR, 199 RDBSTATE, 188 Receiver, 26 Receiver communication channel, 245, 247 Reduced message type, 37 Release security, 15, 19

Remote Function Call \rightarrow RFC, 16 Request, 16 Request IDocs, 41 Response, 16 RFC, 16 queued \rightarrow qRFC, 205 synchronous \rightarrow sRFC, 19 transactional \rightarrow tRFC, 19 RFM (RFC-enabled function module), 18 ROLEID, 212 ROLLBACK, 92 Root segment, 78, 79 RSEOUT00, 59, 196 RSNAST00, 44, 58 RSRLDREL, 215 Rule, 67, 80 general, 83

<u>S</u>

SALE, 25, 31 Sample file, 56 SAP_ALE_EDIQUEUE, 206 SAP_IDOC_001, 227 SAP NetWeaver PI, 205, 243 agency, 244 header mapping, 245 partner, 26 party, 244 SARA, 225 SARI, 228 SBWP, 183 SE11, 124 SE18, 103 SE19, 103, 110 SE80, 111 SE84, 91 Segment definition, 124 filter, 76 qualifying, 145 release, 125 type, 23, 124 version, 84 Sender. 26

Sender field convert, 83 copy, 82 SERDAT, 197 SERIAL, 198, 203 Serialization, 193 business objects, 200 flag, 202 group, 193, 194 timestamp, 198 Shared Master Data Tool \rightarrow SMD, 29 Signature, 154 inbound function module, 162 SLD, 244 SM59, 102 SMD, 29 SNDPOR, 243 SNDPRN, 244 SOAP, 16 Special method call, 48 sRFC, 19 SRRELROLES, 211 Standard inbound processing, 54 Standard interface, 15 Standard outbound processing, 57 Standard task, 178 Status 42<normal>, 57 Status 53<normal>, 188 Status 66, 204 Status 68<normal>, 188 Status 74<normal>, 61 STATUS IDoc, 62, 190 Status record, 21 Success status, 167 SWO1, 176 Synchronous interface, 16 Synchronous RFC \rightarrow sRFC, 19 SYSTAT, 62, 230 System Landscape Directory \rightarrow SLD, 244 System name, logical, 25 System, technical, 244

Τ

Task, general, 181, 184 Technical error, 20 Technical system, 244 Templates, 45 Test flag, 54 function module, 55 transaction, 51 Time, 153 Transactional RFC \rightarrow tRFC, 19 Transaction data, 29 Transaction security, 17 TRANSID, 212 Transmission medium 6, 43, 44 Transmission medium A, 43, 50 Transport IDoc, 23 tRFC, 19

U

UN/EDIFACT → EDIFACT, 22 Unicode, 56 UNIT_OF_MEASURE_ISO_TO_SAP, 165 UNIT_OF_MEASURE_SAP_TO_ISO, 157 Update technique, 19

V

Version conversion, 84, 119 View, 77

W

WE09, 228 WE14, 58 WE16, 60 WE17, 65 WE18, 63 WE19, 52, 65 WE20, 24, 50, 119, 194 WE21, 138 WE24, 45 WE27, 45 WE30, 126, 148 WE31, 124 WE32, 78 WE41, 44 WE42, 169 WE47, 63, 224 WE55, 139 WE57, 133, 168 WE60, 150 WE62, 81 WE81, 151 WE82, 136, 151 WE85, 205 WEDI, 51 WEINBQUEUE, 208 WENOLINKS, 214 WEOUTQUEUE, 206 WMTORD, 46

X

XML, 23 xPath, 246