ALE Introduction and Administration

HELP.BCMIDALEIO

Release 4.6B

Copyright

© Copyright 2000 SAP AG. All rights reserved.

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

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

Microsoft®, WINDOWS®, NT®, EXCEL®, Word® and SQL Server® are registered trademarks of Microsoft Corporation.

IBM®, DB2®, OS/2®, DB2/6000®, Parallel Sysplex®, MVS/ESA®, RS/6000®, AIX®, S/390®, AS/400®, OS/390®, and OS/400® are registered trademarks of IBM Corporation.

ORACLE® is a registered trademark of ORACLE Corporation, California, USA.

INFORMIX®-OnLine for SAP and Informix® Dynamic Server™ are registered trademarks of Informix Software Incorporated.

UNIX®, X/Open®, OSF/1®, and Motif® are registered trademarks of The Open Group.

HTML, DHTML, XML, XHTML are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Laboratory for Computer Science NE43-358, Massachusetts Institute of Technology, 545 Technology Square, Cambridge, MA 02139.

JAVA® is a registered trademark of Sun Microsystems, Inc. , 901 San Antonio Road, Palo Alto, CA 94303 USA.

JAVASCRIPT® is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

SAP, SAP Logo, mySAP.com, mySAP.com Marketplace, mySAP.com Workplace, mySAP.com Business Scenarios, mySAP.com Application Hosting, WebFlow, R/2, R/3, RIVA, ABAP, SAP Business Workflow, SAP EarlyWatch, SAP ArchiveLink, BAPI, SAPPHIRE, Management Cockpit, SEM, are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other products mentioned are trademarks or registered trademarks of their respective companies.

Icons

Icon	Meaning
⚠	Caution
💬	Example
➡	Note
🧭	Recommendation
SYN	Syntax
💡	Tip

Contents

ALE Introduction and Administration

ALE Integration Technology

Purpose

The integration technology *Application Link Enabling* (ALE) is an important middleware tool in SAP's Business Framework Architecture [Ext.] (BFA). BFA is a component-based architecture enabling software components from SAP and from other software vendors to communicate and be integrated with each other.

ALE can integrate business processes between R/3 Systems and non-R/3 systems as well as between R/3 Systems.

Data is exchanged between applications and remains consistent in all applications.

Company-wide applications such as accounting, human resource management and sales planning may be carried out in the company's headquarters, whereas production and materials management may be carried out in decentralized plants.

Application systems are loosely coupled in an ALE integrated system. Data is exchanged asynchronously, whereby the data arrives in the receiver system, even if the receiver system cannot be connected to at the time the data is sent. ALE uses synchronous communication for reading data only.

ALE provides administration, development and testing tools.

To use the ALE tools choose *Tools → ALE.*

ALE business processes are part of the standard R/3 delivery. They are documented in the Library of ALE Business Processes [Ext.].

For more information about the required system settings see the R/3 Implementation Guide.

Tools → AcceleratedSAP → Customizing → Project Management

SAP Reference IMG

Basis → Application Link Enabling (ALE).

For information on programming see the ALE Programming Guide [Ext.].

Implementation Considerations

Decentralized business applications ensuring data consistency are used because:

- The increasing globalization of markets has led to the physical division of organizational units.
- Business processes are not restricted to one organization only and more and more customers and vendors are becoming involved.
- R/3 System performance can be improved by distributing the business applications.

Features

ALE supports the configuration and operation of distributed applications. ALE controls Messaging [Page 10] across loosely coupled R/3 Systems, ensuring that data is always consistent.

ALE Integration Technology

Applications are integrated using a local database rather than a central one. There is no data retention. ALE guarantees the distribution and synchronization of master data, Customizing data and transaction data through asynchronous communication.

Synchronous communication is used in ALE to read data only.

ALE has many benefits:

- Application data can be distributed between different releases of R/3 Systems
- Data can continue to be exchanged after a release upgrade without requiring special maintenance
- Customers can add their own enhancements
- Communication interfaces enable connections to non-SAP systems.
- R/3 and R/2 Systems can communicate with each other.

ALE has functions for monitoring messages flows and handling communication problems.

ALE and the R/3 Procedure Model

The R/3 Procedure Model [Ext.] provides information application-wide for implementing R/3 software. The model is like a phase model, whereby work steps are assigned to each phase and one or more Customizing work steps are associated with each phase.

When implementing ALE functions you should refer to the R/3 Procedure Model to ensure that R/3 is implemented in a well structured way.

If ALE is to be supported in an R/3 System installation, implementation of the R/3 Procedure Model is affected at four points:

- Determining functions and processes
- Specifying global settings
- Modeling company structure
- Modeling control data and master data

If ALE business processes are supported, functions and processes must be defined on a cross-system basis. The message types exchanged between the systems are defined in the distribution model.

When defining organizational units like company codes and plants in the R/3 System, it is important to note that these have to be unique worldwide, across all systems. For example, the plant **0001** can only exist in one R/3 System.

Message Distribution

Purpose

The basis for distributing application functions is asynchronous distribution of messages via ALE outbound and inbound processing.

Prerequisites

Message distribution is based on IDocs [Ext.]. IDocs are created and dispatched in distributed systems using message types and SAP business object methods (BAPIs).

As of Release 4.0 synchronous and asynchronous interfaces should be implemented as BAPIs. If you want to implement a BAPI as an asynchronous interface, you have to generate a BAPI-ALE interface for an existing BAPI. SAP provides these interfaces for many BAPIs. You can also generate such interfaces yourselves.

You can implement ALE business processes with these interfaces.

For more information see the ALE Programming Guide [Ext.] under *Distribution Using BAPIs* and *Distribution Using Message types*.

Process Flow

In the outbound system an IDoc containing the data to be transferred is created and prepared for dispatch in the outbound system.

Then the IDoc is transferred to the target system.

In the target system the IDoc starts inbound processing. The data is processed and then posted in the application either fully automatically or part manually.

Inbound and outbound IDocs can be processed individually or in a packet. In mass processing several IDocs are processed together in one packet.

Distribution Using BAPIs

BAPIs can be called by applications synchronously or asynchronously. ALE functions such as BAPI maintenance in the distribution model and receiver determination can be used for both types of call.

Note that synchronously-called BAPIs are only used for reading external data to avoid database inconsistencies arising from communication errors.

The application synchronously calls a BAPI in the external system to create an FI document. The document is correctly created but the network crashes whilst the BAPI is being executed. An error message is returned to the application and the FI document is created again. The document has been duplicated in the system called.

An application program can implement a two-phase commit by thoroughly checking the data in the external system. An easier solution is to call the BAPI asynchronously, as Error Handling [Ext.] assures that the data remains consistent.

A BAPI should be implemented as an asynchronous interface, if one of the criteria below applies:

- Consistent database changes in both systems

  Data must be updated in the local system as well as on a remote system

- Loose coupling

  An asynchronous interface would represent too narrow a coupling between the client and the server systems. If the connection fails the client system can no longer function correctly.

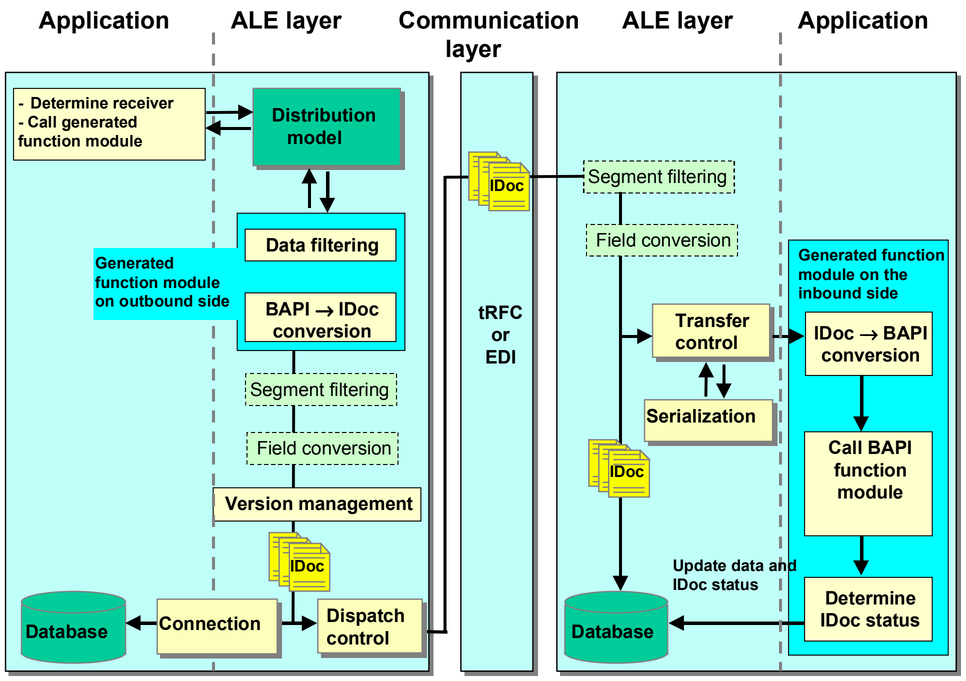
- Performance load

  The interface is used often or it handles large volumes of data. A synchronous interface cannot be used in this situation because performance would be too low.

If you want to implement a BAPI as an asynchronous interface, you have to generate a BAPI-ALE interface for an existing BAPI. For more information see Generating BAPI-ALE Interfaces [Ext.].

Data distribution using BAPIs is illustrated in the graphic below:

Distribution Using BAPIs



The processes in the application layer and the ALE layer are completed on both the inbound and outbound processing sides. The communication layer transfers the data by transactional Remote Function Call (tRFC) or by EDI file interface.

The process can be divided into the following sub-processes:

1. Outbound processing

- Receiver determination
- Call of generated outbound function module
- Conversion of BAPI call into IDoc
- Segment filtering
- Field conversion
- IDoc version change
- Dispatch control

2. IDoc dispatch

IDocs are sent in the communication layer by transactional Remote Function Call (tRFC) or by other file interfaces (for example, EDI).

tRFC guarantees that the data is transferred once only.

3. Inbound Processing

- Segment filtering
- Field conversion
- Transfer control

- Conversion of IDoc into BAPI call
- Call of BAPI function module
- Determination of IDoc status
- Posting of application data and IDoc status
- Error handling

The sub-processes in inbound and outbound processing are described below:

Outbound Processing

On the outbound side first of all the receiver is determined from the distribution model. Then the outbound function module that has been generated from a BAPI as part of the BAPI-ALE interface is called in the application layer (see also Example Programs with Asynchronous BAPI Calls [Ext.]). In the ALE layer the associated IDoc is filled with the filtered data from the BAPI call.

The volume of data and time of the data transfer is controlled by the dispatch control.

The outbound processing consists of the following steps:

Receiver determination

The receivers of a BAPI call are defined in the distribution model in same way as with synchronous BAPI calls.

Before the BAPI or generated BAPI-ALE interface can be called, the receiver must be determined. When the receiver is determined, the filter objects are checked against the specified conditions and the valid receivers are reported back.

If the distribution of the data is also dependent on conditions, these dependencies between BAPIs or between BAPIs and message types are defined as **receiver filters**.

For each of these receiver filters, before the distribution model is defined, a filter object is created whose value at runtimes determines whether the condition is satisfied or not.

For more information see Determining Receivers of BAPIs [Ext.].

Calling the generated outbound function module

If the receivers have been determined, you have to differentiate between local and remote receivers. The BAPI can be called directly for local receivers. For remote calls the generated ALE outbound function module must be executed so that processing is passed to the ALE layer. The data for the BAPI call and the list of allowed logical receiver systems are passed to this function module.

Programming Notes:

After calling the generated function module the application program must contain the command COMMIT WORK. The standard database COMMIT at the end of the transaction is not sufficient. The COMMIT WORK must not be executed immediately after the call, it can be executed at higher call level after the function module has been called several times.

The IDocs created are locked until the transaction has been completed. To unlock them earlier, you can call the following function modules:

`DEQUEUE_ALL` releases all locked objects

`EDI_DOCUMENT_DEQUEUE_LATER` releases individual IDocs whose numbers are transferred to the function module as parameter values.

Data Filtering

Two filtering services can be used - parameter filtering with conditions and unconditional interface reduction.

- If entire parameters have been deactivated for the interface reduction, they are not included in the IDoc. If, on the other hand, only individual fields are not to be included for structured parameters, the entire parameters are still included in the IDoc.
- With parameter filtering, table rows that have been filtered out are not included in the IDoc.

For more information see Filtering Data [Ext.].

Conversion of BAPI call into IDoc

Once the data has been filtered, an IDoc containing the data to be transferred, is created from the BAPI call by the outbound function module

Segment filtering

Once the IDoc has been created, IDoc segments can be filtered again. This filtering is rarely used for BAPIs.

For more information see the R/3 Implementation Guide under:

Basis
Application Link Enabling
Modelling and Implementing Business Processes
Master Data Distribution
Scope of Data for Distribution
Message Reduction

Field conversion

You can define field conversions for specific receivers in the R/3 Implementation Guide:

Basis
Application Link Enabling
Modelling and Implementing Business Processes
Converting Data Between Sender and Receiver

Standard rules can be specified for field conversions. These are important for converting data fields to exchange information between R/2 and R/3 Systems. For example, the field *plant* can be converted from a two character field to a four character field.

Standard Executive Information System (EIS) tools are used to convert fields.

IDoc version change

To guarantee that ALE works correctly between different releases of the R/3 System, IDoc formats can be converted to modify message types to suit different release statuses.

If version change has been completed, the IDocs are stored in the database and the dispatch control is started which decides which of these IDocs are sent immediately.

SAP uses the following rules to convert existing message types:

- Fields can be appended to a segment type
- New segments can be added

ALE Customizing records the version of each message type used in each receiver. The IDoc is created in the correct version in outbound processing.

Dispatch control

Scheduling the dispatch time:

- IDocs can either be sent immediately or in the background. This setting is made in the partner profile.
- If the IDoc is sent in the background, a job has to be scheduled. You can choose how often background jobs are scheduled.

Controlling the amount of data sent:

- IDocs can be dispatched in packets. The packet size is assigned in ALE Customizing in accordance with the partner profile.

Basis
  Application Link Enabling
    Modeling and Implementing Business Processes
      Partner Profiles and Time of Processing
        Maintain Partner Profile Manually
or: *Generate Partner Profiles*

This setting is only effective if you process the IDocs in the background.

Inbound Processing

On the receiver side the ALE layer continues with the inbound processing.

On the application side when the generated inbound function module is executed, the BAPI call is generated from the IDoc, the BAPI function module is called and the IDoc status is determined.

After the BAPI or the entire packet has been processed, the IDoc status records of all IDocs and the application data created from successfully completed BAPIs, are posted together.

The inbound processing consists of the following steps:

Segment filtering

On the inbound side IDoc segments can be filtered the same as they can on the outbound side. This filtering on the inbound side is also rarely used for BAPIs.

Field conversion

As with outbound processing, fields can be converted if the field format is different in the receiving and sending systems.

After the fields have been converted, the IDoc is saved on the database and it is passed to the transfer control for further processing.

Transfer control

The transfer control decides when the application BAPIs are to be called. This may be either immediately when the IDoc arrives or at a later time in background processing.

If several mutually dependent objects are distributed, **serialization** can be used during the transfer control. IDocs can be created, sent and posted in a specified order by distributing message types serially. Errors can then be avoided when processing inbound IDocs.
If BAPIs are used object serialization is used exclusively. This assures that the message sequence of a particular object is always protected.

For more information about used object serialization see ALE Customizing.

Basis
Application Link Enabling
Modeling and Implementing Business Processes
Master Data Distribution
Converting Data Between Sender and Receiver
Serialization by Object Type

When the time arrives for processing the BAPI, the generated inbound function module is called.

Conversion of IDoc into BAPI call

When the BAPI is called, the entire data from the IDoc segments is written to the associated parameters of the BAPI function module. If an interface reduction has been defined for the BAPI, the hidden fields are not filled with the IDoc data.

BAPI function module call

Next the BAPI function module with the filled parameters is executed synchronously. As the BAPI does not execute a COMMIT WORK command, the application data that it has created, modified or deleted is not yet saved in the database.

IDoc status determination

If the function module has been executed, the IDoc status is determined in the inbound function module from the result of the call.

Posting of application data and IDoc status

If each IDoc or BAPI is processed individually, the data is written immediately to the database.

If several IDocs are processed within one packet, the following may happen:

- The application data of the successfully completed BAPI together with all the IDoc status records is updated, provided that no BAPI call has been terminated within the packet.
- As soon as a BAPI call is terminated within the packet, the status of the associated IDoc will indicate an error. Application data will not be updated. Then inbound processing is run again for all the BAPI calls that had been completed successfully. Provided that there is no termination during this run, the application data of BAPIs and all the IDoc status records are updated. This process is repeated if there are further terminations.

Note: Packet processing is only carried out if there is no serialization.

Error handling

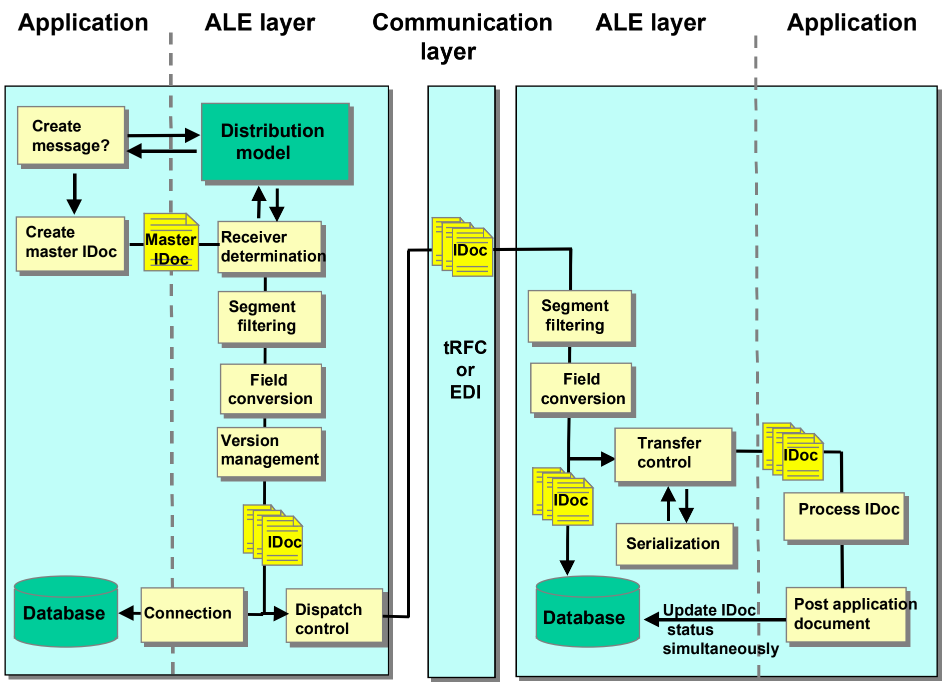
You can use SAP Workflow for ALE error handling:

- The processing of the IDoc or BAPI data causing the error is terminated.
- An event is triggered. This event starts an error task (work item).
- Once the data of the BAPI or IDoc has been successfully updated, an event is triggered that terminates the error task. The work task then disappears from the inbound system.

For more information see Error Handling [Ext.].

Distribution Using Message Types

When message types are used to transfer data asynchronously in ALE:



Outbound Processing

An application function module creates a master IDoc in outbound processing, the so-called master IDoc.

The following steps are carried out in the ALE layer:

- Receiver determination, if this has not already been done by the application
- Data selection
- Filtering segments
- Converting fields
- Version change
- Dispatch Control

The formatted IDoc is passed to the communication layer and from here sent to the system that was called (server) via a transactional remote function call (RFC) or via file interfaces (for example, EDI).

If an error occurs in the ALE layer, the IDoc containing the error is saved and a workflow task is created. The ALE administrator can use this workflow to correct the error.

For information on programming see the Implementing Outbound Processing [Ext.].

The individual steps are explained below.

Receiver Determination

Like a normal letter, an IDoc has a sender and a receiver. If the application does not explicitly specified the receiver, the ALE layer uses the distribution model to help determine the receivers of the message.

The ALE layer can find out from the model whether any distributed systems should receive the message and, if so, then how many. The result may be that one, several or no receivers at all are found.

For each of the distributed systems identified as receiver systems, the data specified by the filter objects in the distribution model is selected from the master IDoc. This data is then entered into an IDoc, and the appropriate system is specified as the receiver.

Segment Filtering

Individual segments can be removed from the IDoc before it is dispatched. If you want to remove IDoc segments, in Customizing for ALE choose:

Modelling and Implementing ALE Business Processes
Master Data Distribution
Scope of Data for Distribution
Filter IDoc Segments

The appropriate setting depends on the sending and receiving logical R/3 System.

Field Conversion

You can define field conversions for specific receivers in ALE Customizing:

Modelling and Implementing ALE Business Processes
Converting Data Between Sender and Receiver

Standard rules can be specified for field conversions. One set of rules is created for each IDoc segment and rules are defined for each segment field. These are important for converting data fields to exchange information between R/2 and R/3 Systems. For example, the field *plant*" can be converted from a two character field to a four character field.

Standard Executive Information System (EIS) tools are used to convert fields.

IDoc Version Change

SAP guarantees that ALE works correctly between different releases of the R/3 System. By changing the IDoc format you can convert message types from different R/3 releases.

SAP uses the following rules to convert existing message types:

- Fields can be appended to a segment type
- New segments can be added

ALE Customizing records the version of each message type used in each receiver. The communication IDoc is created in the correct version in outbound processing.

Dispatch Control

Time and quantity are the factors that control the dispatch of IDocs in the dispatch control.

Scheduling the dispatch time:

IDocs can either be sent immediately or in the background. This setting is made in the partner profile.

If the IDoc is sent in the background, a job has to be scheduled. You can choose how often background jobs are scheduled.

Controlling the amount of data sent:

IDocs can be dispatched in packets. The packet size is assigned in ALE Customizing in accordance with the partner profile.
Modeling and Implementing ALE Business Processes
→ *Partner Profiles and Time of Processing*
→ *Maintain Partner Profiles*

This setting only affects IDocs that are processed in the background.

Inbound Processing

The following processes are carried out on inbound IDocs in the ALE layer:

- Segment filtering
- Field conversion
- Transfer control
- Serialization

For information on programming see the Implementing Inbound Processing [Ext.].

The individual steps are explained below.

Segment filtering

You can filter IDoc segments in inbound processing.

In inbound processing this function is principally the same as in outbound processing.

Field conversion

You can define field conversions for specific receivers in ALE Customizing:

Modelling and Implementing ALE Business Processes
Converting Data Between Sender and Receiver

Standard rules can be specified for field conversions. One set of rules is created for each IDoc segment and rules are defined for each segment field. These are important for converting data fields to exchange information between R/2 and R/3 Systems. For example, the field "plant" can be converted from a two character field to a four character field.

Standard Executive Information System (EIS) tools are used to convert fields.

For reduced message types field values are not overwritten in the receiving R/3 System, if the corresponding IDoc field contains the character "/".

Transfer control

Once the IDocs have been written to the database, they can be posted by the application.

IDocs can be passed to the application either immediately on arrival or at a later time in background processing.

Inbound IDocs can be posted in three ways:

- By calling a function module directly:

  The inbound IDocs are posted directly. An error workflow is started, if an error occurs.

- By starting an SAP Business Workflow. A workflow is the sequence of steps required to post an IDoc.

  Workflows for ALE are not provided.

- By starting a work item

  A single step performs the IDoc posting.

The standard inbound processing setting is for ALE to call a function module directly. For information about the options in SAP Business Workflow see the Inbound Processing Using SAP Workflow [Ext.].

You can specify the people to be notified for handling IDoc processing errors in SAP Business Workflow. Different people can be responsible for each message type.

Mass Processing of IDocs

Mass processing refers to bundling IDocs into packets that are then dispatched and processing in the receiving R/3 System. Only one RFC call is needed to transfer several IDocs. Performance is considerably better if you dispatch optimal packet sizes.

To set mass processing parameters, in ALE Customizing choose: *Modeling and Implementing Business Processes → Partner Profiles and Time of Processing → Generate Partner Profiles.* For a given message type you can define the outbound parameters, packet size and processing mode.

If the processing mode is set to *Collect IDocs and transfer*, outbound IDocs of the same message type and receiver are sent in a scheduled background job in an appropriately sized IDoc packet. The IDocs can be dispatched in a scheduled background job or in ALE Administration.

Some distribution scenarios cannot support mass processing of inbound IDoc packets. This is especially true if the application sending the IDocs uses the ABAP command CALL TRANSACTION USING. In this case the outbound parameter PACKETSIZE must be set to "1".

You can find a list of function modules for mass processing in ALE Development by choosing *IDocs → Inbound processing → Function module → Define attributes.* These function modules have input type 0.

Error handling

For more information about the required settings see ALE Customizing:

R/3 Implementation Guide
Basis
Application Link Enabling (ALE))
Error Handling

Error Handling in ALE Outbound Processing

If an error occurs in the ALE layer, the faulty IDoc is saved and a workflow is generated. The ALE administrator can process the error through the workflow.

Error Handling in ALE Inbound Processing

Any errors that occur during ALE processing are handled as follows:

- The processing of the IDoc causing the error is terminated.
- An event is triggered. This event starts an error task (work item):
  - The employees responsible will find a work task in their workflow inboxes.
  - An error message is displayed when the work task is processed.
  - The error is corrected in another window and the IDoc can then be resubmitted for processing.
  - If the error cannot be corrected, the IDoc can be marked for deletion.
- Once the IDoc has been successfully posted, an event is triggered that terminates the error task. The work task then disappears from the inbox.

Repeating Transfer of IDoc To Application

If the processing of the inbound IDoc resulted in an error (IDoc status 51 - “Application document not posted”, or 63 - “Error passing IDoc to application”), you can use the report RBDMANI2 to resubmit the IDoc.

In this program you can select specific errors. The program can also be scheduled as a periodic job to collect IDocs that could not be posted because of a lock problem.

Modeling Distribution

Purpose

To implement the distribution of business application processes and functions using ALE you need to construct a logical Distribution Model [Page 25] of the whole system. Within the framework of this configuration, customers can specify which applications are to run on which systems and which messages are to be exchanged between the applications.

Prerequisites

The most important questions customers should ask themselves when working out a distribution concept are:

- Where exactly are the distributed R/3 Systems to be installed?
- What applications are to run on what systems?
- What master data and transaction data are to be exchanged between the applications?
- What Customizing data must be known to the distributed systems?

Process Flow

You should model the logical system as though only one single R/3 instance is involved. Applications are then distributed across several physical systems in accordance with this model. Most of the organizational units, such as PLANT or SALES ORGANIZATION must be given unique names. Company codes and business areas are the only exceptions to this.

Distribution Model

Definition

The distribution model describes the ALE message flows between logical systems.

Use

You can specify the relationships between logical systems, message types, BAPIs and filters in the distribution model. Applications and the ALE layer use the model to determine receivers and to control the data distribution.

For more information about maintaining the distribution model see the Customizing (R/3 Implementation Guide):

Basis
Application Link Enabling (ALE).
Modeling and Implementing Business Processes
Maintaining the Distribution Model

Structure

In the distribution model you can specify the messages to be sent to a given logical system. You can also define the conditions for the content and dispatch of messages in the filters [Page 26].

The distribution model consists of one or more views that you can define. With more complex distribution tasks you can assign business sub-areas or groups of logical systems to separate views.

The relevant views of the distribution model must be available in all the logical systems involved in ALE. You can find the functions for distributing views in Customizing for ALE.

Filters

Definition

Filters represent the conditions that message types and BAPIs have to satisfy so that they can be distributed by ALE outbound processing.

Use

ALE outbound processing determines the data content and receivers of messages from the filters. Filters are assigned to methods or message types in the distribution model. A filter is always assigned to a Filter Group [Page 27].

For more information about filters see the R/3 Implementation Guide:

Basis
Distribution (ALE):
Modeling and Implementing Business Processes
Maintaining the Distribution Model

Structure

Message types and BAPIs can be filtered using the following filter types:

- Filter Objects [Page 29]
- Classes [Page 31]
- Dependencies [Page 32]

Filter Groups

Definition

A filter group comprises one or more filters.

Use

Filter groups enable you to define complex conditions for filtering message types. Several filter groups can be defined for one message. A filter is always assigned to a filter group. Provided you are not adding a filter to an existing filter group, the filter group is created automatically.

Filter groups are linked disjunctively (with the logical operator OR).

Each filter group is evaluated independently of other filter groups.

Within a filter group, filters are usually linked conjunctively (with the logical operator AND). Filter objects of the same name in the same filter group are linked with an OR operator. Message type dependencies are linked conjunctively (with the logical operator AND).

Example of filtering in one filter group

The filter group comprises the following filters:

Plant 001
Plant 002
Division 01
Division 02
Distribution across classes
Dependency of message type MATMAS
Dependency of message type CREMAS

These result in the filter conditions:

(Plant 0001 OR plant 0002)
AND
(Division 01 OR division 02)
AND
Distribution via classes
AND
Dependency of message type MATMAS
AND
Dependency of message type CREMAS

Example of filtering with several filter groups

The filter group 1 comprises the following filters:

Globally unique company code GL0001

The filter group 2 comprises the following filters:

Sales organization YYZZ

As the filter groups are evaluated separately, *no* filtering takes place in this example.

Reason:

Filter group 1 contains no condition for the sales organization. So the relevant segment is created, even if the condition does not apply to filter group 2.

Filter group 2 contains no condition for the company code. So the relevant segment is created, even if the condition does not apply to filter group 1.

Filter Objects

Definition

Filter objects are attributes of a message in the distribution model. A filter objects consists of a filter object type (for example, material type) and an assigned object value (for example PROD).

Use

ALE uses filter objects in outbound processing. You can use the filter objects to specify the content of a message and its receivers.

Filter objects function differently for message types and for BAPIs.

- When **message types** are filtered, all the IDoc segments containing a field with an object type name whose value does not match the object value are suppressed. The subsegments of these IDoc segments are also suppressed.

  If the suppressed segment was a mandatory segment and provided that there are no other mandatory segments of the same name on the same IDoc hierarchy level, the higher-level IDoc segment is also suppressed. This process is repeated and the final result depends on the specific IDoc structure, mandatory segments and the segments that have been repeatedly processed.

  If the uppermost mandatory segment of an IDoc still exists after this process, the rest of the IDoc is distributed.

- With **BAPIs** the filter object type corresponds to a parameter name. BAPI filter objects check whether a parameter contains the specified object value. An IDoc is created and distributed via the BAPI, only if this is the case.

  The following distinction is made:

- Receiver determination

  When the receiver is determined, the filter objects are checked against the specified conditions and the valid receivers are reported back.

  For further information see Determining the Receivers of a BAPI [Ext.] in the *ALE Programming Guide.*

- Parameter filtering

  The dataset of the BAPI tables is determined by filtering the BAPI table parameters, similar to filtering IDoc segments.

  For further information see BAPI Parameter Filters [Ext.] in the *ALE Programming Guide.*

Structure

Filter objects comprise a filter object type and an assigned object value.

Filter object types are either field names in IDoc segments or parameter names in BAPI calls.

Classes

Definition

Classes are groups of similar objects with common characteristics.

Use

The R/3 System classification can be used to control ALE distribution. Distributable object classes can be assigned to a message type and used as filters for distributing this message type.

For example, you can classify the following object types:

- Material
- Customer
- Vendor

Structure

A class contains characteristics that describe the attributes of the objects to be classified. You can define the values allowed for the characteristics.

Dependencies

Definition

Dependencies describe the distribution relationship between the interfaces below:

- Between BAPIs
- Between message types
- Between a BAPI and a message type

Use

In some cases business processes and data only need to be distributed, if other processes are also being distributed. If so, you can make the dispatch of one message dependent on the dispatch of another one.

Such a dependency must be implemented using an application function module and it must be defined in ALE Development. For more information see the ALE Programming Guide [Ext.].

When you maintain the distribution model, these defined dependencies are offered as an option for receiver filtering.

Example of dependency between a BAPI and a message type:

The distribution of company addresses has been integrated in the object maintenance for the vendor. The address data is then distributed together with the object data via ALE. The address data is dependent on the object data and is distributed via the BAPI. The object data is distributed in the message type CREMAS.

So a dependency exists between the BAPI and the message type.

In the distribution model an active receiver filter is assigned to the BAPI used for distributing organization adresses (AddressOrg.SaveReplica). The dependency has been activated in the attribute *Dependent distribution* in the filter display.

Based on the receiver determination, the object data with the BAPI addresses are only distributed, if the filter condition for CREMAS is met.

Setting Up the Communication

To set up the communication, a series of steps are required that you have to carry out in Customizing for ALE (Transaction SALE).

More information about ALE communication can be found in the R/3 Implementation Guide (IMG) under *Basis Components → Distribution (ALE)*.

- Define the RFC destinations

  The remote function call is controlled in the parameters of the RFC destination. For further information see Maintaining Remote Destinations [Page 34].

  You can also specify which RFC destinations are to be used for synchronous method calls. These calls may either be BAPIs or Dialog Methods [Ext.]. For further information see RFC Destinations for Synchronous Method Calls [Page 43].

  In the ALE IMG choose:

  Sending and Receiving Systems
  Systems in Network

- Generate partner profiles

  The partner profiles are based on the distribution model.

  In the ALE IMG choose:

  Modeling and Implementing Business Processes
  Partner Profiles and Time of Processing

- Partner profiles and model settings are checked for consistency.

  In addition to checking the consistency of the partner profiles and model settings you can also check the consistency of all the systems and the distribution of the Customizing data.

  In the ALE IMG choose:

  Modeling and Implementing Business Processes
  Partner Profiles and Time of Processing

- Check connections

  When the system is running, you can check whether the messages sent have been received in the receiver system and you can monitor the processing status of the messages.

  In the ALE IMG choose:

  System Monitoring
  IDoc Confirmation in Receiver System (ALE Audit)

Maintaining Remote Destinations

Choose *Tools* → *Administration* → *Administration* → *Network* → *RFC destinations*.

Details are explained in the following topics:

Types of Destinations [Page 39]

Displaying, Maintaining and Testing Destinations [Page 35]

Entering Destination Parameters [Page 37]

Maintaining Group Destinations [Page 42]

Displaying, Maintaining and Testing Destinations

To display, create or modify destinations, choose *Tools* → *Administration* → *Administration* → *Network* → *RFC destinations* or enter transaction code SM59.

Remote Destinations are stored in table RFCDES. The RFCDES table describes logical destinations for remote function calls.

It is not possible to maintain the RFCDES table directly.

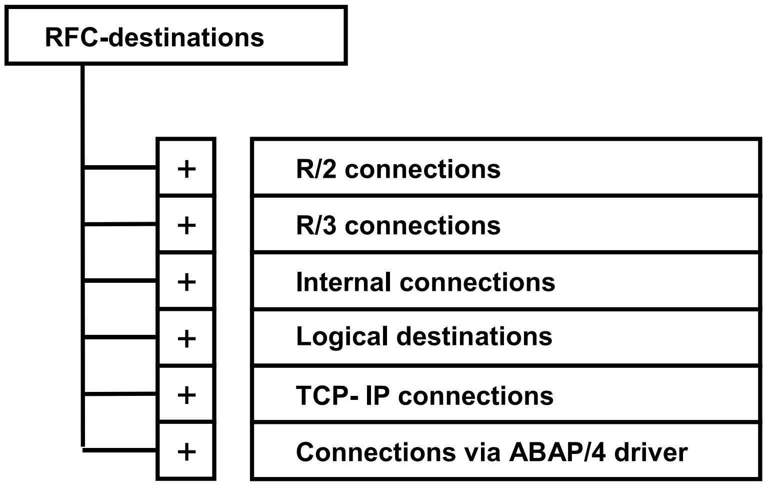
You can also access logical destinations via the Implementation Guide (IMG) by choosing *Tools* → *AcceleratedSAP* → *Customizing* → *Execute Project* → *SAP Reference IMG*.

In the Implementation Guide, expand the following hierarchy structure:

Basis
Application Link Enabling (ALE)
Sending and Receiving Systems
Systems in Network
Define Target Systems for RFC Calls

Displaying Destinations

The initial screen for this transaction displays a tree:



Different connection types (i.e. partner systems or programs) are possible. For further information, see Types of Destinations [Page 39].

To display all information for a given destination, double-click it, or place the cursor on it and press `F2`.

To search for a destination, press the *Find* button and specify your selection. You get a list of all entries matching your selection. Place the cursor on the one you want, and press `F2` or simply double-click the destination. All information for the given entry appears.

Creating Destinations

On the destinations overview screen (transaction code SM59), the connection types and all existing destinations are displayed in a tree structure.

All available connection types are explained in Types of Destinations [Page 39].

To create a new RFC destination, press the *Create* button. A new screen is displayed with empty fields for you to fill in.

If you want to create a new destination

As you create a remote destination, you can specify a particular application server or a group of servers for a balanced distribution of system load.

For details of the destination parameters, see Entering Destination Parameters [Page 37].

Changing Existing Destinations

On the destinations overview screen (transaction code SM59), the connection types and all existing destinations are displayed in a tree structure.

You can display all information for a given destination by double-clicking it or pressing `F2` on it.

To change an existing destination, double-click it, or place the cursor on it and press the *Change* button.

For details of the destination parameters, see Entering Destination Parameters [Page 37].

Testing Destinations

To test a destination, choose the appropriate function from the *Test* menu.

- *Connection* (also available via the *Test connection* pushbutton)
- *Authorization* (checks logon data)
- *Local network* (provides a list of application servers)

Entering Destination Parameters

In addition to the *RFC destination*, you must enter the following information:

Technical settings

- *Connection type*

  Enter an existing connection type or choose one via the field entry help.
  All available connection types are explained in Types of Destinations [Page 39].

- *Trace*

  Mark the *Trace* option to have the RFC communication logged and stored in a file. You can then display the file, both in the calling and receiving system, using report RSRFCTRC.

- *Load balance*

  If you choose load balancing, you must specify the following information:

  – *Target system* (For a list of available servers, log on to the target system and choose *Tools → Administration, Monitor → System Monitor → Servers*.)

  – *Message server* (Log on to the target system and choose *Control → Control Panel* from the CCMS main menu. It is the server that offers the service M.)

  – *Group* (of servers) (see SAP Logon Group of Servers)

  Otherwise, you must specify the following information:

  – *Target host*

    The name of a server host of the target system that you want to use as a port to the system.

  – *System number*

    Communications service used with the target system. To obtain it, choose *Tools → Administration → Monitor → System Monitor → Servers*.

Security Options

The following options are available only with some connection types:

- *Trusted system* (for type 3 only)

  If the target system is a trusted system, choose *Yes*. For details on trusted systems, see Maintaining Trust Relationships Between R/3 Systems [Ext.].

- *SNC* (Secure Network Communications, available for types 3 and T only)

  If you have an active SNC-supported security system, you can activate additional security options which you must set via *Destinations → SNC options*.

Description

Text description of the entry.

Logon

- *Language*

System language to be used

- *Client*

  Client code

- *User*

  User name to be used for remote logon, if different from current user name

- *Password*

  User password

- *Current user*

  The current user name is to be used for remote logon.

- *Unencrypted password (2.0)*

  If the target system is an R/3 System of Release 2.0, the password must not be encrypted.

The *Attributes* section contains creation and change information.

Types of Destinations

Each destination has a connection-type field (*Connection type*), which tells the kind of system connection:

- **R/2 connections (Type 2)**

  Type 2 entries specify R/2 systems. No further specification is required, i.e. when you create a type 2 entry, you only need to give the host name; all communications information is already stored in the sideinfo table in the SAP Gateway host. You can, however, specify logon information if desired.

  Example entry name: K50

- **R/3 connections (Type 3)**

  Type 3 entries specify R/3 systems. When you create a type 3 entry, you must give the host name and communications service. You can also specify logon information if desired. From R/3 Release 3.0 onwards, you can also specify the load-balancing option if desired.

  From R/3 Release 3.0 onwards, it is possible to specify an application server from the R/3 message server. The application server is then determined according to the load-balancing process. This applies both for RFCs between R/3 Systems and external calls to R/3 Systems.

  Example entry name: K11

- **Internal connections (Type I)**

  Type I entries specify R/3 systems connected to the same data base as the current system. These entries are pre-defined and cannot be modified. The entry names are the same as those used in the SAP Message Server (transaction SM51)

  Example entry name: hs0010_K11_24

  – hs0010=host name

  – K11=system name (data base name)

  – 24=TCP-service name

- **Logical destinations (Type L)**

  Instead of specifying a system connection, type L (logical) entries refer to a physical destination. Type L destinations can also refer to other type L entries. A type L entry uses the information in the "referred-to" entry, and adds further information of its own. Typically, the "referred-to" entry gives the host information, and the type-L entry gives logon data. You can also set a user name, an explicit password, a logon language or an explicit client.

  A type L entry can refer to other type L entries.

  Example entry name: K11_SD or K11_01

  – K11=name of RFCDES entry for R/3 system K11

  – SD or 01: for the fields User='SD_INPUT' or Mandant='001'

Types of Destinations

- **Connections via ABAP driver (Type X)**

  Type X entries specify systems where device drivers in ABAP have been specially installed. When you create a type X entry, you must give the name of the ABAP device driver.

- **TCP/IP Connections (Type T)**

  Type T destinations are connections to external programs that use the RFC API to receive RFCs. The activation type can be either *Start* or *Registration*.

  If it is *Start*, you must specify the host name and the pathname of the program to be started.

  Activation Type *Start*

  The communication method depends on how you select the program location:

  – **Explicit host**

    In this case, the program is started either by the default gateway for the system or by the explicitly specified gateway (gwrd) via remote shell.
    Ensure that the computer with the gateway process can access the specified computer by entering **`/etc/ping <host name>`**.

    In order to be able to start a program on another computer using remote shell, the target system must fulfil certain conditions.

    - The user ID of the gateway process must exist and a file called **`.rhosts`** must also be present in the user's home diretory.
    - The file **`.rhosts`** must contain the name of the calling computer.

    To check this, logon to the computer containing the gateway process with the appropriate user ID and enter the command **`remsh <host name> <program name>`**. The <host name> and <program name> must be the same as in SM59. (If you call an RFC server program without any parameters, the RfcAccept call always returns an error code (RFC_HANDLE_NULL) and the program terminates at once.)

  – **Application server**

    On choosing *Application server* and specifying your program, you can start the program from the SAP application server.

    First, ensure that the program can be accessed from the SAP application server and that the SAP application server has the authorization to start the program.

    To check this, logon with the user ID of the SAP application server (e.g. c11adm). If possible, change to the working directory of the SAP application server (/usr/sap/.../D.../work) and try to start the RFC server program manually from there. (As in the above case, if you call an RFC server program without parameters, the *RfcAccept* call always returns an error code (RFC_HANDLE_NULL) and the program terminates at once.)

  – **Front-end workstation**

    On choosing *Front-end workstation* and specifying your program, you can start the program from the SAPGUI.

    Ensure that you can access the program with SAPGUI.

Ensure that SAPGUI has the authorization to start the program.

To check this, simply call the RFC server program in your environment.

The function call can also be transactional (CALL FUNCTION... IN BACKGROUND TASK DESTINATION...).

Activation Type *Registration*

If the activation type is *Registration*, you have to identify a registered RFC program. With an SAP gateway, an RFC server program can be registered under this ID and then wait for RFC calls from different SAP Systems.

Example entry name: SERVER_EXEC

- **Type M**

  Type M entries are asynchronous RFC connections to R/3 Systems via CMC (protocol X.400).

- **Type S**

  Type S corresponds to type 2, except that the destination is SNA or APPC.

Maintaining Group Destinations

To achieve a balanced load distribution in the target system, you must define a group of application servers as an RFC destination. When processing tasks in parallel, you can use the group destination only in connection with the asynchronous RFC [Ext.].

The resources available on each application server depends on the current system load.

To display and maintain the RFC groups, proceed as follows:

1. From the RFC destination overview screen, choose *RFC* → *RFC groups*.

   You'll see:

   – the names of any RFC groups that have already been defined

   – a list of the instances (host, system and instance number) in your R/3 System

   – the current status (active or not) of each server.

2. To define an RFC group, choose *Edit* → *Create*. Enter a server group name and an instance in the dialog window.

   To add instances to an existing group, double click the group name and enter a new instance in the dialog window.

   By creating duplicate entries, you can assign a server to more than one group. In this case, jobs that use the group will compete for free work processes on the shared server(s).

   Usage examples:

   You could use groups to allow different parallel-processed jobs to run at the same time without competing for the same servers. In this case, the different groups used by the jobs would specify different servers.

   You could also use groups to separate processing from servers on which dialog users are active. In this case, the group used for processing would name servers other than those in the logon groups for users.

RFC Destinations for Synchronous Method Calls

Use

As of Release 4.0 you can also use synchronous interfaces in ALE distribution scenarios.

These interfaces may either be BAPIs or dialog methods. In both cases an object method from the BOR is defined as an API method and implemented by an RFC-enabled function module. This function module is then called by synchronous RFC.

You can specify different RFC destinations for different synchronous method calls.

To ensure compatibility with earlier releases, the standard RFC destination for BAPI calls is also used for calling the following RFC-enabled function modules that are not assigned to an object method in the BOR.

- COHR_ORDER_CONF_MAINTAIN_RFC
- COHR_ORDER_CONF_GET_RFC
- COHR_ORDER_CONF_DETAILS_RFC
- RP_REMITTANCE_ACKNOWLEDGEMENT
- HRTIM_AA_DOC_SHOW

Procedure

You can assign RFC destinations for synchronous method calls in ALE Customizing:

Basis
Application Link Enabling (ALE)
Sending and Receiving Systems
Systems in Network
Synchronous Processing

You can define the following types of RFC destinations:

- The **standard RFC destination for BAPI calls** uses a fixed user ID of type CPIC in the server system and is only available through the authorizations to carry out BAPI calls via ALE. The necessary authorizations for this are non-critical. Unauthorized access with this user ID is not possible.
- The **standard RFC destination for dialog calls** uses a fixed user ID of type DIALOG in the server system and is only available through the authorizations that carry out dialog calls via ALE. The necessary authorizations for this are non-critical. Unauthorized access with this user ID is not possible.

  For further information on dialog methods see Integration of Dialog Interfaces [Ext.] in ALE Programming.
- For **special methods** (BAPIs or dialog methods), which must be protected from unauthorized access, a new RFC destination is created on the client system.
  The client system must be specified as a trusted system on the server system (see remote

communications, Trusted Systems: Trust Relationships Between R/3 Systems [Ext.].
No user ID nor password is specified in this RFC destination. This RFC destination is assigned to the methods to be protected. The current user ID in the client system is used to logon to the server system. The method can only be called if the user has authorization for it (authorization object S_RFCACL).

In some cases a different assignment may be required.

Example

The ALE scenario comprises two systems:

- AC

  There are 200 on-line users in the AC system, among them, one with the user ID CEO.

  The AC system is to call some methods synchronously in the HR system.

  Only the user CEO is to be authorized to make the remote call to the method *Document.Display* in the HR system.

- HR

  There is only one on-line user on the HR system, among them, one with the user ID CEO.

  The following methods are to be called on the AC system:

  – *Document.ReadInfo* (synchronous BAPI, called through RFC)

  – *Document.Check* (synchronous BAPI, called through RFC)

  – *Document.Display* (dialog method, called via RFC by the CEO user only)

User Master Records

In addition to the CEO user, an ALE_AC user is created in the HR system.

ALE_AC is of type CPIC, it is not a dialog user. Users of type CPIC cannot usually start dialog transactions. ALE_AC only receives the authorizations that enable the remote call to the methods *Document.ReadInfo* and *Document.Check.*

RFC Destinations

The profile parameter *auth/rfc_authority_check* for the RFC authorization check is set in both systems. An authorization check is carried out on the function group of the function module called (authorization object S_RFC).

- HR system:

  The AC system is set up as the trusted system in the HR system.

  The system name, client, user name and other optional data is searched in the data supplied by the trusted system and checked against the field values of the authorization object S_RFCACL.

- AC system:

  Two RFC destinations are created on the AC system both with the same application server in the HR system.

  – HR_DOC

The user ID ALE_AC and a password are defined for these destinations.

- HR_BLANK

  No user ID is assigned to this destination.

RFC Destinations for Methods

HR_DOC is specified as the standard destination for synchronous BAPI calls from the HR system.

The RFC destination HR_BLANK is specified for calling the HR system method *Document.Display.*

Result

This procedure produces the following results:

- A well-protected HR receiving system.
- The password of user CEO is not transferred.
- No unnecessary on-line users in the HR system.
- As no standard RFC destination is specified for dialog calls, *Document.Display* is the only method that can be called from the AC system.
- When a release is upgraded, methods cannot be unintentionally called in the HR system. The user authorizations in the HR system prevent this.

Further Information

- With asynchronous BAPI calls, the RFC destination in the partner profile is specified for the associated message type.
- For information on trusted systems and assigning destinations see the documentation *Remote Communications*.
  - Assigning Remote Destinations [Page 34]
  - Trusted System: Trust Relationships Between R/3 Systems [Ext.]

Technology of ALE Business Processes

Definition

ALE business processes are application scenarios that use Application Link Enabling.

Use

The ALE business processes in the standard system cover important application areas in which business functions and processes are distributed. They are preconfigured and are easy to use once the appropriate settings have been made in Customizing for ALE and for the individual applications.

ALE business processes are used for the following distribution tasks:

- Synchronizing Customizing Data Between Systems [Page 47]
- Master Data Distribution [Page 51]

ALE also supports R/2 Connections [Page 66] and Connections to Non-SAP Systems [Page 68].

Automatic tests in the CATT environment can check the quality of the ALE layer and ALE business processes. For further information see Automatic Tests [Ext.].

For more information about the processes see Library of ALE Business Processes [Ext.].

Synchronizing Customizing Data Between Systems

Two important aspects:

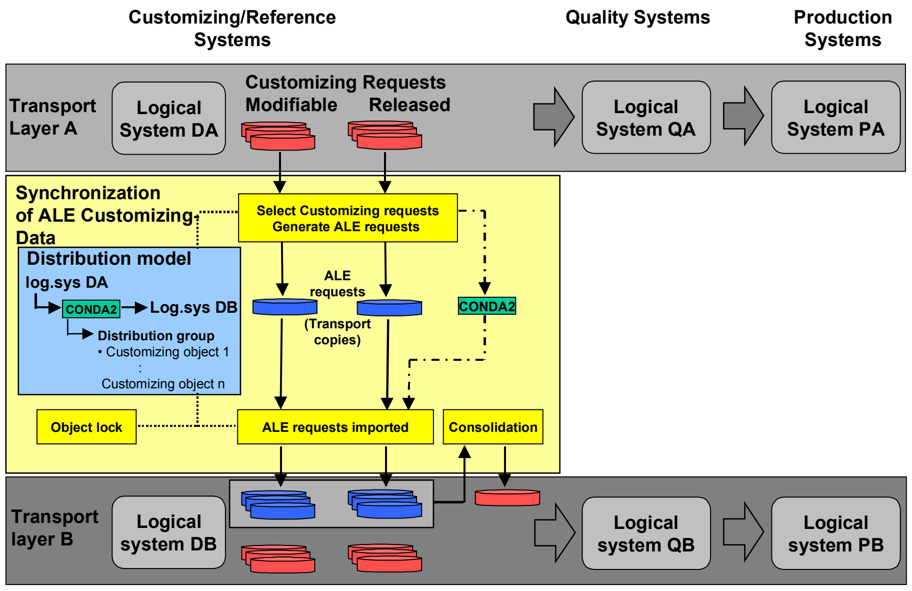
- Which Customizing data has to be synchronized across the different systems?
- How is the synchronization performed?

Customizing data is all the data configured in Customizing.
It includes, for example, organizational units (company code, division, plant), units of measurement and many other parameters that have to be set in the system.

Customizing has to be identical in certain areas in both systems before data can be exchanged between them.

In an ALE integrated system you can specify a central maintenance system in which Customizing data objects can be maintained centrally. Customizing data is transported from the central system to the systems that need to be synchronized.

Synchronization of ALE Customizing Data



The synchronization of Customizing data can be divided into:

- Modeling

  The ALE distribution group is a central modeling element. ALE distribution groups contain client-dependent Customizing objects of category CUST.

  For your own Customizing tables you have to make entries in each of the administration tables of the Customizing objects (Transaction SOBJ). These entries are made automatically, if you create a maintenance interface with the table maintenance generator for your Customizing tables (*Tools* → *Development* → *Dictionary* → *Utilities* → *Table maintenance generator*, or transaction code SE11).

In ALE Customizing you have to define ALE distribution groups in the same system that is specified as the maintenance system of the distribution group:

Basis → Application Link Enabling (ALE)
Modelling and Implementing Business Processes
Synchronization of Customizing Data

In Releases before 4.6A distribution is modeled with the message type CONDAT. Customizing data (control data) for distribution must be specified as the filter objects of this message type.

You cannot use the old and new modeling processes in parallel.

For this reason you have to use a tool in the ALE IMG to generate distribution groups for existing Customizing objects.

You can specify a different maintenance system for each distribution group.

- Transport management

  In ALE Administration you can process ALE transport requests for synchronizing Customizing data between systems as follows:

- In the sending system:

  Generate ALE requests

- In the receiving system:

  Import ALE requests with Workflow connection

  Consolidate and forward ALE requests

Synchronizing Customizing Data

The set of Customizing data objects to be synchronized in a specific distribution depends on the ALE business process used in that distribution.

Another important factor is your company's "distribution philosophy". "Centralizers" usually prefer to maintain too much data centrally rather than too little, whereas "individualists" tend to steer away from central control.

SAP provides you with two methods for comparing Customizing objects:

- System-wide synchronization of Customizing objects

  This is in ALE Administration under (*Services → Customizing data → Customizing Cross-system viewer*)

  If one of the participating systems has got a Release status earlier than 4.6A, you have to use an object synchronization tool in the ALE Implementation Guide. (*Modeling and Implementing ALE Business Processes → Synchronization of Customizing Data → Modeling Before Release 4.6A (with CONDAT) → Check Consistency of Customizing Data Distribution*. Then run a Repository report to get a list of Customizing data objects to be synchronized.

- For every setting you make in the Implementation Guide, you can find out from the system what the associated customizing data object is.

When you are ascertaining the control data objects, take account of the following:

- You should maintain the entire organizational structure of your company centrally if possible.
- You should consider distribution scenarios that you may want to use in the future as early as possible in your planning. Once systems have drifted apart, it is quite difficult to get them back into line.

Once you know which customizing data objects need to be synchronized across the different systems, you have to decide how they will be distributed.
You will probably find that you end up with a heavily centralized view of the system – in other words, you will have a central maintenance system for all the customizing data that has to be synchronized across the systems.

The Distribution Process Before 4.X

To distribute Customizing data, filter objects for the CONDAT message type are entered in the distribution model. When you maintain the Customizing data distribution in the distribution model and then distribute the model:

- The Correction and Transport System distributes the customizing data from the maintenance system to the receiver systems specified in the distribution.
- The maintenance of customizing data objects is locked in the systems specified as the receivers of the customizing data objects. Only those customizing data objects to be distributed to a specific system are not affected.

Maintaining Customizing Data

When you maintain Customizing data, your changes will be recorded by the Correction and Transport System. A transport request is required for this. After you have maintained the Customizing data, the change request is released and is ready for transport.

When the transport request is released, the ALE distribution model is checked to ascertain whether other systems need to know about the changed customizing data objects. If so, Transaction BD77 is called with the parameter transport request and a list of all the objects is displayed. The transport request is then processed further in one of two ways:

1. A transport request is generated for each system, containing the exact set of objects that are defined by the intersection of the distribution model (message type CONDAT) and the request just released. This transport request must then be released and transported.
2. An IDoc for the message type CONDAT can be sent to each system, which then generates a work flow item in the target system. The person processing the work flow item can display a list of all these objects (the same as with Transaction BD77) and can start Transaction SM30 or SCUO to start the standard tables synchronization for each object. The synchronization must be carried out manually.

We recommended you choose the first option and to notify the administrator of option 2 so that he/she does not have to synchronize the tables manually.

Master Data Distribution

Rather than distributing the complete master data information, views of the master data can be distributed (for example, material sales data, material purchase data). Each view of the master data is stored in a separate message type.

Users can specify which data elements in a master record are to be distributed.

Various distribution strategies [Page 52] are supported:

- Cross-system master data can be maintained centrally and then distributed. The final values are assigned locally.
- A view of the master data can be maintained locally. In this case there is always one maintenance system for each view. The master data is transferred to a central R/3 System and distributed from there.

Types of Distribution

- Active distribution (PUSH)

  If the master data is changed (for example, new data, changes or deletions), a master data IDoc is created in the original system and is distributed by class as specified in the distribution model.

- Requests (PULL)

  A request occurs when a client system needs information about master data held in the system. You can select specific information about the master data, for example, the plant data for a material.

  If you want to be notified of all subsequent changes to the master data, this has to be set up “manually” between the two systems. It is not yet possible for this to be done automatically in the distribution mechanism in the original system.

Transferring the Master Data

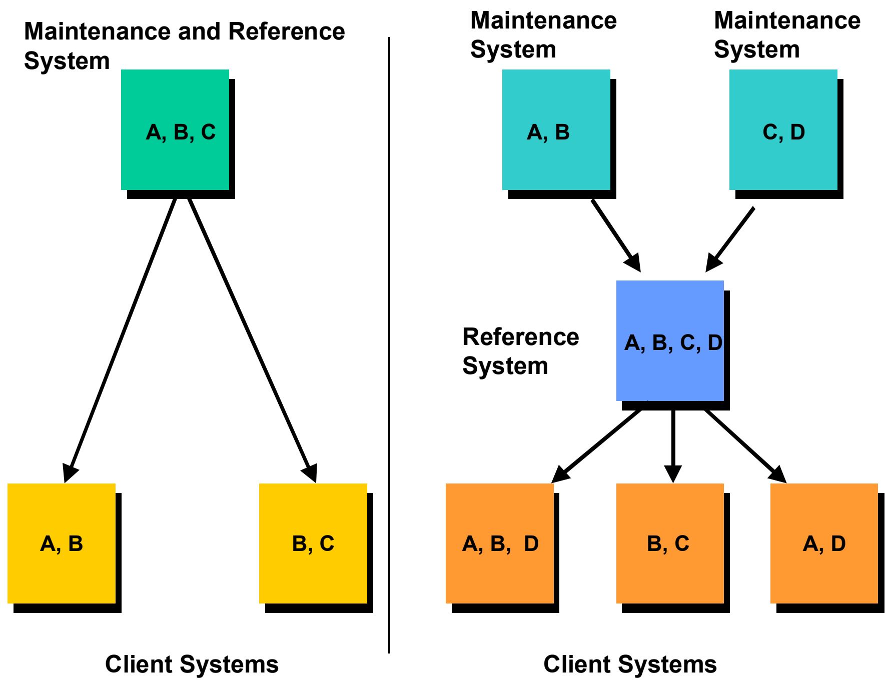
A distinction is made between transferring the **entire** master data and transferring only **changes** to the master data.

If the entire master data is transferred, a master IDoc is created for the object to be distributed in response to a direct request from a system. Only the data that was actually requested is read and then sent. The customer specifies the size of the object to be distributed in a request report.

If only changes are transferred, the master IDoc is created based on the logged changes.

Ways to Distribute Master Data

The graphic below is an example of how master data is distributed:



The graphic on the left shows the simplest case:

- All the master data is created in a central system and is then distributed to the local systems.

  For example, the materials A, B and C are created centrally; A and B are then distributed to one decentralized system, whilst B and C are distributed to another decentralized system.

The graphic on the right shows a hierarchy of three layers:

- Master data can be created in several different systems. A central R/3 System is used as a reference point for the other systems.

  For example, the materials A and B are created in one system, whilst C and D are created in a second system. In each case the materials are also distributed to the reference system. They are distributed to the client systems.

In principle, master data can be freely distributed between all the R/3 Systems. A central R/3 system is not necessary, but it does make the distribution of the master data easier.

Distributing Master Data Using Classes

Classes are used in master data management when distribution rules cannot be mapped across organizational units. There are classes for material, customer and vendor master data.

As cost centers and G/L accounts are not classified, they cannot be distributed in classes. Since the quantities of data in these cases are small, all the data can be distributed to all the systems. You can also filter the data by organizational unit.

Classes can be used to define the master data objects which a given system needs to know about.

The Hamburg system needs to know all the purchasing data in the WHOLE class (range of wholesale goods) and also all the sales data in the SOUTH class (materials sold in the south of the country).

For each object type a (customer-defined) class type can be specified as relevant for ALE. All classes belonging to this class type can be used.

Classes are maintained in master data transactions. To add a master data object to a class, you simply classify it.

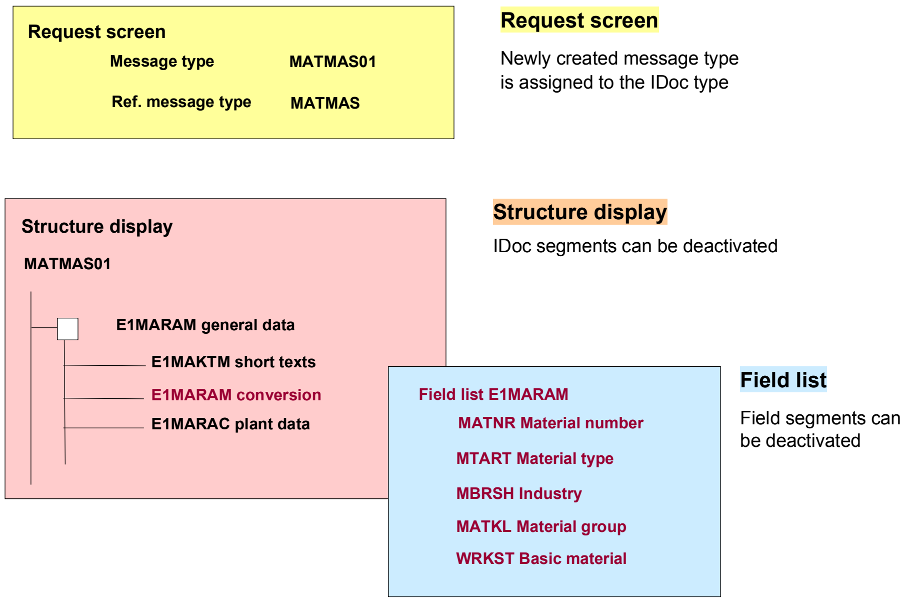
Master Data IDocs

Maximal IDocs

A maximal IDoc specified by SAP contains all the information required for distributing an object. The structure is similar to the R/3 table structure.

Creating a Reduced IDoc

A reduced IDoc can be created from a maximal IDoc.



The structure of the IDoc is displayed in one of the transactions started from ALE Customizing. You can deactivate segments and fields. Mandatory fields cannot be reduced.

When a reduced IDoc is created from a maximal IDoc, a new message type is created and assigned to this IDoc. The reference to the message type of the maximal IDoc stays the same (for example, the new message type for material is MATSMD and the reference message type is MATMAS.)

This IDoc with the new message type is now used to transfer data rather than the maximal IDoc. It is automatically assigned to the change documents.

When sending IDocs of type MATMAS02 , the E1MKALM segments (production version) are filled and sent, but are ignored by the receiving R/3 System. This data may be used by non-SAP systems without any restrictions.

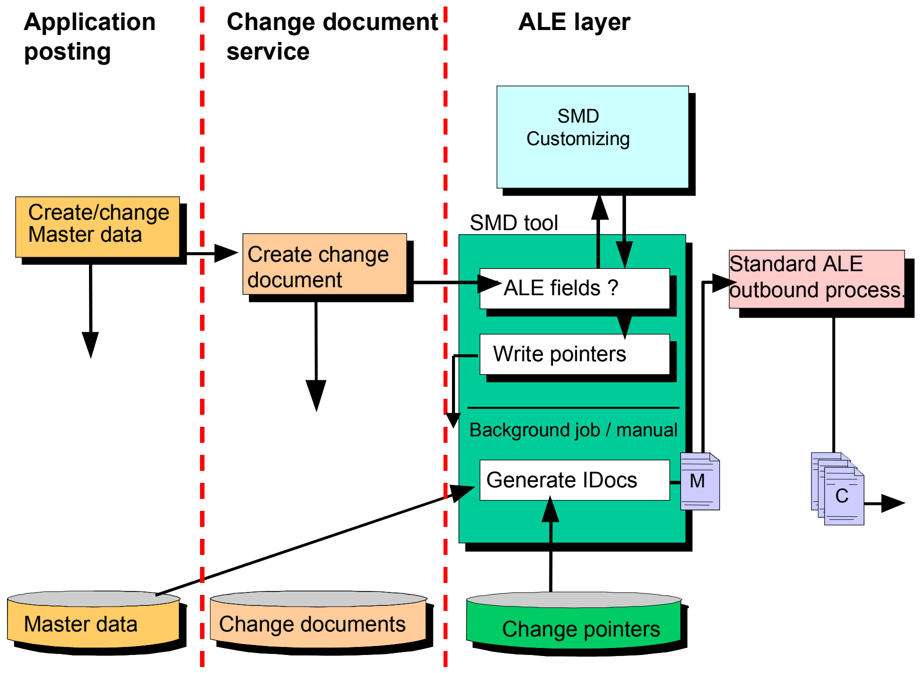
If short texts or long texts belonging to master data are changed, they are distributed and posted in the receiving system. Deleted texts are not transferred.

Distributing Master Data with the SMD Tool

Changes to master data objects are managed using the Shared Master Data (SMD) tool. The SMD tool distributes master data changes to the decentralized systems.

The SMD tool groups changes together according to their content and the time they were changed:

- Firstly, if several changes to a master data object have been made by different R/3 transactions and stored in different tables, they are combined into one single change to the master data object.
- Secondly, if several changes are made in close succession, they are also combined into a single change and distributed as one change.

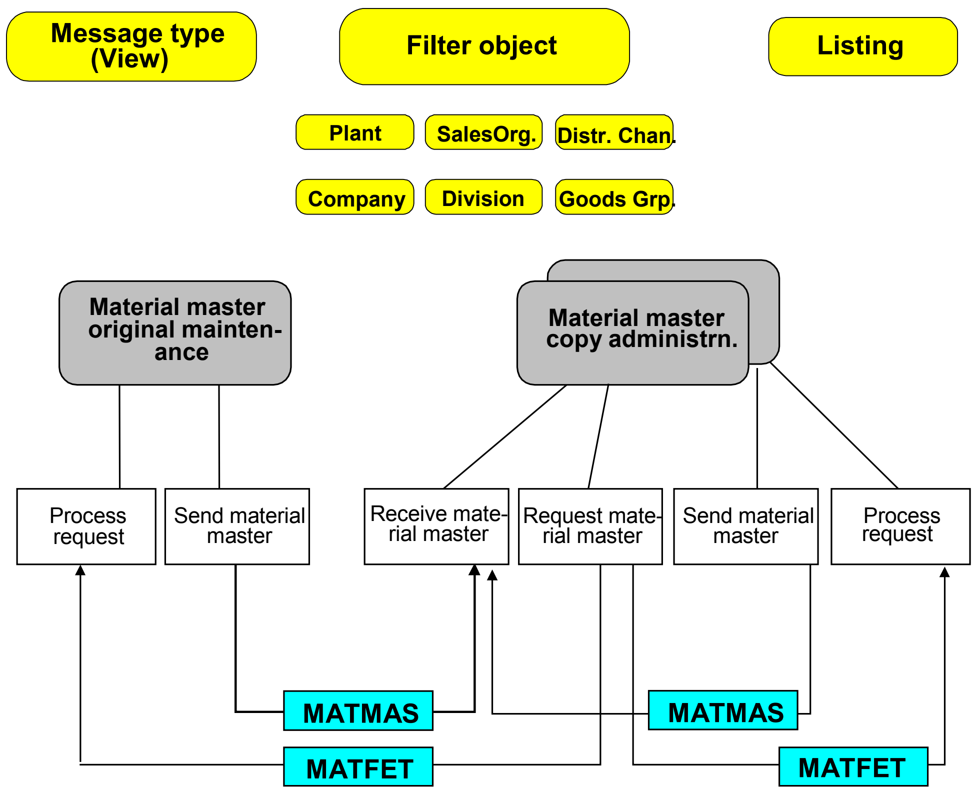


The SMD tool is connected to the change document interface. If the master data changes are to be distributed, the application writes a change document. The contents of this are passed to the SMD tool. The tool writes change pointers, reads the application data and creates the master IDoc.

The master IDoc is then passed to the ALE layer, which sends it to all interested systems.

Example of Master Data Distribution

The material master maintenance is used here as an example to explain how master data is distributed. All the scenarios for master data distribution are constructed in a similar way to the material master data distribution.



The material master maintenance and material master maintenance (copy) function types are located in distributed systems.

If maintenance is central, copy management is performed directly in the decentralized systems. If there are several maintenance systems (for example, one for each view of the material), copy management is handled in the reference system. The reference system then distributes the master data changes to all the other decentralized systems.

Messages can be sent in one two ways:

- If PUSH type distribution is used, data is exchanged using the MATMAS message type. (This message contains the material master data to be distributed).
- If PULL type distribution is used, the MATFET message type is sent from the material master copy management to the material master maintenance original. (This message is a request to the original to send material master data.)

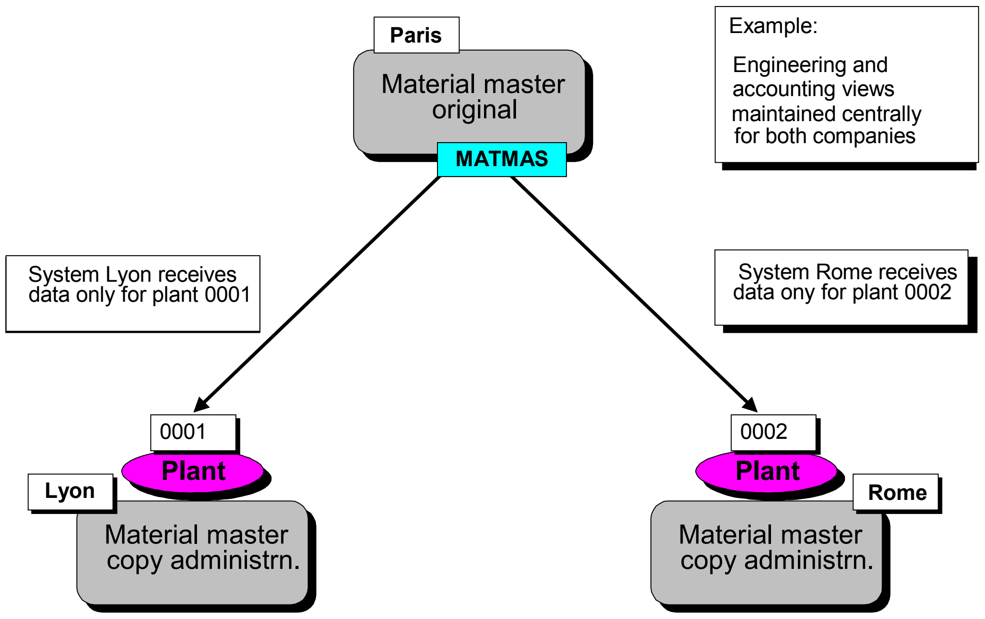
The following filters control the distribution:

- Classes [Page 53] (for example, distributing classified material numbers to specific systems).
- Filter objects: organizational units (sales organization, plant).

Example of Master Data Distribution

As part of the ALE Customizing process the customer specifies in the distribution model which fields in which application table are relevant to the distribution for each message type.

Distribution Model - Material Master Data Distribution. The “plant” filter object is used in the graphic above. Since plant 001 is managed in the LYON system, LYON receives the material master data for plant 001.


Paris
Material master original
MATMAS
Example:
Engineering and accounting views maintained centrally for both companies
System Lyon receives data only for plant 0001
System Rome receives data ony for plant 0002
0001
Plant
Lyon
Material master copy administrn.
0002
Plant
Rome
Material master copy administrn.

Checking Whether Distribution Is Required

The contents of the change document are passed from the change document system to the SMD Tool. For each document the system checks that the modified field and the table are assigned to a message type.

Writing Change Pointers

If the fields are required for distribution, the SMD tool writes change pointers and stores them in the BDCP and BDCPS tables. Change pointers are basically the key fields of the table that contains the changed field. In ALE Customizing, customers can specify the fields that need to be distributed.

Change pointers are created only if both ALE and the message type are active.

Sending Changes

To distribute changed master data change pointers have to be processed. Depending on the message type, the program RBDMIDOC creates the master IDocs and passes them to the ALE layer for dispatch.

For this you should use program RBDMIDOC. RBDMIDOC is usually automatically executed in the background. A background job should be scheduled for each message type.

Depending on the settings in the partner profiles, it may be necessary to send IDocs directly by executing the program RSEOUT00. The background job that executes the RBDMIDOC can do this in a second step. For more information about maintaining the distribution model see the R/3 Implementation Guide.

Basis
Application Link Enabling (ALE)
Modelling and Implementing ALE Business Processes
Master Data Distribution
Replication of Modified Data
Creating IDocs from Change Pointers

When creating a master IDoc ensure that:

- The master IDoc contains all the IDoc segments, whose fields have changed.
- The master IDoc contains all the mandatory IDoc segments
- The parent segments of these segments are dispatched
- IDoc segments are sent complete, in other words, data must be entered in all fields.

Notes on Reading Changes

When inbound master data is processed, you can prevent the contents of sender fields overwriting fields in the receiver system. You can find out how to make this setting in the IMG:

Basis
Application Link Enabling (ALE)
Modelling and Implementing ALE Business Processes
Converting Data Between Sender and Receiver

For IDocs with message types MATMAS, DEBMAS or CREMAS and all other IDocs which use CALL TRANSACTION for inbound processing, no fields will be overwritten if the constant “/” is specified in the IDoc field.

Distributable Master Data Objects

The following master data objects can be distributed:

- Change numbers
- Article master record
- User master record
- Purchasing info record
- Business process
- Classification, class and characteristics
- Conditions
- Cost center
- Cost center group
- Cost element group
- Cost element
- Customer master record
- Activity type
- Activity master record
- Activity type group
- Vendor master record
- Material master record
- Unit of measure for cost center and cost element combination
- Source list
- Human resources: HR master data, organizational data
- Profit center
- G/L account
- Bill of material (materials and documents)
- Activity price of cost center and cost element combination

Some of these master data objects are described in more detail below:

Change Numbers

If a change number is distributed using the message type ECMMAS, it contains:

- Change master record (in long text if available)
- Alternative dates
- Validity

- Object types for change master record
- Object management records (in long text if available)

If the distributed change number does not already exist in the target system, it is created here. It is not yet possible to delete a change number in the target system nor to distribute it using a change pointer.

Article Master Record

For further information about distributing articles refer to the topic below in the document *ISR - SAP Retail:*
Article: Distributing Master Data [Ext.]

User Master Record

For further information about the central administration of user master data, refer to the ALE Implementation Guide (Transaction SALE).

Modelling and Implementing Business Processes
Predefined ALE Business Processes
Cross-Application Business Processes
Central user administration

Purchasing Info Record

When Purchasing Info Records [Ext.] are distributed in the message INFREC the following data is transferred:

- General data and texts
- Purchasing organizational data and texts for purchasing organizational data

The following data is not distributed:

- Customs tariff preference data
- Order price history

The prerequisite is that the master records of the vendors and materials in question have already been distributed

The number ranges for purchasing info records need to be synchronized system-wide.

- When the IDoc for purchasing info records of a stock material is posted, if the purchasing info record for the material and the appropriate supplier does not already exist in the receiver system, the number of the purchasing info record is copied from the central system. Otherwise, the existing purchasing info record is updated.
- For purchasing info records for non-stock materials, the purchasing info record number is always copied from the central system. If a purchasing info record with the same number already exists, the system checks whether it has the same supplier and material group as the info record from the central system.

When a purchasing info record is posted, its conditions are not changed. However, the NETPR (net price) and EFFPR (effective price) fields are updated. In other words, the values are copied from the central system.

Conditions for info records must be transferred separately using the standard conditions distribution (message COND_A).

If the *PLANT* filter object is used in modeling the distribution of purchasing info records, the plant field must contain the value SPACE when you are working with purchasing info records which deal with more than one plant.

If the *Material* or *Material listing* filter objects are used for modeling the distribution of info records, the *Material* field must contain the value SPACE when you are working with purchasing info records for non-stock materials.

The *supplier listing* filter object cannot be used in the distribution of purchasing info records. Only material listings are allowed to be used as filter objects in the distribution of purchasing info records.

SAP provides the following enhancements:

- MMAL0003: ALE purchasing info record distribution: Outbound Processing
- MMAL0004: ALE purchasing info record distribution: Inbound Processing

For information on purchasing info records refer to the documentation *MM - Purchasing* in Purchasing Info Records [Ext.].

Business Process

For further information about distributing business processes refer to the documentation *CO - Process Cost Accounting* under the topic:
Master Data in Process Cost Accounting [Ext.]

Classification

For the message type CLFMAS the dependencies of other message types, such as MATMAS and CREMAS must be maintained in different filter groups.

Conditions

The message type COND_A is used to distribute Conditions [Ext.].

Fields which are reduced are set to initial in the receiver system. If you reduce whole segments, these are deleted in the target system.

Conditions with use E (= bonus, conditions for later calculation) can only be distributed manually and not with the change document system.

Cost Center

When cost centers are distributed, the field CV_OTYPE for the joint venture object type is not distributed with them.

Customer Master Record

Addresses of contact persons are not distributed.

Long texts are not included in the change document system.

Activity Master Record

When Activity Master Records [Ext.] are distributed, the following data is transferred with the message:

- Activity basic data

- Short texts (multi-language)
- Short texts (multi-language)

The activity conditions must be distributed separately with the message COND_A.

If you want to distribute activity contracts or maintain the same master data centrally so that you can use it in other R/3 Systems, then you should distribute activity master records.

Activity master records or any changes made to them are distributed with the report RBDSESRV or with the SMD tool.

For more information see the documentation *MM - Services* in Service Master Records [Ext.].

Vendor Master Record

When distributing vendor master data, reminder data is posted in inbound processing only to the default dunning area.

Material Master Record

Changes to the material type of a material cannot be distributed. The material type of a material (field MTART) can only be transferred to the receiving R/3 System, if you create a material, not if you change a material. A single material may therefore have different material types in different systems.

The production version segment (E1MKALM) is only used in outbound processing. You cannot import the information into an R/3 System. This information can, however, be processed by non-SAP systems.

For further information about distributing material master records see the SAP Library in the documentation *LO - Administration of Material Master Data* under IDoc Types for Distributing Master Data [Ext.].

Source List

The message SRCLST is used to distribute Source List Records [Ext.].

The prerequisite is that the master records of the vendors, materials and possibly contracts have already been distributed

Source list records referring to a scheduling agreement cannot be distributed, because scheduling agreements cannot be distributed using ALE.

If you use the *purchasing organization* filter object when modeling the distribution of source lists, the supplier field must contain a space if you are working with source lists which span more than one purchasing organization.

If you use the *Supplier* filter object when modeling the distribution of source lists, the supplier field must contain a space if you are working with source list records which are supplier-independent.

The *supplier listing* filter object cannot be used in the distribution of source lists. Only material listings are allowed to be used as filter objects.

SAP provides the following enhancements:

- MMAL0001: ALE source list distribution: Outbound Processing

- MMAL0002: ALE source list distribution: Inbound Processing

For more information see the documentation *MM - Purchasing* in Maintaining the Source List [Ext.].

Human Resources: HR Master Data, Organizational Data

For more information see Master Data Distribution (Human Resources) [Ext.].

Profit Center

Statistical key figures are not distributed.

Master data can only be maintained in the local system of the controlling area.

G/L Account

The EXTERNAL ACCOUNTING INFORMATION field in the company code segment is only included if new G/L accounts are created. Changes to this field are not posted. For security purposes, you have to do this manually.

Bill of Materials

Bills of material (BOM) are distributed with all items valid on the key date and, if applicable, with local object dependencies. Only simple bills of material can be distributed.

The distribution of bills of material does not include:

- Long texts for headers, alternatives and items
- Sub-items
- Global object dependencies these must exist in the system
- BOM history - only the status of the BOM on the key date is distributed.
- the BOM item objects, for example materials, documents and classes. These objects must be distributed separately from, and before, the BOM.
- If applicable, the change number. The change number must exist in the system.

If you send bills of material directly, the user receives a list of all bills of material selected by the system which can be manually edited afterwards. Because only simple BOMs can be distributed, the user cannot select relevant alternative and variant BOMs, plant assignments and material variants for distribution.

You can overwrite the selection date displayed on the selection screen in the detail list. You can also replace it with a change number. For maintenance in the target system, you can also give a valid-from date or a change number (engineering change management). If you do not change it, the valid-from date or change number is copied from the sending system.

Here are the message variants for message type BOMMAT:

- CNG (change, standard): Changes the BOM items.

  The BOM header data remains unchanged. It creates the BOM in the target system, if it is not already there. This is the default if you do not select a message variant.

- CRE (create): Creates the BOM.

If the BOM already exists in the target system, the IDoc is not posted.

- DEL (delete): Deletes the BOM.

  You should use this function with the utmost caution, since the deletion of the BOM also deletes references from other applications (such as routing and production orders).

Changes to simple material BOMs or BOM items can also be distributed with the SMD tool. This way you can identify items in the target system using the fields, item category, item number, sort string and object. The object field is dependent on item category material, document data and class data.

R/2 Connections

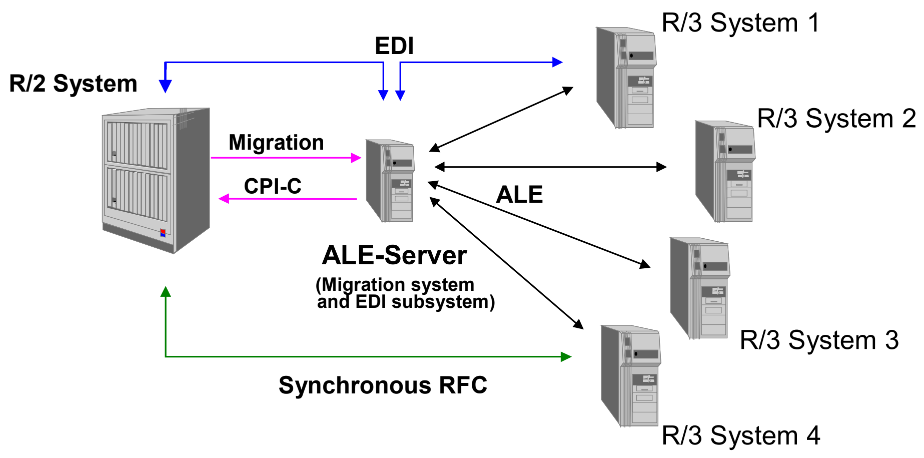
Distributing Master Data From R/2 to R/3

Migration tools are used to distribute data between R/2 and R/3 Systems. These tools are necessary because a rather complicated set of rules is required to map R/2 data structures to R/3 data structures.

These migration rules are used to control the distribution across an ALE server to an R/3 System. (The migration guide contains further details.)

The distribution process can be broken down into two stages. The first stage consists of a permanent migration of the objects to be distributed from an R/2 system to an R/3 System (the ALE server), whilst the second stage is the distribution of the data from the ALE server to one or more target R/3 Systems.

Distributing Data Between Coupled R/2 And R/3 Systems



The ALE server converts the data format and transfers the data between the host system and the R/3 instances.

Distributing Transaction Data From R/3 To R/2

When distributing data in the reverse direction, the data to be sent to the R/2 System is addressed to the R/2 System. (specify the R/2 System as the logical system.) The data is first transferred to the ALE server via the port specification.

Firstly, carry out the normal inbound processing steps in the ALE server.

A comparison between the target logical system and your logical system shows that the inbound IDoc has to be sent to the R/2 System:

- The new port is determined.
- The IDoc is placed in the outbox.
- The dispatch function module is called. It transmits data to R/2 via CPI-C.

- The R/2 System recognizes the data as an inbound IDoc.
- An application module posts the IDoc in R/2.

The data is imported into the R/2 System via appropriate interfaces. These could be batch input interfaces. Alternatively, if an application has an adequate number of EDI interfaces in both directions (usually the case for document data), then these may be used for the distribution.

R/2 System Release 5.0F or later is required for the EDI communication between R/2 and R/3.

Connections to Non-SAP Systems

ALE is not restricted to communication between SAP systems, it can also be used for connecting R/3 Systems to non-SAP systems.

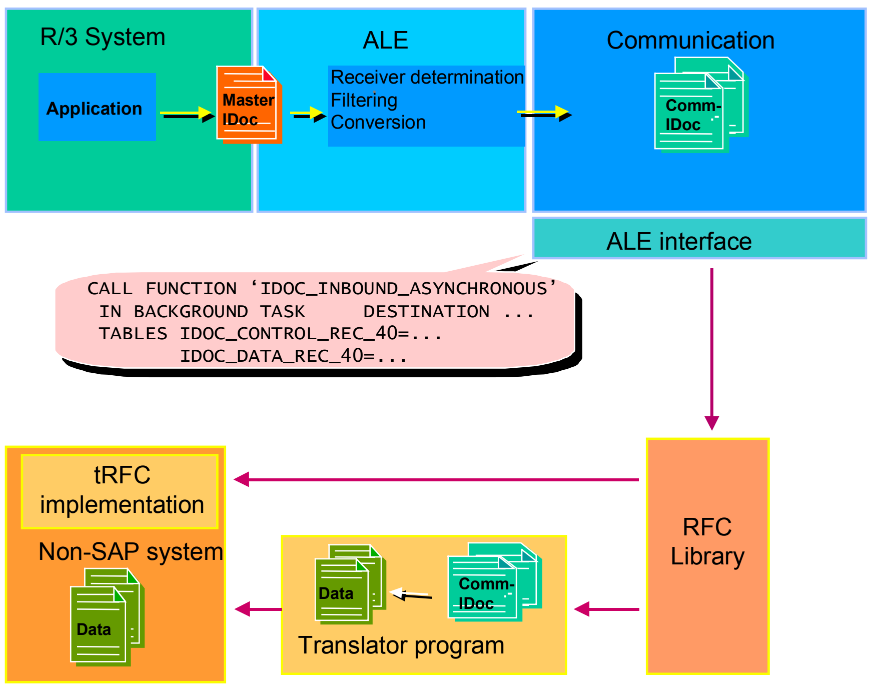
By using IDocs as universal information containers, ALE can reduce the number of different application interfaces to one single interface that can either send IDocs from an R/3 System or receive IDocs in an R/3 System.

SAP certified Translator Programs [Page 70] can convert IDoc structures into customer-defined structures.

Alternatively, the RFC interface for sending and receiving IDocs can be used in non-SAP systems.

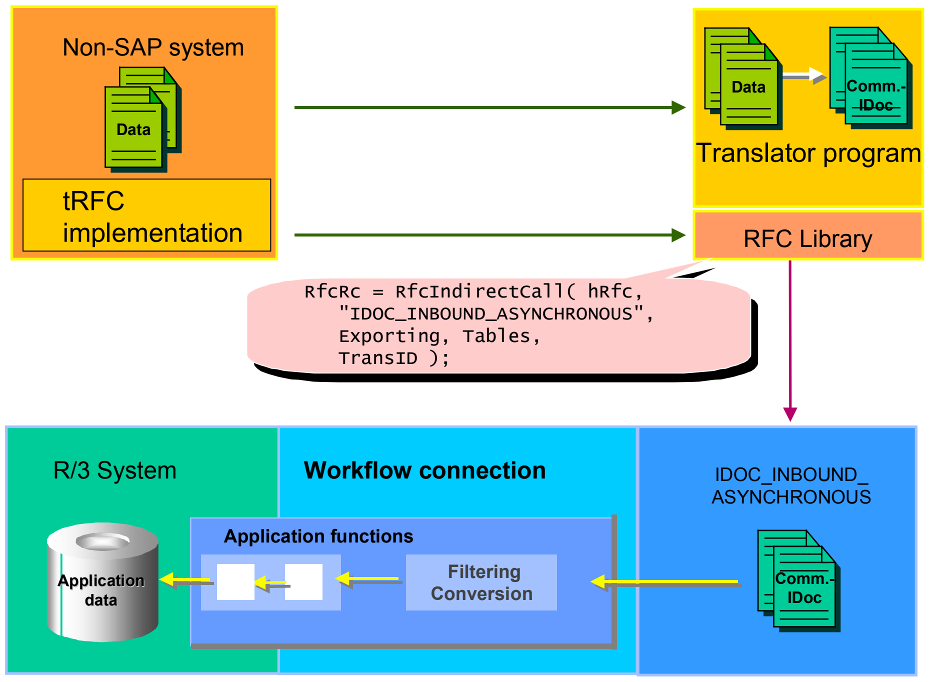
In both cases you need the *RFC Library* of the *RFC Software Development Kit* (RFC-SDK).

Communication from an R/3 System to a Non-SAP System



```
CALL FUNCTION ‘IDOC_INBOUND_ASYNCHRONOUS’
 IN BACKGROUND TASK      DESTINATION ...
 TABLES IDOC_CONTROL_REC_40=...
        IDOC_DATA_REC_40=...
```

Communication from a Non-SAP System to an R/3 System



```
RfcRc = RfcIndirectCall( hRfc,
    "IDOC_INBOUND_ASYNCHRONOUS",
    Exporting, Tables,
    TransID );
```

You can find an example of an IDoc interface to non-R/3 Systems in the documentation Interfaces to Link Mobile Data Entry and Warehouse Control Unit [Ext.].

- For information on the technical implementation see ALE Programming Guide [Ext.].
- You can find the requirements for the certification of interfaces in SAPnet under **http://www.sap.com/csp/scenarios**.
  Choose *Cross Application*, *CA-ALE* and *CA-AMS.*

Translator Programs for Communication

Definition

Translators programs are used to connect non-SAP systems to ALE. They must be certified by SAP.

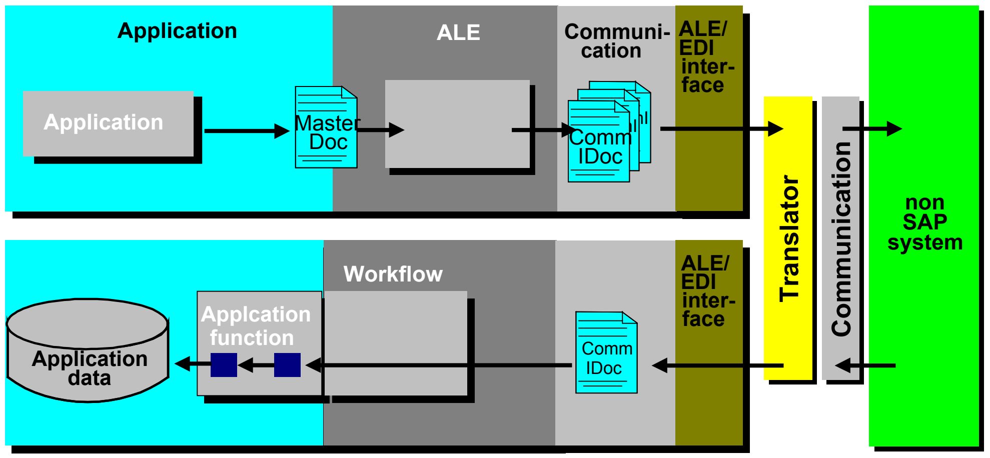
Use

Translators are typically used for:

- Mapping IDocs to any structure required in non-SAP systems
- Controlling communication such as establishing and restarting connections.

Structure

Using A Translator Between R/3 Systems And Non-SAP Systems



Integration

Translators are supplied from external vendors. SAP certifies the programs to ensure that communication between the ALE interface and the translator is functioning correctly.

The following criteria is checked for the certification:

- Can the translator automatically copy the IDoc structures into its own repository?
- Can the translator take an IDoc from an R/3 System and interpret the information based on its repository data?
- Does the translator have adequate mapping functionality?
  - Can the translator pass the IDoc created back to R/3?

The certification itself does not evaluate the functions provided in the program.

Administration of ALE Functions

Purpose

Application Link Enabling provides a range of administration tools:

- Monitoring of IDoc processing and tRFC communication
- Transport of Customizing data from maintenance systems to receiving systems
- Optimization of ALE performance
- Serialization of messages

You can find the administration functions under *ALE Administration.*

Information about optimizing the performance of ALE often refers to settings in R/3 Customizing. You can find ALE Customizing under *Basis* → *Distribution (ALE)*

In the sections below you will find general information, step-by-step procedures and information about the settings required in Customizing for ALE.

Monitoring Message Exchange

In *ALE Administration* you can use various tools to monitor Messaging [Page 10].

You can display an overview of the communication and the processing status of IDocs.

You have to make the settings required for monitoring in Customizing for ALE. The sections below contain details of the procedure.

Central Monitoring Using the ALE CCMS Monitor

Use

You can monitor several R/3 Systems in an ALE integrated system at the same time. You use the CCMS Alert Monitor [Ext.].

The alert monitor gives you an overview of the following R/3 System performance attributes which are important for ALE:

- IDoc change pointers
- Processing of outbound IDocs
- tRFC queue
- Processing of inbound IDocs

You can monitor any number of R/3 Systems from any one system. The number of systems that can be monitored is, however, restricted by technical factors, such as the speed of the network and the level of traffic in the network. From our experience such restrictions only present a problem in very large integrated systems.

Integration

The ALE monitor sets up a connection to the CCMS monitoring. Data suppliers provide the CCMS identification numbers. These data suppliers are integrated in the CCMS (by Customizing).

ALE monitoring objects are defined in the ALE Customizing.

Prerequisites

You have defined the ALE Monitoring objects in Customizing for ALE (*Basis Components → Distribution (ALE) → Monitoring Systems → Central Monitoring of All Systems*).

For analysis purposes, ALE monitoring objects form a group of associated selection options based on IDoc attributes.

Individual objects are assigned values based on the current system status and the assignment of selection options from IDoc attributes.

Activities

To check the current status and the open alerts of your R/3 System:

In ALE Administration choose: *Monitoring → ALE Monitor in CCMS*.

You can identify the current status by the performance values and status messages recently forwarded to the alert monitor. Older alerts that are still open (not yet processed) are not included in the color coding.

Follow these steps:

1. Check the color coding in the monitoring tree.

   The color of nodes or MTEs (monitoring tree elements) mean:

   Green: The component is running normally. All is OK.

Yellow: A warning message has been issued - a “yellow alert”.

Red: A problem or critical status message has been issued - a “red alert”.

Gray: No data available for this node. (Check in the CCMS in the *Self-monitoring* monitor why the *Collection tool* is not available for this node).

Note: To display the legend of colors and symbols used in the alert monitor, choose Extras → *Legend.*

The alert monitor passes the highest alert level in the monitoring tree to the highest node. For example, if the node with the name of your R/3 System is green, this means that all the components in the monitoring tree of this system have a “green” status. All is OK.

To start the analysis method, select a node. The analysis method displays details of the current status of the node.

You can select the display option, **automatic refresh.** Choose *Extras* → *Display options.* Select the *General* register. In the group box, *Refresh display,* select the option *Yes* and enter an updating interval. Recommended value: 300 seconds or longer

If the automatic refresh is switched off, the alert monitor displays the data available when it was started.

2. Check what has been going on in your system.

   In the standard toolbar select *Open alerts.*

   Instead of the current system status, the color marking shows where there are open alerts in the system. Open alerts are alerts that have not yet been analyzed and set to *Completed.*

   When you start your work in the morning or return to your workplace after the lunch break, you can check in the *Open Alerts* view, whether any problems occurred in the system during your absence. The monitor records all alerts, even if the status of the alert has been corrected in the meantime.

3. Respond to an alert.

   In the monitoring tree, yellow entries mean warnings and red entries mean errors:

   To deal with them:

a) Select *Open alerts* to switch to the alert display, provided it is still active.

   The monitor now displays the number of alerts for each monitoring tree element (MTE). The most important alert messages that still need to be resolved are also displayed.

b) Select a yellow or red monitoring tree element and select *Display alerts*.

   The systems opens the alert browser and display the open alerts. The browser contains all the alerts in the tree node you selected. Move the cursor further up the monitoring tree to display a wider range of alerts. Select an MTE on the lowest level of the display to display only those alerts that affect this MTE.

c) Analyze an alert.

Each line in the alert browser displays an alert message (far right) as well as details of the alert.

The browser provides two further sources of information. Select an alert checkbox and the functions (using icons) below:

- **Start analysis method**
  This starts the problem analysis transaction or similar tool associated with the alert.
- **Display details**
  The system shows details of the monitoring tree element. This includes the latest values or status messages, alert threshold values and performance data for the last entry time (only for performance monitoring tree elements) You can display the data graphically by selecting the relevant line and *selecting Display performance values graphically.*

4) When you have dealt with the alert, you can change its processing status.

   Once you have analyzed the problem and either corrected it or are sure that it is safe to ignore it, select the alert and select *Alerts completed*.
   The alert monitor deletes the alert from the list of open alerts.

Further Functions

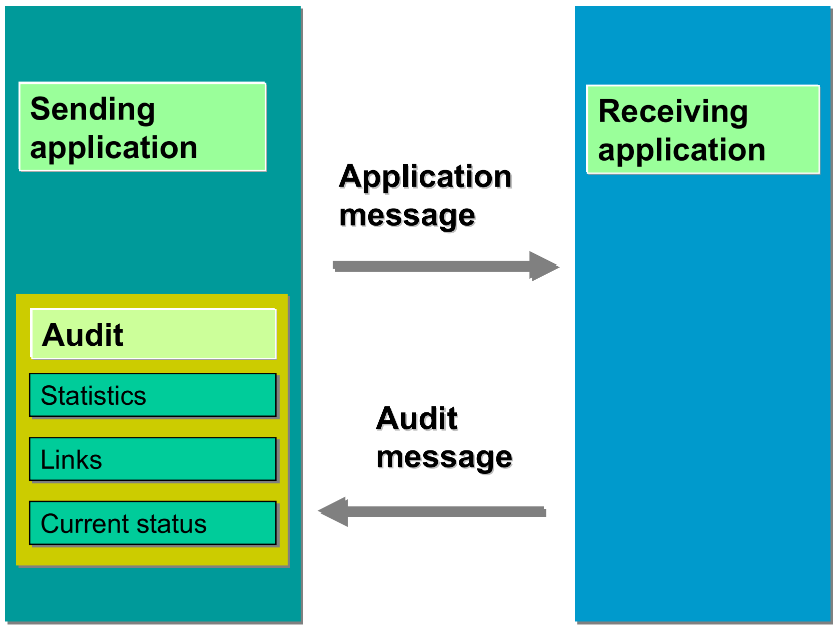
The topics below describe how you can configure the alert monitor to your requirements and how you can use more of its functions:

- Monitoring: How-To Instructions [Ext.]:
  Instructions for tasks you may have to carry out in the alert monitor.
- Creating Your Own Monitors [Ext.]:
  Instead of working with a predefined monitor, you can create your own special monitors. You can include only those monitoring tree elements that you really have to display.
- Customizing the Alert Monitor [Ext.]:
  All monitoring tree elements have approved standard threshold values for deleting alerts and standard assignments for analysis methods.
  You can change the threshold values and the assigned analyze methods, if required.

IDoc Status Reports (ALE Audit)

ALE audit enables the sending R/3 system to monitor the processing status of dispatched IDocs in the receiving system. The receiving R/3 System periodically sends confirmation messages to the sending system. These confirmations are logged in IDoc status records and in separate audit tables in the sending system.

A report program on the sending system analyzes the audit database.



As well as ALE audit there are other monitoring tools in *ALE Administration*.

See also:

Dispatch of Audit Confirmations [Page 78]

Analysis of the Audit Database [Page 82]

ALE Audit IDocs [Page 87]

Comparison of ALE Audit With the Standard IDoc Monitor [Page 77]

ALE Audit and ALE Tracing

The differences between ALE Audit and system-wide IDoc Tracing are:

- The audit database only contains details of IDocs that have errors.
- As a rule, audit reporting is usually more efficient than IDoc monitoring because the dataset to be analyzed is smaller.
- The audit database is only updated periodically. Although this time interval is variable, newly created IDocs only appear in the audit report after a time delay.
- Unlike IDoc monitoring, ALE audit only works with logical systems and not with EDI communication partners such as customers and vendors.

With ALE audit you can easily monitor the IDoc transfer from the sending system provided that time is not an essential criterion for the transfer.

Dispatch of Audit Confirmations

To improve performance, confirmations are dispatched periodically for packets of IDocs rather than for individual IDocs. Confirmations have a special ALEAUD01 IDoc with message type ALEAUD.

The RBDSTATE report dispatches the audit confirmations. You can also start this program in *ALE Administration* by choosing *Monitoring → ALE Audit → Send confirmations.*

Confirmations can only be dispatched, if a message flow for the message type ALEAUD has been specified in the distribution model. The filter object MESTYP provides a more detailed specification. You can use the filter object to set the message types for generating audit confirmations. As a rule, you do not have to send confirmations for all message types. For example, it is quite likely that confirmations will not be necessary for master data. If the MESTYP filter object is not used all IDocs are confirmed by ALEAUD in the receiver system.

See also:

Parameters in Program RBDSTATE [Page 79]

Parameters of Program RBDSTATE

The selection parameters allow you to regulate in which systems and for which application message types, confirmations should be generated in the current report run. The program generates IDocs from the message type ALEAUD that contains information about the processing state of inbound IDocs in your own system. An audit IDoc contains confirmations for up to 500 IDocs. If there are more IDocs to be confirmed, several audit IDocs are generated. A list of the generated IDocs is displayed. If an IDoc cannot be generated or an error occurs, a message is displayed.

The program selects IDocs whose statuses have recently changed. Because almost every IDoc activity (e.g. creation, successful posting/ unsuccessful posting in an application) alters the status of the IDoc, it is these IDocs which have in some way or other been processed. You can specify the repeat interval for these selections with the last selection parameter.

If the report is to be run online, the parameter is mandatory.

This is not the case with background processing where there are two processing modes: Daily or Less Frequent Dispatch of Confirmations [Page 80] and Frequent Dispatch of Confirmations [Page 81] .

Daily Dispatch of Confirmations

This processing mode is preferred for confirmations that are not time-critical. You can dispatch confirmations for IDocs created the previous day or for changes that go back even further. When you enter the variants for background processing, chose a date variable for the date change selection parameter. You can set the days flexibly (for example, yesterday) on which IDoc status changes are to be confirmed.

When scheduling a background job (for example, for sending changes made the previous day), make sure you do not schedule the job at exactly 00.00 hours. Schedule it at least a few minutes later, otherwise the previous day's IDocs which were not posted by 00.00 hours will not be confirmed.

Hourly Dispatch of Confirmations

This processing mode is used for time-critical confirmations, for example, those which should be reported every hour. Do not enter a value for the change date selection parameter. Leave the change date selection parameter empty, as the program verifies the date and time of the last run, or in the case of a first run, the date and time it was scheduled. This time less 5 minutes is the left selection limit for the IDoc changes. The background job start time less 4 minutes is the right selection limit for the IDoc changes.

Evaluating the Audit Database

You can evaluate the audit database in ALE Administration. *Monitoring* → *ALE Audit* → *Evaluate database.*

You can display the messages sent and details of any IDocs containing errors. deters include the IDoc number in the receiving system, the current status and the error message in the receiving system.

The status records, based on the confirmation messages, may have the following values:

- "39" - IDoc is in receiving system (ALE service) and is still being processed.
- "40" - Application document not created in receiving system. (Processing could not be completed in the receiving system).
- "41" - Application document created in receiving system. (Processing has been completed.) The message in the status record is the same as the corresponding message in the IDoc status record in the receiving system.

See also:

Overview of All Statistics [Page 83]

Overview of Individual Statistics [Page 84]

List of IDocs with a Specific Status Value [Page 85]

IDoc Display [Page 86]

Segments of Audit IDoc ALEAUD01 [Page 87]

Overview of Statistics

Here you can display an overview of all the audit statistics you have selected. Each receiver system, message type, and IDoc creation date are recorded. In the column *Last IDoc* you can tell whether all the current day's IDocs are already included in the statistics. If the value of the entry is "24:00:00", they will all be included. If an earlier time is displayed, only those IDocs created up until this time are included in the statistics.

The column *IDocs total* contains the total number of IDocs in the statistics database. The column *Queue Outbound* contains the number of IDocs still being processed in your own system (not yet dispatched). The column *Queue Inbound* contains the number of IDocs received by the target system but still being processed at the time of the last confirmation. If you double-click on a statistic, a status overview is displayed.

Overview of Individual Statistics

All IDocs for the selected statistic are grouped by status and listed separately for outbound and inbound IDocs. If you select a status value, an IDoc list for this status value is displayed. The IDoc list only shows status values which are not yet finalized. It does not show, for example, status 53 - "application document posted" because ALE audit only has information about IDocs that are still being processed.

List of IDocs with Specific Status Value

All of the IDocs with a specific status value can be displayed here. Any linked application objects are also displayed. If the IDocs are already in the receiver system, the IDoc number in the receiver system is also displayed. To display the associated local IDoc, double click on a line.

IDoc Display

This is the IDoc individual display and is also used in IDoc monitoring. You can display the information that exists in your own system for this IDoc. Status records with values 39 and 40 indicate error messages in the receiving system.

Segments of Audit IDoc ALEAUD01

The audit IDoc ALEAUD01 contains three segments: E1ADHDR, ESTATE and E1PRTOB.

One IDoc contains a maximum of 500 IDoc confirmations. The segments comprise the following:

- Segment E1ADHDR

  E1ADHDR contains the message type, the message code and the message function for the ensuing confirmations.

  The field MESTYP_LNG in the segment E1ADHDR has been extended in Release 4.0 because message type names have been increased from 6 to 30 characters. Furthermore MESTYP contains the message type name in a field six characters long; MESTYP_LNG contains the message type name in a field 30 characters long. If possible, values are entered in both fields when an audit IDoc is dispatched. The field MESTYP is left empty for message types created in Release 4.0 onwards and which have not been allocated a six character short text in Customizing for IDoc Basis.

- Segment E1STATE

  E1STATE contains the IDoc number of the sending system (Field DOCNUM), the current status in the receiving system and the message fields from the current status record in the receiving system. In a remote system the DOCNUM value depends on how the IDocs were previously dispatched to the R/3 System. The sending system IDoc number is the number assigned when the INBOUND_IDOC_PROCESS in field EDI_DC-DOCNUM was called. The R/3 System saves this number unless it is the initial value.

  If the audit IDoc is going to be sent to an R/3 System, the *DOCNUM* field must contain a valid R/3 IDoc number. If there is not an IDoc for the given number, this entry is ignored. The *Status* field must contain a valid R/3 status value for inbound IDocs.

  In Release 4.0 more fields have been added to the segment because the data format of IDoc status records in R/3 has changed. The number of lines in the message parameters (from field STAPA1 to field STAPA4) have been increased from 20 to 50 lines. This is why the new fields (from STAPA1_LNG to STAPA4_LNG) have been added. The contents of the field STACOD have been distributed to four fields, STATYP, STAMQU, STAMID and STAMNO.

  When an audit IDoc is created, values should be entered in both field groups. However, this is not always possible for new message types created in Release 4.0.

- Segment E1PRTOB

  E1PRTOB contains the receiver's IDoc number and the application object generated in inbound processing, if one exists. The application object has the same format as the BOR link tools, that is, it comprises a BOR object type, object ID and logical owner system.

IDoc Tracing

Use

You can use the system-wide IDoc tracing to monitor the processing status of IDocs in the sending and receiving systems.

Activities

In ALE Administration choose *Monitoring* → *IDoc tracing*. From here:

1. Enter the message type, receiving system and period of time to be traced and execute the function.

    An overview of all the IDocs in the receiving system is displayed.

    It contains information about the number of IDocs with a particular status. The column *Description* indicates the respective status. (IDocs with the status *Not yet transferred* are still in the tRFC queue and can be displayed in Transaction SM58).

2. Select a status line and choose *Display linked IDocs* (or double-click).

    A list of IDoc pairs in the sending and receiving systems appears.

3. You can display individual IDocs with their control, data and status records by double clicking on a line. The same applies to the IDocs in the receiving system.

Transporting Customizing Data

Use

In *ALE Administration* you can transport *Customizing data* for Customizing objects in receiving systems using ALE transport requests.

Customizing objects are assigned to distribution groups and transported to the participating systems in transport requests. You have to define distribution groups in Customizing for ALE.

For more information about this topic see Synchronizing Customizing Data Between Systems [Page 47].

Prerequisites

You have created distribution groups in Customizing for ALE and carried out modeling.

Modeling and Implementing Business Processes
Synchronization of Customizing Data

Procedure

To transport Customizing data you have to execute the functions below in both the sending and the receiving systems: Choose *Tools* → *ALE* → *Services* → *Customizing Data* → *ALE Requests:*

In the sending system:

- **Display outbound ALE Requests**

  Under *Display outbound requests* you can display the status details of ALE transport requests for a specified time period for each distribution group and target system.

- **Generate ALE requests**

  Under *Generate* using the selection options and parameters you can generate ALE transport requests and IDocs that notify the receiving systems.

  Note on field *CTS dummy system*

  The CTS requires information on the target system, even though in this ALE case, this information is not used because ALE controls the distribution through the info-IDoc itself. Enter an "empty pseudo system" to which usually no transports are made.

  Notes on the objects:

  Customizing data is assigned to Customizing objects and types are assigned to Customizing objects. Depending on this type, problems could arise when transport requests are generated:

Object Type	Notes
L Logical transport object	No problems
S Table (with text table)	No problems

T Single transaction object	Works with restrictions (no standard type of request generation defined, no line selection)
V View	Usually no problems but no line selection possible with objects that are not linked (e.g. SADR* tables).

The objects selected in the system from the Customizing requests which were included in the ALE request and recorded in the request documentation.

In the receiving system:

- **Display inbound ALE requests**

  In *Display inbound requests* you can display the ALE requests and their import status for a specified time period for each distribution group and source system.

- **Import ALE requests**

  Under *Import* you can transport source requests for a specified time period for each distribution group.

- **Consolidate and forward ALE requests**

  Under *Consolidate* you can create a consolidation request from several ALE requests. The consolidation request is used to transport Customizing data imported by ALE requests into quality and production systems.

  You can only execute this function in Customizing Systems in which Customizing data has been imported by ALE transports.

  The function generates a standard Customizing request that has not been released. The user of the function is also the owner of the consolidation request.

Result

The Customizing objects are identical in all individual systems in your ALE integrated system.

Processing IDocs Manually

Use

IDocs can be processed manually in inbound and outbound processing in logical systems.

Features

From *ALE Administration* choose *Monitoring* → *Process IDocs manually*.

On the screen *Select IDocs* you can enter selection options to limit the number of IDocs. If you want to prevent all the IDocs in the database from being counted, which reduces system performance, select the checkbox *No IDoc count* and execute the function.

A hierarchy of the IDoc status information appears:

```
Status groups
   Individual status
      Message type
        Error message
```

You can select information for each of these levels.

You can display the selected IDocs, process them immediately, or first restrict the number of IDocs by status group and then process them.

To restrict the number of IDocs to be displayed or processed, you can filter the status display on the screen *Select IDocs* by choosing I*Doc number, Date created, Message type* and *Partner system.*

You can not process communication errors.

Note that it takes a long time to expand the sub-tree *Communication errors.* You should therefore only expand this sub-tree, if you really need to.

Optimizing ALE Performance

IDocs are fundamental to Application Link Enabling. Optimal system performance in ALE processing is synonymous with optimal performance of IDocs.

In ALE processing IDocs are:

- Created in the R/3 sending system
- Transferred to the communication layer in the R/3 sending system
- Used when R/3 sending and receiving systems communicate
- Processed in the receiver system

You can optimize IDoc processing in one or more of the following ways:

- Controlling IDoc Events [Page 94]
- Processing IDocs in Parallel [Page 98]
- Processing IDocs in Packets [Page 102]
- Administration of IDoc Communication [Page 106]

Controlling IDoc Events

You can specify the times IDocs are created, transferred to the communication layer, and posted in the receiving system in ALE Customizing.

It is best to process time consuming ALE tasks in the background at times when the load on the system is minimal.

See also:

Scheduling IDoc Creation [Page 95]

Scheduling IDoc Transfer to the Communication Layer [Page 96]

Scheduling IDoc Posting [Page 97]

Scheduling IDoc Creation

You can specify the time an IDoc is created for distributing **master data** only. When transaction data is distributed, IDocs are created at the same time as the data is saved in the application.

Master data IDocs can be created in two ways:

- By **processing change pointers** created when master data is created or changed.
  To do this, in ALE Customizing choose:
  Modelling and Implementing ALE Business Processes
  Master Data Distribution
  Replication of Modified Data
  Creating IDocs from Change Pointers
- By **directly sending** or **requesting** master data in the ALE master data menu area or by calling an appropriate program.

You should create change pointers to log master data changes. This avoids having to keep an external list of master data changes for synchronizing data later by sending it directly.

If the same data is changed several times, only the last change pointer is used to create the master data IDoc. This minimizes the number of IDocs dispatched.

To create change pointers, in ALE Customizing you have to activate change pointers generally as well as for individual message types:

Modelling and Implementing ALE Business Processes
Master Data Distribution
Replication of Modified Data
Activate Change Pointers - Generally
Activate Change Pointers for Message Types

It is useful to send or request master data directly, if an R/3 system is to be initialized or synchronized. In *ALE Administration* choose the required master data and function under *Master Data Distribution*.

Scheduling IDoc Transfer to Communication Layer

When an IDoc is transferred to the communication layer, a transactional RFC (tRFC) forwards it to an external system. The ALE layer transfers the IDoc.

The ALE layer processes the outbound IDoc and controls its dispatch. Depending on the partner profile settings in Customizing for *ALE,* IDocs can be transferred in the following ways:

- IDocs assigned to the same partner and message type are collected and the RSEOUT00 program later forwards them to the communication layer.
  To do this, in ALE Customizing choose:
  Basis
  Application Link Enabling (ALE)
  Modelling and Implementing ALE Business Processes
  Partner Profiles and Time of Processing
  Schedule IDoc Dispatch in Packets

  You can process IDocs that are ready for transfer manually in ALE Administration (*IDocs → Monitoring → Process IDocs manually*).
- The IDoc is forwarded to the communication layer immediately.

You should use the outbound processing mode *Collect IDocs,* especially if you are distributing master data. System performance is better.

You can also Process IDocs in Packets [Page 102] in the "collect" processing mode.

To go to the ALE administration functions, from the SAP menu choose

Modelling and Implementing Business Processes
Partner Profiles and Time of Processing
Maintain Partner Profiles Manually or
Generate Partner Profiles.

Under outbound parameters choose the output mode *Collect IDocs and transfer*.

Scheduling IDoc Posting

There are two ways of posting IDocs in ALE inbound processing:

- Immediate processing:

  Upon receipt inbound IDocs are immediately released for posting. ALE inbound processing splits the IDoc packets into individual IDocs.

- Background processing

  Inbound IDocs and IDoc packets are first saved in the database. IDoc packets are split into single IDocs beforehand.

  The program RBDAPP01 later releases the saved IDocs for processing. Single IDocs can be put into packets and then processed.

  To do this carry out the steps below in ALE Customizing:

1. Set up background processing:

   Modelling and Implementing Business Processes
   Partner Profiles and Time of Processing
   Maintain Partner Profiles Manually

   Then the required setting is: In the detail screen *Inbound Parameters* select the option *Trigger by background program.*

2. Schedule posting:

   Modeling and Implementing Business Processes
   Partner Profiles and Time of Processing
   Scheduling IDoc Processing in Receiving System

   You can also process the IDocs manually by passing them to the posting function module. In ALE Administration choose *Monitoring → Process IDocs manually*, select the IDocs and then select *Process.*

   You should choose background processing, especially if large data volumes are to be distributed. System performance is better.

   Packet Processing [Page 102] can be used to process individual inbound IDocs in the background. A function module capable of mass processing is required for this. Processing IDocs in Parallel [Page 98] has further advantages for inbound processing.

Processing IDocs in Parallel

Several IDocs can be processed at the same time in different dialog processes. This is particularly advantageous for sending mass data.

See also:

Creating IDocs in Parallel [Page 99]

Sending IDocs in Parallel [Page 100]

Posting IDocs in Parallel [Page 101]

Creating IDocs in Parallel

IDocs can only be created in parallel in different dialog processes, if master data is sent directly. In contrast, the program RBDMIDOC to process change pointers uses only one dialog process.

There are no benefits of creating IDocs in parallel for distributing transaction data in ALE, because this mainly involves single events which cannot be accelerated by running dialog processes at the same time.

You must specify a server group for creating IDocs in parallel. The server group comprises the application servers, whose available dialog processes are used to create IDocs. Two dialog processes remain unused on every application server in a server group.

There might be only one application server in a server group. Local dialog processes create the IDocs in parallel.

To define the server group in ALE Customizing choose:

Sending and Receiving Systems
  Systems in Network
    Define Target Systems for RFC Calls

Then to create groups, choose RFC → *RFC Groups*

It is better to create IDocs in parallel when sending large quantities of master data.

Choose *Tools* → *ALE* → *Master data distribution* → *Cross-application* → *Material* → *Send,* then the required master data and function.

You can only create IDocs in parallel, if the name of the required server group is entered on the *Send material* screen.

Because two dialog processes remain unused on every application server in a server group, make sure there are enough dialog processes available on the application servers.

Sending IDocs in Parallel

IDoc communication with transactional RFC (tRFC) uses all available dialog processes on the application server that transfers the IDocs to the communication layer. With tRFC parallel communication, it does not matter whether IDocs are first collected or transferred immediately.

IDocs should be transferred to the communication layer on an application server with a sufficient number of dialog processes.

Make sure you specify a sufficient number of dialog processes on the receiving R/3 System. The number of dialog processes should not be less than the number on the sending application server.

Posting IDocs in Parallel

This is used for:

- Immediate processing

  When IDocs are processed immediately, all dialog processes on the receiving application system are used for posting. This could block the application server. You cannot distribute inbound IDoc packets among one server group.

  An IDoc packet may consist of one or more IDocs. A dialog process to execute the posting function module is started on the application server for every IDoc packet. If the IDoc packet consists of several IDocs, and if the posting function module is not capable of mass processing, the ALE layer calls the function module several times in the same dialog process and each time transfers a single IDoc for processing.

- Background processing

  Inbound IDoc packets are split into individual IDocs and are stored in the database.

  You can process IDocs that are ready for transfer (status 64) manually in ALE Administration (*IDocs → Monitoring → Process IDocs manually*).

  Alternatively, IDocs can be processed in the background.
  To do this, in ALE Customizing choose:
  Modeling and Implementing Business Processes
  Partner Profiles and Time of Processing
  Scheduling IDoc Processing in Receiving System

  All application servers in a server group can be used in parallel for updating IDocs in the background. There might be only one application server in a server group. If you do not specify a server group, all dialog processes in the local server are used in parallel. This could block the application server.

  For mass data it is better to process IDocs in parallel.

  You must select the option *Parallel posting* on the screen *Inbound processing of IDocs ready for transfer* in the program RBDAPP01.

  Specify a server group for parallel updating on several applications.

  Because two dialog processes remain unused on every application server in a server group, make sure there are enough dialog processes available on the application servers.

Processing IDoc Packets

Packet processing enables one program or one function module to process batches of data of the same type. This reduces the number of dialog processes called and improves system performance.

You can use packet processing in ALE for:

- Creating IDoc Packets [Page 103]
- Sending IDoc Packets [Page 104]
- Posting IDoc Packets [Page 105]

Creating IDoc Packets

If you send master data **directly**, a function module creates an IDoc for each master data object, for example, material. In most cases more than one master data object can be passed to the function modules for IDoc creation.

If you send master data directly, you can send more master data objects in each process. As a guide you can send between 20 and 100 objects.

This reduces the number of dialog processes used and the administrative load on the R/3 System.

Packet processing and parallelism complement each other, although in some situations they may compete with each other. If there are too many master data objects in each process, possibly not all available dialog processes will be used.

Sending IDoc Packets

Several IDocs can be grouped into a packet and sent in one transactional Remote Function Call (tRFC). This has the following benefits:

- The fewer administrative tasks reduce the load on the system
- tRFC uses less dialog processes in the sending R/3 System
- tRFC uses less dialog processes in the receiving R/3 System

You can specify the packet size of message types in Customizing for ALE.
Modeling and Implementing Business Processes
Partner Profiles and Time of Processing
Generate Partner Profiles

As a guide, packet size should be between 10 and 200 IDocs. The smaller the IDocs are, the larger the packet size can be. For message types which contain large volumes of data, for example, GLROLL or ALEAUD, the packet size should be between 1 and 10 IDocs.

If the IDocs are not going to be posted immediately in the receiving R/3 System, an IDoc packet can contain up to about 10000 data records. For example, if each IDoc contains 10 data records, a packet can contain up to 1000 IDocs.

Posting IDoc Packets

Two groups of function modules are used to post IDocs:

- Function modules which **process IDocs in mass**. These transfer packets of IDocs whose individual IDocs are updated in the same Logical Unit of Work (LUW).
- Function modules which **process one IDoc** per call**.**
- INPUTTYP contains the code for posting function modules.

To display the function module's INPUTTYP, on the ALE Development screen, choose *IDoc* → *Inbound* → *Function module attributes.*

INPUTTYP can contain the following values:

- "0", for function modules which process IDocs in packets.

"1" and "2" for function modules which process one IDoc per call.

If you post the IDocs **immediately**, the R/3 sending system determines the packet size. ALE inbound processing can recognize if the posting function module allows packet processing and if so, passes the IDoc packet to it. If not, the IDoc packet is split into individual IDocs.

If IDocs are posted **in the background**, you can specify the size of the IDocs to be generated in the program RBDAPP01.

If you use function modules that can process IDocs in mass, the database load is reduced.

If you group IDocs into packets, this may also be practical for function modules that post inbound IDocs one at a time, because the ALE layer calls the function module several times in the same dialog process, thereby reducing the administrative load on the R/3 System.

If program RBDAPP01 carries out the background processing, as a guide, you should use a packet size of between 20 and 100 IDocs.

Packet processing and parallelism complement one other. Packet processing and parallelism complement each other, although in some situations they may compete with each other. If the size of the packet is too big, this may mean that not all the available dialog processes are being used.

Administration of IDoc Communication

IDoc communication is based on transactional Remote Function Call (tRFC).

You can optimize the performance of ALE processing by using the following settings and procedures.

- Suppressing Background Processing [Page 107]
- Setting Dispatch Status to OK [Page 108]
- Checking the tRFC Status [Page 109]

Suppressing Background Processing

In the standard R/3 System, if tRFC errors occur, a background job is generated to re-establish the connection. In certain circumstances this could result in a large number of background jobs being started that completely block background processing.

To cancel a background job if tRFC errors occur use program RSARFCEX to restart tRFC.

In ALE Customizing select a connection to change:
  Sending and Receiving Systems
    Systems in Network
      Define Target Systems for RFC Calls

Then choose *Destination* → *tRFC options* and select *Suppress background job if connection error*. Alternatively from the R/3 initial screen by choosing *Tools* → *Administration,* then *Administration*→ *Network* → *RFC Destinations*.

Setting Dispatch Status to *Dispatch OK*

When an IDoc is passed to the communication layer, it is assigned a globally unique transaction identifier (TID). If the IDoc has been successfully dispatched, this information is not automatically passed to the ALE communication layer. So that ALE processing sets the IDoc status to *Dispatch OK,* you should compare the TID numbers in the communication layer and the ALE layer.

The program RBDMOIND checks that the IDocs passed to the transactional RFC have already been sent to the receiving R/3 System. If they have been sent, the IDoc status is changed to "12" - *Dispatch OK*.

To set the IDoc dispatch status, run the program RBDOIND after the IDocs have been passed to the communication layer.

Check the tRFC Status

To check the status of tRFC calls select *Tools* → *ALE* → *Administration* → *Services* → *Communication* → *Transactional RFC* → *Display incorrect calls*. tRFC calls that connect IDocs use the function module INBOUND_IDOC_PROCESS in the receiving system. INBOUND_IDOC_PROCESS).

If an IDoc in the sending system has been passed to tRFC (IDoc status "03"), but has not yet been input in the receiving system, this means that the tRFC call has not yet been executed.

The program RSARFCEX restarts unsuccessful tRFC calls.

You cannot choose the option *is being executed* in background processing.

Serialization of Messages

Use

Serialization plays an important role in distributing interdependent objects, especially when master data is being distributed.

IDocs can be created, sent and posted in a specified order by distributing message types serially.

Errors can then be avoided when processing inbound IDocs.

Features

Interdependent messages can be serially distributed in the following ways:

- Serialization by Object Type [Ext.]
- Serialization by Message Type [Ext.]
- Serialization at IDoc Level [Page 114]
  (not for IDocs from generated BAPI-ALE interfaces)

Serialization Using Object Types

Use

Serialized messages may be of different types (for example, create, change, cancel messages). All messages here relate to one special application object.

With object serialization the messages belonging to a given object are always processed in the correct order on the receiver system. The order messages are posted in on the receiver system is the same as they were created on the sender system.

Prerequisites

You have to activate serialized distribution in both systems in ALE Customizing:

Tools → AcceleratedSAP → Customizing → Edit Project

SAP Reference IMG

Basis Components
Distribution (ALE)
Modeling and Implementing Business Processes
Master Data Distribution
Serialization for Sending and Receiving Data
Serialization Using Object Types

Features

Object type serialization is carried out using object channels.

In an object channel all messages are processed in the target system in the same sequence they were created in the source system. An *object channel* contains a number of ordered IDocs and is defined by an object type (BOR) and precisely one channel number. A *channel number* is a message attribute. It is generated by the function module ALE_SERIAL_KEY2CHANNEL.

All messages with the same channel number and object type are serialized when the messages are processed.

The current number of each object channel is recorded. This process is takes place in what is known as the *registry*. There is an outbound registry and an inbound registry. *Serialization* must be activated in both registries (see prerequisites).

Outbound processing (in source system):

When outbound IDocs are processed, for each object channel (field CHNUM) a unique serial number is assigned to each IDoc created (field CHCOU). This number and the object channel are transferred with the IDoc in the SERIAL field.

An unique serial number is assigned to each message for the application object in question.

Inbound processing (in target system)

When inbound IDocs are processed, a unique serial number is generated for each object channel and communication link. The ALE layer determines whether a given IDoc can now be posted or whether other IDocs have to be posted first. The serial number for each received IDoc is exactly one whole number lower. Any gaps in the sequence will mean that IDocs are missing, either because the transfer did not work, or because earlier IDocs were not posted successfully.

In this case the IDoc is assigned status 66 and must be posted again with the program RBDAPP01.

Objects are assigned to messages and channels by the application.

Transfer errors (IDoc sequence mixed up) and inbound posting errors (IDoc cannot be posted due to Customizing errors) no longer affect the sequential order.

Serialization By Message Type

Use

IDocs can be created, sent and posted in a specified order by distributing message types serially.

Object interdependency is important at the message type level.

Consider a purchasing info record with a vendor and a material. To avoid any processing errors, the vendor and material must be created in the receiving system before the purchasing info record.

Prerequisites

You have to activate serialized distribution of message types in both systems in ALE Customizing (Transaction SALE).

Basis Components
  Distribution (ALE)
    Modeling and Implementing Business Processes
      Master Data Distribution
        Serialization for Sending and Receiving Data
          Serialization by Message Type

Features

Serialized distribution is only used to transfer changes to master data. IDoc message types are assigned to serialization groups according to the order specified for their transfer. Master data is distributed in exactly the same order. If all the IDocs belonging to the same serialization group are dispatched successfully, the sending system sends a special control message to the receiving system. This control message contains the order IDocs are to be processed in and starts inbound processing in the receiving systems.

Serialized distribution of message types is not a completely new way of distributing master data; it uses existing ALE distribution mechanisms whilst adhering to a specified order of message type. This distribution could also be carried out manually using existing ALE programs. However, serialized distribution automates the single steps and can schedule them in a batch job.

In the serialization menu selection criteria determine how certain parts of the serialized distribution will be processed, for example, control message dispatch and inbound processing.

Serialization at IDoc Level

Use

Delays in transferring IDocs may result in an IDoc containing data belonging to a specific object arriving at its destination before an "older" IDoc that contains different data belonging to the same object. Applications can use the ALE Serialization API to specify the order IDocs of the same message type are processed in and to prevent old IDocs from being posted if processing is repeated.

Prerequisites

IDocs generated by BAPI interfaces cannot be serialized at IDoc level because the function module for inbound processing does not use the ALE Serialization API.

Features

ALE provides two function modules to serialize IDocs which the posting function module has to invoke:

- IDOC_SERIALIZATION_CHECK
  checks the time stamps in the serialization field of the IDoc header.
- IDOC_SERIAL_POST
  updates the serialization table.

Converting Logical System Names

Purpose

Due to the increasing componentization and openness of R/3, logical systems must be able to communicate with each other beyond the boundaries of ALE integrated systems, for example, for in central user administration. Logical systems must therefore have unique names, even those outside the ALE integrated system.

As of Release 4.5A SAP provides a tool to automatically convert the names of logical systems in all application tables, master data tables and control data tables.

As of R/3 Release 4.6C this tool is integrated in the client copy.

This includes both client-dependent and client-independent tables.

- You must not under any circumstances convert logical system names in a production system.
- When logical system names are converted, in exceptional cases inconsistencies may result after the database has been saved.
- IDocs not yet processed that contain the names of the logical systems, must be processed before you start the conversion.
- While the conversion program is running, no other activities can be carried out in the system.
- The conversion must be carried out in all the partner systems. The conversion is carried out in the current client for all client dependent and client-independent table entries.

Prerequisites

Users of the report must have the authorization object B_ALE_SYS in their user profile.

We recommend that only the ALE system administrator is given this authorization.

Process Flow

To carry out the conversion, start Transaction `BDLS`.

The default name displayed is the old logical system name.

Enter the new name of the logical system.

You can simulate the conversion in the report's test mode. Here only the tables in question and the number of the tables entries to be converted are displayed. The table entries in the database are not converted. You can evaluate the results and assess the runtime.

The report carries out the tasks below:

1. Determines all the active, transparent database tables, whose fields have references to the domains:
   - LOGSYS
   - EDI_PARTNUM
2. Converts the field values to the new logical system names

3. Updates the database

Transporting Changes

The table contents are not transported.

Conversion of EDI Tables

When EDI tables are converted, the report also determines the table fields containing the value LS (logical system).for the partner type. The associated logical system names are converted too.

Exceptions

If the new logical system name is maintained on the local R/3 System (where the names are being converted) in table TBDLS, the logical system name cannot be converted. You must assume that the new logical system name is already being used (for example in the ALE distribution model).

ALE Recovery for Data Consistency

Use

In distributed environments you can also recover an R/3 System following a system failure caused by a database error. Using backup and point-in-time recovery, you can reset the database to the status at the time the system failed and continue working with it.

In this case, transactions may still have taken place after the time the system is reset to, which are subsequently lost and have to be carried out again. External interactions (for example, ALE, EDI, mail, fax, telex), possibly carried out under a different key, are duplicated and require special processing.

For example, if, a process sent a message to another system and started another process here during the time between when the error occurred and when it was discovered on the recovered system, the data related to this process will no longer exist on the recovered system. The results of the second process carried out on the receiving system still exist in this system. The inconsistent data in the two systems can be put right by canceling the data resulting from the communication in the receiving system.

In the case of such point-in-time recoveries, certain documents must be canceled in the communication partners' systems and messages sent earlier from the receiving system must be sent again. You have to determine all the actions that need to be carried out in the recovery process.

You can use ALE recovery tools for this.

ALE recovery tools can also be used for data exchanged with non-SAP systems.

Prerequisites

To recover a system, the following is required:

- Database that allows a point-in-time recovery
- Continuous database backup
- The database of the recovered system has already been restarted at the status before the error occurred

Features

ALE Recovery tools perform the required analyses and help you carry out the necessary tasks.

1. The recovered system prompts its communication partners to provide details of the IDocs exchanged between the systems during the time affected.
2. The communication partners of the recovered system report back this information.
3. The information reported back is compared to the information in the recovered system. This process determines what further activities are required to recover a consistent status in the entire distributed environment. The status of IDocs is updated in the recovered system.
4. A list of documents to be canceled in the communication partner systems is created. If necessary, IDocs are sent again automatically.

Not all inconsistencies can be automatically put right by the recovery tools. You may have to remove some of the inconsistencies manually. This especially applies to canceling application documents.

Activities

To remove inconsistencies, you have to carry out the following R/3 transactions:

- Determine recovery objects: Transaction code BDRC
- Process recovery objects: Transaction code BDRL

For more information, see the program documentation.

The following functions are also available:

- Application log for ALE recovery procedure: Transaction code BDR1

  The relevant parameters specified in the transactions *Determine recovery objects* and *Process recovery objects,* are recorded in a log. You can display the log in Transaction BDR1.

- Reorganization of the data for recovery objects: Transaction code BDR2

  The data generated in the transactions *Determine recovery objects* and *Process recovery objects,* is used to identify the status of the recovered objects.

This data is deleted when it is reorganized if the following conditions apply:

- The recovered objects do not need to be processed further.
- The recovered objects have already been processed.