Enterprise Services for SAP ERP Enhancement Package 2

July 2007

Copyright

© Copyright 2007 SAP AG. All rights reserved.

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

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

Microsoft, Windows, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.
IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, Informix, i5/OS, POWER, POWER5, OpenPower and PowerPC are trademarks or registered trademarks of IBM Corporation.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.
Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.
Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

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

MaxDB is a trademark of MySQL AB, Sweden.

SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

Icons in Body Text

Icon	Meaning
	Caution
	Example
	Note
	Recommendation
	Syntax

Additional icons are used in SAP Library documentation to help you identify different types of information at a glance. For more information, see *Help on Help* → *General Information Classes and Information Classes for Business Information Warehouse* on the first page of any version of *SAP Library*.

Typographic Conventions

Type Style	Description
Example text	Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options.
	Cross-references to other documentation.
`Example text`	Emphasized words or phrases in body text, graphic titles, and table titles.
EXAMPLE TEXT	Technical names of system objects. These include report names, program names, transaction codes, table names, and key concepts of a programming language when they are surrounded by body text, for example, SELECT and INCLUDE.
`Example text`	Output on the screen. This includes file and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.
`Example text`	Exact user entry. These are words or characters that you enter in the system exactly as they appear in the documentation.
`<Example text>`	Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.
`EXAMPLE TEXT`	Keys on the keyboard, for example, `F2` or `ENTER`.

Enterprise Services for SAP ERP Enhancement Packages 11
SAP Enterprise Services 11
mySAP Business Suite: Service Processing 12
Introduction to Enterprise Service-Oriented Architecture 12
Composite Application 14
Service 14
Web Service 15
Service-Oriented Architecture 16
Enterprise Service 17
SAP's Enterprise Service-Oriented Architecture (SOA) 18
mySAP Service Provisioning: Runtime 19
Runtime Architecture Overview 19
HTTP Communication Layer 21
Web Service Enabling Layer 22
Invocation and Execution of Synchronous Web Services 24
Dispatching a Request 25
Processing a Request 25
Sending a Response 26
mySAP as Service Provider and Service Consumer 26
Communication Pattern 26
Proxy Type 28
Data Conversion and Customer Modifications 31
Proxy Meta Data 32
Application Level Mapping of Data Types 33
Data Access 34
Concurrency Control 34
Overwrite 36
Delta Update 36
Optimistic Locking 36
Reliable Messaging 38
Restartable Services 39
Verification Services 39
Idempotent Services 41
Important Facts for Implementers 42
Commit Procedures 42
Commit by XI Only 43
No Commit by Framework 44
Exceptions and Errors 45
Exception 46

Error Message ... 47
Security ... 48
Providing Secure Transportation ... 49
Providing Authentication in SAP Web AS ... 49
Providing Authorization within the mySAP Business Suite ... 50
Web Service Invocation via XI Message Broker ... 50
SAP Exchange Infrastructure ... 50
XI Protocol ... 52
XI Local Integration Engine ... 52
mySAP Service Provisioning: Design Time ... 52
Business Object and Service Modeling ... 53
Business Object Modeling ... 53
Process Component ... 54
Business Object ... 54
Service Interface Modeling ... 56
Proxy Generation in the ABAP Stack ... 58
mySAP Service Provisioning: Configuration ... 59
Releasing a Web Service ... 60
Service Definition and Service Variant ... 61
Performing Web Service Administration ... 61
Configuring Idempotent Services ... 62
Example: Service Operation ... 63
Modeling of Process Components in ARIS ... 63
Business Object Modeling ... 65
Modeling of Service Interface in ESR ... 66
Proxy Implementation ... 67
RFC: PT_ESA_REQUEST_CREATE ... 73
Stability of Released Enterprise Services ... 73
Glossary ... 74
A2A ... 75
ABAP ... 75
BAPI ... 75
B2B ... 75
HTTP ... 75
HTTPS ... 75
Java ... 75
RFC ... 76
SAML ... 76
SSL ... 76
XML ... 76

XML Encryption ... 76
XML Signature ... 77
About This Documentation ... 77
SAP: Important Disclaimers and Legal Information ... 79
Enterprise Services for SAP ERP Cross-Application Components ... 80
Business Partner Data Management ... 80
Employee ... 80
Employee E-Mail Data Management In ... 81
Maintain E-Mail Address ... 81
Employee Payment Card Management In ... 83
Create Payment Card Details ... 84
Cancel Payment Card Details ... 86
Customer ... 88
Manage Customer In ... 89
Update Customer Bank Details ... 89
Update Customer Basic Data ... 92
Create Customer Relationship Contact Person ... 93
Update Customer Relationship Contact Person ... 94
Cancel Customer Relationship Contact Person ... 96
Sales Arrangement ... 97
Manage Sales Arrangement In ... 97
Create Sales Arrangement ... 98
Update Sales Arrangement ... 99
Enterprise Services for Kanban Processing ... 100
Kanban ... 101
Query Kanban Out ... 101
Query Kanban Tag for Printing ... 102
Price Master Data Management ... 103
Sales Price Information ... 103
Sales Price Information Out ... 103
Request Sales Price Information as Bulk ... 104
Sales Price Specification Calculation ... 104
Calculating Sales Price Specification In ... 104
Create Sales Price Specification Calculation ... 105
Retail Event Management ... 105
Retail Event ... 105
Manage Retail Event In ... 106
Create Retail Event ... 106
Retail Merchandise Distribution Management ... 107
Retail Allocation ... 107

Create Retail Allocation ... 108
Trade Item Catalogue Authoring ... 109
Trade Item Catalogue ... 109
Trade Item Catalogue Subscription Sending In ... 110
Change Trade Item Catalogue based on Confirmation ... 111
Trade Item Catalogue Subscription Sending Out ... 112
Request Trade Item Catalogue Subscription ... 112
Request Trade Item Catalogue Subscription Cancellation ... 113
Trade Item Catalogue Transmission Receiving In ... 114
Maintain Trade Item Catalogue ... 115
Trade Item Catalogue Transmission Receiving Out ... 116
Confirm Trade Item Catalogue Transmission ... 117
Trade Item Catalogue Transmission Request Out ... 118
Request for Trade Item Catalogue Transmission Request ... 118
Trade Item Catalogue Publishing ... 119
Published Trade Item Catalogue ... 120
Accepting Trade Item Catalogue Out ... 121
Notify of Trade Item Catalogue Acceptance ... 121
Enterprise Services for SAP ERP Logistics ... 122
Outbound Delivery Processing ... 122
Outbound Delivery ... 123
Manage Outbound Delivery ... 124
Change Outbound Delivery ... 124
Production ... 126
Production Order ... 127
Manage Production Order In ... 127
Change Production Order ... 128
Update Production Order ... 129
Sales Contract Processing ... 130
Trade Price Specification Contract ... 130
Trade Price Specification Contracting In ... 130
Cancel Trade Price Specification Contract ... 131
Maintain Trade Price Specification Contract ... 131
Trade Price Specification Contracting Out ... 131
Confirm Trade Price Specification Contract ... 132
Sales Order Processing ... 132
Sales Order ... 133
Query Sales Order In ... 133
Find Sales Order Basic Data by Elements ... 134
Manage Sales Order In ... 135

Create Sales Order V1 ... 136
Read Sales Order V2 ... 138
Change Sales Order ... 139
Change Sales Order Item ... 146
Enterprise Services for SAP ERP Financials ... 150
CashPointProcessing ... 150
CashPointPayment ... 151
Cash Point Payment Notifying In ... 151
Create Cash Point Payment ... 152
Cancel Cash Point Payment ... 153
CashPointOpenItemSummary ... 153
Query Cash Point Open Item Summary In ... 154
Find Cash Point Open Item Summary by Elements ... 154
CashPointClosingDocument ... 155
Cash Point Document Notifying In ... 155
Create Closing Document ... 156
Contract Accounts Receivables Payables Processing ... 156
ContractAccount ... 157
Query Contract Account In ... 158
Find Contract Account Balance ... 158
Find Contract Account Split Item Group by Business Partner ID ... 159
PaymentDirective ... 161
Query PaymentDirective In ... 161
Find Payment Directive by Business Partner ID ... 162
Find Payment Directive by Receivables Payables Register Item Split Item ID ... 163
Manage PaymentDirective In ... 163
Create Payment Directive ... 164
Due Item Processing ... 165
TradeReceivablesPayablesAccount ... 166
Query Trade Receivables Payables Account In ... 167
Find Trade Receivables Payables Account Split Item Group by Customer ID ... 168
Trade Receivables Payables Account Action In ... 168
Release Trade Receivables Payables Account Split Item Group for Payment ... 169
Expense and Reimbursement Management ... 170
Expense Arrangement External Input ... 171
Expense Arrangement External Input Transmission Receiving In ... 171
Maintain Expense Arrangement External Input ... 172
Expense Arrangement External Input Transmission Receiving Out ... 174
Confirm Expense Arrangement External Input Transmission ... 175
Payment Processing ... 176

Payment Order ... 176
Collective Payment Order In ... 177
Collective Payment Order Notification ... 177
PaymentRegister ... 178
Manage Payment Register In ... 178
Read Payment Register Item ... 179
Travel Management ... 180
Travel Planning Arrangement ... 180
Travel Planning Arrangement In ... 181
Change Arrangement Based on Flight Arrangement Created Notification ... 182
Change Arrangement Based on Hotel Arrangement Created Notification ... 184
Change Arrangement Based on Car Rental Arrangement Created Notification ... 185
Change Arrangement Based on Train Arrangement Created Notification ... 186
Change Arrangement Based on Customer Loyalty Programme Created Notification ... 188
Travel Planning Arrangement Out ... 189
Notify of Travel Planning Arrangement Employee ... 190
Enterprise Services for SAP ERP Human Capital Management ... 192
Personnel Administration ... 192
Personnel Leaving ... 193
Personnel Leaving Out ... 193
Notify of Personnel Leaving ... 194
Enterprise Services for Trading Industries ... 196
Trading Order Processing ... 196
Trading Order ... 196
Manage Trading Order ... 196
Read Trading Order ... 197
Query Trading Order In ... 197
Find Trading Order By Elements ... 197
Trading Ordering In ... 198
Cancel Trading Order ... 198
Maintain Trading Order ... 198
Release Trading Order ... 199
Trading Ordering Out ... 199
Confirm Trading Order ... 200
Notify of Trading Order ... 200
Notify of Trading Order Cancellation ... 200
Notify of Trading Order Release ... 201
Analytical View of Trading Order ... 201
Analytical View of Trading Ordering Out ... 202

Notify of Analytical View of Trading Order 202
Copyright 203
SAP - Important Disclaimers and Legal Information 205

Enterprise Services for SAP ERP Enhancement Packages

SAP Enterprise Services

Purpose

Enterprise Services from SAP afford you many new opportunities to automate your business processes and react flexibly to changes in your business processes. You can convert new business processes much more easily. You can also implement the *Enterprise Services* in heterogeneous, distributed system landscapes by implementing standardized Web technologies.

Integration

- A large number of *Enterprise Services* for typical business processes are already available in the *SAP Business Suite* applications. You can implement these services so that they communicate directly with other applications, client programs, user interfaces, and so on.
- Alternatively, you can use the *Enterprises Services* from SAP using the *SAP NetWeaver Exchange Infrastructure* (*SAP NetWeaver XI*). In this case, *Enterprise Services* communication occurs using the *Integration Broker* of the *SAP NetWeaver Exchange Infrastructure*, which contains additional functions for the processing and creation of messages. In the *Enterprise Services Repository* in *SAP NetWeaver XI* you can find descriptions of the interfaces of all the services that SAP has developed for communication with SAP applications. You can also define your own *Enterprise Services* using the *SAP NetWeaver Exchange Infrastructure*.

Features

- *Enterprise Services* are standardized interfaces for specific functions within an application, for example, for functions that read the business partner's address or that update reports. They simultaneously display descriptions of process steps and technical modules. In this way, and due to their very specific syntax and strictly defined naming convention, the *Enterprise Services* represent a shared language for communication between different business areas (for example, marketing and IT).
- You can use the *Enterprise Services* from SAP to create user interfaces independently of the underlying individual applications in the back-end system. These interfaces can be exactly adjusted to the workflows and work organization of your company.

  You can therefore provide user interfaces that are tailored exactly to the tasks and knowledge of the employees in your company. You use the same services and process the same data regardless of whether you create user interfaces for experts or for occasional users. The *Enterprise Service* works in the background on the technical details and data entry and processing is considerably simplified.
- With the help of the *Enterprise Services* from SAP, you can trigger automated processes, enter and process data, and exchange data with other systems and applications. In contrast to simple Web services, *Enterprise Services* from SAP are generally logical steps of a business process and can thus be used to support cross-application and cross-departmental business processes.
- You can implement *Enterprise Services* from SAP to automate flows for which you previously could not use any standard SAP solution or for which there previously was

no integration possible. SAP does not only develop *Enterprise Services* with the aim of providing individual interfaces for communication between individual application functions. *Enterprise Services* from SAP also aim to enable central business processes using several *SAP Business Suit*e applications. For this purpose, *Enterprise Services* are bundled into *Enterprise Services Bundles* (ES Bundles). The *ES Bundles* represent templates for the processing of *Enterprise Services* for the illustration of business processes such as Order to Cash. You can find descriptions of the *Enterprise Services Bundles* in the *Enterprise Services Workplace* (ES Workplace) by choosing **`sdn.sap.com`** → *Enterprise SOA* → *ES Workplace* → *ES Bundles Wiki.*

- The *Enterprise Services* are used to split the complex functions of a business process into reusable elements. These elements – the individual *Enterprise Services* – can then be reused, exchanged, combined into new combinations, or incorporated into completely different business processes according to your own needs.

  Let us assume that you use the *Enterprise Services* to provide both an application with which your employees can submit their expenses and a number of additional services for the distribution of expenses. At the moment, the data is probably sent to particular administrators in the relative departments. However, you already anticipate that a central administration center will take over the processing in the near future. An automatic system for processing expenses or outsourcing to a foreign company is probably under discussion.

  If you use *Enterprise Services* you need only perform such changes in a few critical places. In addition, nothing will change on the employee's user interface, regardless of how you construct the distribution in the background. The function for sending expenses remains unchanged for employees and they are thus not required to understand the various flows in the background.

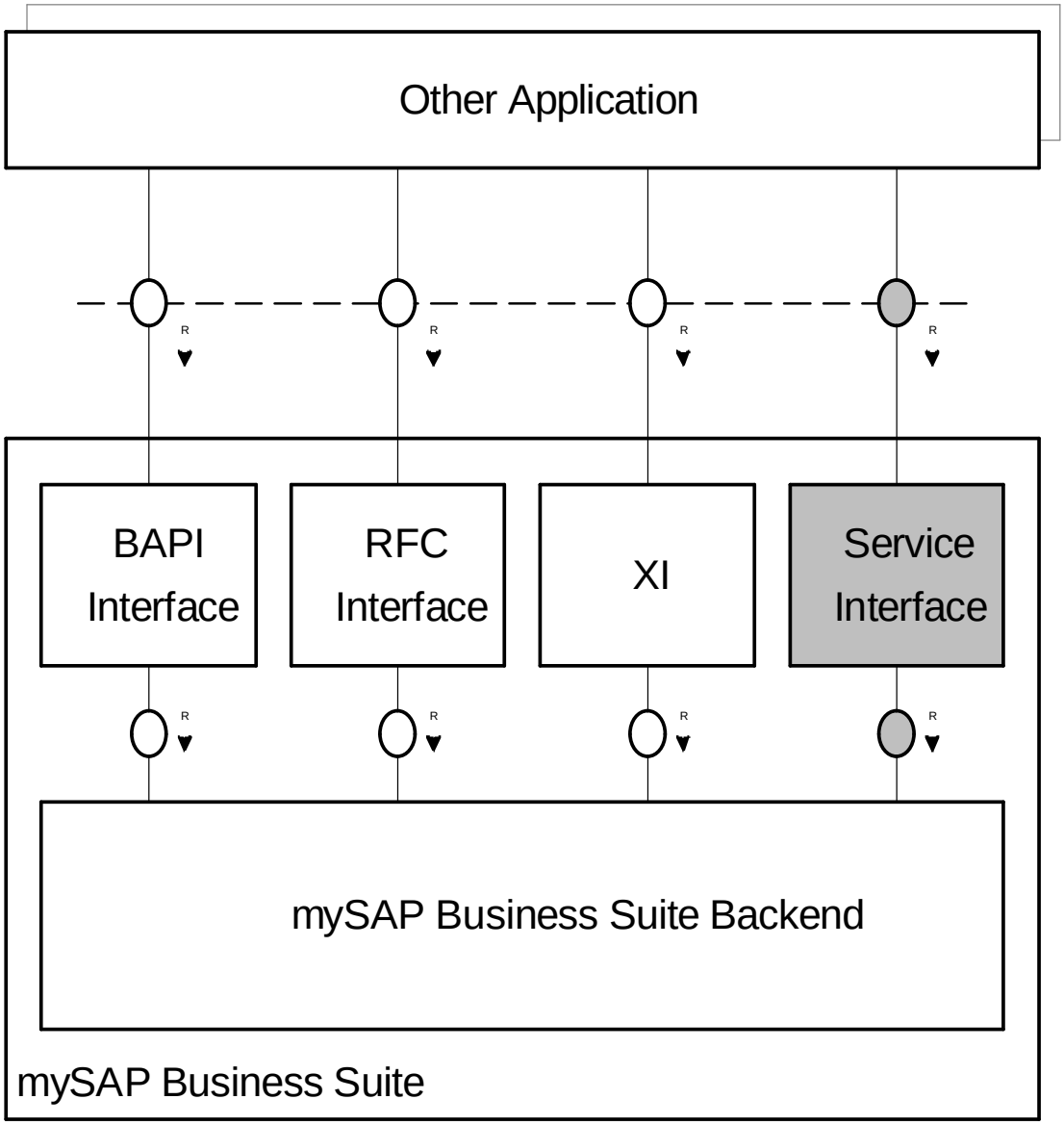
mySAP Business Suite: Service Processing

With the product enterprise services packages for the mySAP Business Suite 2005, SAP is moving the mySAP Business Suite 2005 toward Enterprise Service-Oriented Architecture (SOA). Introducing Enterprise SOA into the mySAP Business Suite 2005 naturally has implications for its software architecture. This document describes the implications and the concepts used to meet the challenges that arise.

Introduction to Enterprise Service-Oriented Architecture

The introduction of Enterprise Service-Oriented Architecture (SOA) essentially constitutes a new, more standardized way of accessing mySAP Business Suite functionality from other applications, alongside conventional access mechanisms like Business Application Programming Interfaces (BAPIs), Remote Function Calls (RFCs) and the SAP Exchange Infrastructure (SAP XI). Another application in this context may be an independent third-party application, a composite application or another SAP application like another mySAP Business Suite instance.

Introduction of Enterprise SOA into the mySAP Business Suite



Implementation Considerations

Enterprise SOA sets requirements for functionality access both on a technical and semantic level. To meet the requirements, which are detailed in the following sections, SAP has built a service interface layer on top of the mySAP Business Suite as shown in the figure above.

When discussing the SOA-enabling of the mySAP Business Suite, three different dimensions need to be distinguished and addressed:

- *Runtime:* the running system including all mechanisms in use to process a request via the service interface layer

  See mySAP Service Provisioning: Runtime

- *Design time:* procedures and methods applied by SAP during application design and development to create the software in the first place

  See mySAP Service Provisioning: Design Time

- *Configuration:* all concepts concerning activation and publication of functionality provided by the service interface layer

See mySAP Service Provisioning: Configuration

Note

It is expected that the reader is familiar with the structure and the basic architecture concepts of the mySAP Business Suite. For detailed information on the mySAP Business Suitesee the SAP help portal (help.sap.com *mySAP Business Suite*) or the SAP Developer Network (SDN) (sdn.sap.com).

.

Features

The remainder of this section defines the relevant terms and introduces the concept of enterprise SOA:

- Service
- Web Service
- Enterprise Service
- Service-Oriented Architecture
- SAP's Enterprise SOA

Example

The section Example: Service Operation 'Create Employee Leave Request' provides a walk-through of service provision.

Composite Application

A composite application is an application that is built upon the mySAP Business Suite and uses and extends the functionality of the mySAP Business Suite. Composite applications are developed by both SAP and external companies.

Service

A service is a well-defined piece of functionality which can be invoked via a stable and implementation-independent interface.

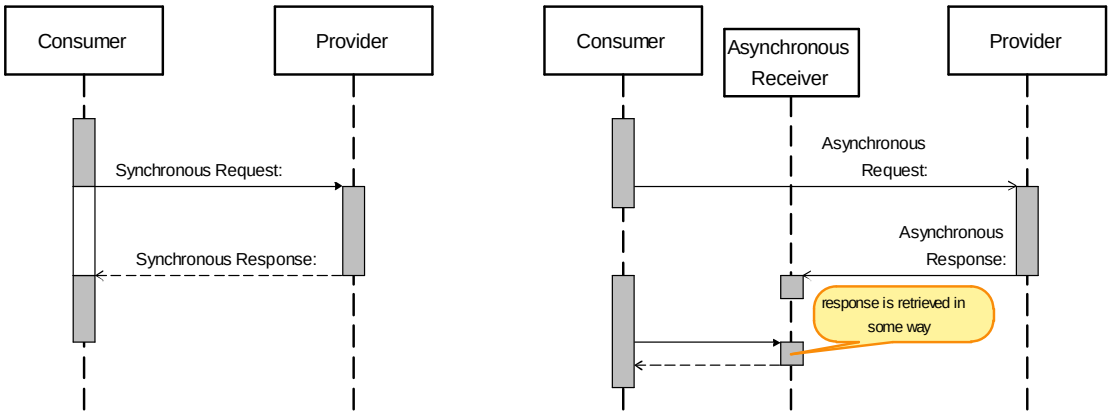
The entity that offers a service to other entities is called *service provider*. An entity that makes use of a service by triggering its execution and – if appropriate – receiving its result, is called *service consumer*.

Synchronous/Asynchronous Communication

Services may be invoked either *synchronously* or *asynchronously*. In the synchronous case, the service consumer waits for the result after triggering the service provider, and is blocked until the result has arrived. In the asynchronous case, the service consumer continues its work after triggering the service provider, and may go back to the original task, as soon as the

answer arrives. For the asynchronous case, an additional agent for receiving the response and a well-defined interface for the return of the response are required.

Synchronous Versus Asynchronous Service Call



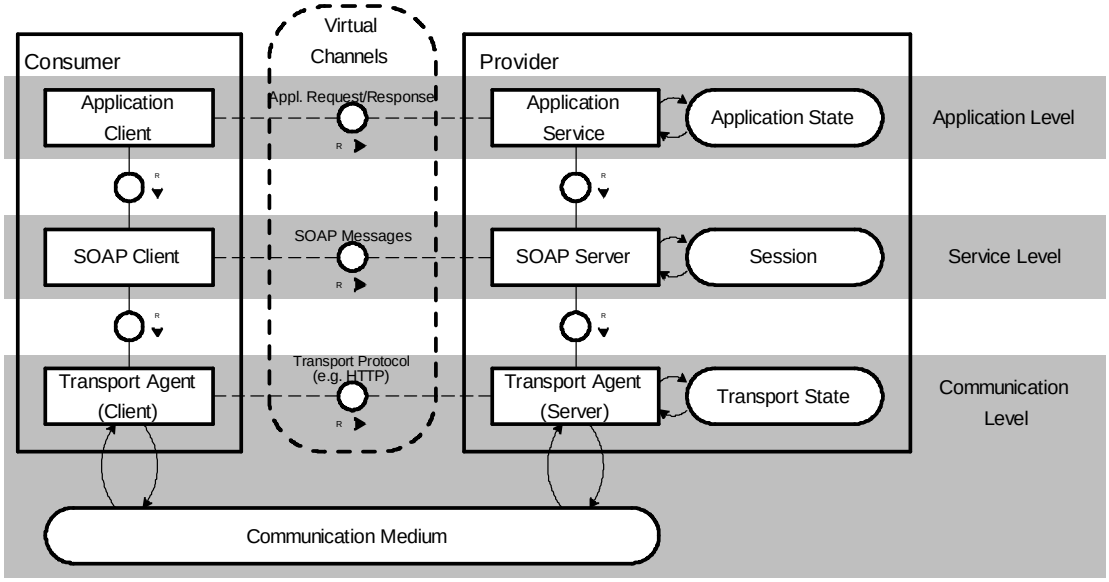
Statefulness/Stateless

Services may be *stateful* or *stateless*. Statefulness means that context information is stored on the provider side between successive service calls. State may be established on different levels as shown in the figure below.

- On the *communication level*, by means of a stateful protocol.
- On the *service level*, by combining successive service calls from the same consumer to a session (for example by exchanging session IDs).
- On the *application level*, by using application-specific mechanisms.

A stateless service, as defined by SAP, is a service that is stateless on the service level. However, it may establish state on the communication level or application level.

Levels of Statefulness



Web Service

A Web service is a special service which can be triggered via a network and conforms to the following Web standards:

- *Web Service Definition Language (WSDL)* is a standardized XML format for describing a service. In WSDL, the term service refers to a concrete instance of a service, provided by a service provider at a specified network endpoint. A service description in WSDL therefore comprises a service interface description and a concrete binding.

  The service interface is the abstract technology-independent description of the service. In WSDL, a service may comprise several service operations which can be invoked separately. For each operation WSDL describes the parameters needed to invoke the service and the result that will be returned.

  The binding determines the technical parameters, such as the network endpoint that acts as the service provider, the protocol used to communicate with the service provider and the format in which messages are exchanged.

- *Universal Description, Discovery and Integration (UDDI)* is the name of a directory service which can be used to publish the WSDL descriptions of the service instances that a service provider wants to make available to other applications.

- *SOAP* is a standardized, message-based protocol that is used by Web services for communication between the service consumer and the service provider. A SOAP message comprises a header part, which contains communication control parameters, and a body part, which contains the payload

The requirement of adherence to these three standards makes a Web service a standardized form of an Internet-based service.

Service-Oriented Architecture

Service-Oriented Architecture (SOA) is the term for software architecture that is based on the concept of services.

Structure

In SOA there are one or more *service providers* on one side, and one or more *service consumers* on the other side.

The service providers offer certain functionality as services and publish their interfaces which are designed to be stable.

Service consumers and service providers are only loosely coupled. The interplay of consumer and provider is based on the service interface definition as a kind of contract. The concrete implementation of a service is hidden from the consumer.

The services should be defined in such a way that they can be found and used easily in various applications. Consumers usually make use of several services, which may come from multiple service providers to create new functionality.

Benefits

Following the principles of SOA when building an application provides several advantages:

- *Flexibility*

Since service interfaces according to SOA are stable, and service implementations are transparent for the consumers, SOA provides the prerequisites for applications in which higher-level functionality realized by the service consumer can be modified without having to change the underlying service functionality.

- *Productivity*

  If the services are well defined, it is very efficient to build and advance SOA-based applications once a pool of services is available.

- *Adaptability*

  SOA facilitates the integration of different applications, thanks to the use of open and non-proprietary standards. This helps customers to integrate applications from different application vendors into their IT landscape.

Enterprise Service

An enterprise service is a service which provides business functionality and which is published by SAP in the enterprise services workplace in the SAP Developer Network (SDN) (at sdn.sap.com *Enterprise SOA* *ES Workplace*). In general, enterprise services have the following features:

- *Business semantics:* Enterprise services are structured according to a harmonized enterprise model based on process components, business objects and global data types (GDTs). They are defined using an outside-in approach: common business rules and know-how, rather than SAP-specific implementations, are the guideline for defining the business content of SAP applications.

- *Quality and stability:* Enterprise services ensure a stable interface for future versions (backward compatibility). Their behavior, prerequisites, dependencies of usage and configuration possibilities are well documented.

- *Standards:* Enterprise services are based on open standards. The interfaces are described according to WSDL. They are created out of global data types which are based on UN/CEFACT CCTS (Core Component Technical Specification).

There are different application areas for enterprise services, for example application-to-application (A2A) and business-to-business (B2B) integration or user interfaces.

Integration

Enterprise Services for the mySAP Business Suite 2005

A first wave of enterprise services for the mySAP Business Suite was released in the product enterprise services packages for the mySAP Business Suite.

While enterprise services can have different technical characteristics, most of the enterprise services in the enterprise services packages are optimized for a usage in which SAP makes no assumptions about the service consumer. As a consequence, these enterprise services are as simple and easy to use as possible, which results in the following characteristics:

- *Atomic transaction:* The execution of a service must always result in consistent business data. It shall not be necessary to execute any further services subsequently to achieve a consistent state on the database.

- *Statelessness:* No context information is stored on the service level between service calls. In addition, enterprise services for the mySAP Business Suite do not store

transient context information on the application level either. Hence, one service call only influences subsequent service calls by means of the persistent data in the database.

Furthermore, these enterprise services for the mySAP Business Suite are defined as synchronous services. This means, in particular, that they can be used without the SAP Exchange Infrastructure (SAP XI).

These enterprise services can be combined to form new business processes (service orchestration). User interfaces as well as composite applications can be built upon them.

SAP's Enterprise Service-Oriented Architecture (SOA)

SAP has been able to move the mySAP Business Suite toward enterprise SOA firstly by realizing the technical prerequisites for Web services and secondly, by introducing enterprise services.

1. *Generated Web Services*

   By means of SAP Web Application Server (SAP Web AS), SAP has supported the concept of Web services since SAP Web AS 6.40.

   SAP Web AS is the basis for all SAP applications. It comprises a framework for processing ABAP and Java programs and basic functionality for communications, transaction handling, database access, and other functions. The conventional way of calling SAP Web AS from other applications is via Business Application Programming Interfaces (BAPIs) or Remote Function Calls (RFCs). Both are proprietary SAP methods for inter-application communication.

   In SAP Web AS 6.40, SAP Web AS can act as a Web service provider, and there is a way of generating Web services based on any BAPI or RFC function.

   These generated Web services conform to the technical requirements of a Web service. However, their semantics are still those of the underlying BAPIs or RFCs. Hence, from a semantic point of view, they are not optimized for SOA.

2. *Enterprise Services*

   In order to take the step towards SOA on a semantic level as well, SAP introduced the notion of enterprise services. SOA based on enterprise services together with the required infrastructure is called enterprise SOA.

Prerequisites

To transform the mySAP Business Suite into an enterprise SOA solution, two main objectives have to be accomplished.

1. A technical framework has to be set up to allow service publication and the processing of service calls.

   To allow for as much reuse as possible and to avoid re-implementation of the business functionality of the mySAP Business Suite, the technical framework has been realized by means of a service interface layer on top of the mySAP Business Suite, as shown in the figure below.

2. The functionality of the mySAP Business Suite has to be optimized for use in a service-oriented application

The second task is to provide the business content of the mySAP Business Suite in a form that meets the principles of enterprise SOA. Here, SAP follows an outside-in, model-based approach. All services, and the business objects that these services operate on, are modeled according to current business practice.

mySAP Service Provisioning: Runtime

This section describes the concepts required to understand the execution of a Web service offered by the mySAP Business Suite.

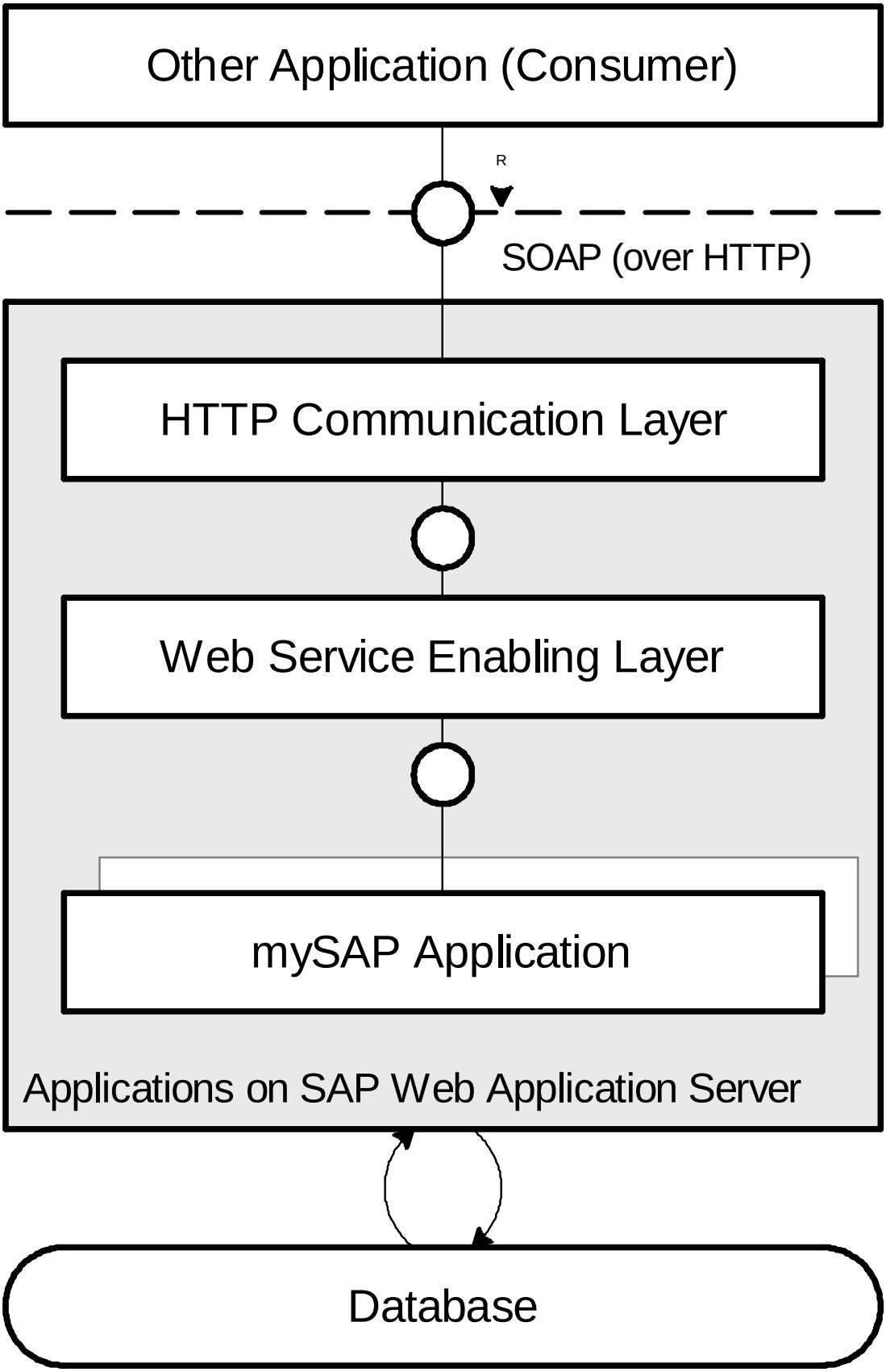
- The Runtime Architecture Overview describes the components of SAP Web AS that are involved in the execution of a Web service. It focuses only on the point-to-point scenario of executing Web services. In this scenario, a Web service offered by a provider is directly called by a consumer.
- The following specific concepts related to the invocation of synchronous Web services are also described:
  - mySAP as Service Provider and Service Consumer
  - Data Conversion and Custom Modifications
  - Data Access
  - Exceptions and Errors
  - Security

  Some of these concepts are not necessary to understand the services contained in the enterprise services packages, but they provide a complete picture of what service provisioning means at SAP.
- Web Service Invocation via XI Message Broker: An alternative to the point-to-point scenario is to use the SAP Exchange Infrastructure (XI) integration server as a message broker between the consumer and the provider. XI is not required for executing the synchronous Web services offered in the enterprise services packages, but only for asynchronous services. This scenario is therefore described separately.

Runtime Architecture Overview

The mySAP Business Suite is a family of applications running on SAP Web AS. In order to call the Web services offered by the mySAP Business Suite, another application acting as service consumer sends SOAP requests to SAP Web Application Server, which acts as a service provider. The following figure shows an overview of such a system.

Web Service-Enabled mySAP Business Suite



The applications which form the mySAP Business Suite can be distributed across different instances of SAP Web AS. This fact is omitted in the described architecture, as this bluebook does not focus on the implementation of SAP Web AS. It only describes components as necessary, to explain the execution of a Web service.

Structure

In order to allow the execution of Web services, two layers of functionality in SAP Web AS are required as shown above:

- The HTTP Communication Layer
- The Web Service Enabling Layer

Process

The majority of Web services provided by the enterprise services packages are invoked synchronously.

For more information on this process, see Invocation and Execution of Synchronous Web Services.

Note

The enterprise services packages for the mySAP Business Suite on SAP NetWeaver 2004s focus on providing synchronous Web services. However, Web services provided or called by an mySAP application need not be invoked synchronously. The underlying technical infrastructure of SAP Web AS also supports asynchronous service invocations. However, in order to use them reasonably, a networking infrastructure providing reliable messaging is required (see Web Service Invocation via XI Message Broker).

.

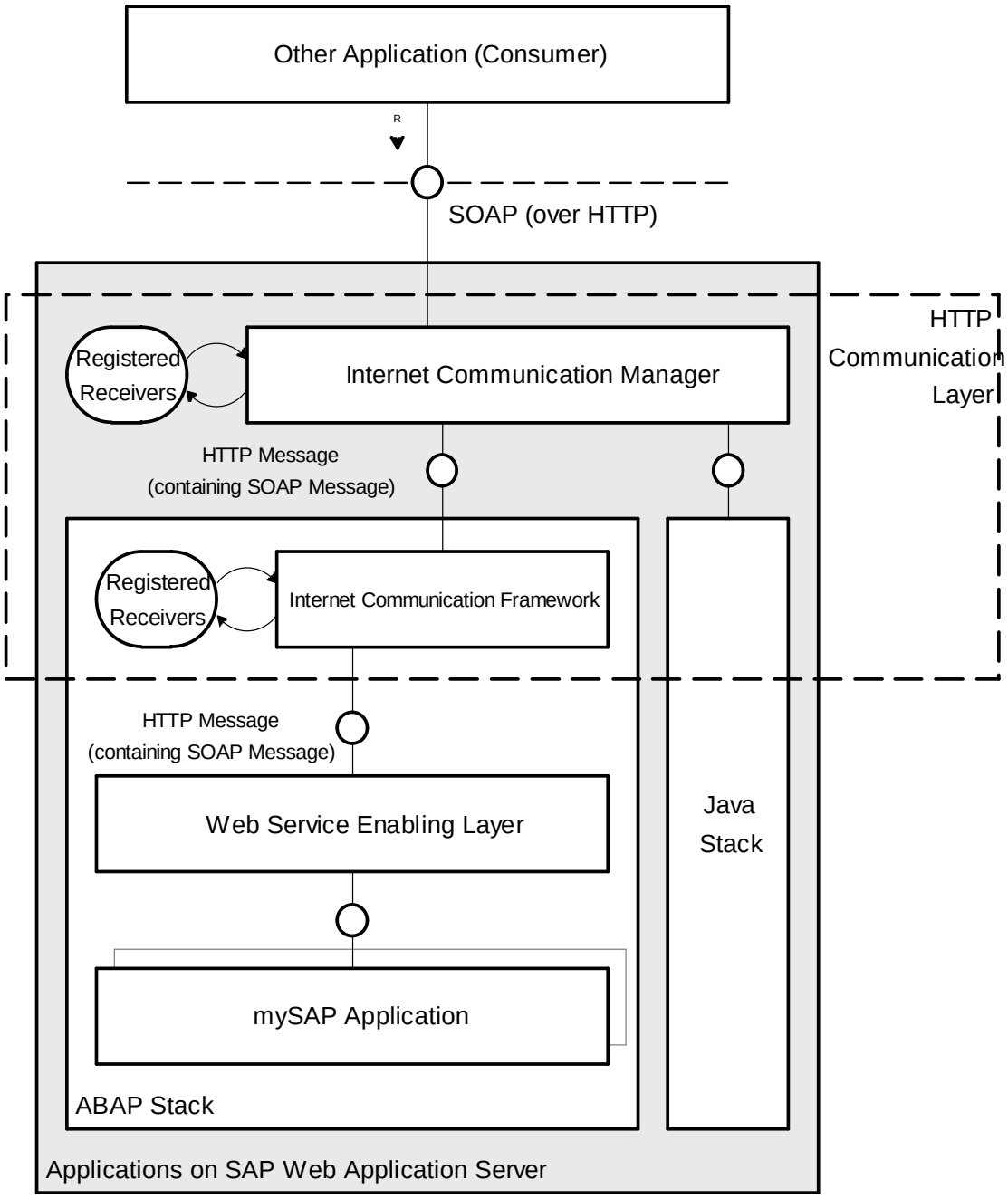
HTTP Communication Layer

The HTTP communication layer is one of two layers of functionality in SAP Web AS required for the execution of Web services.

The HTTP communication layer of the SAP Web AS handles all HTTP-based communication, which includes Web service communication. SAP Web AS can, for example, also act as a regular Web server, providing HTML Web pages. This layer is necessary for Web service invocations, because currently at SAP, all SOAP messages are sent via HTTP.

The main task of the layer is to accept incoming HTTP requests and dispatch them to the appropriate receiver within SAP Web AS. Dispatching is performed in two consecutive steps by the Internet Communication Manager and the Internet Communication Framework.

HTTP Communication Layer



Note

According to the W3C recommendation, SOAP does not necessarily need to be based on HTTP, but can use other transport protocols, such as SMTP. However, as transport protocols other than HTTP are rarely used, SAP Web AS only supports SOAP communication via HTTP or HTTPS.

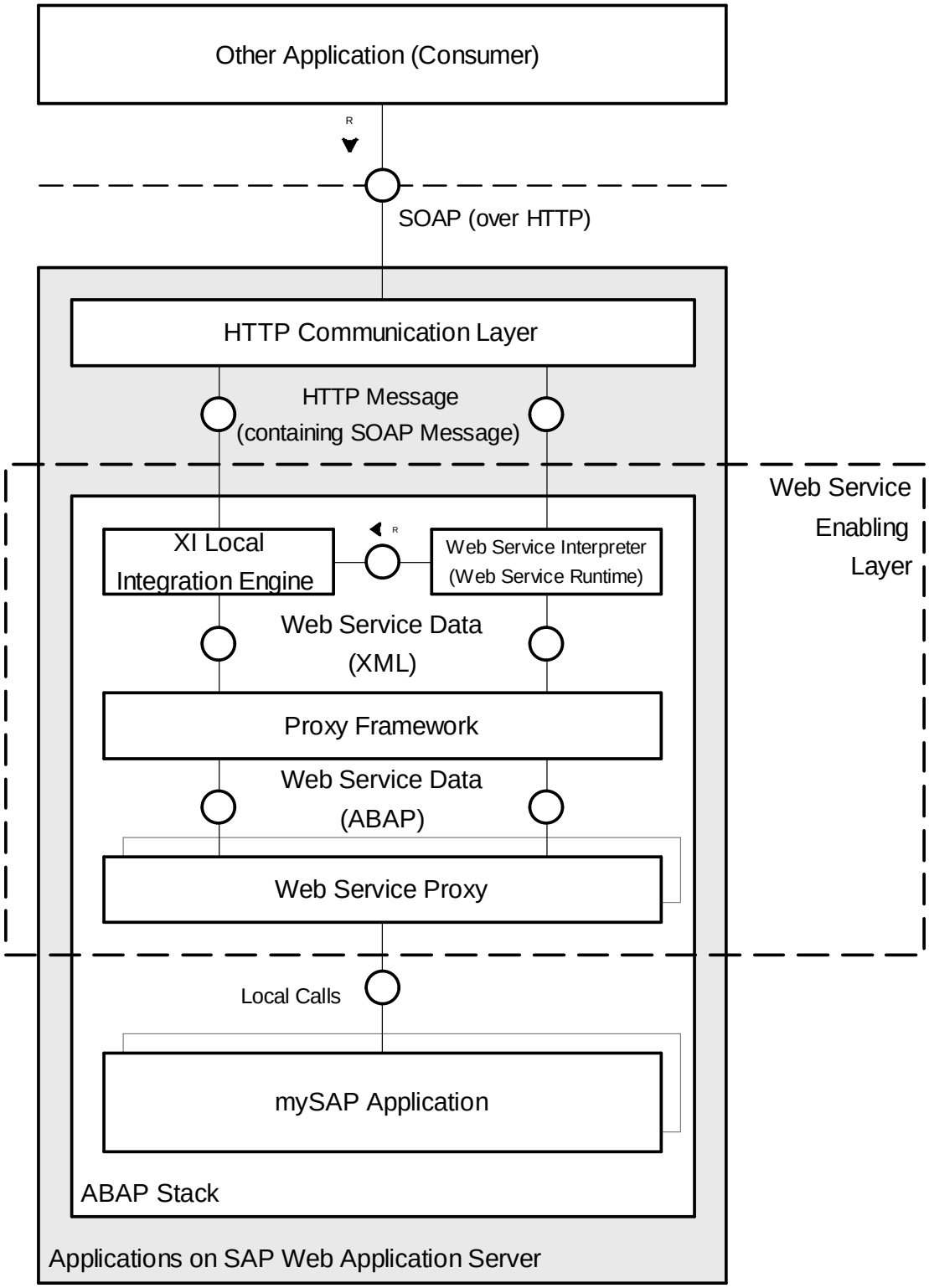
.

Web Service Enabling Layer

The Web Service Enabling Layer is one of two layers of functionality in SAP Web AS required for the execution of Web services.

The Web Service Enabling Layer receives HTTP-based SOAP messages from the HTTP Communication Layer. It interprets the SOAP messages and extracts the parameters for calling a Web service. The layer also contains the Web service *proxies* (as seen in the figure below), which are the implementation components that implement the Web service interface by forwarding calls to existing mySAP applications. In addition they can provide new functionality.

Web Service Enabling Layer



In addition to the Web service interpreter for interpreting point-to-point Web service calls, the figure above also depicts the *XI Local Integration Engine*. It is the counterpart of the Web service interpreter if Web services are executed via the XI infrastructure (see Web Service Invocation via XI Message Broker). However, it is also used in the point-to-point scenario, as it provides some functionality, such as logging, that is used by the Web service interpreter.

The Web Service Enabling Layer fulfills two tasks:

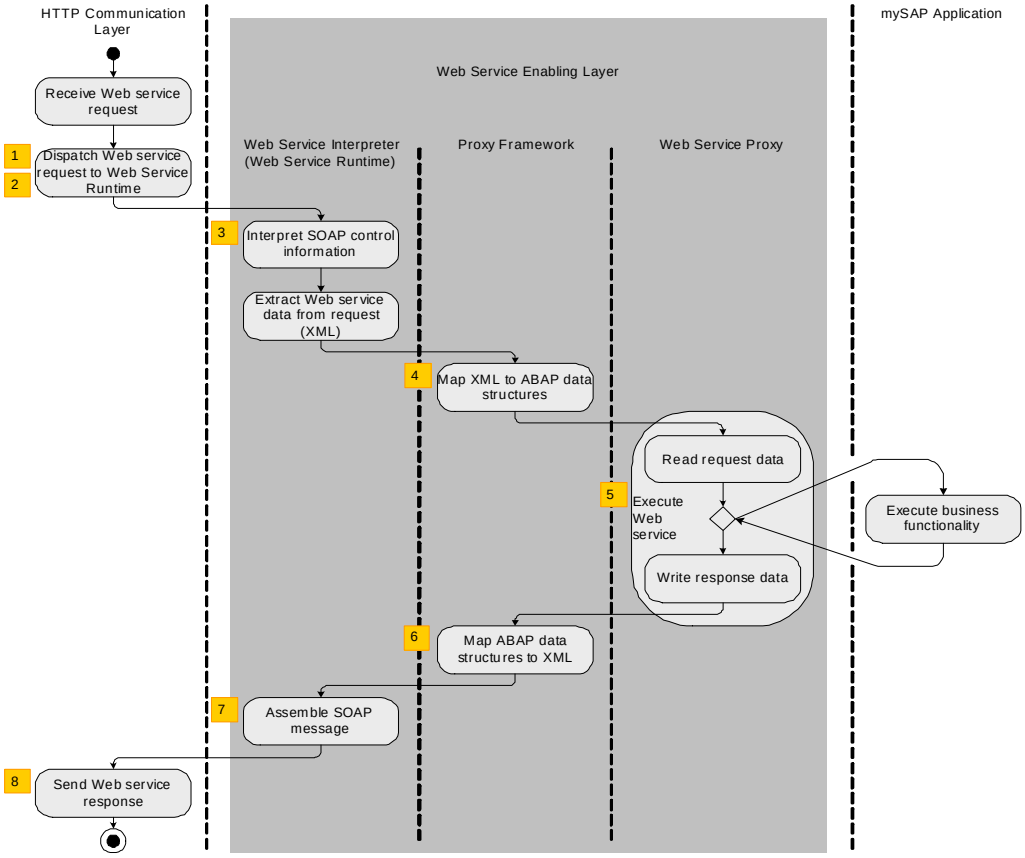
- Interpreting the SOAP messages and mapping the contained Web service data to a format that can be processed by existing applications on the SAP Web AS.
- Providing the Web services, meaning that this layer contains the implementations of the Web service interfaces. Such an implementation builds on existing functionality provided by mySAP applications.

Invocation and Execution of Synchronous Web Services

The majority of Web services provided by the enterprise services packages are invoked synchronously.

The following figure shows the execution of a synchronous web service. For the sake of clarity, the Web service consumer and the components of the HTTP Communication Layer are omitted.

Invocation and Execution of Synchronous Web Services



Process

The execution of such a service takes eight steps. Steps one to five deal with dispatching the incoming message, mapping the contained data, and triggering the appropriate Web service proxy. The remaining steps are required to return the Web service result to the service consumer.

- Dispatching a request (steps 1–3)
- Processing a request (steps 4–5)
- Sending a response (steps 6–8)

The steps described are processed in every invocation of a Web service offered by the enterprise services packages.

Dispatching a Request

Procedure

1. The Internet Communication Manager (ICM) decides whether to forward an incoming request to the ABAP stack or the Java stack. All services contained in the enterprise services packages are implemented in ABAP, which is why a request for the execution of such an enterprise service is forwarded to the ABAP stack

2. The Internet Communication Framework (ICF) receives all HTTP requests which are sent to the ABAP stack and forwards the request to the appropriate application. SOAP requests for the execution of Web services are forwarded to the Web Service Enabling Layer.

   The decisions made in both steps are based on the URL which is in the request. Each receiver is registered for a specific URL, or for a part of a URL, to specify the type of requests that it can accept.

   To determine whether the ABAP stack or the Java stack is addressed, the Internet Communication Manager analyzes the first part of the URL, which identifies the virtual host, for example https://www.myCompany.com:1234/WebServices/myService. The Internet Communication Framework analyzes the rest of the URL, for example https://www.myCompany.com:1234/WebServices/myService, to determine the application that is to receive the request.

   The HTTP Communication Layer does not modify the HTTP messages it forwards.

Processing a Request

Procedure

All incoming Web service requests are processed by the components of the Web Service Enabling Layer in three sequential steps.

1. The Web service interpreter reads the SOAP message provided by the HTTP Communication Layer. It reads the information in the SOAP header and extracts the Web service data from the body of the message. The SOAP header is used to extend the standard SOAP protocol with information required for additional protocol services, such as reliable messaging. The Web service interpreter component, which is called Web service runtime in SAP Web AS, must interpret this information in order to provide the additional services.

2. The extracted XML Web service data is forwarded to the proxy framework. The proxy framework maps the data from XML to ABAP data structures and invokes the appropriate Web service proxy.

3. The Web service proxy provides functionality of the Web service by placing local calls to function modules, BAPIs and RFCs of mySAP applications. As a Web service proxy is a standard ABAP class, it can also contain further coding that adds to existing mySAP functionality.

Note

- Web service proxies can be used for both point-to-point and XI-based invocation of Web services.

- The Web service proxy described above is only one of the proxy types available in SAP Web AS and is the proxy for synchronous Web services in the inbound scenarios. For more information on all the proxy types available see Proxy Type.

.

Sending a Response

Procedure

The steps required to send a response are as follows:

1. When the processing of the Web service proxy is finished, the proxy triggers the proxy framework to map the response data from ABAP data structures to XML.

2. The SOAP message containing the Web service response is assembled by the Web service interpreter, meaning that it adds SOAP control information to the header of the message and Web service data to the body of the message.

3. The HTTP Communication Layer sends the response to the specified receiver as an HTTP message.

mySAP as Service Provider and Service Consumer

Web service-based communication in which the mySAP application acts as a service provider, is referred to as an *inbound scenario*. Communication in which the mySAP application acts as the service consumer is referred to as an *outbound scenario*.

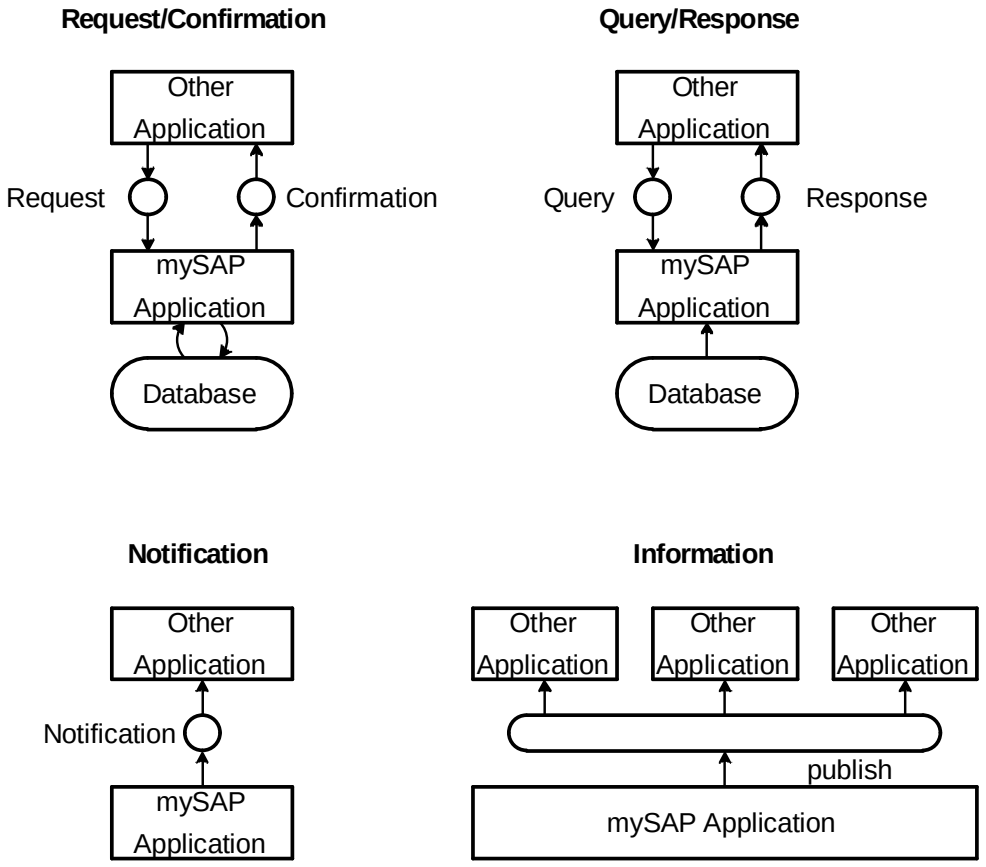
The communication patterns that can be used in these scenarios are described in the section Communication Pattern.

The Web service proxy described in the previous sections is only one of the proxy types available in SAP Web AS and is the proxy for synchronous Web services in the inbound scenarios. All the proxy types that are available are described in the section Proxy Type.

Communication Pattern

A communication pattern defines the choreography of a single interaction between the communication partners and defines which messages types are exchanged as request and response. Four patterns for communication between a mySAP application and another application have been implemented, as can be seen in the following figure.

Communication Patterns



The patterns can be categorized into patterns for transactional processing of data and patterns for data retrieval or information syndication. Patterns of the first category are used to change data of mySAP applications, while patterns of the second category are only used for retrieving data or information.

Transactional Data Processing

- *Request/Confirmation*: The basic pattern for transactional processing is called request/confirmation. It assumes that a Web service request is sent from another application to a mySAP application. The confirmation of the execution of the Web service is returned either immediately (synchronously) or later (asynchronously).

- *Notification*: A second pattern for transactional processing is called notification. Here, the other application sends a notification message to the mySAP application. This message leads to data changes. In contrast to the request/confirmation pattern, no response is sent.

Data Retrieval and Information Syndication

- *Query/Response*: The basic data provisioning pattern is called query/response. Another application requests some data from a mySAP application in a read-only

form, specified by a query. Query/response communication can be either synchronous or asynchronous.

- *Information*: A second data provisioning pattern is the information pattern, which informs other registered applications of data changes in a mySAP application. The difference to the notification pattern is that the information message is typically sent to a list of subscribed receivers, and the receivers do not take immediate action upon the information received. No response to the information message is expected.

All communication patterns can be used in the outbound scenario as well. Communication patterns are realized by invoking Web services. Therefore, an application which is a consumer (or provider respectively) in a communication pattern need not always be a consumer on the level of Web service invocations but may also provide Web services that are required for the communication pattern (see figure).

The example in the above figure shows the asynchronous realization of a query/response communication pattern. On communication pattern level, a consumer sends a query to a provider and receives a response. On the Web service invocation level both communication partner consume and provide Web service.

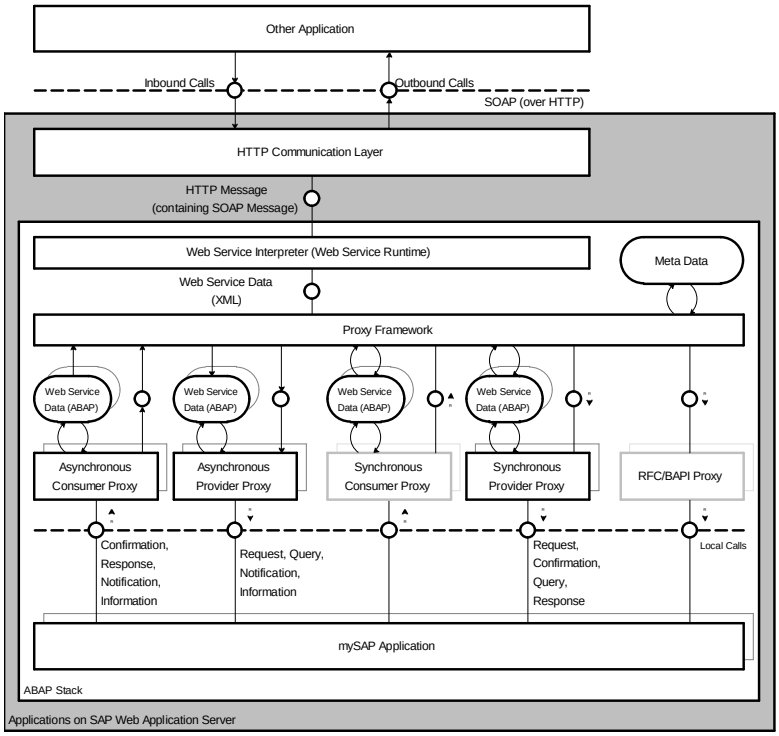
To distinguish between the two levels, in this document *inbound (or outbound) scenario* refers to the *communication pattern level* and *inbound (or outbound) call* refers to the *Web service invocation level*. An inbound (or outbound) scenario may comprise inbound and outbound calls.

Proxy Type

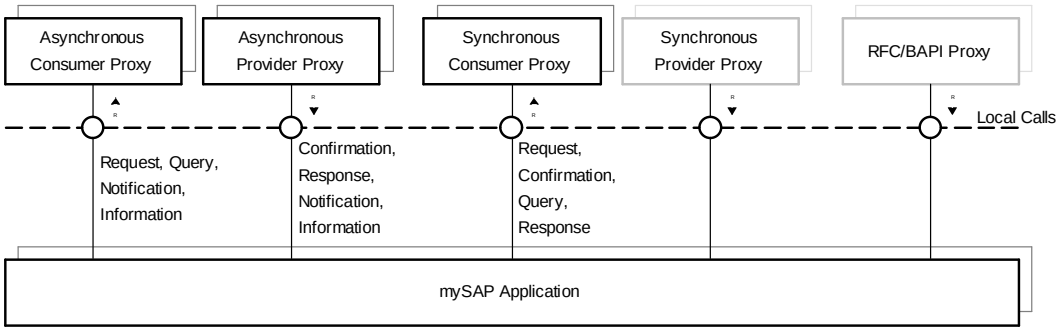
A Web service proxy, or proxy, is the implementation of a Web service interface in the Web Service Enabling Layer. Different types of proxies are used to implement the communication patterns, introduced in the previous section, in the inbound and outbound scenarios.

The first figure illustrates the proxy types used in the inbound scenario. The second figure shows a slightly modified version of the lower part of the first. It depicts the proxies used in the outbound scenario.

Proxy Types Used in the Inbound Scenario



Proxy Types Used in the Outbound Scenario



Note

The names of the proxy types used in this document are the names used in SAP NetWeaver NewYork. In SAP NetWeaver 2004s the proxy types are called client (consumer) and server (provider) proxy.

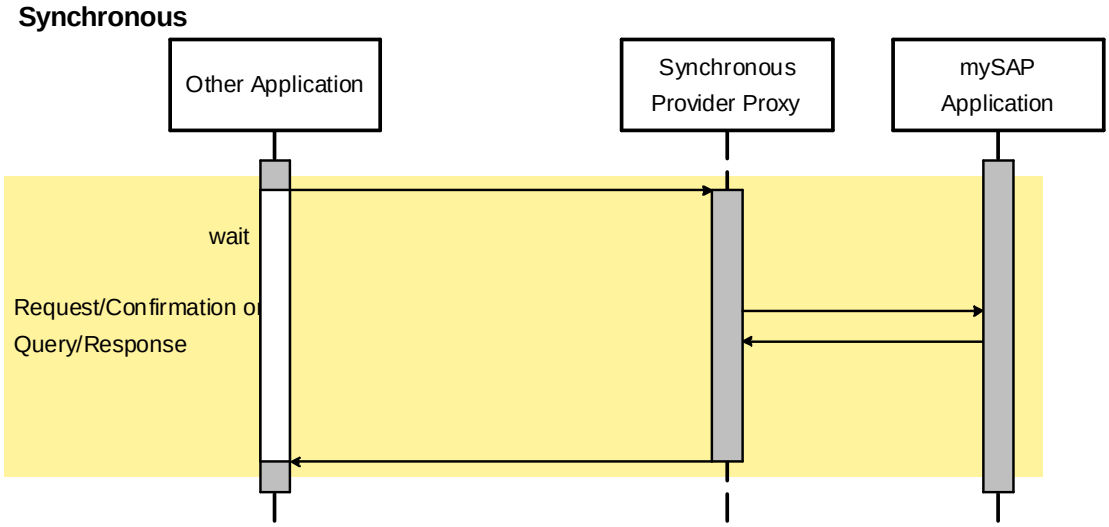
Proxy Type	Proxy
Asynchronous Proxies	These proxy types send or receive asynchronous Web service calls. • Asynchronous Consumer Proxy: This proxy type sends asynchronous Web service calls to a provider. • Asynchronous Provider Proxy: This proxy type receives asynchronous Web service calls. The Web services provided by the proxy are based on the functionality of existing applications within SAP Web AS.
Synchronous	These proxy types send and receive synchronous Web service calls.

Proxy Type	Proxy
Proxies	• Synchronous Consumer Proxy: This type of proxy sends synchronous Web service calls to a provider and waits for the response. • Synchronous Provider Proxy: This type of proxy receives synchronous Web service calls and sends a response to the consumer once processing has finished.
2 RFC/BAPI Proxies	This proxy type wraps an existing RFC/BAPI or function group as Web service. Therefore, an RFC/BAPI proxy also is a synchronous provider proxy. However, this proxy is depicted separately in Figure 2–7 Proxy Types used in the Inbound Scenario Note RFC/BAPI proxies are generated automatically without any further need of implementation effort. This is why they are not visible in the development environment. However, it is necessary to define a virtual interface for these proxies. This allows to hide specific parameters of an RFC/BAPI call in a Web service. It is, for example, possible to set default values for parameters which should not be available in the Web service.

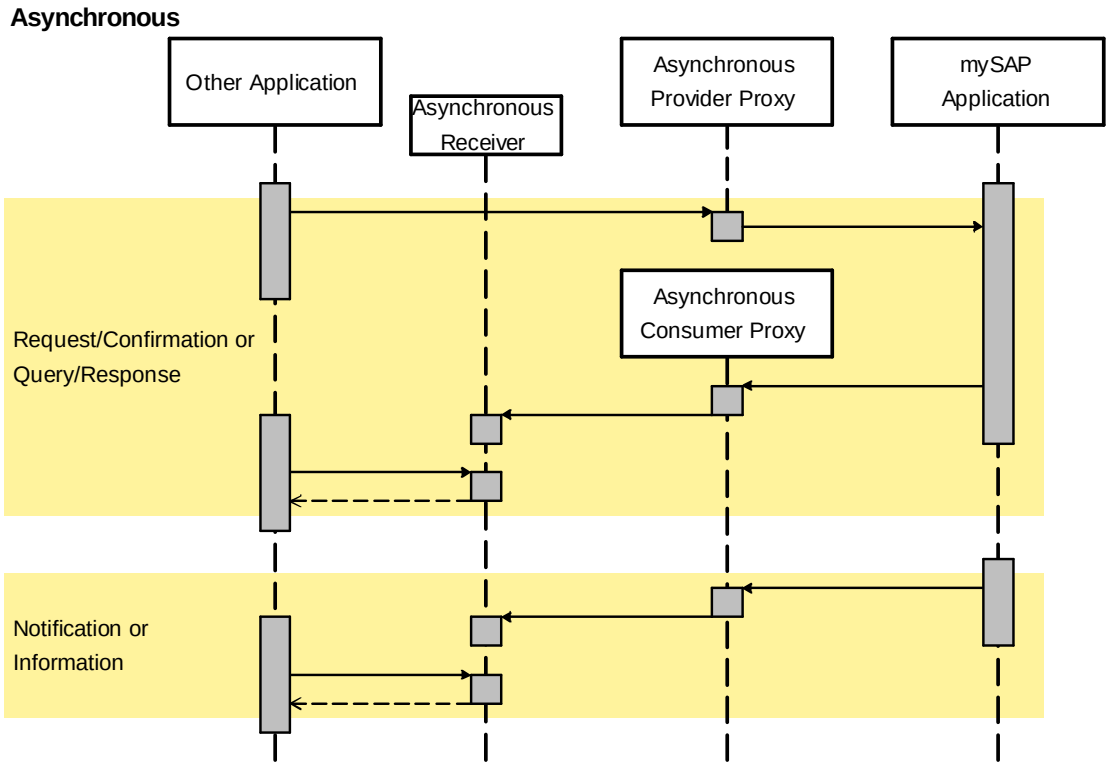
In the two figures above, the messages types (see Communication Pattern) that can be sent or received by each proxy type are shown next to the channels connecting the proxies and the mySAP applications. The usage of the proxies in the communication patterns are shown in the two figures below. All patterns, except the notification and information patterns, are depicted in the inbound scenario. If used in the opposite direction, the consumer/provider proxies have to be replaced with their counterparts.

The first figure below shows the communication patterns available in the synchronous scenario, while the second shows the communication patterns available in the asynchronous scenario.

Usage of Proxy Types in Synchronous Scenarios



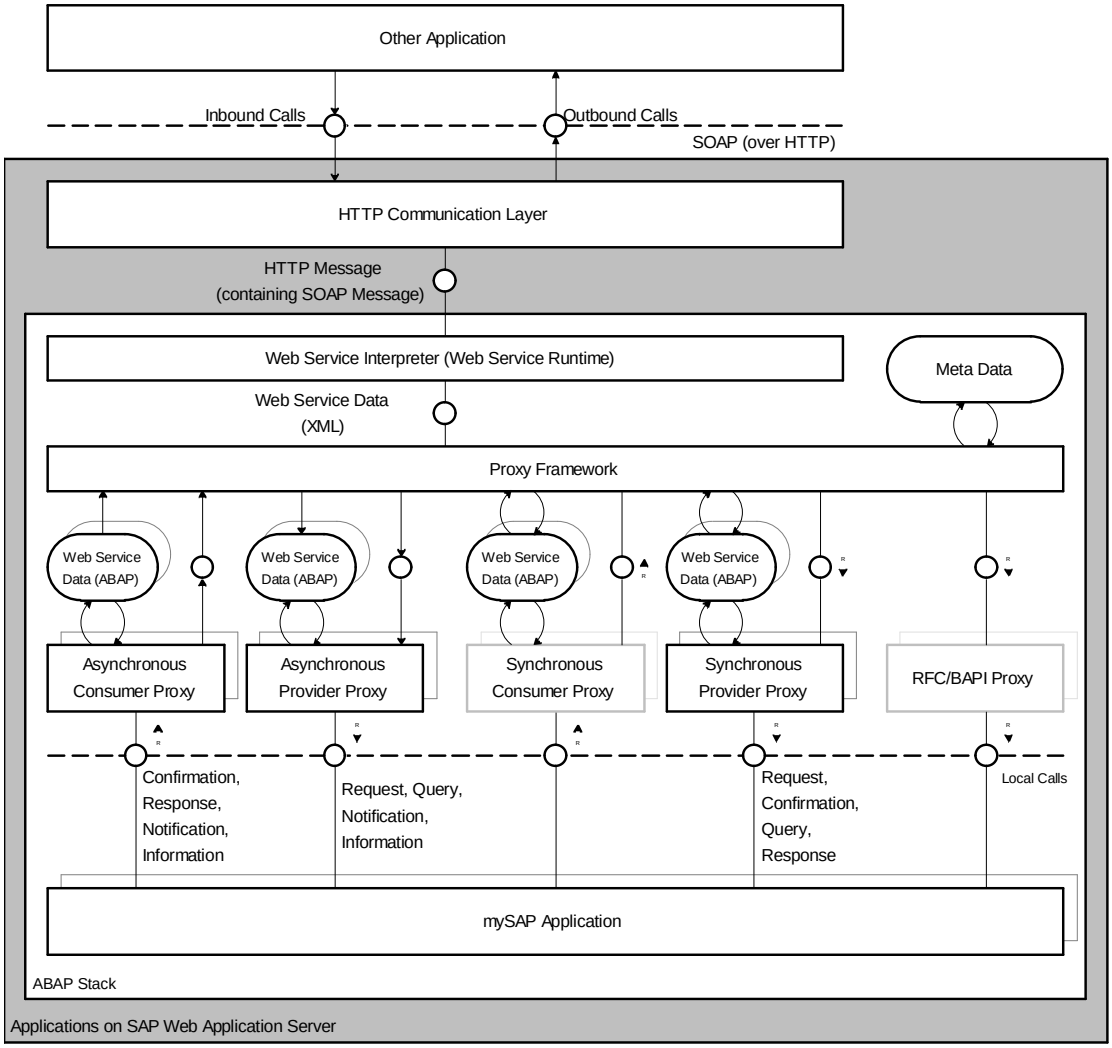
Use of Proxy Types in Asynchronous Scenarios



Data Conversion and Customer Modifications

The data contained in a Web service call is represented by XML data types. To allow ABAP applications to process the data, it has to be mapped to ABAP data types. The ABAP data of a Web service call is depicted by the Web service data storages in the following figure.

Proxy Types Used in the Inbound Scenario



Depending on the type of proxy, these storages contain either a request or a response only, or both requests and responses. The ABAP data structures implementing these Web service data storages are provided by the proxies. The proxy framework knows how to map the XML data types to an ABAP representation and vice versa.

- Proxy Meta Data: The information required to do the mapping is contained in the proxy meta data.
- Application Level Mapping of Data Types: The technical mapping between XML and ABAP performed in the proxy framework is not sufficient. A conversion of data types on the application level is required for specific data types.

Proxy Meta Data

The data contained in a Web service call is represented by XML data types. To allow ABAP applications to process the data, it has to be mapped to ABAP data types. The information required to do the mapping is contained in the proxy meta data.

The proxy framework creates mapping programs, called *Simple Transformations*, based on the given mapping information specified for the data types. For incoming messages, a mapping program uses the XML parameters as input and fills the ABAP data structure. For outgoing messages, a different mapping program with ABAP as input and XML as output is

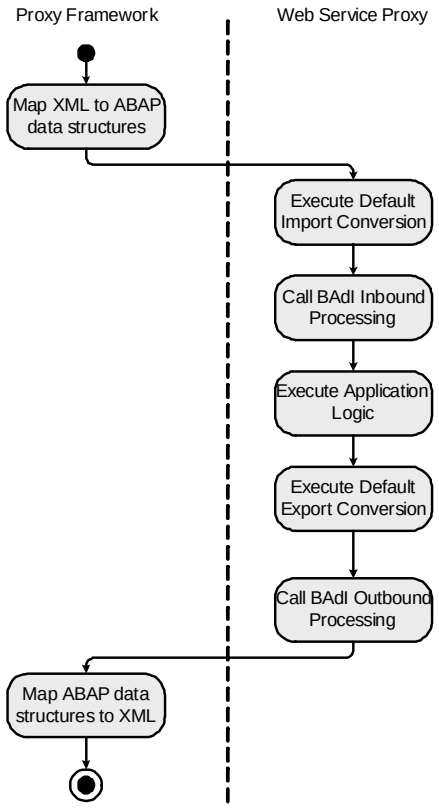
executed. The information contained in the proxy meta data is derived from the abstract service definitions during design time (see mySAP Service Provisioning: Design Time.

Application Level Mapping of Data Types

The technical mapping between XML and ABAP performed in the proxy framework is not sufficient. A conversion of data types on the application level is required for specific data types, too. Examples of such data types are codes, units, and time-related data types. In a Web service call, such data types are typically specified in a standardized format, such as ISO codes. However, mySAP applications have a different format. Therefore, the Web service proxy has to convert the data types into the correct format before it can begin to execute the application logic.

Process

Data Conversion



Business Add-In (BadI)

As the data conversion on the application level also affects data types that can be extended by customers, such as code lists, a Business Add-In (BAdI) is provided for adding custom import and export data conversion (see figure).

In addition to data conversion alone, the methods for inbound and outbound processing provided by the BAdI, can be used by customers to modify the behavior of a Web service.

Example

Per default, a Web service that retrieves information about all goods that can be ordered from a supplier by a retailer does not filter the returned information. However, business cases exist, in which the returned information must be filtered according to certain criteria. If, for example, the retailer is a store, only information about those goods should be retrieved that are part of the retailer's assortment of goods.

The BAdI for the import and export processing in Web service calls uses a sorter concept for the registered BAdIs implementations. Each implementation is categorized as SAP, partner or customer implementation. The implementations are executed in this order.

The BAdI methods can access and modify all parameters of a Web service, even parameters that have been added through custom extensions.

Note

The data structures implementing the Web service data (ABAP) storages in the figure *Proxy Types used in the Inbound Scenario* (see Proxy Type) are called *Data Type Proxies* in the development environment

Data Access

Consistency and correctness of the data in the database have to be ensured for all services in the mySAP Business Suite.

Services for the enterprise services packages for the mySAP Business Suite 2005 are stateless services. On the service level, no context information is stored on the provider side between service calls. This means that each service call is an atomic transaction. The execution of a single mySAP Business Suite service always transfers the database from one consistent state to another. It is not allowed for a service to finish with an inconsistent database state, requiring that further service calls will fix it. Therefore, it is not possible to have transactions on the database level that span several service calls.

The following consequences have to be considered:

- *Snapshot data*: Data retrieval and subsequent data modification cannot be combined in one transaction. Hence, data captured by stateless services is not real-time data. Since data is only locked as long as service processing takes place, the service consumer retrieves data as a snapshot taken at a certain point in time. For more information see Concurrency Control.
- *Reliable messaging*: Since the standard SOAP protocol does not provide a means for reliable messaging, protection against loss and multiple execution of web service requests has to be provided. For more information see Reliable Messaging.
- *Commit*: The work has to be committed to the database at the end of each service execution that involves data modification. For more information see Commit Procedures.

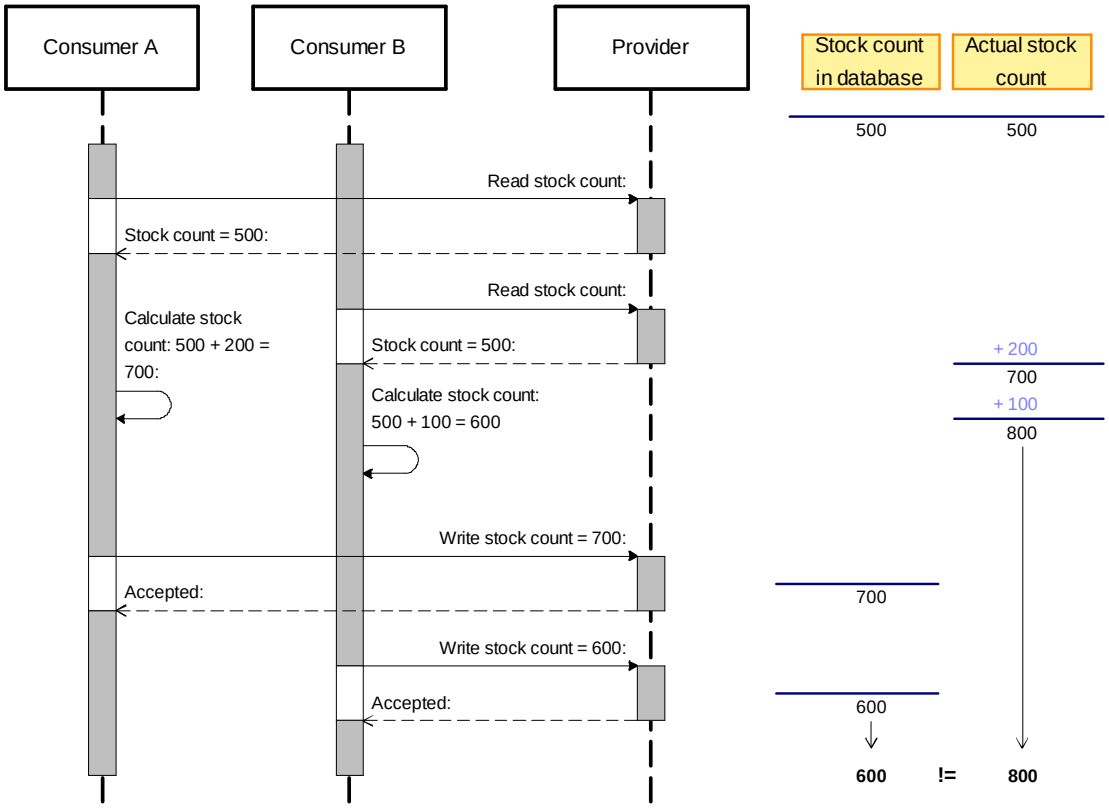
Concurrency Control

Each service in the enterprise services packages constitutes an atomic transaction. That is, a single service call always transfers the global data in the database from one consistent state to another.

However, though the database state is always consistent, it may still become semantically incorrect if multiple service consumers are interacting concurrently on a pair of services for retrieving or modifying a piece of data respectively.

Example

Overwrite Strategy Leading to Incorrect Database Content



Two consumer applications A and B are about to independently post the receipt of a delivery of screws. After querying the current quantity of screws in the warehouse, they calculate the new quantity of screws based on the current quantity and the delivery size. Subsequently they write the result in the sequence A-B.

If no precautions are taken, the new value in the database will be the amount calculated by application B, as B was the last one to write the value. Though there is no consistency problem, the value in the database is still not the right value from a business point of view.

To avoid such problems due to the interleaved execution of corresponding data retrieval and data modification Web services, it is necessary to implement adequate *prevention strategies*.

Basically, there are three different conciliation strategies which may be appropriate depending on the business semantics of the set of data in question. These are:

- Overwrite
- Delta Update
- Optimistic Locking

Overwrite

For the overwrite strategy, also called *last-one-wins* strategy, each database modification attempt is accepted and the data provided simply overwrites the old data in the database.

In the example in the previous section *Concurrency Control*, this strategy is applied and leads to incorrect data. However, there are business cases for which this strategy is appropriate. An example is the recording of the stock count as determined during a physical inventory. It is desirable that the result of the last physical inventory, which is the most recent stock count, is always stored in the database

Services for which the last-one-wins strategy is appropriate are called change services. The term “change” indicates that the service overwrites the old data with new up-to-date data, ignoring the old data.

Delta Update

For the delta update strategy, also called *everybody-wins* strategy, each database modification attempt is accepted as well, but in contrast to the overwrite strategy, the data provided by different consumers is accumulated in the database.

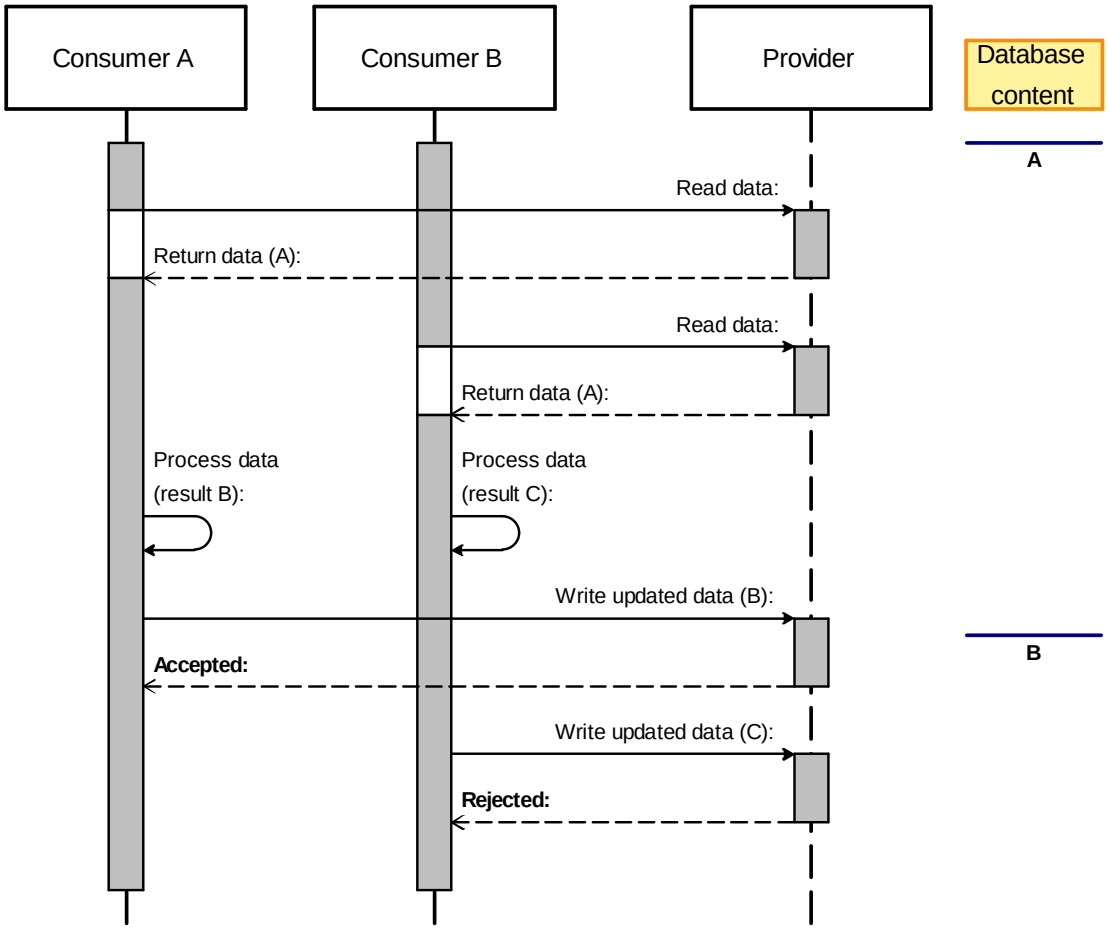
For instance, in the example in the previous section *Concurrency Control*, the applications could provide the changes to stock quantity instead of the new total stock count. All provided changes can then simply be added to the current stock count in the database with the database always containing the correct stock count.

The feasibility and implementation constraints of delta updates depend on the concrete business semantics of the service.

Optimistic Locking

For the optimistic locking strategy, also called *first-one-wins* strategy, only the first one of a sequence of consecutive database update attempts is successful, so the data provided in the first attempt will persist in the database, whereas further attempts fail. According to this strategy, the second database update attempt fails, and the second application is forced to query the current stock amount again and recalculate the new total stock amount.

Optimistic Locking Strategy



Update Services

The term optimistic locking refers to the fact that, in the first place, no locking actually takes place at all, assuming that in most cases no conflicts occur. Instead, for each modification attempt there is a check as to whether a conflict has occurred, and the modification attempt is rejected if need be.

Services for which the first-one-wins strategy is appropriate are called update services. The term “update” indicates that the service provides renewed data, the determination of which has been based upon the data read from the database before.

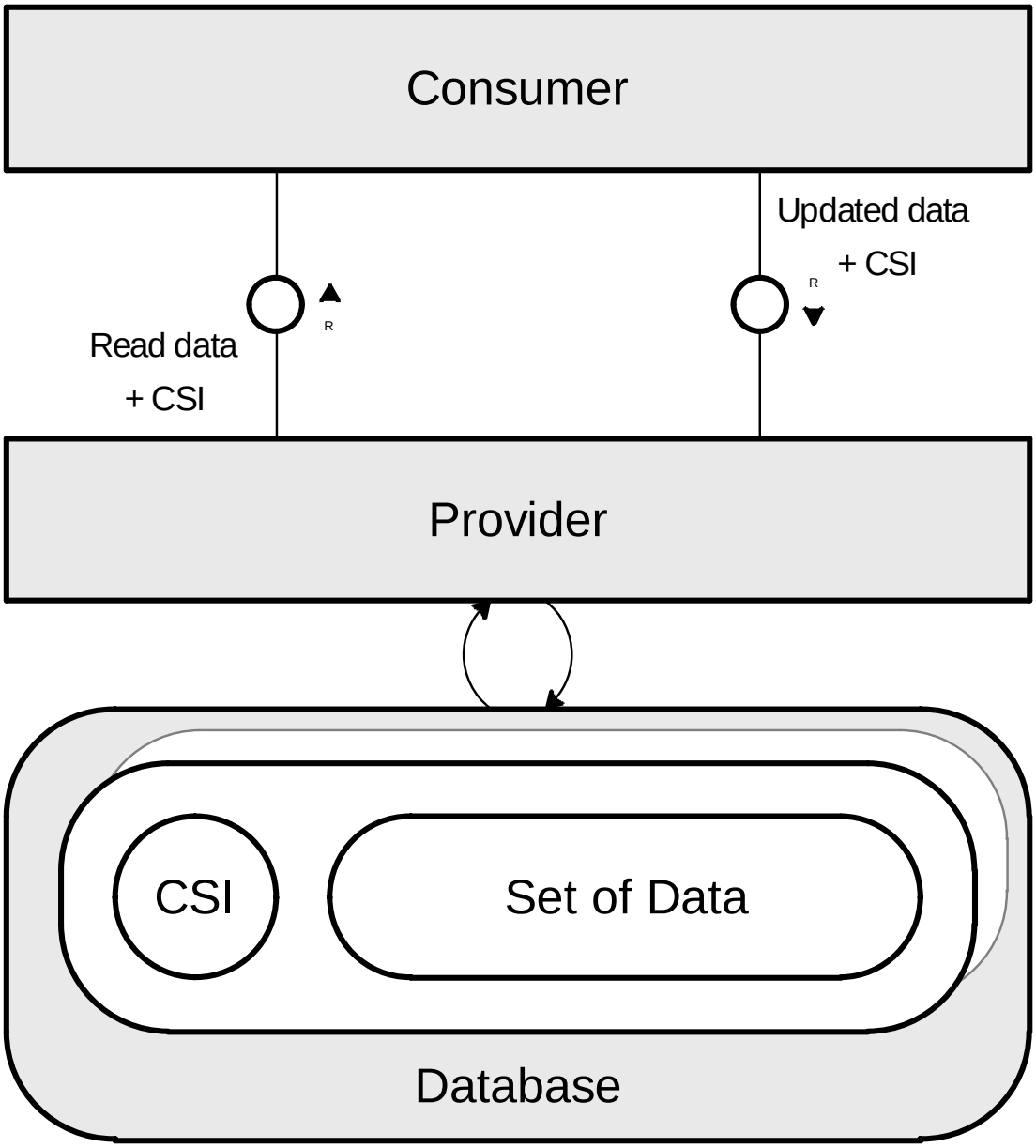
As explained above, the optimistic locking strategy is to ensure that, of all the service consumers which have read a certain database state, only the first one can commit its updated data. The same strategy can be expressed the other way round as well: Whenever a service consumer wants to store some updated data, it has to be ensured that the data it has read before is still the current data in the database. Otherwise it may have based its data processing on obsolete data.

Change State Identifier

To realize this, the service consumer must be able to indicate which data it has read originally. Therefore, an identifier is assigned to all pieces of data that are to be read, which uniquely identifies the state of the data. This identifier is called change state identifier (CSI). The change state identifier must be guaranteed to change with each database update.

As illustrated in the figure below, the consumer always receives the change state identifier together with the requested data. In a subsequent update request, the consumer can use the identifier to specify the state of the data which the update request refers to. If the specified state is no longer the current state, the update request is denied. In this case, the consumer has to read the new state and repeat the update request.

Change State Identifier



The sole requirement for a change state identifier is that it must change with each database update. For the enterprise services for the mySAP Business Suite, an existing database field will be used if an appropriate one is available (such as a change document number or the time stamp of the last document modification). Otherwise, a hash value generated from the current data is used as the change state identifier. As a hash value can be generated from the current data whenever needed, it does not need to be stored in the database. This is an advantage compared to other possible implementations such as an update counter or an update time stamp.

The consumer is not allowed to interpret the change state identifier. For the consumer, it is a token in the form of an arbitrary character string without semantics, because this allows for the underlying implementation to be changed later on.

Reliable Messaging

While SAP XI ensures reliable messaging for broker-based service calls via SAP XI (see Web Service Invocation via XI Message Broker), the standard SOAP protocol does not include any means of reliable messaging. Hence, in the point-to-point scenario, where Web services are called directly using the SOAP protocol, there is no guarantee that each request message sent by the consumer will arrive exactly once at the provider. It may get lost or may arrive several times in case of network problems. Even if the request message arrives at the provider, the response message may get lost during network transport as well. Hence the consumer might assume that its request did not arrive and re-send it.

Protection against loss and multiple execution of web service requests must therefore be provided. Three mechanisms are available. These are:

- Restartable Services
- Verification Services
- Idempotent Services

Restartable Services

It may be possible to implement some services in such a way that multiple invocations do not result in incorrect data in the database. For example, if a service that changes the address information of a customer is invoked several times with the same address as a parameter, this will still result in the correct address being stored in the database. Such services are called *restartable services* or said to have *restart capability*. In the case of restartable services, the consumer can repeat any request without causing the data to be incorrect. Change services are typically restartable.

For many services, however, it is crucial that the service is executed exactly once when requested. For example, if a new purchase order is to be created via a service invocation, it is neither acceptable to have no purchase order created nor to have two or more duplicates created.

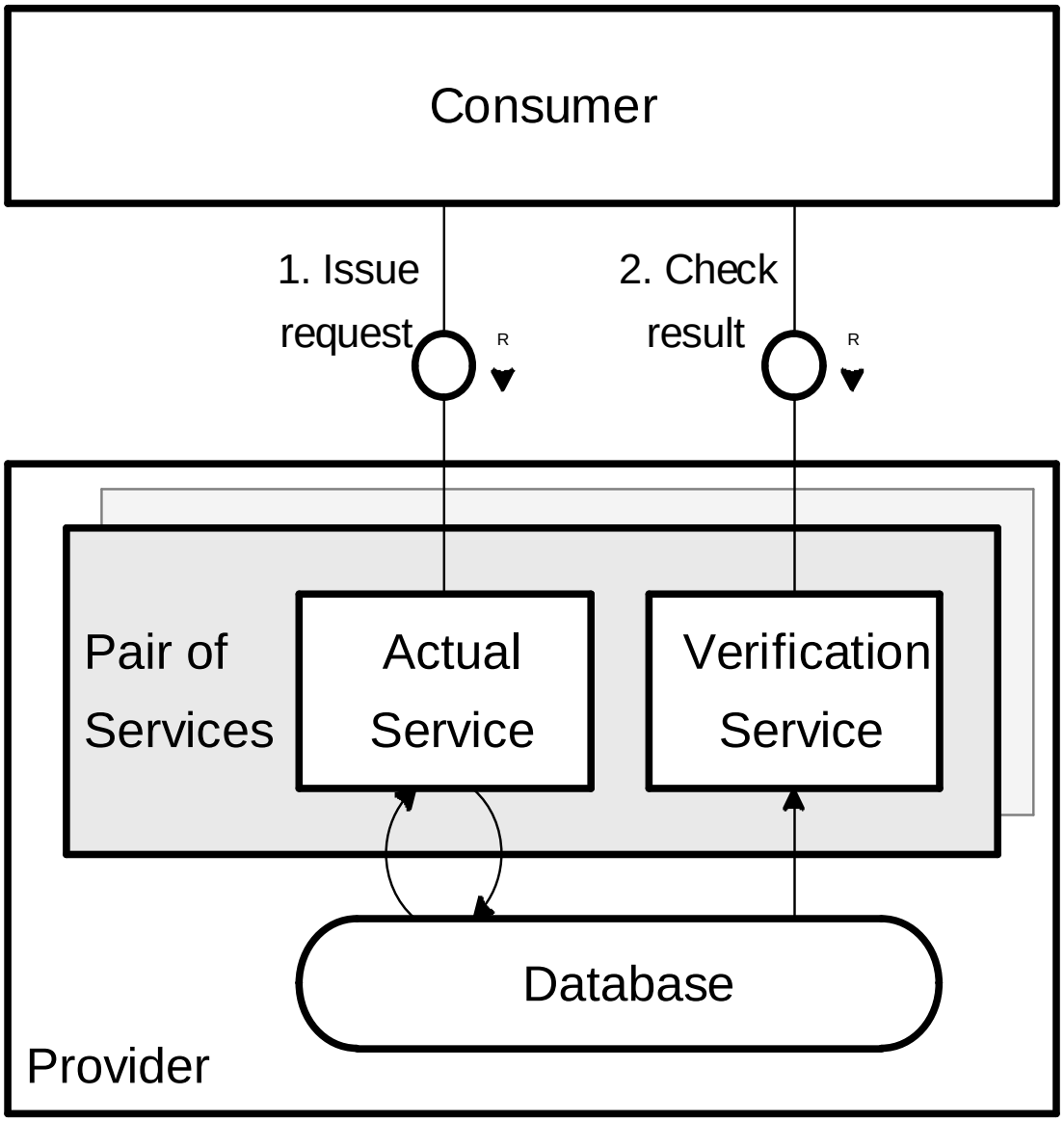
For these services, there are two mechanisms available for ensuring that a service is executed exactly once when it is requested by a consumer: Verification Services and Idempotent Services.

Verification Services

For many services, it is crucial that the service is executed exactly once when requested. For example, if a new purchase order is to be created via a service invocation, it is neither acceptable to have no purchase order created nor to have two or more duplicates created.

One mechanism, which can be used for the services in the enterprise services packages for the mySAP Business Suite 2005, provides the consumer with a way of ensuring that a service is executed exactly once when it is requested by a consumer. The idea is to create an additional service, a *verification service*, accompanying the actual service.

Verification Services



The verification service is a service that does not modify the database and has such semantics that it can be used to check whether a request to the actual service has been successful or not. In many cases, verification services do not need to be implemented explicitly, because an appropriate query service exists anyway.

In the example of a purchase order creation service, a verification service would retrieve purchase orders according to unique identifying parameters, such as a creation date within a certain time period and a specific creator.

Integration

This mechanism has two disadvantages:

- It may not be reasonable to create a matching verification service for every service.
- It requires the consumer to cope with up to twice the amount of services it actually needs, which implies some additional configuration effort.

Another mechanism for ensuring that a service is executed exactly once when it is requested by a consumer is provided by Idempotent Services.

Idempotent Services

For many services, it is crucial that the service is executed exactly once when requested. For example, if a new purchase order is to be created via a service invocation, it is neither acceptable to have no purchase order created nor to have two or more duplicates created.

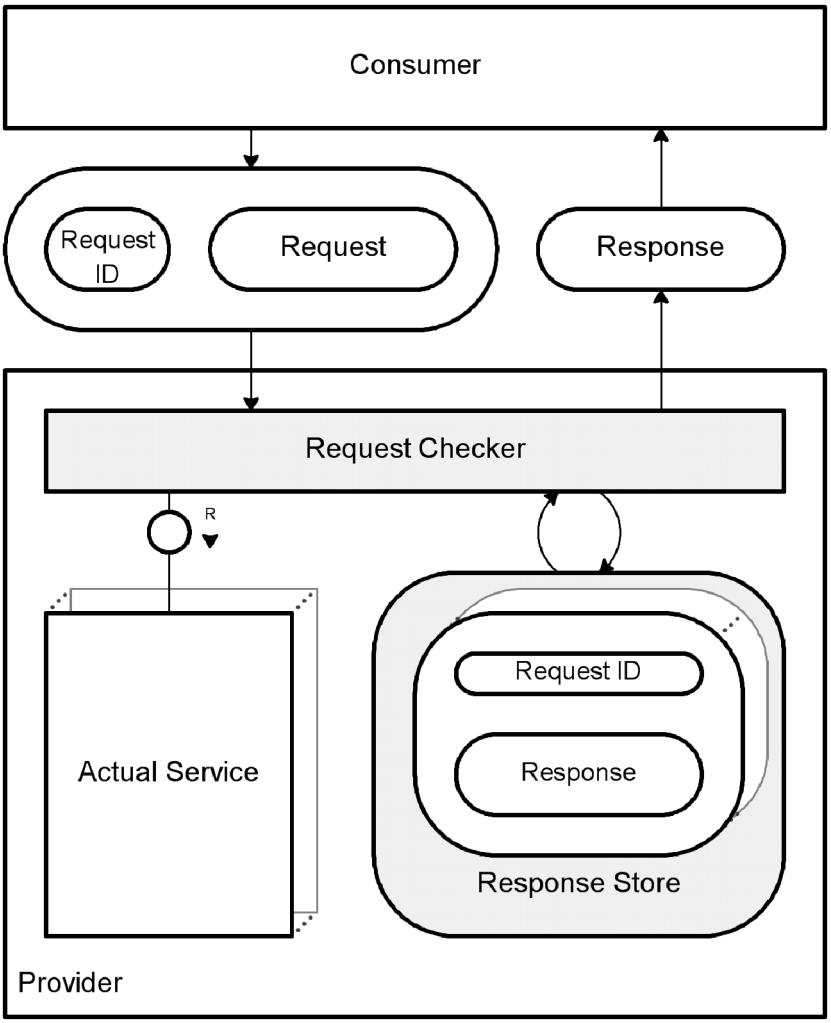
With Netweaver 2004S SPS09 / ECC SE 600 SP03, a framework is provided that ensures on the provider side that service calls are executed exactly once. Such services are called *idempotent services* or said to have *idempotency*.

The framework is used to store a message created in response to an incoming service call before the response is actually sent to the consumer. The response is identified by the request message's ID. For each incoming request, the provider can thus determine if the exact same request has already been processed by checking if a response for the request ID has been stored, and if yes directly return the earlier created response without really re-processing the request.

The figure below shows how a service provider implementation using the framework looks like conceptually ("Request ID" corresponds to the message ID).

The framework takes care of locking the response store, and of removing stored responses after a certain period such that the database isn't filled up unnecessarily.

Idempotent Services



Integration

See Important Facts for Service Implementers

Important Facts for Implementers

Implementation Considerations

The framework or the “request checker” as illustrated in the section Idempotent Services is not called automatically by the web service infrastructure or the local integration engine. Each proxy implementation that has to ensure exactly-once processing must actively call the framework.

You must manually configure the time periods governing for how long messages and information about sent messages are saved. You can find a description of how to do this in Configuring Idempotent Services.

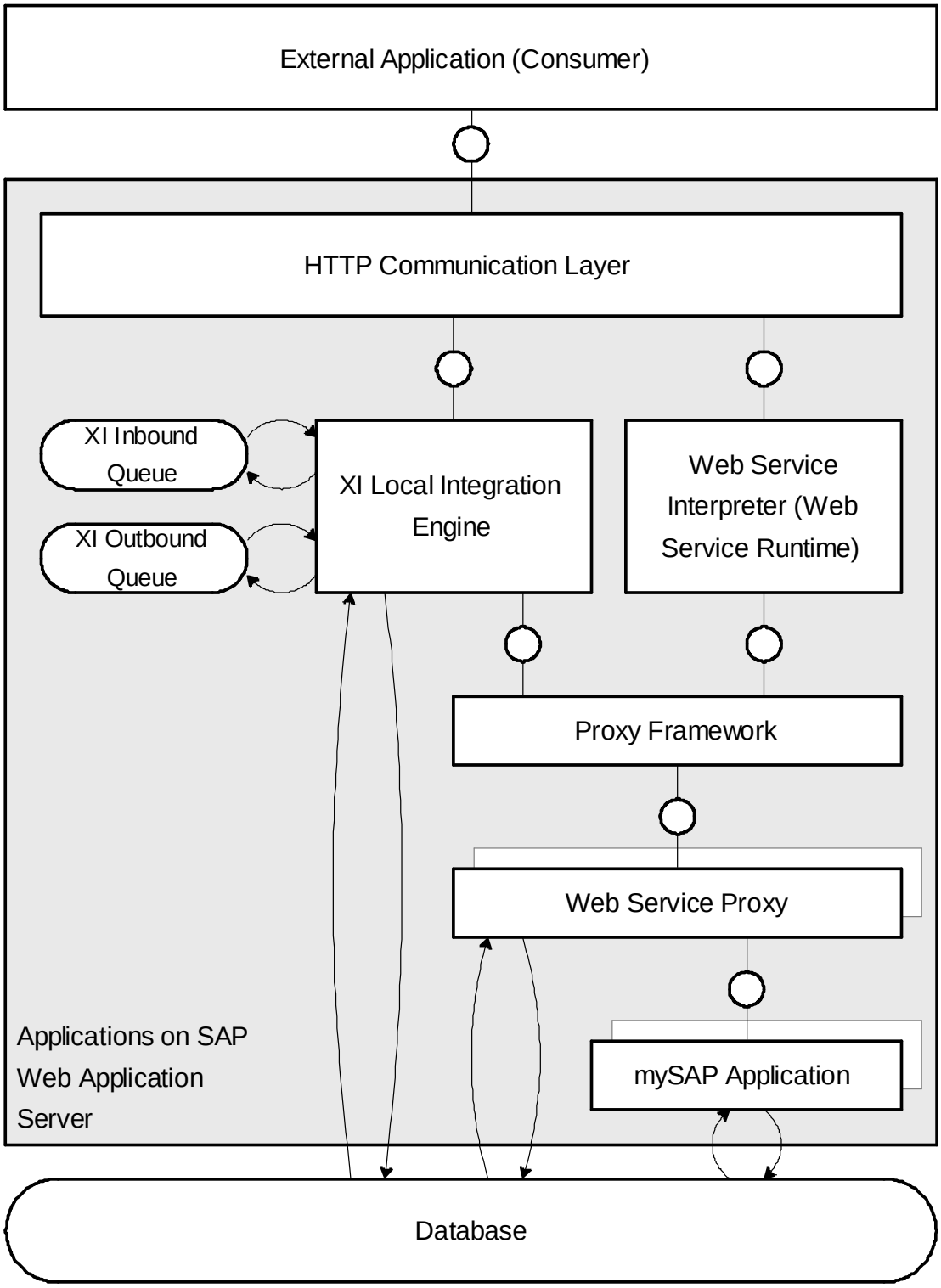
To employ services as idempotent services, you must use the MessageHeader element and its subelement UUID in the inbound and outbound message types. The UUID values must be globally recognizable and satisfy the format that is specified, for example, in IETF RFC 4122 (http://www.ietf.org/rfc/rfc4122.txt). If you do not use both elements when you call the services, the services do not behave in an idempotent manner.

Commit Procedures

The figure below shows all the relevant agents involved in the execution of a Web service call. As can be seen from the diagram, in addition to the mySAP Business Suite applications, two different agents have modifying access to the database: the *XI Local Integration Engine* and the *service proxy*. While the mySAP Business Suite applications need modifying access in order to write their data modifications to the database, these other two agents are responsible for committing the data modifications to the database in different scenarios.

- Asynchronous scenario: Commit by XI Only
- Synchronous scenario: No Commit by Framework

Commit Procedures



Commit by XI Only

Procedure

In the case of asynchronous service requests via SAP XI, the XI Local Integration Engine takes care of the transaction handling. It always opens a transaction before passing control to the proxy framework, and triggers a commit when the request processing is completed by the service proxy and control is handed back by the proxy framework.

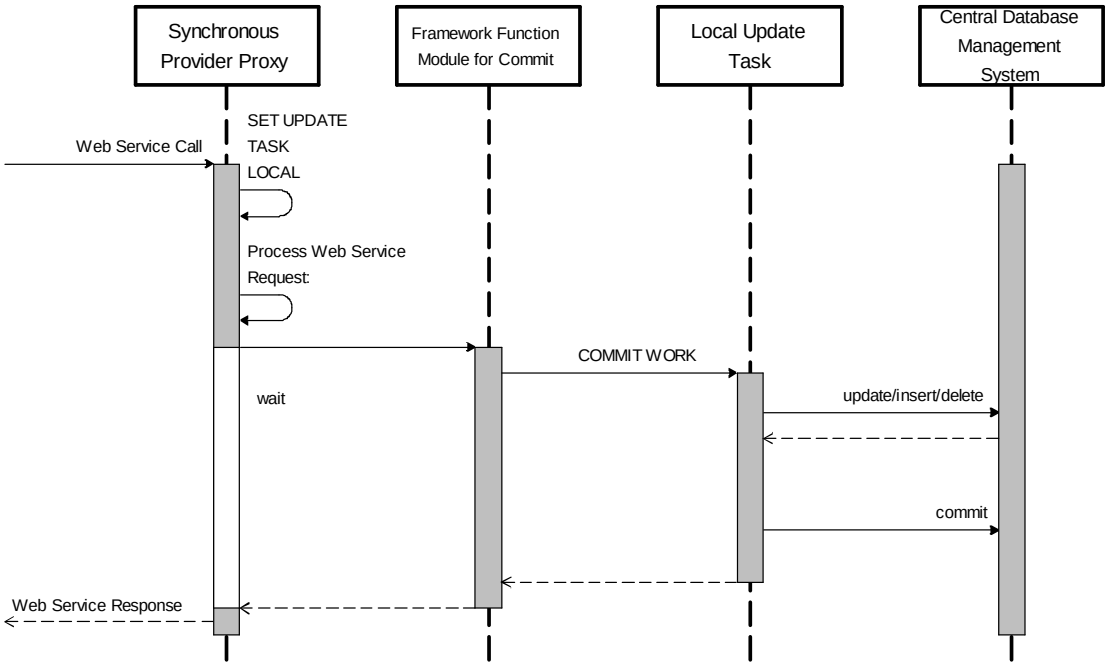
Therefore, asynchronous service proxies must not perform a commit or call any function module which implicitly does so.

No Commit by Framework

Procedure

For synchronous and point-to-point service calls, the commit is not carried out by any framework agent. Even if the service is called synchronously via SAP XI, the XI Local Integration Engine assumes that a synchronous call has the semantics of a query-response call, meaning that there is no modifying database access. Hence it will not trigger a commit. Therefore, in the case of synchronous services, the commit has to be implemented explicitly in the service proxy. The following figure shows the general process.

Regular Process Flow in Service Proxy Implementation



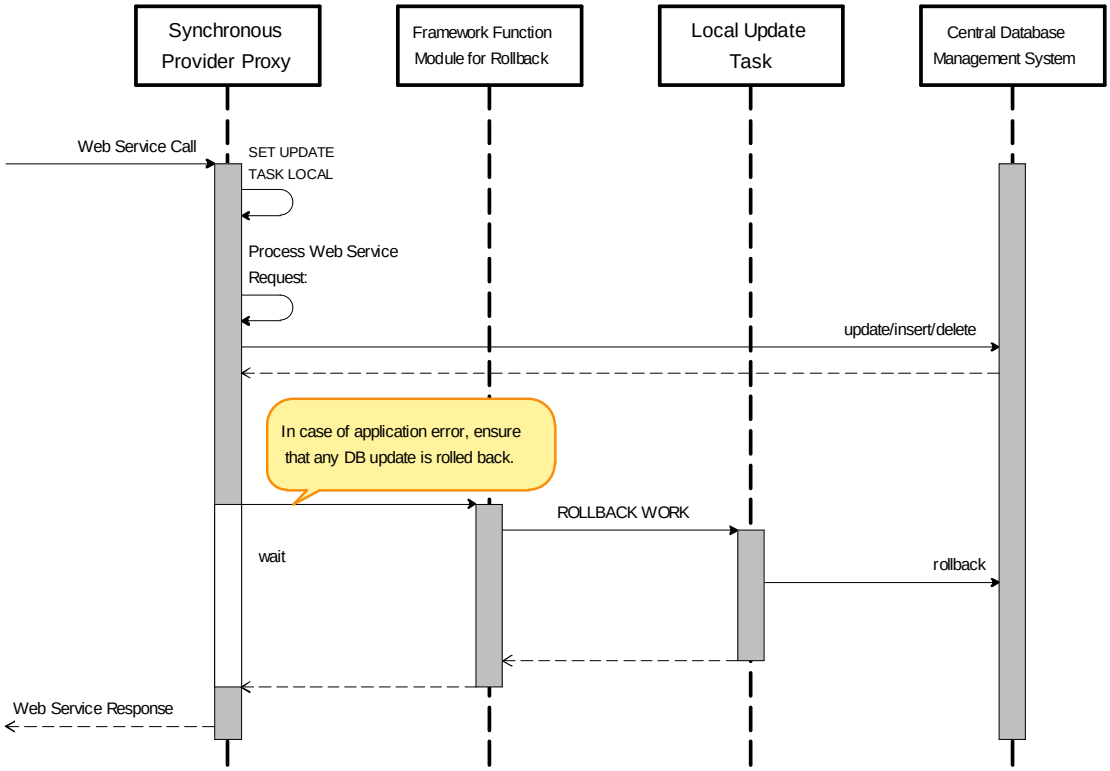
After the actual processing of the Web service request, the service proxy triggers the commit. However, it does not trigger the commit process directly. The invocation of the commit process is encapsulated in a framework function module. This provides the flexibility to alter the transaction handling later on without having to change each service proxy implementation. For example, it leaves room for providing general transaction handling in the proxy framework at a later stage.

At the beginning of processing, the service proxy calls the ABAP statement SET UPDATE TASK LOCAL. This causes the commit to be carried out in the so-called local update task, synchronously within the same operating system process (called “work process” in SAP Web AS). The advantage of a synchronous commit is that the commit is guaranteed to be completed before the response is returned. Hence, information about the success of the commit can be included in the response message and it is ensured that in a subsequent reading call up-to-date data is read. Additionally, no separate operating system process (called “update work process”) is needed, and thus overhead is reduced.

In some cases, direct database access during the actual processing of the Web service request cannot be avoided. For example, if a new object is created, it might need a unique identifier, so that it can be put in relation to existing objects. When such a unique identifier is to be drawn from a continuously increasing number schema without gaps, it must persist in the database immediately to make sure that nobody else assigns the same unique identifier

to another newly created object of the same type. This is depicted in the following figure by the direct call from the synchronous server proxy on the left to the database management system on the right.

Regular Process Flow in a Service Proxy Implementation



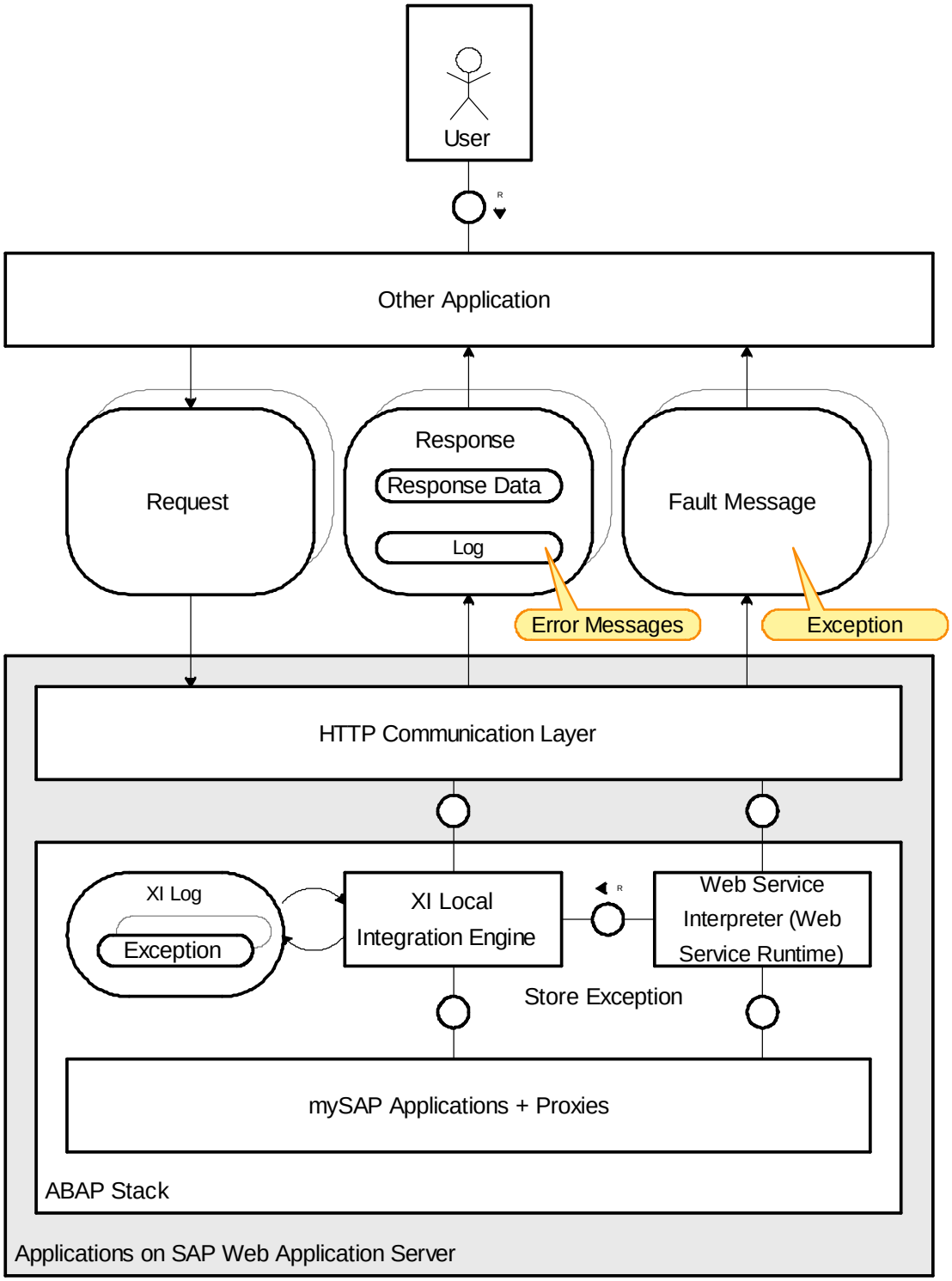
If an error occurs during subsequent processing, all database modifications of the current Web service execution need to be rolled back. For this purpose, the service proxy has to call a framework function module in case of errors. This function module takes care of the database rollback. Again, the rollback is encapsulated in a separate function module to allow for alteration of the transaction handling in the future.

Exceptions and Errors

During the execution of a Web service provided by mySAP applications, different types of errors can occur. Depending on the nature of the error, the other application is notified via either:

- An exception
- An error message

Exceptions and error messages are returned in different ways. Error messages are contained in the normal response, whereas exceptions are sent instead of the normal response.



Exception

An exception is raised when the error cannot be solved by the user, for example code errors, or inconsistent database data. Exceptions are sent instead of the normal response, unlike errors which are contained in the normal response.

XI Log

Exceptions are not only sent to the other application in the synchronous case, but are also stored in the XI log. This is especially important for invoking Web services asynchronously, as no response is sent in this case. This log is managed by the XI Local Integration Engine (see Web Service Invocation via XI Message Broker). The stored exceptions can be accessed and viewed at anytime by using the XI monitoring tools.

Fault Message

In the definition of a service, it is possible to specify fault messages which are sent back instead of the normal response message. Such a fault message is used to return an exception. A standardized fault message, called StandardMessageFault, is used for all Web services provided in the enterprise services packages. Again, in the asynchronous case the fault message is not directly sent to the calling application but is stored in the XI Log for later retrieval.

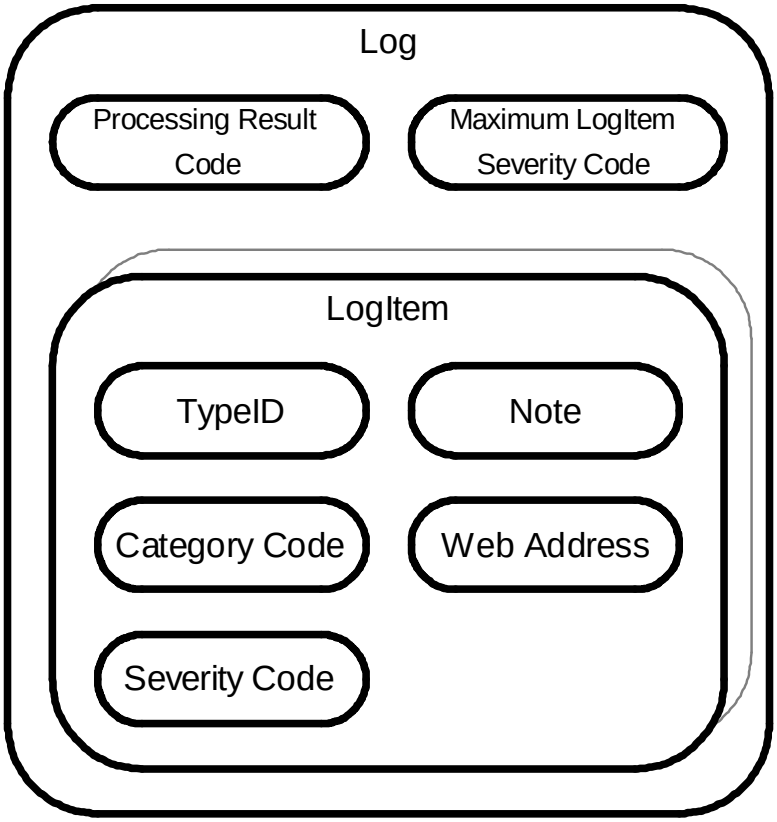
Error Message

Error messages are returned for problems that can be solved by the user, such as invalid input of data, authorization error, or locking error. Errors are contained in the normal response, unlike exceptions which are sent instead of the normal response.

Log Data Type

Including error messages in the normal response message is possible due to a convention. All return messages contain a specific data field, called log. This field is typed with a complex data type that is shown in the figure below.

Log Data Type



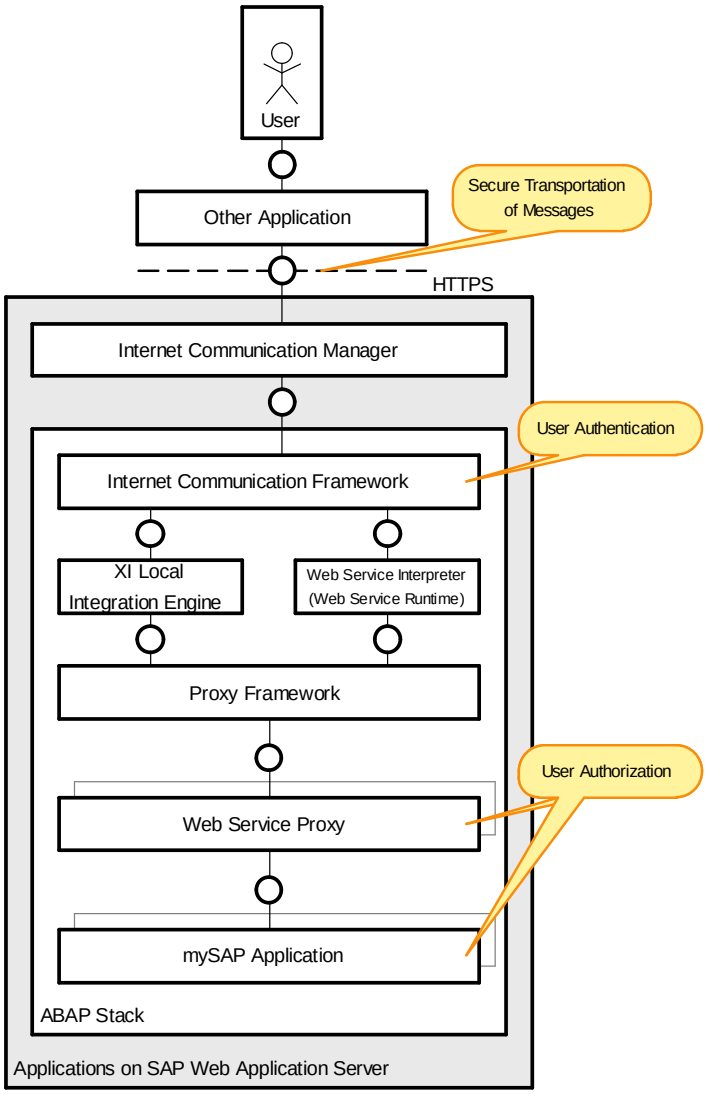
The processing result code specifies whether the Web service execution was successful, partially successful or failed. Depending on the code, the receiving other application can read and interpret the contained information and error messages. Each error message is transmitted as a single log item of the log. A log item can be assigned to an application specific type and category. The note of the log item contains the description of the error message. Any additional information about the error which is available on the Web is specified via the given Web address. Each log item is assigned to a severity level, which can be information, warning, error and abort. The highest level of all log items contained in a log is also stored in the log itself.

Security

The security of a Web service in mySAP Business Suite is accomplished in three areas:

- secure transportation of Web services messages
- authentication of Web service consumers in SAP Web AS
- authorization of Web service consumers within the mySAP Business Suite

Security in mySAP Web Services



More Information

As the subjects covered in these areas are not particular for Web services, the following sections only give an overview. For more information on security at SAP, see the SAP Service Marketplace at service.sap.com/security.

Providing Secure Transportation

This process is used to ensure secure transportation of Web service messages.

Process

To ensure secure transportation of Web service messages, a Secure Sockets Layer (SSL) connection for HTTP messages (HTTPS) can be used. This is the recommended option for secure message transportation in SAP NetWeaver 2004s. SAP NetWeaver 2004s partially supports the OASIS Web Services Security standard (XML Signature and authentication using UsernameToken). Future releases will fully support the OASIS Web Services Security standards, including XML-Signature and XML Encryption for securing the message content and Security Assertion Markup Language (SAML) 1.1 for single sign-on.

Providing Authentication in SAP Web AS

This process is used to authenticate Web service consumers in SAP Web AS.

Process

Consumers of ABAP Web services are authorized by the Internet Communication Framework. As the ICF only interprets HTTP and not SOAP, the information required for authentication must be contained in the HTTP part of a message.

SAP NetWeaver 2004s supports three different options for authentication of services:

- *Username and Password:* Service request messages contain the user name and password. ICF checks whether the user may access the system and the given password is correct.
- *SSL and Client Certificate:* This authentication option is based on a certificate that is used to prove the other application's (client) identity on the SSL connection from the other application to the SAP Web AS. The certificate contains a description of the identity of the other application and a public key that can be used by SAP Web AS to encrypt messages. With this key, SAP Web AS can establish the SSL connection with the other application and authenticate the user.
- *SAP Logon Ticket:* This authentication option is used for user authentication between different SAP Web AS instances. The different applications of the mySAP Business Suite may run on different instances of SAP Web AS. A user that is authorized in one instance of SAP Web AS does not have to authenticate itself again in another instance. Therefore, a ticket containing authentication information for Single Sign-On, called mySAP SSO2, is contained in the HTTP header of messages that are exchanged between SAP Web AS instances.

If the authentication of the Web service consumer is successful, the request is executed under this user's identity. This identity is used for authorization checks in subsequent calls to application functionality (see Providing Authorization within the mySAP Business Suite).

Public Web Services

It is also possible to define default users for Web services. In this case, the other application does not include any user data in the Web service message. The ICF assigns a predefined user to the incoming requests. Web services with default users are also called Public Web services, as everybody can access them without restriction.

Note

In future releases of SAP NetWeaver, other authentication options, such as user name and password in a SOAP message, XML-Signature and SAML will be supported. Some of these authentication options can only be provided by the Web service interpreter, as ICF cannot read the SOAP content of HTTP messages. For this reason, a standard user is assigned by the ICF during the normal authentication process. The Web service interpreter substitutes the real user for this standard user if the authentication is successful.

Providing Authorization within the mySAP Business Suite

This process is used to authorize Web service consumers within the mySAP Business Suite.

Process

Accessing mySAP functionality via Web services follows the standard mySAP authorization concept. This concept is based on authorizations for specific authorization objects. If a user does not have an authorization for an authorization object, which is checked during the execution of a Web service, the execution is terminated, and an error message is returned (see Exceptions and Errors).

Web Service Invocation via XI Message Broker

The enterprise services packages for the mySAP Business Suite on SAP NetWeaver 2004s focus on providing synchronous Web services. However, Web services provided or called by an mySAP application need not be invoked synchronously. The underlying technical infrastructure of SAP Web AS also supports asynchronous service invocations. However, in order to use them reasonably, a networking infrastructure that provides reliable messaging is required.

Features

- SAP Exchange Infrastructure (SAP XI)
- XI Protocol
- XI Local Integration Engine

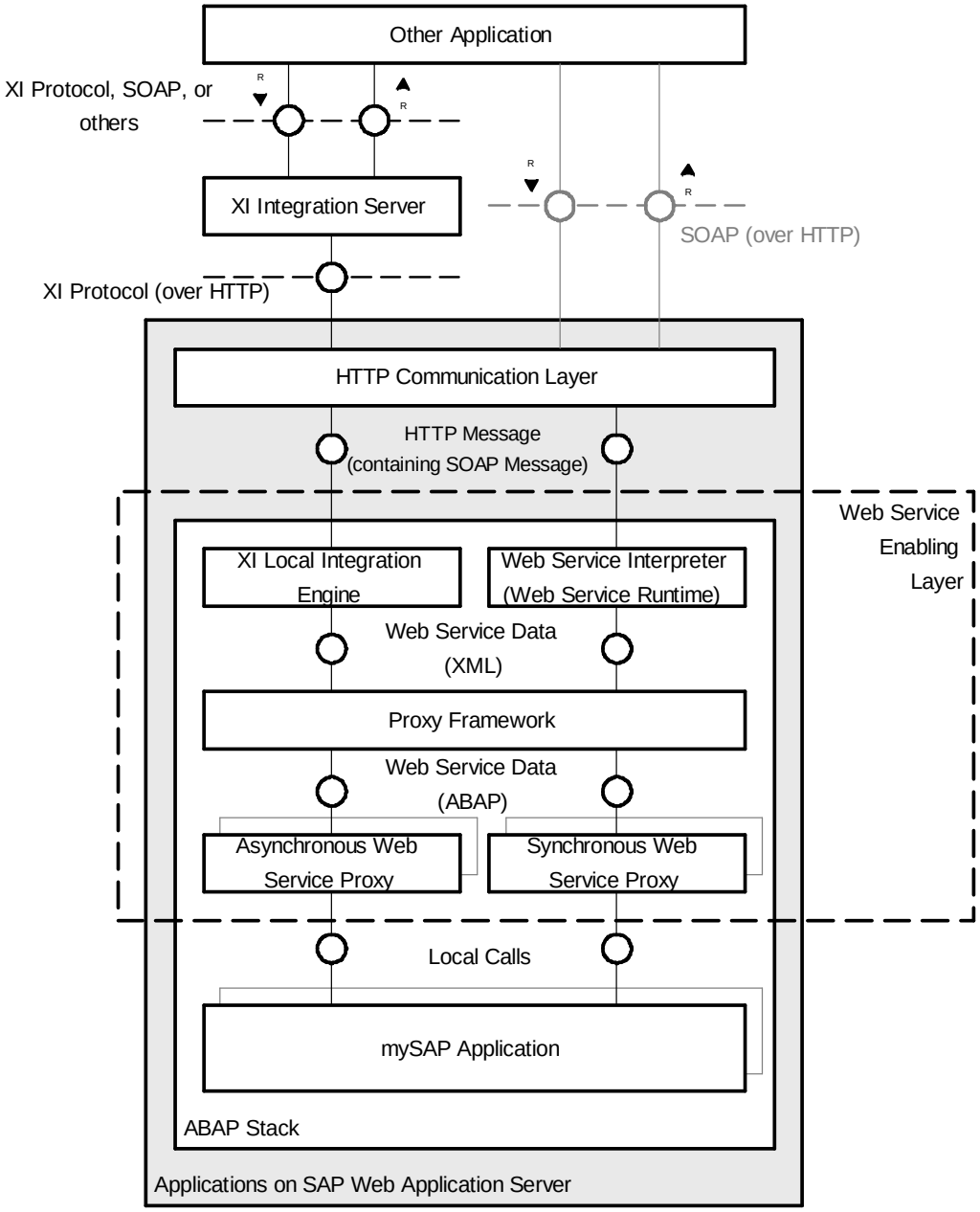
SAP Exchange Infrastructure

SAP Exchange Infrastructure (SAP XI) is an integration technology and platform. XI uses by default an extended SOAP protocol for connecting applications. In addition, other protocols, such as JMS, JDBC, RFC and IDOC, can be used with adapters. XI provides communication services such as message routing and mapping of messages between different formats and contains a component for cross-component business process management (ccBPM). This component makes it possible to model and execute business processes on XI.

SAP XI provides, among other things, reliable messaging for asynchronous communication. Therefore, the mySAP application does not implement such functionality on its own for the asynchronous invocation of Web services but requires SAP XI-based communication in this scenario.

The figure below shows the XI scenario for invoking Web services by refining the architecture introduced in the previous sections. The different types of proxies described in the section Proxy Type are not shown (for clarity reasons).

XI Based Web Service Invocation



More Information

For more information about SAP XI, see the SAP help portal at help.sap.com *SAP NetWeaver.*

XI Protocol

The XI Integration Server understands all common network communication protocols, and application protocols, such as HTTP and SOAP. In addition, it offers the specific XI protocol on the application level. This protocol is based on SOAP and enhances it with information required to provide network infrastructure services, such as message routing, reliable messaging, and message monitoring. These services are not available in point-to-point scenarios that rely on the standard SOAP protocol. By default, the XI Integration Server is accessed via the XI protocol. However, other protocols can be utilized by using adapters.

Communication between the XI Integration Server and SAP Web AS is always carried out using the XI protocol. As this protocol is an enhanced SOAP protocol and based on HTTP, the HTTP Communication Layer can also handle XI protocol messages without modification.

XI Local Integration Engine

The XI Local Integration Engine is the receiver of the XI messages in the Web Service Enabling Layer, instead of the Web service interpreter.

This local counterpart to the XI integration server speaks to the XI protocol and extracts the Web service data from the message. As the XI protocol also provides services such as reliable messaging, the XI Local Integration Engine does not only transform and forward the incoming message, but may also trigger the XI integration server to retransmit messages that have been lost.

Once the Web service data has been extracted from the XI message, the rest of the Web service execution is identical to the execution in the point-to-point scenario (see Runtime Architecture Overview).

Note

By using standardized extensions to the SOAP protocol, it is possible that some of the mentioned services, such as reliable messaging, can be provided in point-to-point scenarios. However, these standardized extensions are not supported in SAP NetWeaver 2004s, but in later releases only. The most common standardized extensions are Web Service Reliable Messaging, Web Service Addressing and Web Service Security.

mySAP Service Provisioning: Design Time

Services in the enterprise services packages are created in two steps during design time.

Process

First, implementation independent service interfaces are defined. These service interfaces are aligned through a common object model representing the business content of the mySAP Business Suite.

See Business Object and Service Modeling

In order to realize the defined services as Web services working on mySAP applications on the ABAP stack, corresponding ABAP entities have to be created and implemented.

See Proxy Generation in the ABAP Stack

Business Object and Service Modeling

This process is the first step in creating services in the enterprise services packages.

Process

This process involves the definition of implementation-independent service interfaces by creating a model of the business functionality to be offered, determining the functions to be realized, and deriving the appropriate service interface definitions.

- Business Object Modeling
- Service Interface Modeling

Governance Process

Since a sound model of the business content is crucial for SAP's success in service provisioning, SAP has set up a governance process for the modeling activities. The governance process ensures that domain experts from all concerned divisions of SAP have reviewed the model and that the model is harmonized across SAP as far as possible.

ESR

The approved service interface definitions are stored in the Enterprise Service Repository (ESR). The ESR is SAP's design time repository for reusable type definitions.

ARIS

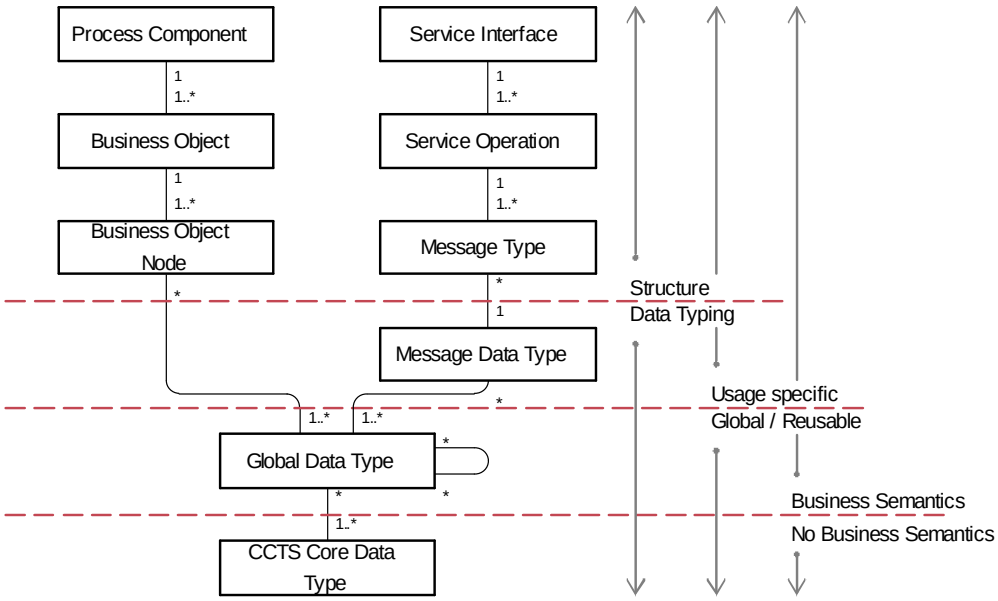
High-level models are kept and maintained in the modeling tool ARIS and can be viewed in the Enterprise Services Workplace in the SAP Developer Network (SDN) (at sdn.sap.com *sdn.sap.com* *ES Workplace*).

Business Object Modeling

This process is used to create a model of the business functionality to be offered, determining the functions to be offered.

The entities in the business model of SAP application functionality are shown in the figure below.

Entities of Service Interfaces and Business Object Models



Process

Process components and business objects are modeled in the ARIS modeling tool and can thus be viewed in the Enterprise Services Workplace. The internal structure of the business objects is only modeled SAP-internally and not released to the customers.

1. Process Component
2. Business Object

Process Component

A process component comprises all functionality offered by a service-oriented SAP application that realizes a certain business process, where a business process is regarded to be a sequence of activities transforming a defined business input into a defined business outcome.

Integration

A set of business objects belongs to each process component.

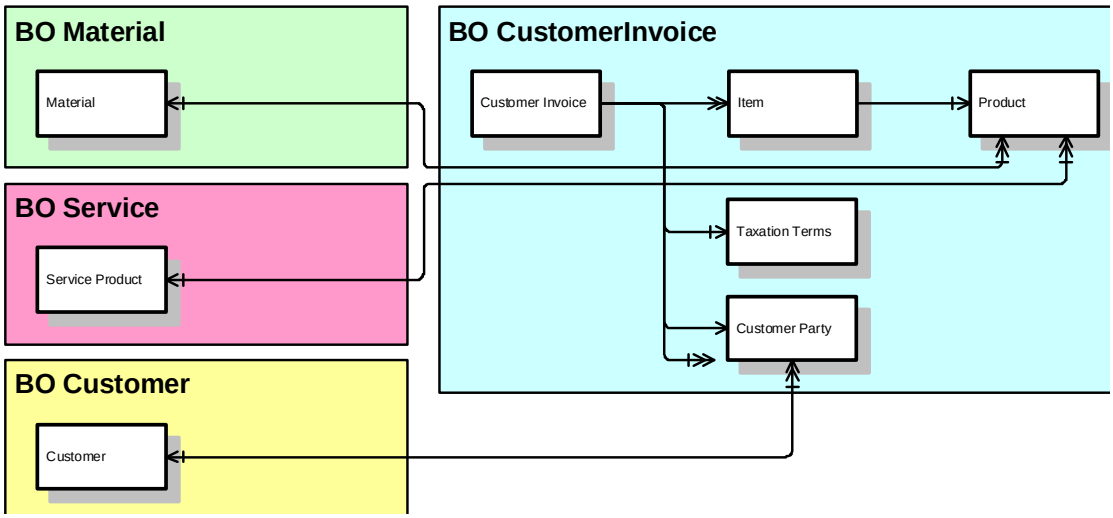
Example

An example for a process component is “customer invoice processing”.

Business Object

Business objects are entities from the business world that are relevant for a business process. A business object may be of diverse nature. Examples for business objects are documents, such as “customer invoice” or “employment” contract, legal entities, such as “customer” or “bank”, or physical objects such as “material” or “inventory” item.

Example of a Business Object Model



Structure

Business Object Node

Each business object is modeled as a hierarchical structure of nodes. As an example, the figure above shows an extract of the business object model of the customer invoice.

Nodes may have different cardinalities depending on their business semantics. For some nodes an arbitrary number of node instances is possible, like for line items in a customer invoice. For other nodes only a limited number of instances may exist. For example, each line item of a customer invoice represents only a single product, which is to be invoiced. In addition to the internal structure of a business object, the associations between related business objects are modeled as well in the business object model. For example, the customer invoice depends on a customer and a material or a service product.

Global Data Type (GDT)

Each business object node is made up of data fields, which are typed with so-called global data types (GDTs). GDTs are allowed and intended to be reused in different business object nodes. The idea is that for each basic business fact a GDT is defined and used wherever the business fact has to be represented in a data field.

GDTs may be either unstructured (for example, ProjectName and TaxableIndicator) or structured, that is a record of two or more other GDTs (for example, DeliveryTerms and Address).

CCTS

Unstructured GDTs are either qualified versions of other unstructured GDTs or represent a core data type as defined in the Core Components Technical Specification (CCTS) by UN/CEFACT (http://www.unece.org/cefact/ebxml/CCTS_V2-01_Final.pdf).

Core component types are basic data type definitions on a more technical level, which have no business semantics on their own. Examples for core data types are identifier (a character string uniquely identifying a certain object within a certain context), indicator (a Boolean value) or amount (an amount of money together with the currency). By adhering to a public standard for elementary data types, it is easier to exchange data with third party applications.

Example

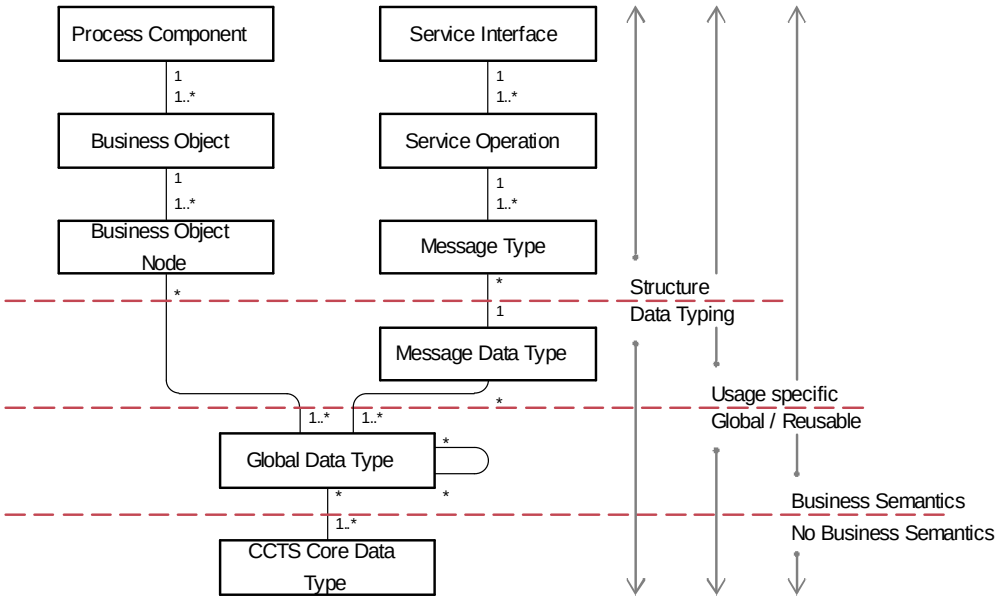
Examples for business objects are documents, such as “customer invoice” or “employment” contract, legal entities, such as “customer” or “bank”, or physical objects such as “material” or “inventory” item.

Service Interface Modeling

This process is used to derive the appropriate service interface definitions.

Service interface definitions for enterprise services have a uniform hierarchical structure, as shown below.

Entities of Service Interfaces and Business Object Models



Service Operation / Message Interface

For each operation, up to two parameters may be required, an input parameter and an output parameter. The input parameter has to be provided when calling the service. The output parameter is determined during the service execution and returned to the caller.

In addition to these two parameters per operation, an additional universal fault parameter exists. This fault parameter is sent back to the consumer in case of exceptions instead of the normal output parameter (see Exceptions and Errors).

Service Operations are also called Message Interfaces.

Message Type

The parameters are also called messages, since they make up the payload of the SOAP messages transferred between the consumer and the provider. For each service, the message required as input parameter and/or the message provided as output parameter must have a certain message type. The universal fault parameter has its own separate message type. Except for the fault message type, message types must not be reused between service operations, but are unique to one service operation.

Message Data Type

The message types are typed with a message data type, which is a record of data fields typed with GDTs, analogous to the data fields of business object nodes. Message data types unlike message types may be reused.

Process

Leading Business Object

All services are grouped around the business objects. Each service operation is associated with the business object that is central for the service operation's business purpose. For example the service operation customer invoice creation request is associated with the business object customer invoice. There may be multiple business objects relevant for the execution of a service operation. In the customer invoice example, the customer and the products to be invoiced are relevant as well. The service operation will be assigned to the central business object, which has to be identified among all relevant business objects. This central business object is also called *leading business object* for the service operation.

Message data types for a service operation are always derived from the leading business object as described in the following paragraphs. This ensures that service operations that are associated with the same central business object always use compatible message data types.

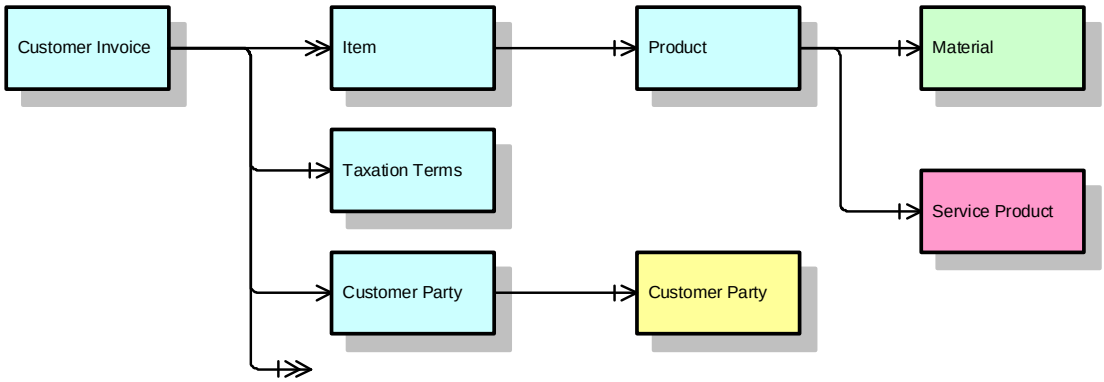
A message data type consists of up to three parts:

- a generic message header
- a so-called business document object
- for output messages: a generic log structure, which is used to return administrative information about the service execution.

Business Document Object

The business document object contains the specific business content of the message, that is the business facts relevant for the service execution or the business facts that show the result of the service execution. The business document object is the part of the message data type that is derived from the business object model. The business document object is a trimmed extract from the business object model containing all relevant parts of the leading business object of the service interface and all relevant parts of related business objects. The following figure shows the business document object corresponding to the customer invoice example above.

Example of a Business Document Object



Since the business document object is always a part of the message data type, it has to be structured hierarchically. The root node of the leading business object becomes the root node of the business document object. Below the root node, all relevant parts of the leading business object are included in the business document object according to the business object model. The required parts of associated business objects are linked below those nodes of the leading business object, with which they are associated in the business object model. The extent to which associated business objects are included in the business document object depends on the semantics of the service operation. In some cases an identifier of the

object is sufficient. In other cases one or more nodes or even the whole business object may be required.

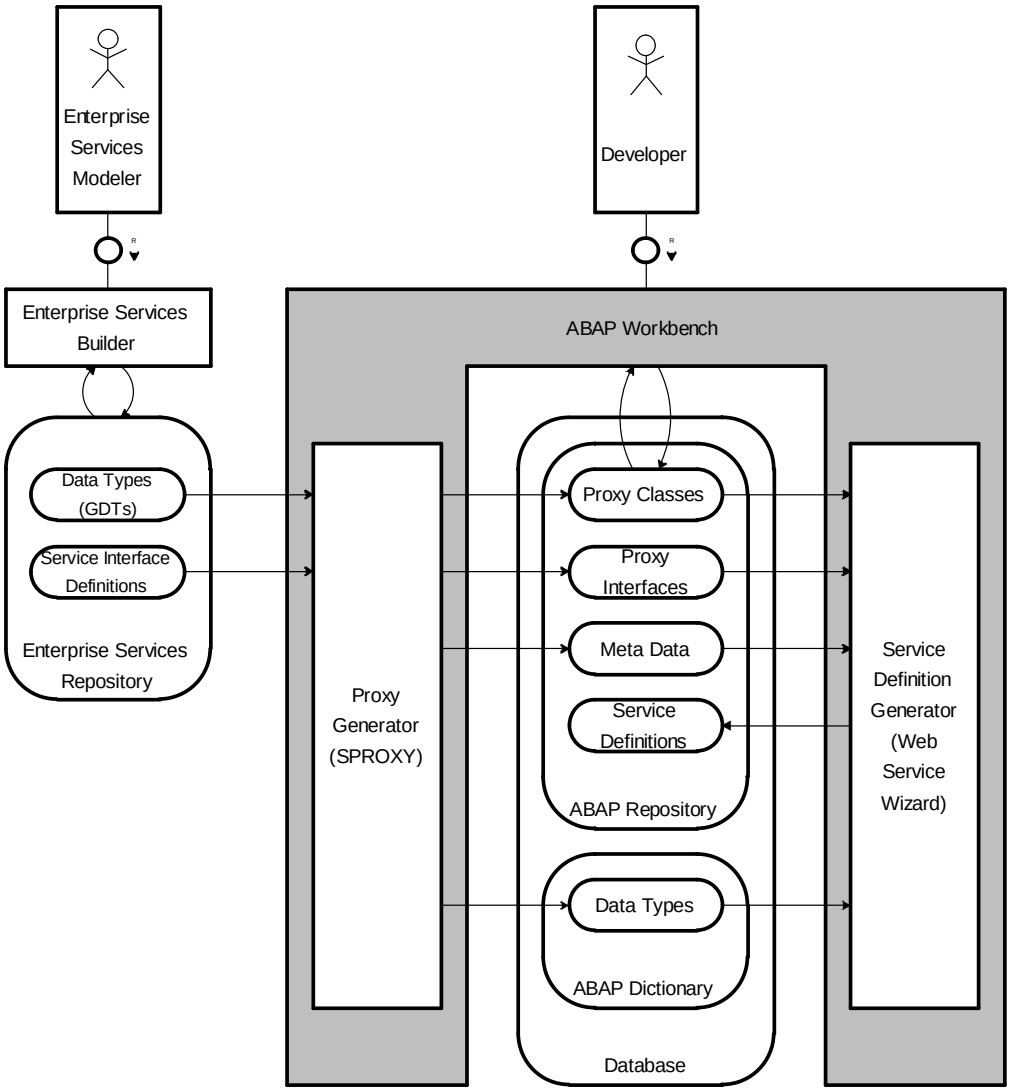
Proxy Generation in the ABAP Stack

A service interface definition in ESR is an abstract definition that does not depend on a specific implementation. In order to use the abstract definitions in the ABAP stack, corresponding ABAP entities have to be created and implemented in SAP Web AS.

ABAP Entities and Meta Data

In the ESR, the structure of a Web service is defined by specifying a service interface and the data types that are used in this interface (see Business Object and Service Modeling). Corresponding ABAP entities for the service interfaces and data types defined in ESR are ABAP interfaces, ABAP classes and ABAP data types. In addition meta data of the ABAP entities are required in the ABAP stack in order to execute a Web service. The meta data contains information about the entities that have been created based on ESR content. This information specifies for example how to map data types from the XML representation in ESR to the ABAP representation.

Design Time Architecture



Process

1. *Enterprise Services Builder*

   The figure above shows the systems involved in designing service interfaces in ESR and creating ABAP entities in the ABAP Stack on SAP Web AS for providing mySAP Web services. Service interface definitions and data types are created in ESR with the enterprise services builder.

2. *Proxy Generator*

   Based on the ESR content, the proxy generator creates all necessary ABAP entities for a service interface. It creates, on the one hand, the required ABAP data types in the ABAP dictionary, on the other hand, the ABAP interface and the ABAP class for the proxy which implements the service interface.

3. *Automatic and Manual Proxy Implementation*

   The implementation of a consumer proxy can be generated automatically.

   A provider proxy which relies on existing functionality must be implemented by a developer, as the logic which calls the existing functionality can not be created automatically. Therefore, the proxy generator only creates an empty ABAP class for provider proxies. The methods of this class have to be implemented by a developer. Each method corresponds to an operation of the Web service.

4. *Service Definition Generator*

   Once the proxy classes have been implemented, a service definition has to be created with the Web service definition generator, before a Web service can be released (see Releasing a Web Service). A service definition contains the design time properties of a Web service. The properties specify preferred security settings which should be used when the Web service is released during configuration. Design time properties, which are created with the service definition generator, are stored in a service definition entity.

mySAP Service Provisioning: Configuration

This process is used to configure services created by SAP. Once the services have been created by SAP, they are delivered with the enterprise services packages to the customer. The customer needs to configure the services according to his or her needs and the available IT infrastructure.

Business Configuration

The business configuration of the service-enabled mySAP Business Suite does not differ from that of the conventional mySAP Business Suite. Details of the business configuration of the mySAP Business Suite can be found in its Implementation Guide (IMG). The implementation guide is a tool for configuring the SAP system to meet customer requirements. It is accessed from the SAP menu of the SAP system by choosing *Tools* *Customizing* *IMG*.

SAP XI

For services that are to be called via SAP XI, the conventional XI configuration has to be carried out. For more information about SAP XI, see the SAP help portal at help.sap.com *SAP NetWeaver*.

Process

- Releasing a Web Service
- Performing Web Service Administration
- Configuring Idempotent Services

Releasing a Web Service

This is the first step of the configuration process.

Note

You perform the configuration described in these steps using the WSCONFIG transaction. For more information, see the SAP Help Portal under help.sap.com *SAP NetWeaver Releasing for the SOAP Runtime*.

Procedure

1. Activate the required services

   You need to decide which of the services delivered by SAP should be active in your installation of the mySAP Business Suite. These services have to be activated explicitly. The activation process is also referred to as the release of the service.

2. Apply variants

   For each service delivered, a service definition is provided. The service definition consists of a service interface description and one or more variants. When a service is to be released, the next step is to decide which of the variants available is applied.

   See Service Definition and Service Variant

3. Specify the following additional information:

   - *Name and release text:* Name and textual description of the service that will be published to the consumers.
   - *Virtual host:* The name of the SAP system which acts as the service provider. The virtual host implicitly determines the transfer protocol.
   - *URL:* The relative URL of the service in relation to the virtual host.
   - *Security profiles:* Security settings specifying the authorizations required for the execution of the Web service.

4. Generate a runtime configuration

   The service configuration utility takes the service interface description, the chosen variant and the additional information to generate a WSDL description and a runtime configuration. The WSDL description can be published to the service consumers, while the runtime configuration is needed by the Web service interpreter and the proxy framework to process calls to the service.

Service Definition and Service Variant

For each service delivered, a service definition is provided. The service definition consists of a service interface description and one or more variants. A variant describes the features of the service. Features in this context are properties of the service on a non-technical level, which restrict the choice of host and protocol in the configuration process. For example, if the variant specifies that the service needs authentication by certificates, it is not possible to use HTTP as a transfer protocol. HTTPS should be used instead.

Features

There are three feature types for services:

- *Unique message ID:* A unique message ID should be assigned to each message.
- *Authentication level*: Requirements for the authentication mechanism. Possible values are: “no authentication required”, “password authentication required” and “certificate authentication required”.
- *Transport guarantee level:* Requirements for the transport mechanism. It specifies whether integrity and/or confidentiality have to be ensured. Integrity is the guarantee that transferred messages arrive unmodified at the receiver. Confidentiality is the guarantee that only the receiver can access the unencrypted content of a transferred message.

Activities

When a service is to be released, the first step is to decide which of the variants available is applied.

Performing Web Service Administration

Procedure

The administration of Web services is the part of the configuration which takes place when the Web services are released.

1. Make the following two administration settings for an SAP Web service
   - *Trace and Log:* The extent to which information about the use of a service is recorded. It is possible, for example, to record information about calls to the service received, errors that occur during service execution, and performance measures.
   - *UDDI publication:* The WSDL description of released Web services can be published in UDDI repositories.
2. Test the Web service

   SAP’s Web service administration utility provides a test Web site, which allows invoking the Web service with test parameters. This can be used to check the configuration settings applied.

Note

You perform the administration described below using the WSADMIN transaction. For more information, see the SAP Help Portal under help.sap.com *SAP NetWeaver Administration for the SOAP Runtime.*

Configuring Idempotent Services

You use this method to set how long the system stores saved response messages or saved information about responses that have been sent. An initialization process takes you through the steps the first time you make these settings.

The initialization process incorporates the following steps:

- The system creates several database objects that are necessary for saving response messages.
- The system schedules the following background jobs for the regular deletion of saved response messages and message IDs:
  - SAP_BC_IDP_WS_SWITCH_BD
  - SAP_BC_IDP_WS_SWITCH_BDID

To ensure, for performance reasons, that the responses or the information about sent responses are deleted as soon as possible after the desired period, the system uses table pairs:

- two tables per client for response messages
- two tables per client for information about response messages that have been sent

The two tables are filled alternately. When changing from the first table to the second table, the data in the first table is kept, but the data from the second table is deleted before the system adds new content. You decide how often the background jobs change tables using the procedure described here. This results in a minimum storage duration for the data.

Prerequisites

You must be using at least SAP NetWeaver 2004s SPS09.

Procedure

1. Start transaction WSIDPADMIN.
2. In the *Document* area, enter the minimum storage period for response messages. In addition, enter how often the tables must be changed.
3. In the *Document ID* area, enter the minimum storage period for information about response messages that have already been sent. In addition, enter how often the Document ID tables must be changed. We recommend you choose a longer time interval than you did for the response messages. Note that a repeated request message results in an error if the corresponding response message has already been deleted but the information about the sent response message is still available.
4. Execute the transaction by choosing *Schedule* or pressing F8.
5. Repeat the settings in every client in which you want to call Enterprise Services.

Example

You use a time period of two hours for changing the document tables. This means that response messages are stored for at least two hours but at most for just less than four hours.

You use a time period of one day for changing the document ID tables. This means that information about sent response messages is stored for a maximum of just less than two days.

Example: Service Operation “Create Employee Leave Request”

This section provides a walk-through of service provision using the example of the service operation “Create Employee Leave Request”. This service operation is associated with the business object “EmployeeLeaveRequest”, which belongs to the process component “Time and Labour Management”.

Process

1. Modeling of Process Components in ARIS
2. Business Object Modeling
3. Modeling of Service Interfaces in ESR
4. Proxy Implementation

Modeling of Process Components in ARIS

The first step in the process of service creation is the development of the process component model in the modeling tool ARIS.

Procedure

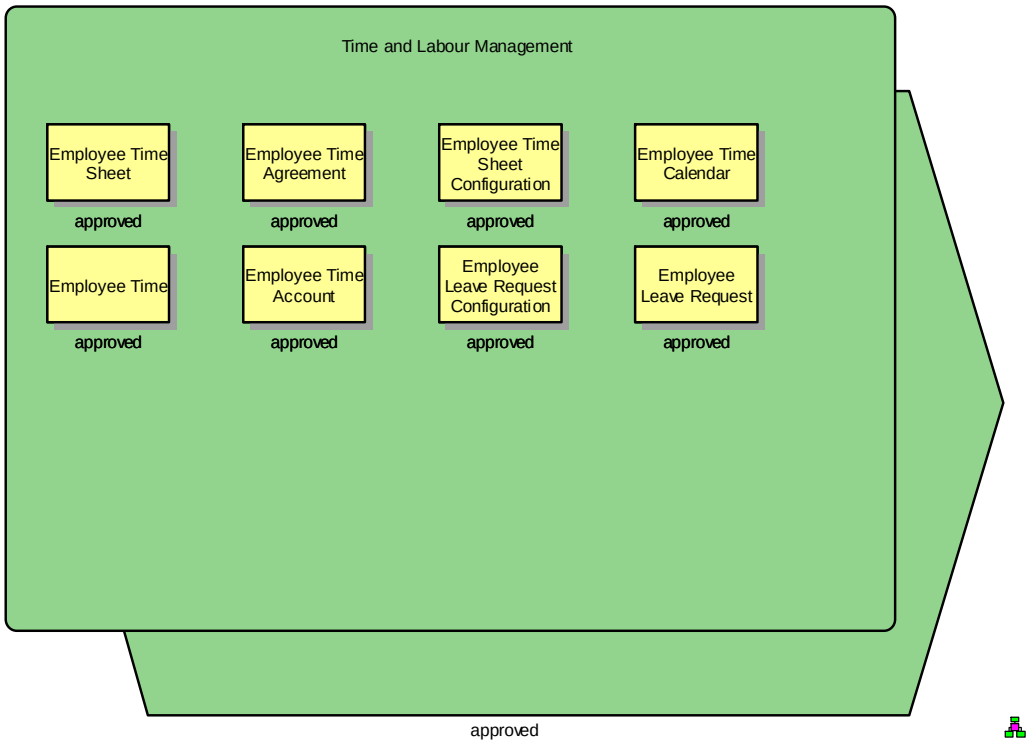
The first figure below shows the process component “Time and Labour Management” as an extract from the business object map, which is an overview of all process components and business objects.

Time and labour management supports the definition of employees' planned working time as well as the recording of the actual working times and absences and their evaluation.

Inside the process component, the contained business objects are shown. One of them is the business object EmployeeLeaveRequest.

An EmployeeLeaveRequest is the application form used by an employee to request leave. The request contains information about the desired leave besides request information such as submission date or the selected approver. The approver receives information about the desired leave and (depending on the type of desired leave and its configuration) has the possibility to approve or reject the request.

Process Component & Business Objects

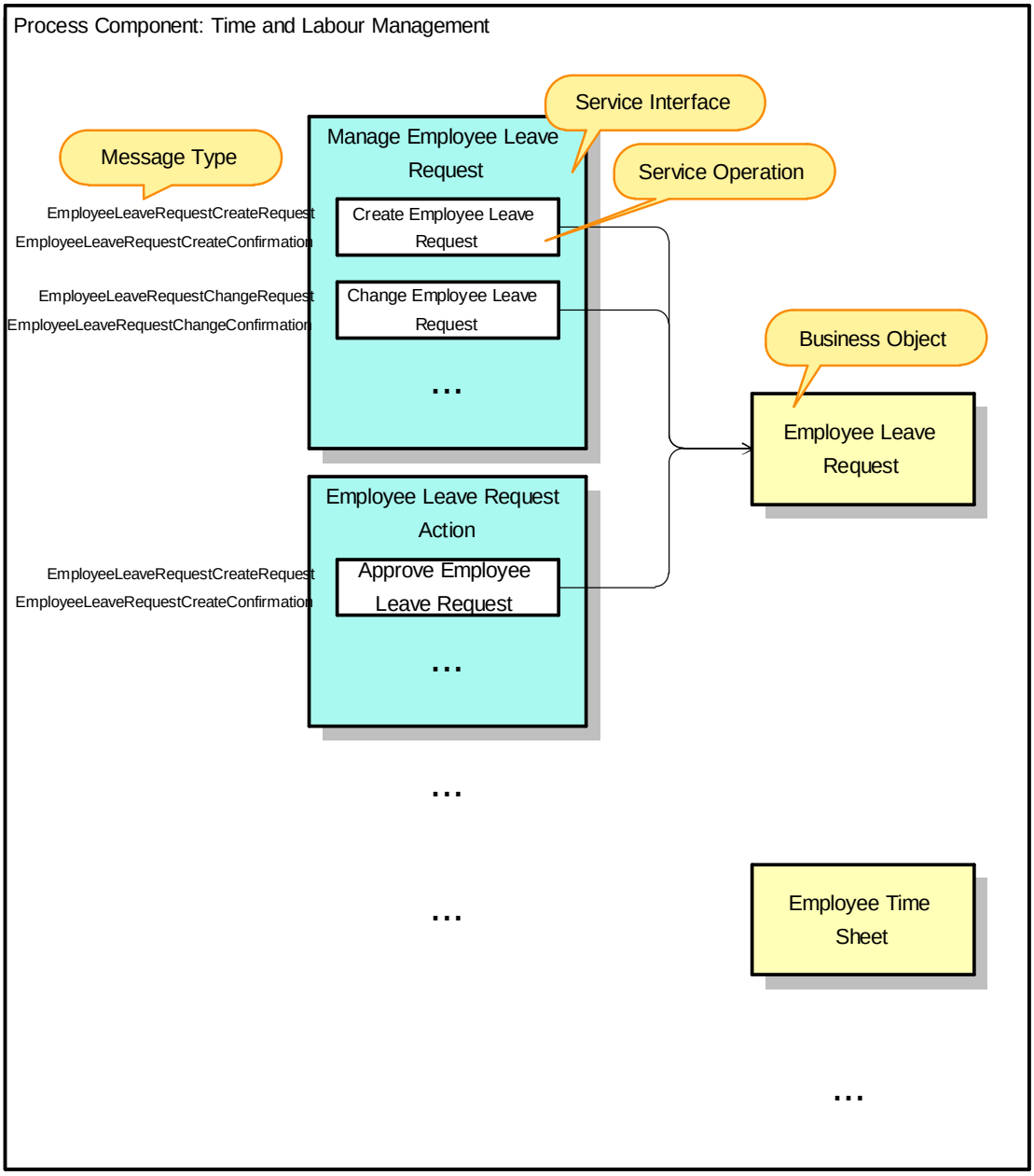


The next figure below shows an extract of the process component model of the process component “Time and Labour Management”. The process component model particularizes the business object map. For one process component, it shows all contained business objects together with the related service interfaces. For each service interface the operations and messages are shown.

For the business object EmployeeLeaveRequest, one of the service interfaces is the interface “Manage Employee Leave Request”, which contains all operations necessary to write, change and delete employee leave requests. Another service interface for the business object EmployeeLeaveRequest is “Employee Leave Request Action”, which contains operations for approving and denying employee leave requests.

One of the operations in the service interface “Manage Employee Leave Request” is the operation “Create Employee Leave Request”, which triggers the creation of a new employee leave request. The operation makes use of two messages, the inbound message EmployeeLeaveRequestCreateRequest and the outbound message EmployeeLeaveRequestCreateConfirmation.

Process Component Model



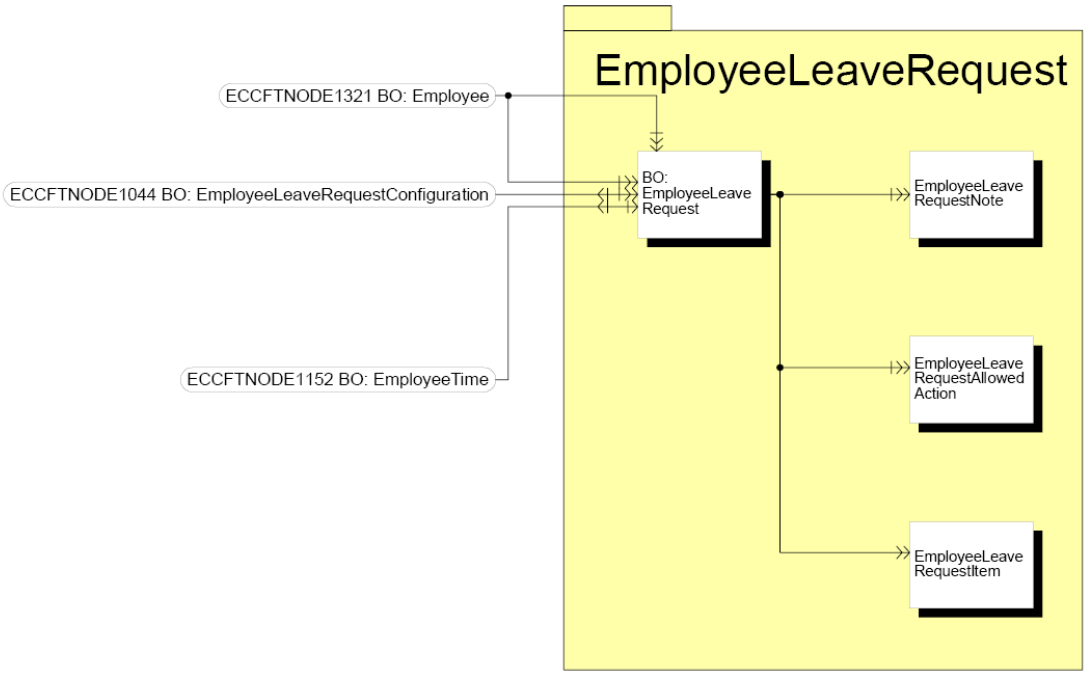
Business Object Modeling

The second step in the process of service creation is to define the inner structure of the business objects in a business object model.

Procedure

The figure below shows the business object model for the business object EmployeeLeaveRequest. Besides the root node, there are three sub nodes: Note, AllowedAction and Item.

Business Object Model



A *note* is a free text together with information about its author and the date and time of creation.

The *allowed action* is a coded representation of the way in which the transmitted document can be processed, for example “delete”, “modify” or “approve”.

An *item* of an employee leave request is a document item describing the details of an employee’s requested leave, e.g. the kind of leave and the planned period.

The business object EmployeeLeaveRequest has four associations with other business objects: to EmployeeTime, to EmployeeLeaveRequestConfiguration and two associations to Employee.

The linked employee time entry is an existing one, which is the result or the subject of this employee leave request.

The linked employee leave request configuration provides information about the functional and UI arrangement of employee leave requests concerning the individual business needs of the company.

One of the linked employees is the requester. The other linked employees are further participants, for example the responsible approver of the request.

The data fields contained in the business object nodes are not discussed here individually, but can be seen from the message structure in the next section about service interface modeling.

Modeling of Service Interface in ESR

The third step in the process of service creation is to derive the structure of the messages from the business object model and register the service operations together with the required messages in ESR.

Procedure

The figure below shows the message structure for the “Employee Leave Request Create Request” message as it is registered in the ESR.

Message Structure

SAP - Service Interface: Element Structure for EmployeeLeaveRequestCreateRequestMessage

Package	level 1	level 2	level 3	level 4	level 5	Cardinality	Type	Name
EmployeeLeaveRequestCreateRequest/EmployeeLeaveRequestCreateCheckQuery	EmployeeLeaveRequestCreateRequest/EmployeeLeaveRequestCreateCheckQuery						MDT	EmployeeLeaveRequestCreateRequest/EmployeeLeaveRequestCreateCheckQuery
EmployeeLeaveRequest		EmployeeLeaveRequest				1	XXX	EmployeeLeaveRequest
EmployeeLeaveRequestHeade			Participant			0..n	XXX	Participant
				RoleCode		1	GDT	EmployeeLeaveRequestParticipantRoleCode
				WorkAgreementID		1	GDT	WorkAgreementID
			Note			0..1	XXX	Note
				Text		1	GDT	Text
BusinessTransactionDocumentReference			LeaveEmployeeTimeReference			0..1	XXX	LeaveEmployeeTimeReference
				ActionCode		1	GDT	ActionCode
				LeaveEmployeeTimeReference		1	GDT	BusinessTransactionDocumentReference
EmployeeTimeItem			LeaveEmployeeTimeItem			0..n	XXX	LeaveEmployeeTimeItem
				CategoryCode		1	GDT	EmployeeTimeItemCategoryCode
				TypeCode		1	GDT	EmployeeTimeItemTypeCode
				Validity		1	GDT	EmployeeTimeItemValidity
				FullWorkingDayIndicator		0..1	GDT	FullWorkingDayIndicator
				Valuation Terms		0..1	XXX	ValuationTerms
					EmployeeTimeValuationTerms	0..1	GDT	EmployeeTimeValuationTerms
				ActivityAllocationAndCostAssignment		0..1	XXX	ActivityAllocationAndCostAssignment
					WorkBreakDownStructureElementID	0..1	GDT	ProjectElementID
				ServiceProvisionAndResourceConsumption		0..1	XXX	ServiceProvisionAndResourceConsumption
					ReceivingCostCentreID	0..1	GDT	CostCentreID
				LogsticConfirmation		0..1	XXX	LogsticConfirmation
					OrderID	0..1	GDT	BusinessTransactionDocumentID
				DifferentPayment		0..1	XXX	DifferentPayment
					PositionID	0..1	GDT	PositionID
				ValuatedDuration		0..n	XXX	ValuatedDuration
					DurationCalculationMethodCode	1	GDT	EmployeeTimeItemDurationCalculationMethodCode
					Duration	1	GDT	Duration
				AdditionalInformation		0..1	XXX	AdditionalInformation
					FirstAdditionalInformationText	0..1	GDT	AdditionalInformationText
					SecondAdditionalInformationText	0..1	GDT	AdditionalInformationText
					ThirdAdditionalInformationText	0..1	GDT	AdditionalInformationText
					FourthAdditionalInformationText	0..1	GDT	AdditionalInformationText
					FifthAdditionalInformationText	0..1	GDT	AdditionalInformationText
					SixthAdditionalInformationText	0..1	GDT	AdditionalInformationText
					SeventhAdditionalInformationText	0..1	GDT	AdditionalInformationText
					EighthAdditionalInformationText	0..1	GDT	AdditionalInformationText
					NinthAdditionalInformationText	0..1	GDT	AdditionalInformationText
					TenthAdditionalInformationText	0..1	GDT	AdditionalInformationText

Level 1 represents the complete message, which (on level two) consists only of the business document object for “EmployeeLeaveRequest”. The message header and log are not required in this case.

The business document consists of four parts. The participant, the note, the employee time reference and the employee time item.

The participant is the employee linked to the business object, which is relevant for the service operation. In the case of the create operation the relevant employee is the requester.

The note is the note from the business object.

The employee time reference is a reference to the existing employee time, which is the subject of this leave request. This reference is only needed if an existing (planned) employee time entry is requested to be changed or canceled.

The employee time item corresponds to the employee leave request item and describes the requested leave.

For all parts of the message the contained data fields are listed together with the global data types with which they are typed.

Proxy Implementation

In order to provide the Web service on the ABAP stack, ABAP entities have to be created with the proxy generator (see Proxy Generation in the ABAP Stack).

Procedure

The message interface in ESR which defines the Web service is,

`EmployeeLeaveRequestCreateRequestConfirmation_In`

Based on this message interface, the proxy generator creates the following ABAP entities:

- ABAP data types
- ABAP Interface: II_TIM_EMPLOYEELR_CRTRC
- ABAP class for the Web service proxy which implements the interface: `CL_TIM_EMPLOYEELR_CRTRC`

The implementation of the proxy is shown in the following in more detail. To keep the example clear, the created ABAP data types are not depicted in this example.

When a Web service call is received, the proxy framework triggers the proxy by calling the `II_TIM_EMPLOYEELR_CRTRC` interface method `EXECUTE_SYNCHRONOUS`. It calls the following methods in the specified order:

1. `INPUT_MAPPING:` Performs the application level mapping of input parameters (see Data Conversion and Custom Modifications)
2. `BUSINESS_LOGIC:` Executes the business logic by calling the existing RFC (see RFC: PT_ESA_REQUEST_CREATE)
3. `OUTPUT_MAPPING:` Performs the application level mapping of output parameters (see Data Conversion and Custom Modifications)

Example

The following is an excerpt of the source code of the EXECUTE_SYNCHRONOUS method.

Execute Synchronous

```
METHOD ii_tim_employeelr_crtrc~execute_synchronous.
```

Syntax

```
1.   * EmployeeLeaveRequestCreateRequestConfirmation_In DATA
2.   lv_employeenumber TYPE pernr_d.
3.   DATA lv_leave_id       TYPE tim_req_id.
4.   DATA lt_messages       TYPE bapirettab.
5.   DATA ...
6.
7.   * database updates are done synchronously
8.   SET UPDATE TASK LOCAL.
9.
10.  * Input Mapping: map proxy input parameters to function
     module (RFC) input parameters
11.  CALL METHOD me->input_mapping
12.  EXPORTING
```

```
13.    is_input = input "/input format according to proxy ABAP
   datatype
14.  IMPORTING
15.    ev_employeenumber = lv_employeenumber "/internal format
16.    ev_leave_id       = lv_leave_id
17.    ev_operationid    = lv_operationid.
18.
19.  * Business logic call
20.  CALL METHOD me->business_logic
21.  EXPORTING
22.    iv_employeenumber = lv_employeenumber
23.    iv_leave_id       = lv_leave_id.
24.  IMPORTING
25.    ev_request_detail = lv_request_detail       "/result from
   business logic processing
26.    ev_success        = lv_success
27.    et_messages       = lt_messages. "/ application message
   table -
28.                                     internal format
29.
30.  * Map application errors table to format of global data type
   Log
31.  CALL METHOD cl_proxy_log=>fill
32.  EXPORTING
33.    bapireturn_tab = lt_messages
34.  IMPORTING
35.    log            = output-... .
36.  EXIT.
37.  ENDIF.
38.
39.  * Output Mapping: map function module (RFC) output parameters
   to proxy output parameters
40.  CALL METHOD me->output_mapping
41.  EXPORTING
42.    iv_request_detail = lv_request_detail
43.  IMPORTING
44.    rs_output = output... . "/ input format according to proxy
   ABAP
45.                            datatype
46.
47.  * Commit
48.  cl_soap_commit_rollback=>commit( ).
```

```
49.
50.  ENDMETHOD. "ii_tim_employeelr_crtrc~execute_synchronous
51.
```

The methods called in this excerpt are shown in more detail in the following. The methods for input and output mapping show the mapping of parameters based on a few examples. The method which executes the business logic basically only calls the existing RFC (see RFC: PT_ESA_REQUEST_CREATE)

Input Mapping

```
METHOD input_mapping.
```

Syntax

```
1.   DATA wa_participant TYPE LINE OF
     hrsef_employeelr_crt_req_msg-employee_leave_request-
     participant.
2.   DATA lt_messages    TYPE bapirettab.
3.   ...
4.
5.   * Mapping employee numbers
6.   LOOP AT is_input-employee_leave_request_create-
     employee_leave_request-participant INTO wa_participant.
7.     CASE wa_participant-role_code-value.
8.       WHEN cl_tim_proxy_constants=>participant_role_owner.
9.         ev_employeenumber = wa_participant-work_agreement_id-
     value.
10.        ...
11.    ENDCASE.
12.  ENDLOOP.
13.  ...
14.
15.  * Mapping LeaveID and ActionCode
16.  CALL METHOD
     cl_tim_leave_proxy_mappings=>time_ref_input_mapping
17.  EXPORTING
18.    is_proxy_time_ref  = is_input-
     employee_leave_request_create-
19.                         employee_leave_request-
20.                         leave_employee_time_reference
21.  IMPORTING
22.    ev_fm_leave_id     = ev_leave_id
23.    ev_fm_operation_id = ev_operationid
24.    et_messages        = lt_messages.
25.
26.  APPEND LINES OF lt_messages TO et_messages.
```

```
27.  ...
28.
29.  ENDMETHOD. "input_mapping
```

Business Logic

METHOD business_logic.

Syntax

```
1.   DATA : lv_pernr    TYPE pernr_d.
2.   DATA : lt_messages TYPE bapirettab.
3.
4.   lv_pernr  =  iv_employeenumber.
5.
6.   *** If owner pernr is not supplied, use the pernr of the user
     who is invoking the service
7.     IF iv_employeenumber IS INITIAL.
8.
9.       CALL FUNCTION 'PAD_EMPLOYEENUMBERS_GET_LIST'
10.      EXPORTING
11.        iv_username        = sy-uname
12.        iv_key_date        = sy-datum
13.      IMPORTING
14.        ev_employee_number = lv_pernr
15.      TABLES
16.        t_messages_out     = lt_messages.
17.
18.      IF NOT lt_messages IS INITIAL.
19.        APPEND LINES OF lt_messages TO et_messages.
20.        RETURN.
21.      ENDIF.
22.
23.    ENDIF.
24.
25.
26.  CALL FUNCTION 'PT_ESA_REQUEST_CREATE'
27.  EXPORTING
28.    iv_employee_number      = lv_pernr
29.    iv_leave_id             = iv_leave_id
30.    iv_oparation_id         = iv_operationid
31.    iv_leave_request        = iv_requestdetail
32.  IMPORTING
33.    ev_request_detail       = ev_request_detail
```

```
  ev_success_confirmation = ev_success
TABLES
  et_messages             = et_messages.

ENDMETHOD.

```

Output Mapping

```
METHOD output_mapping
```

Syntax

```
  DATA ls_participant       TYPE ptesa_actor_struc.
  DATA ls_proxy_participant TYPE hrsef_employee_leave_request13.
  DATA ls_time_item         TYPE hrsef_employee_leave_request10.
  DATA lv_owner             TYPE pernr_d.
  ...

  * Participant table
  LOOP AT iv_request_detail-participant INTO ls_participant.
    CALL METHOD cl_tim_leave_proxy_mappings=>participant_output_mapping
      EXPORTING
        is_fm_participant    = ls_participant
      IMPORTING
        ev_owner             = lv_owner
        es_proxy_participant = ls_proxy_participant.

    IF NOT ls_proxy_participant IS INITIAL.
      APPEND ls_proxy_participant TO rs_output-
             employee_leave_request_create-employee_leave_request-
             participant.
    ENDIF.
  ENDLOOP.
  ...

  * Time item table
  CALL METHOD cl_tim_leave_proxy_mappings=>item_output_mapping
  EXPORTING
    iv_pernr      = lv_owner
    is_fm_item    = iv_request_detail-item
```

```
29.  IMPORTING
30.    es_proxy_item = ls_time_item.
31.
32.  APPEND ls_time_item TO rs_output-
   employee_leave_request_create-
33.         employee_leave_request-leave_employee_time_item.
34.
35.  ENDMETHOD. "output_mapping
36.
```

RFC: PT_ESA_REQUEST_CREATE

The RFC used to implement the Web service in the ABAP stack is PT_ESA_REQUEST_CREATE. The RFC not only supports the creation of an employee leave request but also makes it possible to change and delete employee leave requests. Therefore other Web services can be created that are based on the same RFC. As in our example, only employee leave requests are created, the proxy implementing the Web service only uses part of the available functionality.

Basically the RCF takes the employee number, the leave request and an operation type as input. In case of the creation of a request (determined by the operation type) and if executed successfully, it returns the created leave request containing additional detailed information that was added to the original request during creation.

Stability of Released Enterprise Services

On the one hand, it is necessary to further develop enterprise services to satisfy new or changed requirements. On the other hand, it is essential that as soon as enterprise services are released, they remain stable. This means they should only be further developed to be upward compatible so that existing consumers do not have to make any adjustments after an upgrade of the enterprise service.

Compatible Further Developments of Enterprise Services

Compatible further developments must necessarily be *syntactically* compatible. This means that the interfaces of a service, especially the XML schema of the input and output message types, may only be enhanced in a particular way. For example, no elements may be renamed, nor may type or length be changed.

A simplified example is only the *syntactically downward-compatible* addition of new optional elements and attributes. An element or attribute is optional if it does not have to be contained in an XML document that is exchanged at runtime. A new optional element or attribute can be added at any place in the interface as a sub-element or attribute of an existing element, as long as that element already has at least one sub-element or attribute.

Note

Since this rule is also valid for response messages, it implies requirements regarding the way enterprise service consumers must process response messages. A consumer must ignore unknown elements or attributes when receiving a response message, meaning they should

not strictly validate the messages. In this sense, enterprise services must be *consumer-upward compatible*.

At first glance, additional forms of *syntactically* compatible enhancements would be feasible — especially for input message types. However, they frequently indicate changes that are *incompatible in content* or they are incompatible from the perspective of customer-defined extensions of the data conversion (see BAdIs).

Changes in content arise, for example, if the existing implementation of the service is changed or enhanced, thus causing the service or some of the elements in its interface to receive a modified meaning, without the structure of the interface itself being changed. In addition to the syntactical downward compatibility, *content* compatibility is also essential for service consumers.

If the further development of a service provides for syntactical compatibility as well as content compatibility, the released service itself is enhanced, and a new service is not developed. After an upgrade of the enterprise service, consumer automatically use its enhanced version.

If an incompatible change cannot be avoided, one or more new enterprise services are offered to replace the existing one. However, exceptions to this rule are possible.

New Enterprise Services to Replace Enterprise Services to be Discontinued

- If the incompatible change is a change in content, the new enterprise services will correspondingly have a different name. Thus, the new services must not necessarily have the same name as the replaced services.
- If the change is not a change in *content*, the new enterprise service will essentially keep the name of the replaced service. However, both enterprise services belong to the same namespace, so that the new service cannot have the exact same name. It can be considered to be a new version of the replaced service and therefore has an attached version number "_V1". If there are additional incompatible changes, the version number increases accordingly.

The enterprise service to be replaced is not removed immediately, but rather flagged as “Deprecated” (obsolete). It is still supported in the change release (the release in which the subsequent service is rolled out) and its functionality is retained.

In the subsequent release, the actual implementation of the discontinued service is removed and replaced by coding which merely returns an exception message. At the same time, the enterprise service changes its status to “Revoked.”

The replaced service can only be removed after at least one more release when the discontinuation phase has been entirely finished.

As long as it exists, the discontinued enterprise service refers to its successor services and there is documentation on how existing applications that use the discontinued service can be converted to the new services.

Glossary

Terms used in Enterprise Service Oriented Architecture

A2A

A2A refers to communication between different software applications within the same company. Interfaces for A2A communication, unlike B2B interfaces, may be specific to a certain software application. (Compare B2B)

ABAP

ABAP (Advanced Business Application Programming) is a 4th generation programming language created by SAP. It is currently positioned as the language for programming the SAP Web Application Server, part of the SAP NetWeaver platform for building business applications.

BAPI

A standardized and stable programming interface that facilitates external access to business processes and data in the SAP System. BAPIs offer an object-oriented view of business components in the SAP system.

B2B

B2B refers to communication between different companies. Interfaces for B2B communication must be independent of specific software applications. Usually B2B interfaces comply with international e-business standards. (Compare A2A)

HTTP

Hypertext Transfer Protocol (HTTP) is the protocol used to transfer or convey information on the World Wide Web. It is a patented open internet protocol whose original purpose was to provide a way to publish and receive HTML pages.

More Information

HTTP/1.1 specification (RFC 2616, June 1999)

HTTPS

HTTPS refers to the combination of normal HTTP interaction over an encrypted secure socket layer transport mechanism.

Java

Java refers to a number of computer software products and specifications from Sun Microsystems (the Java™ technology) that together provide a system for developing and deploying cross-platform applications in an object-oriented way.

More Information

http://java.sun.com

RFC

A Remote Function Call is a call of an SAP function module that runs in a different system (destination) from the calling program. Connections are possible between different SAP systems and between an SAP system and a non-SAP system. In non-SAP systems, instead of function modules, special programmed functions are called, whose interface simulates a function module. The term RFC sometimes also refers to the called function module.

SAML

Security Assertion Markup Language (SAML), developed by the Security Services Technical Committee of OASIS, is an XML-based framework for communicating user authentication, entitlement, and attribute information. SAML allows business entities to make assertions regarding the identity, attributes, and entitlements of a subject (an entity that is often a human user) to other entities, such as a partner company or another enterprise application.

More Information

http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security

SSL

Secure Sockets Layer (SSL) is a cryptographic protocol which provides secure communication on the Internet.

XML

Extended Markup Language (XML) is a generic language for defining new markup languages specified by the World Wide Web Consortium (W3C). Markup languages (such as HTML) are used to represent documents with a nested, treelike structure.

More Information

http://www.w3.org/XML/

XML Encryption

XML Encryption is a specification that defines how to encrypt the content of an XML element, as recommended by the World Wide Web Consortium (W3C).

More Information

http://www.w3.org/Encryption/2001/

XML Signature

XML Signature is a specification that defines an XML syntax for digital signatures, as recommended by the World Wide Web Consortium (W3C).

More Information

http://www.w3.org/TR/xmldsig-core/

About This Documentation

Documentation Target Groups

The documentation is aimed primarily at the following target groups:

- Business consultants who design processes and applications for customers and who want to learn about the features of the individual services
- Business consultants for service providers who set up a back-end system such that the services can be used in the conventional way
- Software developers who use the services when creating applications (user interfaces) to enable data communication with the back-end system
- Members of the support team who provide support to the above target groups on how to use the service

Documentation Structure

Title

The titles of the documents for the individual SOA entities correspond to the names that were used in the business models to describe the objects.

In following releases, the *message interfaces* are called *operations*. For this reason, the *message interfaces* are now called *(service) operations* in the documentation.

Technical Data

The technical data of the operations includes the names of the operations as objects in the *Enterprise Service Repository* (ESR) and specifies what the associated Web service definitions are called.

Documentation

In addition to the technical data, the documentation includes the definitions of the relevant SOA entities as they can be found in the business models. The following sections describe the business use of the objects and their position within the business model. If the data structures of the operation's message types have special features, these are described in the Requirements, Structure, and Error Handling sections. Information that is valid for all objects

of a process component, business object, or service interface is grouped together for each object.

All special features that are based on the implementation of the service and the processing of messages in the SAP back-end system are described in the *Notes for SAP Back-End System Administration* section.

Tables of Technical Data

The following tables explain the meaning of the individual entries in the tables of technical data.

Entity Type	Entry in the Table	Meaning
Process component	Software Component	Name of the software component in the SAP back-end system with the release number as of which the implementation of the services of this process component is available
Business object	Software Component	Name of the software component in the SAP back-end system with the release number as of which the implementation of the services of this business object is available
Service interface	Software Component	Name of the software component in the SAP back-end system with the release number as of which the implementation of the services of this service interface is available
	Category	Information indicating whether this is an inbound or outbound interface
	Mode	Information indicating whether the interface supports synchronous or asynchronous

		processing
Service operation	Technical Name in Enterprise Service Repository (ESR)	Unique name of the operation in ESR
	Namespace in ESR	The namespace indicates the SAP back-end software component in which the specified business logic of the operation is stored.
	Software Component Version in ESR	The software component version indicates the release status of the SAP back-end as of which the operation is available.
	Category	Information indicating whether this is an inbound or outbound operation
	Mode	Information indicating whether the operation is used for synchronous or asynchronous processing
	Related Web Service Definition	Name of the Web service definition

SAP: Important Disclaimers and Legal Information

The services delivered by SAP for *mySAP Business Suite* are developed in two steps. First, the services are defined independent of an actual implementation. To do this, they are presented uniformly in a business object model that is valid in general. The models represent the business content of *mySAP Business Suite* in a simple and structured form.

You can view the models on the *Enterprise Service Workplace* in *SAP Developer Network* (SDN) at **`sdn.sap.com`** → **`Enterprise SOA`** → **`ES Workplace`**.

Second, SAP implements the objects required for realizing the services in the SAP system as ABAP entities.

The uniform and structured view that the business models provide for the functions of *mySAP Business Suite* is also used for structuring the documentation. Therefore, objects are described in the documentation that do not have a direct correspondent in an SAP system. These consist of the following objects: process components, business objects, and service

interfaces. The description of these objects provides you with an overview of the actual business process in which you can use the services delivered by SAP.

The implementations of service operations are available as part of the delivery of *mySAP Business Suite enterprise services package*.

Enterprise Services for SAP ERP Cross-Application Components

Business Partner Data Management

The *Business Partner Data Management* process component is a part of the ERP foundation. The business content described here refers only to the enterprise services relevant to SAP ECC.

Definition

Includes all basic data on employees and business partners.

Use

You can use the *Business Partner Data Management* process component to manage all business partner data that is necessary for controlling business processes such as Sales and Distribution, Purchasing, and Accounting. You can also use this process component to provide services for queries related to person-specific information on employees, as well as information along the organizational structure.

Notes for SAP Back-End System Administration

Processing

Take into account the following behavior of the change and update operations of this process component when you use it:

The data transferred by request messages overwrites only those affected objects in the databases of the back-end system that are also filled in the message structure. This also applies to the individual items of list-like elements.

This means that if optional elements are not filled in a request message, the back-end system leaves the values as they are in the affected fields in the database. The only exception to this behavior is when these values are automatically updated due to changes to other data.

Employee

The *Employee* business object is part of the ERP foundation. The business content described here refers only to the enterprise services relevant to SAP ECC.

Definition

Person who contributes, or has contributed to, the creation of goods and services within the enterprise.

This includes internal and external employees. As opposed to external employees, internal employees are bound by instructions and are subject to job-related organizational control.

Use

The *Employee* business object is a fundamental object that cannot be assigned to a specific process component.

From a Personnel Administration perspective, the business object includes data such as name, address data, communication data, and the employee's photo.

Each employee is also part of an organizational structure within the company. From an Organisational Management perspective, the business object includes data such as the *Organisational Centre* that the employee belongs to, the responsible manager, and all employees that belong to this *Organisational Centre*.

Related Business Objects

Work Agreement [Extern]

Employment [Extern]

An employee's tasks and responsibilities are specified in the *Work Agreement*. This work agreement establishes *Employment*. It forms the basis for further agreements such as working times and salary data that are specified in other business objects.

Employee E-Mail Data Management In

Technical Data

Entity Type	Service interface
Software Component	*SAP ERP enterprise services package* (as of EA-HR 602)
Category	Inbound
Mode	Asynchronous

Use

This inbound service interface contains an operation that creates or deletes an employee's e-mail address.

Integration

The inbound service interface belongs to the Business Partner Data Management [Page 80] process component and the Employee [Page 80] business object.

Maintain E-Mail Address

Technical Data

Entity Type	Operation

Technical Name in Enterprise Services Repository (ESR)	EmployeeCommunicationDataEMailAddressRequest_In
Namespace in ESR	http://sap.com/xi/EA-HR/Global2
Software Component Version in ESR	As of EA-HR 602
Category	Inbound
Mode	Asynchronous

Use

This inbound operation is used to maintain the e-mail addresses of an employee for one or more of the employee's *Work Agreements*.

You can use this service to transfer e-mail addresses from external e-mail systems, for example, to use them in *Travel Management* (standalone system). In *Travel Management*, e-mail addresses are required for notifications from external reservation systems, for example.

Features

The inbound operation processes the following message type:

`EmployeeCommunicationDataEMailAddressRequest`

The following fields are available in the operation:

- *EmployeeID*
- *CommunicationData*

  Contains communication data that is not address-dependent, such as e-mail address or cell phone number.

  - *@actionCode*

    Contains the encrypted representation of an instruction for the recipient of a message that contains information about how the transferred element is to be processed. The message executes the *Create* and *Delete* operations. Changing e-mail addresses is not supported. Therefore, you can enter the following values for *@actionCode*:

    - **`01`** – Create
    - **`03`** – Delete

    Other values are not permitted.

- *CommunicationDataAssignment*

  - *ValidityPeriod*

    Specifies the validity period for the assignment of the *Communication Data* to a *Work Agreement*.

- *WorkAgreement*

  In this message, you can enter several of the WorkAgreementIDs belonging to the *Employee* for which you want to maintain the employee's e-mail address.

  If you do not enter a WorkAgreementID, the system maintains the e-mail address for all WorkAgreementIDs belonging to the EmployeeID.

- *EMail*

  - EmailURI

Contains the e-mail address.

If you do not enter a validity period for the message, the system uses the following values:

Start date = <sy-datum>

End date = 31.12.9999

Prerequisites

You must activate the business function HCM_SRV_CI_1 *HCM, Enterprise Services 01* to have this inbound operation available.

Integration

This inbound operation belongs to the Employee E-Mail Data Management In [Page 81] inbound service interface.

Notes for SAP Back-End System Administration

Prerequisites

If you use this inbound operation for *Travel Management* (standalone system), you must make sure that the *Communication* infotype (0105), subtype *E-Mail* (010), is not maintained in *Travel Management*. You use a suitable authorization concept for this.

Processing

- If an error occurred when processing *@Action-Code* 01 (*Create*), the processing is discontinued immediately by an exception. The message is then identified in XI as containing errors and can be updated retroactively. A rollback is triggered for all processed operations and no data is saved in the back-end system for the message.
- If the data to be deleted is not found in the back-end system when processing *@Action-Code* 03 (*Delete*), processing is discontinued and the status of the message is set to With Errors in XI.
- The following Business Add-Ins (BAdI) are available for this operation:
  - PAD_SE_EECDATAEMAILADD_CONV (Mapping for EmployeeCommunicationDataEMailAddressRequest_In)
  - PAD_SE_SUBTY_CONV (*Subtype Mapping for E-Mail and Credit Cards*)

    In the standard system, the e-mail address of an employee is saved in the *E-Mail* subtype (010) of the *Communication* infotype (0105). If you want to change the subtype for the e-mail address, you can implement this BAdI.

  For more information about these BAdIs, see the Implementation Guide for *Personnel Administration* and choose *Enterprise Services* → *Business Partner Data Management*.

Employee Payment Card Management In

Technical Data

Entity Type	Service interface
Software Component	*SAP ERP enterprise services package* (as of EA-HR 602)
Category	Inbound
Mode	Asynchronous

Use

This inbound service interface groups operations that maintain the *Payment Card* (credit card) data of an employee.

Credit card data is required, for example, in *Travel Management* for credit card clearing whereby a file with the payment transactions is transported into the back-end system. For this process, the employee's credit card number must be saved in the *Communication* infotype (0105), subtype *Credit card number(s)* (11).

Change information for credit card data (credit card number, validity, and so on) must also be imported regularly into the Travel Management system so that the credit card clearing process can be executed continuously.

Integration

The inbound service interface belongs to the Business Partner Data Management [Page 80] process component and the Employee [Page 80] business object.

Create Payment Card Details

Technical Data

Entity Type	Operation
Technical Name in Enterprise Services Repository (ESR)	EmployeePaymentCardDetailsCreateRequest_In
Namespace in ESR	http://sap.com/xi/EA-HR/Global2
Software Component Version in ESR	As of EA-HR 602
Category	Inbound
Mode	Asynchronous

Use

This inbound operation is used to create the *Payment Card* (credit card) data of an employee for one or more of the employee's *Work Agreements*.

Features

The inbound operation processes the following message type:

`EmployeePaymentCardDetailsCreateRequest`

In this message, you can enter several of the *WorkAgreementIDs* belonging to the *Employee* for which you want to create the *Payment Card* (credit card) data.

If you do not enter a *WorkAgreementID*, the system creates the *Payment Card* data for all *WorkAgreementIDs* belonging to the *EmployeeID*.

The request message contains the following fields that you need to fill:

- *PaymentCardDetails*
  - *PaymentCardID*

    The *PaymentCardID* specifies the number of the credit card.
  - *PaymentCardTypeCode*

    The *PaymentCardTypeCode* specifies the type of credit card.

The standard system contains the following *PaymentCardTypeCodes*:

1 American Express

2 VISA

3 Mastercard

4 Diners

5 maestro

- *EmployeePaymentCardAssignment*

  The *EmployeePaymentCardAssignment* is the assignment of the employee's *Payment Card* to a *Work Agreement*.

  - *ValidityPeriod*

    The *ValidityPeriod* specifies the validity of the credit card.

If you do not enter a validity period for the message, the system uses the following values:

Start date = <sy-datum>

End date = 31.12.9999

Prerequisites

You must activate the business function HCM_SRV_CI_1 *HCM, Enterprise Services 01* to have this inbound operation available.

Integration

This inbound operation belongs to the Employee Payment Card Management In [Page 83] inbound service interface.

Notes for SAP Back-End System Administration

Prerequisites

You have defined the permitted credit card companies in the Customizing settings for *Travel Management* by choosing *Financial Accounting (New)* → *Travel Management* → *Travel Planning* → *Master Data* → *Define Credit Card Providers for Travel Planning*.

Mapping of the GDT values to the back-end values

GDT:PaymentCardTypeCode	**Values in the Back-End System**	
	ID	**Credit Card Company**
1	AX	American Express Company
2	VI	VISA International
3	MC	Master Card
4	DC	Diners Club International
5	EC	Euro Card

If you use this inbound operation for *Travel Management* (standalone system), you must decide whether the leading system for the *Communication* infotype (0105), subtype *Credit card number(s)* (11), is the external system or the Travel Management system. You use a suitable authorization concept for this.

Processing

If the *WorkAgreementID* entered does not belong to the *EmployeeID*, an exception message is sent to the XI.

If the transferred data contains errors or if an error occurred when processing the transferred data, processing is immediately discontinued by an exception. The message is then identified in XI as containing errors and can be updated retroactively. A rollback Work is triggered for all processed operations for other *WorkAgreementIDs* that were processed successfully, and no data is saved in the back-end system for the message.

The following Business Add-Ins (BAdI) are available for this operation:

- PAD_SE_SUBTY_CONV (*Subtype Mapping for E-Mail and Credit Cards*)
- PAD_SE_PAYMENTCARD_CONV (*Mapping for Credit Cards*)

For more information about these BAdIs, see the Implementation Guide for *Personnel Administration* and choose *Enterprise Services* → *Business Partner Data Management*.

Cancel Payment Card Details

Technical Data

Entity Type	Operation
Technical Name in Enterprise Services Repository (ESR)	EmployeePaymentCardDetailsCancelRequest_In
Namespace in ESR	http://sap.com/xi/EA-HR/Global2
Software Component Version in ESR	As of EA-HR 602
Category	Inbound
Mode	Asynchronous

Use

This inbound operation is used to delete the *Payment Card* (credit card) data of an employee for one or more of the employee's *Work Agreements*.

Features

The inbound operation processes the following message type:

`EmployeePaymentCardDetailsCancelRequest`

In this message, you can enter several of the *WorkAgreementIDs* belonging to the *Employee* for which you want to delete the *Payment Card* (credit card) data.

If you do not enter a *WorkAgreementID*, the system deletes the *Payment Card* data for all *WorkAgreementIDs* belonging to the *EmployeeID*.

The request message contains the following fields that you need to fill:

- *PaymentCardDetails*
  - *PaymentCardID*

    The *PaymentCardID* specifies the number of the credit card.
  - *PaymentCardTypeCode*

    The *PaymentCardTypeCode* specifies the type of credit card.

The standard system contains the following *PaymentCardTypeCodes*:

1 American Express

2 VISA

3 Mastercard

4 Diners

5 maestro

- *EmployeePaymentCardAssignment*

  The *EmployeePaymentCardAssignment* is the assignment of the employee's *Payment Card* to a *Work Agreement*.

  - *ValidityPeriod*

    The *ValidityPeriod* specifies the validity of the credit card.

If you do not enter a validity period for the message, the system uses the following values:

Start date = <sy-datum>

End date = 31.12.9999

Prerequisites

You must activate the business function HCM_SRV_CI_1 *HCM, Enterprise Services 01* to have this inbound operation available.

Integration

This inbound operation belongs to the Employee Payment Card Management In [Page 83] inbound service interface.

Notes for SAP Back-End System Administration

Prerequisites

You have defined the permitted credit card companies in the Customizing settings for *Travel Management* by choosing *Financial Accounting (New)* → *Travel Management* → *Travel Planning* → *Master Data* → *Define Credit Card Providers for Travel Planning*.

Mapping of the GDT values to the back-end values

GDT:PaymentCardTypeCode	**Values in the Back-End System**	
	ID	**Credit Card Company**
1	AX	American Express Company
2	VI	VISA International
3	MC	Master Card
4	DC	Diners Club International
5	EC	Euro Card

If you use this inbound operation for *Travel Management* (standalone system), you must decide whether the leading system for the *Communication* infotype (0105), subtype *Credit card number(s)* (11), is the external system or the Travel Management system. You use a suitable authorization concept for this.

Processing

If the *WorkAgreementID* entered does not belong to the *EmployeeID*, an exception message is sent to the XI.

If the transferred data contains errors or if an error occurred when processing the transferred data, processing is immediately discontinued by an exception. The message is then identified in XI as containing errors and can be updated retroactively. A rollback Work is triggered for all processed operations for other WorkAgreementIDs that were processed successfully, and no data is saved in the back-end system for the message.

The following Business Add-Ins (BAdI) are available for this operation:

- PAD_SE_SUBTY_CONV (*Subtype Mapping for E-Mail and Credit Cards*)
- PAD_SE_PAYMENTCARD_CONV (*Mapping for Credit Cards*)

For more information about these BAdIs, see the Implementation Guide for *Personnel Administration* and choose *Enterprise Services* → *Business Partner Data Management*.

Customer

Technical Data

Entity Type	Business Object
Software Component	*SAP ERP enterprise services package* (as of ECC-SE 600) SAP APPL (as of SAP_APPL 602)

Definition

A *Customer* is a business partner who is offered or sold materials or services.

Use

Sales employees need to have information about customers so they can create sales documents (for example, queries or contracts), get a quick overview of a customer, or simply contact a customer.

Information about whether a customer exists already, and basic information including address and contact details can be crucial for a business transaction.

The business object *Customer* is a projection of the business object *Business Partner*. The following element displays the business object.

- ID (Mandatory)

  The ID uniquely identifies the customer.

Notes for SAP Back-End System Administration

Prerequisites

To use the operations in this business object, you must implement the software component *SAP_APPL* as well as the component *Sales (SD-SLS)*. You have also performed the basic Customizing settings for the component.

Manage Customer In

Technical Data

Entity Type	Service interface
Software Component	*SAP ERP enterprise services package* (as of ECC-SE 600) SAP APPL (as of SAP_APPL 602)
Category	Inbound
Mode	Synchronous

Definition

Group of operations that provide information about a customer.

Use

The inbound service interface *Manage Customer In* groups operations that sales employees can use to read or process data about a customer.

Integration

The inbound service interface *Manage Customer In* is part of the Business Partner Data Management [Page 80] process component and the Customer [Page 88] business object.

Update Customer Bank Details

Technical Data

Entity Type	Service operation
Technical Name in Enterprise Services Repository (ESR)	CustomerERPBankDetailsUpdateRequest Confirmation_In
Namespace in ESR	http://sap.com/xi/APPL/Global2
Software Component Version in ESR	As of SAP_APPL 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	ECC_CUSTBANKDETAILSUPDRC

Use

Sales employees can use this inbound operation to update a customer's bank details. You can create new bank details or change and delete existing bank details.

Features

This inbound operation processes the following message types:

- CustomerERPBankDetailsUpdateRequest_sync
- CustomerERPBankDetailsUpdateConfirmation_sync

The following values must be supplied:

- CustomerID
- ChangeStateID

The system distinguishes between two different update queries:

- Create/change/delete bank details for which all bank detail data is **known** (country, bank key, account number).

  The following elements must be filled:

  - Bank-CountryCode
  - Bank-InternalID
  - BankAccountID

  These elements uniquely identify the bank details.

- Create/change/delete bank details for which the account number is **not known**.

  In this case, you can just enter the IBAN and create, change, or delete the corresponding bank details. To create, the "IBAN without account number" function must be activated.

  The following element must be filled:

  - BankAccountStandardID

You can process several sets of bank details using the update query. The action you need to carry out is exactly defined for each set of bank details. You use the actionCode attribute for this.

actionCode can have the following values:

- 01: Create Bank Details
- 02: Change Existing Bank Detals
- 03: Delete Existing Bank Details

When you **change** bank details, there are some elements that may not be changed. This depends on the type of bank details:

- Create bank details for which all bank detail data is **known** (country, bank key, account number).

  The following elements may not be changed:

  - BankAccountIDCheckDigitValue: If an IBAN exists, you may not change this information.
  - Bank-StandardID
  - Bank-BankName

- Change/delete bank details for which the account number is **not known** (IBAN without account number)

  The following elements may not be changed:

  - BankAccountID
  - BankAccountIDCheckDigitValue
  - Bank-InternalID
  - Bank-StandardID

- Bank-CountryCode
- Bank-BankName

When you **create** bank details, there are some elements that may not be changed:

- Element BankAccountIDCheckDigitValue: If you want to create bank details for which the account number is not known (IBAN without account number), do not enter the bank control key. If you fill this element anyway, an error message appears.

Integration

This inbound operation belongs to the inbound service interface Manage Customer In [Page 89].

Constraints

Field Name	Meaning	Maximum No. of Characters in GDT	Maximum No. of Characters in Backend
CustomerID	Customer Number	16	10 Only entries with 10 characters or less are valid.
BankAccountID	Account Number	35	18
BankAccountHolderName	Account Holder	80	60
BankInternalID	Bank Key	18	15

Action 01 - Create

- Element BusinessPartnerBankDetailsID:
  The partner bank category must be unique, meaning two sets of bank details cannot use the same partner bank category. That means that when you create bank details, you cannot enter a partner bank category that already exists. If you do, an error message appears.
- Element Bank-StandardID:
  The SWIFT code cannot be determined on an individual basis. The system always uses the SWIFT code from the master record of the respective bank. If you enter a SWIFT code that is different than the one in the master record, an error message appears.
- Element Bank-BankName:
  You can enter the bank names individually. However, the system might not consider the name you enter because it always considers the name from the bank's master record. The message that then appears does not, however, explain this behavior.

Notes for SAP Back-End System Administration

Prerequisites

You are using the *Master Data Synchronization (AP-MD-BF-SYN)* application component.

If you want to create bank details without an account number and you enter an IBAN, you must activate the "IBAN without Account Number" function. To activate this function, follow this Customizing path:

Financial Accounting (New) → Accounts Receivable and Accounts Payable → Activate IBAN without Bank Account Number

Processing

You can use the Business Add-In (BAdI) BPC_APPL_SE_CUSTBANKDETUPDRC for this operation.

Update Customer Basic Data

Technical Data

Entity Type	Service operation
Technical Name in Enterprise Services Repository (ESR)	CustomerERPBasicDataUpdateRequestConfirmation_In
Namespace in ESR	http://sap.com/xi/APPL/Global2
Software Component Version in ESR	As of SAP_APPL 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	ECC_CUSTOMERBASICDATAUPDRC

Use

Sales employees can use this inbound operation to update a customer's master data. The master data contains the general data about a customer as well as address and communication data.

Features

This inbound operation processes the following message types:

- CustomerERPBasicDataUpdateRequest_sync
- CustomerERPBasicDataUpdateConfirmation_sync

The following values must be supplied:

- CustomerID
- ChangeStateID

Integration

This inbound operation belongs to the inbound service interface Manage Customer In [Page 89].

Constraints

Field Name	Meaning	Maximum No. of Characters in GDT	Maximum No. of Characters in Backend
CustomerID	Customer Number	16	Only entries with 10 characters or less are valid.

RegionCode	Region	6	3

Only standard fields are supported for the update.

Values in fields that are not supported are not processed further.

Supported Standard Field	Non Supported Fields
PhoneNumber	CountryDialingCode, CountryName
MobilePhoneNumber	CountryDialingCode, CountryName
FacsimileNumber	CountryDialingCode, CountryName
Standard E-mail address	--
Standard Internet address	--

Notes for SAP Back-End System Administration

Prerequisites

For more information, see Customer [Page 88].

Processing

You can use the Business Add-In (BAdI) BPC_APPL_SE_CUSTBDUPDRC for this operation.

Create Customer Relationship Contact Person

Technical Data

Entity Type	Service operation
Technical Name in Enterprise Services Repository (ESR)	CustomerERPRelationshipContactPerson CreateRequestConfirmation_In
Namespace in ESR	http://sap.com/xi/APPL/Global2
Software Component Version in ESR	As of SAP_APPL 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	ECC_SALESARRANGEMENTCRTRC

Use

Sales employees use this inbound operation to create the contact person of a customer relationship.

Features

This inbound operation processes the following message types:

- CustomerERPRelationshipContactPersonCreateRequest_sync
- CustomerERPRelationshipContactPersonCreateConfirmation_sync

The following value must be supplied:

- CustomerID

Do not supply the following value; it is generated as soon as a contact person is created for a customer:

- ContactPerson-EmployeeID

Integration

This inbound operation belongs to the inbound service interface Manage Customer In [Page 89].

Constraints

Field Name	Meaning	Maximum No. of Characters in GDT	Maximum No. of Characters in Backend
CustomerID	Customer Number	16	10

For telephone numbers and Internet connections, only values in the standard fields are supported:

Supported Standard Field	Non Supported Fields
Standard Phone Number	CountryDialingCode, CountryName
Standard Mobile Phone Number	CountryDialingCode, CountryName
Standard Fax Number	CountryDialingCode, CountryName
Standard E-mail address	
Standard Internet address	

Notes for SAP Back-End System Administration

Prerequisites

You have activated the contact person assignment in Customizing.

Processing

You can use the Business Add-In (BAdI) BPC_APPL_SE_CUSTRLCNTPERSCRTRC for this operation.

Update Customer Relationship Contact Person

Technical Data

Entity Type	Service operation
Technical Name in Enterprise Services Repository (ESR)	CustomerERPRelationshipContactPerson UpdateRequestConfirmation_In
Namespace in ESR	http://sap.com/xi/APPL/Global2
Software Component Version in ESR	As of SAP_APPL 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	ECC_CUSTOMERCONTACTPERSONUPDRC

Use

Sales employees use this inbound operation to update the contact person of a customer relationship.

Features

This inbound operation processes the following message types:

- CustomerERPRelationshipContactPersonUpdateRequest_sync
- CustomerERPRelationshipContactPersonUpdateConfirmation_sync

The following values must be supplied:

- ID
- ChangeStateID
- InternalID

Integration

This inbound operation belongs to the inbound service interface Manage Customer In [Page 89].

Constraints

Field Name	Maximum No. of Characters in GDT	Maximum No. of Characters in Backend
CustomerID	16	10
ChangeStatefield	40	10
ContactPersonInternalID	60	10
EmployeeID	20	10
HigherContactPersonInternalID	32	10
ContactPersonFunctionalAreaCode	1	2

For telephone numbers and Internet connections, only values in the standard fields are supported:

Supported Standard Field	Non Supported Fields
Standard Phone Number	CountryDialingCode, CountryName
Standard Mobile Phone Number	CountryDialingCode, CountryName
Standard Fax Number	CountryDialingCode, CountryName
Standard E-mail address	

Notes for SAP Back-End System Administration

Prerequisites

You are using the *Master Data Synchronization (AP-MD-BF-SYN)* application component.

You have activated the contact person assignment in Customizing.

Processing

You can use the Business Add-In (BAdI) BPC_APPL_SE_CUSTRLCNTPERSUPDRC with enhancement spot BPC_SPOT_APPL_SE_CUSTOMER for this operation.

Cancel Customer Relationship Contact Person

Technical Data

Entity Type	Service operation
Technical Name in Enterprise Services Repository (ESR)	CustomerERPRelationshipContactPerson CancelRequestConfirmation_In
Namespace in ESR	http://sap.com/xi/APPL/Global2
Software Component Version in ESR	As of SAP_APPL 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	ECC_CUSTOMERCONTACTPERSONCNCRC

Use

Sales employees use this inbound operation to cancel the contact person of a customer relationship.

Features

This inbound operation processes the following message types:

- CustomerERPRelationshipContactPersonCancelRequest_sync
- CustomerERPRelationshipContactPersonCancelConfirmation_sync

The following values must be supplied:

- ID
- InternalID

Integration

This inbound operation belongs to the inbound service interface Manage Customer In [Page 89].

Constraints

Field Name	Meaning	Maximum No. of Characters in GDT	Maximum No. of Characters in Backend
CustomerID	Customer Number	16	10
ContactPersonInternalID	Contact Person (number)	60	10

Notes for SAP Back-End System Administration

Prerequisites

You are using the *Master Data Synchronization (AP-MD-BF-SYN)* application component.

You have activated the contact person assignment in Customizing.

Processing

You can use the Business Add-In (BAdI) BPC_APPL_SE_CUSTRLCNTPERSCNC with enhancement spot BPC_SPOT_APPL_SE_CUSTOMER for this operation.

Sales Arrangement

Technical Data

Entity Type	Business Object
Software Component	*SAP ERP enterprise services package* (as of ECC-SE 600) SAP APPL (as of SAP_APPL 602)

Definition

Arrangement between a sales department and a customer for transacting sales. The arrangement deals with, for example, payment conditions, invoice currency, and Incoterms. The arrangement in no way represents a contract with the customer.

Use

A sales employee needs the detailed information from a sales arrangement to plan and carry out the sales transaction with a customer.

Related Business Objects

- *Customer*
- *Sales Unit*

Notes for SAP Back-End System Administration

To use the operations contained in the business object, you must implement *Sales and Distribution (SD)* and have knowledge concerning *Master Data* → *Business Partners* → *Customers* and the *Sales Summary.*

Manage Sales Arrangement In

Technical Data

Entity Type	Service interface
Software Component	*SAP ERP enterprise services package* (as of ECC-SE 600) SAP APPL (as of SAP_APPL 602)
Category	Inbound
Mode	Synchronous

Use

This inbound service interface contains operations that the sales employees use to create, read, and update sales arrangements.

Prerequisites

The users must be very familiar with the nomenclature of the field names in the interfaces used.

Integration

This inbound service interface belongs to the process component Business Partner Data Management [Page 80] and the business object Sales Arrangement [Page 97].

Constraints

These service interface operations support most of the standard fields. Unessential fields are not supported.

Example: The standard telephone number is currently supported.

Notes for SAP Back-End System Administration

Standard Customizing is sufficient for using the operations in the backend.

Create Sales Arrangement

Technical Data

Entity Type	Service operation
Technical Name in Enterprise Services Repository (ESR)	SalesArrangementERPCreateRequestConfirmation_In
Namespace in ESR	http://sap.com/xi/APPL/Global2
Software Component Version in ESR	As of SAP_APPL 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	ECC_SALESARRANGEMENTCRTRC

Use

Sales employees can use this inbound operation to create a sales arrangement in a specific sales area.

Features

This inbound operation processes the following message types:

- SalesArrangementERPCreateRequest_sync
- SalesArrangementERPCreateConfirmation_sync

The user requires detailed knowledge of each and every field so that they can fill them with the correct data and further process them correctly.

All the codes are standard SAP codes used in *Create Sales Arrangement.*

The following values must be supplied:

- *CustomerID*
- *SalesOrganisationPartyInternalID*
- *DistributionChannelCode*
- *DivisionCode*

You can configure additional fields as required fields in the backend. You must then fill these fields as well.

Integration

This inbound operation belongs to the inbound service interface Manage Sales Arrangement In [Page 97].

Notes for SAP Back-End System Administration

Prerequisites

The following conditions must be fulfilled so you can process the data:

- You have carried out standard Customizing.
- You have filled the required entry fields.
- You have filled the required entry fields configured in Customizing.
- You are using the *Sales and Distribution (SD)* application component.

Processing

There are four required entry fields. You can configure all the other fields as required entry fields in Customizing.

The backend system can fill data records with additional data. This data is stored in Customizing tables.

You can use the Business Add-In (BAdI) BPC_APPL_SE_SALESARRGMENTCRTRC for this operation.

Update Sales Arrangement

Technical Data

Entity Type	Service operation
Technical Name in Enterprise Services Repository (ESR)	SalesArrangementERPUpdateRequest Confirmation_In
Namespace in ESR	http://sap.com/xi/APPL/Global2
Software Component Version in ESR	As of SAP_APPL 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	ECC_SALESARRANGEMENTUPDRC

Use

Sales employees can use this inbound operation to update a sales arrangement in a specific sales area.

Features

This inbound operation processes the following message types:

- SalesArrangementERPUpdateRequest_sync
- SalesArrangementERPUpdateConfirmation_sync

The following values must be supplied:

- *CustomerID*
- *SalesOrganisationPartyInternalID*
- *DistributionChannelCode*
- *DivisionCode*

You can configure additional fields as required entry fields in the backend. You must then fill them.

Integration

This inbound operation belongs to the inbound service interface Manage Sales Arrangement In [Page 97].

Notes for SAP Back-End System Administration

Prerequisites

The following conditions must be fulfilled so you can process the data:

- You have carried out standard Customizing.
- You have filled the required entry fields.
- You have filled the required entry fields configured in Customizing.
- You are using the *Sales and Distribution (SD)* application component.

Processing

You can use the Business Add-In (BAdI) BPC_APPL_SE_SAARRANGEMENTUPDRC for this operation.

Enterprise Services for Kanban Processing

Technical Data

Entity Type	Process Component
Software Component	*SAP ERP enterprise services package* (as of SAP_APPL 602)

Definition

Kanban Processing enables production control and material flow based on the actual stock quantity in production.

Kanban is a method for controlling production and material flow based on the actual stock quantity in production. Material that is required on a regular basis is continually provided in small quantities in production. Replenishment or the production of a material is only triggered when a higher production level actually requires the material. In Kanban, the signal for material replenishment can be triggered, for example, by the work center that requires the material (demand source) by sending a card to the work center that is responsible for manufacturing the material (supply source). This card describes which material is required, the quantity of the material required and where the material is to be delivered.

Use

This process component is used to invoke Kanban operations in the ERP system. You can use this process component to retrieve Kanbans and Kanban Control Cycles based on the search criteria. Also, you can change the status of Kanbans.

Notes for SAP Back-End System Administration

The Kanban Processing process component contains the business object, Kanban.

Kanban

Technical Data

Entity Type	Business object
Software Component	*SAP ERP enterprise services package* (as of SAP_APPL 602)

Definition

Kanban is a means of signalling the demand status for a certain material. Kanban is the information that is exchanged between the demand and the supply source in a pull environment. It contains all information (plant, material, quantity, delivery information) that is needed by the demand source and the supply source to process replenishment. The information can then be obtained in printed or electronic form.

Use

The *Kanban* business object contains operations with which you can get the details of kanbans and the status of kanbans.

Notes for SAP Back-End System Administration

Prerequisites

You have to use the SAP ERP system for the operations in this process component.

Query Kanban Out

Technical Data

Entity Type	Service interface
Software Component	*SAP ERP enterprise services package* (as of SAP_APPL 602)
Category	Outbound

Mode	Synchronous

Use

The *Query Kanban Out* service interface enables you to obtain the `TagID` information of a Kanban from SAP Auto-ID Infrastructure system (SAP AII).

Integration

The *Query Kanban Out* outbound service interface belongs to Kanban Processing [Page 100] process component and Kanban [Page 101] business object.

Structure

This outbound service interface contains the outbound operation, `AutomaticIdentificationLabelByElementsQueryResponse_Out` (Query Kanban Tag for Printing [Extern])

Query Kanban Tag for Printing

Technical Data

Entity Type	Operation
Technical Name in Enterprise Services Repository (ESR)	AutomaticIdentificationLabelByElementsQueryResponse_Out
Namespace in ESR	http://sap.com/xi/APPL/Global2
Software Component Version in ESR	As of SAP_APPL 602
Category	Outbound
Mode	Synchronous

Use

This *Query Kanban Tag for Printing* outbound operation allows you to obtain the RFID `TagID` information for a given `KanbanCardID`. You have to select the following:

- `ReferenceObjectID` (Kanban Card Identifier)
- `SchemeAgencyID` (Logical System Name)
- `ReferenceObjectType` (282 for Kanban)

Features

This outbound operation processes the following messages:

- *AutomaticIdentificationLabelByElementsQueryMessage_sync*

  This message is a query to Kanban Processing process component to retrieve the RFID `TagID` of a Kanban from the SAP Auto-ID Infrastructure system (SAP AII).

- *AutomaticIdentificationLabelByElementsResponseMessage_sync*

  This message retrieves RFID `TagID` retrieved from the SAP AII system.

Although this service's input message type contains an element *MessageHeader/UUID*, the service does **not** support idempotent behavior as described in the *mySAP Service*

Provisioning handbook. Since support for idempotent behavior may be added in the future, we recommend that you do not use this UUID element for the time being. For more information, see the *Service Provisioning Handbook*.

Integration

The *Query Kanban Tag for Printing* operation belongs to *Query Kanban Out* service interface.

Notes for SAP Back-End System Administration

Prerequisites

You must have Kanban data in the system.

Price Master Data Management

Technical Data

Entity Type	Process component
Software Component	EA-RETAIL 602

This process component contains the base business objects for price and promotion optimization. Price Master Data Management enables a company to manage price-related data both for sales and procurement processes. It includes management of prices, discounts and surcharges.

Sales Price Information

Technical Data

Entity Type	Business object
Software Component	EA-RETAIL 602

This business object provides information regarding a product's retail sales price (for example, the price a consumer has to pay for a product in a specific location).

Sales Price Information Out

Technical Data

Entity Type	Service interface
Software Component	EA-RETAIL 602
Category	Asynchronous
Mode	Outbound

This outbound service interface contains an operation that requests price master data from an external Demand Management System.

Request Sales Price Information as Bulk

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	SalesPriceInformationBulkRequest_Out
Software Component Version in ESR	EA-RETAIL 602
Category	Outbound
Mode	Asynchronous
Related Web Service Definition	Not applicable

This outbound operation generates a request to an external demand management system to transmit price master data.

Sales Price Specification Calculation

Technical Data

Entity Type	Business object
Software Component	EA-RETAIL 602

This business object contains calculated sales prices based on the factors influencing pricing for specific data retention levels.

Calculating Sales Price Specification In

Technical Data

Entity Type	Service interface
Software Component	EA-RETAIL 602
Category	Asynchronous
Mode	Inbound

This inbound service interface contains an operation that requests price master data changes from an external Demand Management System.

Create Sales Price Specification Calculation

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	SalesPriceSpecificationCalculationCreationRequest_In
Software Component Version in ESR	EA-RETAIL 602
Category	Inbound
Mode	Asynchronous
Related Web Service Definition	Not applicable

This inbound operation generates a request to an external demand management system to transmit changes in price master data during optimization.

Retail Event Management

Technical Data

Entity Type	Process component
Software Component	EA-RETAIL 602

Retail Event

Technical Data

Entity Type	Business object
Software Component	EA-RETAIL 602

A RetailEvent is a Retail specific business activity to increase sales, for example store opening, sports event, promotion. Potentially affected business processes (procurement, logistics, sales prices, media, arrangement, vendor collaboration etc.) are associated to the RetailEvent.

In today's world retailer face increasing competition and eroding margins, customers have become more and more price sensitive and are trained to wait for promotional deals. Many retailers have resorted to spending more money on promotions and have increased the frequency of them. In certain segments, like grocery and drug, there is usually a promotion running every week. Although some retailers pursue an everyday low cost pricing strategy and do not promote very much, some retailers opt for a Hi-Lo pricing strategy and promotions are a key component of their business model.

Well planned and executed promotions can have a major impact on sales volume in the retail channel. In fact, it is not uncommon to generate many months worth of sales in a single week's promotion. But there are risks involved with the promotional process. If product demand is higher than expected, consumers can walk away empty-handed, which results in missed sales opportunities and dissatisfied shoppers. Conversely, if retailers overestimate demand, they are left with excess inventory, which means lower overall profitability for these items. So, if a retailer can do this well by using applications that support them, they can create a competitive advantage. One key success factor is a strong decision support to guide the user in deciding which products to include in a promotion, which quantities to order and of course for which prices to sell the products promoted to achieve an optimal result of the promotion. Promotion optimization tools make use of sophisticated statistical demand and profit modelling to elevate the role of consumer demand in the decision support process. They provide the possibility to simulate the most profitable combinations of products and prices to include in a promotion. These tools also enable 'what-if" simulations that take price elastic ties, cross-elastic ties of products, cannibalization, and halo effects into account all within the context of defined constraints and business rules.

Manage Retail Event In

Technical Data

Entity Type	Service interface
Software Component	EA-RETAIL 602
Categorie	Inbound
Mode	Asynchronous

Create Retail Event

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	RetailEventCreateRequest_In
Namespace in ESR	http://sap.com/xi/EA-RETAIL/Global2
Software Component Version in ESR	EA-RETAIL 602

Category	Asynchronous
Mode	Inbound
Related Web Service definition	Asynchronous_XI_Defintion_Only

A RetailEventCreateRequest is a request from Retail Demand Management Retail Event Management to a retailer's Retail Event Management with the purpose of creating a retail event to increase sales.

Features

A Retail Event is a retail specific business activity to increase sales. Allowing the creation of retail events in ERP from the outside enables the use of 3rd party planning and optimization tools (demand management systems). This will improve the efficiency of the retail events and therefore will increase sales.

Retail Merchandise Distribution Management

Technical Data

Entity Type	Process component
Software Component	EA-RETAIL 602

Retail Merchandise Distribution Management comprises Business Objects related to supplying and logistical processes to distribute products to stores in the retail industry.

Retail Allocation

Technical Data

Entity Type	Business object
Software Component	EA-RETAIL 602

Retail Allocation plans and controls distribution of merchandise as a part of the procurement process between external suppliers, distribution centers, and stores in the retail industry.

Retailers often have to centrally procure and distribute merchandise for a large number of recipients (for example, stores). Via allocations, it is possible to plan and control relevant quantities, dates, delivery relationships and delivery phases.

There are various scenarios

- merchandise is delivered directly from the vendor to the recipient

- from the vendor to a location and then to the recipient
- from a location to a recipient

Processes following the push as well as the pull principle can be implemented:

- push principle: an overall quantity of a product is distributed according to rules/ratios/strategies to the recipients
- pull principle: demand originating from the recipients is bundled and procured

The operational part of the business process (procuring or delivering the goods) is executed via appropriate documents, that can be automatically created as follow-on documents with reference to the Retail Allocation e.g. Purchase Orders (procurement for ShipFromLocations or for stores (direct store delivery)).

Furthermore, Retail Allocations allow to plan and trigger distribution processes relevant in retail business processes, for example cross-docking and flow-through. There are various ways to create a Retail Allocation automatically, or manually, depending on the business process, e.g.

- Promotion:

  In a promotion (business object) products, quantities and participating stores are planned on a high level; an allocation is created automatically as a follow-on document of the promotion to plan and control the procurement aspect on a more detailed level.

- Purchasing contract/purchase order/shipping notification:

  Process following the push principle starting with a procurement document (vendor to distribution center).

- Merchandise and Assortment Planning:

  A Retail Allocation is created based on planning data in MAP.

- Markdown Optimization: an Retail Allocation is created based on planned allocation data of an markdown optimization tool.

Create Retail Allocation

Technical Data

Entity Type	Operation
Namespace in ESR	http://sap.com/xi/EA-RETAIL/Global2
Software Component Version in ESR	EA-GLTRADE 602
Category	Inbound
Mode	Asynchronous
Related Web Service definition	Asynchronous_XI_Defintion_Only

Retail Allocation Create Request is a request from Retail Demand Management Merchandise Distribution Management to a retailer's Retail Merchandise Distribution Management with the purpose of creating a retail allocation to distribute materials from locations to stores.

Features

RetailAllocation plans and controls distribution of merchandise as a part of the procurement process between external suppliers, ShipFromLocations, and stores in the retail industry. Allowing the creation of retail allocations in ERP from the outside enables to trigger the follow-on processes by using external data e.g. optimized planned distributed quantities of materials for stores.

Trade Item Catalogue Authoring

The trade item catalogue authoring describes the process for a retailer to request, receive and check the quality of the received catalog content and determine when, and to what extent the catalogs are published for use in other business processes.

This process component (together with Trade Item Catalogue Publishing) is part of the Global Data Synchronization ES bundle.

This new ES bundle allows retailers to connect to many manufacturers easily and receive only the data they need in a single easy-to-use format. It eliminates the retail headache of manually tracking and updating information on hundreds or even thousands of products. The Global Data Synchronization ES bundle allows a retailer to subscribe to a particular product or group of products from a manufacturer through a data pool or on a peer-to-peer basis, directly with a manufacturer. Whenever the product size, availability, or other feature changes, the latest information is automatically sent to the retailer. The Global Data Synchronization ES bundle leverages enterprise SOA by connecting a retailer's ERP data to that of his manufacturers and suppliers using XML messaging through GDSN. The Global Data Synchronization ES bundle can be used by retail companies of any size and type. It is especially useful for retailers with ever-changing product assortments. The Global Data Synchronization ES bundle is typically operated out of a company's headquarters, purchasing departments, and so on. The Global Data Synchronization ES bundle is useful for any retailer seeking to streamline and automate its data management systems.

Constraints

Notes for SAP Back-End System Administration

More Information

For more informations, see SAP Library

SAP Retail ▸ Distributed Data Processing ▸ Master Data Exchange Using GDSN and EANCOM-PRICAT Messages ▸ Sending and Receiving GDSN Retail Messages

Trade Item Catalogue

A structured directory of catalog items, where each catalog item represents a product or price specification and provides information about it.

TradeItemCatalogue is used to exchange data about available products between vendors and retailers through Recipient Data Pool. It can contain items (for example unit of measure, description) or item price information.

Related Business Object: Published Trade Item Catalogue

Constraints

Notes for SAP Back-End System Administration

More Information

For more informations, see SAP Library

SAP Retail ▸ Distributed Data Processing ▸ Master Data Exchange Using GDSN and EANCOM-PRICAT Messages ▸ Sending and Receiving GDSN Retail Messages

Trade Item Catalogue Subscription Sending In

Technical Data

Entity Type	Service-Interface
Software Component	EA-RETAIL 700
Category	Inbound
Mode	Asynchronous

This interface is used for receiving success or error messages for sent subscriptions of the retailer. This can be used in trading industry. This inbound service interface contains an operation that is the confirmation to TradeItemCatalogueAuthoring of a TradeItemCatalogueSubscriptionRequest or TradeItemCatalogueSubscriptionCancelRequest by the Recipient Data Pool.

Prerequisites

Integration

This inbound service interface belongs to process component Trade Item Catalogue Authoring and business object Trade Item Catalogue. This inbound service interface encompasses the following inbound operation: Change Trade Item Catalogue based on Confirmation.

Constraints

Notes for SAP Back-End System Administration

More Information

For more informations, see SAP Library

SAP Retail ▸ Distributed Data Processing ▸ Master Data Exchange Using GDSN and EANCOM-PRICAT Messages ▸ Sending and Receiving GDSN Retail Messages

Change Trade Item Catalogue based on Confirmation

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	TradeItemCatalogueSubscriptionConfirmationMessage_async_In
Namespace in ESR	http://sap.com/xi/EA-RETAIL/VP
Software Component Version in ESR	EA_RETAIL 700
Category	Inbound
Mode	Asynchronous
Related Web Service definition	

This is a confirmation of the sent subscription or cancel subscription message. A TradeItemCatalogueSubscriptionConfirmation is the confirmation to TradeItemCatalogueAuthoring of a TradeItemCatalogueSubscriptionRequest or TradeItemCatalogueSubscriptionCancelRequest by the Recipient Data Pool. The structure of the message type TradeItemCatalogueSubscriptionConfirmation is specified by the message data type TradeItem.

Features

This inbound operation processes the following message type: TradeItemCatalogueSubscriptionConfirmationMessage_async.

Prerequisites

Integration

This inbound operation belongs to the inbound service interface Trade Item Catalogue Subscription Sending In.

Constraints

Notes for SAP Back-End System Administration

More Information

For more informations, see SAP Library

SAP Retail ▸ Distributed Data Processing ▸ Master Data Exchange Using GDSN and EANCOM-PRICAT Messages ▸ Sending and Receiving GDSN Retail Messages

Trade Item Catalogue Subscription Sending Out

Technical Data

Entity Type	Service-Interface
Software Component	EA-RETAIL 700
Category	Outbound
Mode	Asynchronous

This interface is used for sending out a subscription or request from the retailer's ERP to the external data pool to subscribe or request selected vendor data. This outbound service interface groups operations that are requests from Trade Item Catalogue Authoring for a subscription or to cancel a subscription.

Prerequisites

Integration

This outbound service interface belongs to process component Trade Item Catalogue Authoring and business object Trade Item Catalogue.

This outbound service interface encompasses the following outbound operations: Request Trade Item Catalogue Subscription and Request Trade Item Catalogue Subscription Cancellation.

Constraints

Notes for SAP Back-End System Administration

More Information

For more informations, see SAP Library

SAP Retail ▸ Distributed Data Processing ▸ Master Data Exchange Using GDSN and EANCOM-PRICAT Messages ▸ Sending and Receiving GDSN Retail Messages

Request Trade Item Catalogue Subscription

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	TradeItemCatalogueSubscriptionRequestMessage_async_Out
Namespace in ESR	http://sap.com/xi/EA-RETAIL/VP
Software Component Version in ESR	EA_RETAIL 700

Category	Outbound
Mode	Asynchronous
Related Web Service definition	

A TradeItemCatalogueSubscriptionRequest is a request from TradeItemCatalogueAuthoring sent to a recipient data pool for a subscription. The structure of the message type TradeItemCatalogueSubscriptionRequest is specified by the message data type TradeItemCatalogue. This operation is used to subscribe to the information of selected vendor products. It is used for Trading industry.

Features

This outbound operation processes the following message type: TradeItemCatalogueSubscriptionRequestMessage_async.

Prerequisites

Integration

This outbound operation belongs to the outbound service interface Trade Item Catalogue Subscription Sending Out.

Constraints

Notes for SAP Back-End System Administration

More Information

For more informations, see SAP Library

SAP Retail ▸ Distributed Data Processing ▸ Master Data Exchange Using GDSN and EANCOM-PRICAT Messages ▸ Sending and Receiving GDSN Retail Messages

Request Trade Item Catalogue Subscription Cancellation

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	TradeItemCatalogueSubscriptionCancelRequestMessage_async_Out
Namespace in ESR	http://sap.com/xi/EA-RETAIL/VP
Software Component Version in ESR	EA_RETAIL 700

Category	Outbound
Mode	Asynchronous
Related Web Service definition	

A TradeItemCatalogueSubscriptionCancelRequest is a request from TradeItemCatalogueAuthoring to cancel a subscription. The structure of the message type TradeItemCatalogueSubscriptionCancelRequest is specified by the message data type TradeItemCatalogue.

Features

This outbound operation processes the following message type: TradeItemCatalogueSubscriptionCancelRequestMessage_asyncMTMT

Prerequisites

Integration

This outbound operation belongs to the outbound service interface Trade Item Catalogue Subscription Sending Out.

Constraints

Notes for SAP Back-End System Administration

More Information

For more informations, see SAP Library

SAP Retail ▸ Distributed Data Processing ▸ Master Data Exchange Using GDSN and EANCOM-PRICAT Messages ▸ Sending and Receiving GDSN Retail Messages

Trade Item Catalogue Transmission Receiving In

Technical Data

Entity Type	Service-Interface
Software Component	EA-RETAIL 700
Category	Inbound
Mode	Asynchronous

This inbound service interface contains an operation that is a request for the creation of Trade Item Catalogue to Trade Item Catalogue Authoring by the recipient data Pool.

Prerequisites

Integration

This inbound service interface belongs to process component Trade Item Catalogue Authoring and business object Trade Item Catalogue.

This inbound service interface encompasses the following inbound operation: Maintain Trade Item Catalogue.

Constraints

Hinweise für die SAP-Backend-Systemverwaltung

More Information

For more informations, see SAP Library

SAP Retail ▸ Distributed Data Processing ▸ Master Data Exchange Using GDSN and EANCOM-PRICAT Messages ▸ Sending and Receiving GDSN Retail Messages

Maintain Trade Item Catalogue

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	TradeItemCatalogueTransmissionRequestMessage_async_In
Namespace in ESR	http://sap.com/xi/EA-RETAIL/VP
Software Component Version in ESR	EA_RETAIL 700
Category	Inbound
Mode	Asynchronous
Related Web Service definition	

The TradeItemCatalogueTransmissionRequest contains information of new or updated vendor products and is stored as catalogue in the receiving system. A TradeItemCatalogueTransmissionRequest is a request for the creation of TradeItemCatalogue to TradeItemCatalogueAuthoring by the recipient Data Pool. The structure of the message type TradeItemCatalogueTransmissionRequest is specified by the message data type TradeItemCatalogue.

Features

This inbound operation processes the following message type: TradeItemCatalogueTransmissionRequestMessage_async.

Prerequisites

Integration

This inbound operation belongs to the inbound service interface Trade Item Catalogue Transmission Receiving In.

Constraints

Notes for SAP Back-End System Administration

More Information

For more informations, see SAP Library

SAP Retail ▸ Distributed Data Processing ▸ Master Data Exchange Using GDSN and EANCOM-PRICAT Messages ▸ Sending and Receiving GDSN Retail Messages

Trade Item Catalogue Transmission Receiving Out

Technical Data

Entitätstyp	Service-Interface
Softwarekomponente	EA-RETAIL 700
Kategorie	Outbound
Modus	Asynchronous

This outbound service interface contains an operation that is a confirmation from Trade Item Catalogue Authoring of the correct transmission of requested Trade Item Catalogues.

Prerequisites

Integration

This outbound service interface belongs to process component Trade Item Catalogue Authoring and business object Trade Item Catalogue.

This outbound service interface encompasses the following outbound operation: Confirm Trade Item Catalogue Transmission.

Constraints

Notes for SAP Back-End System Administration

More Information

For more informations, see SAP Library

SAP Retail ▸ Distributed Data Processing ▸ Master Data Exchange Using GDSN and EANCOM-PRICAT Messages ▸ Sending and Receiving GDSN Retail Messages

Confirm Trade Item Catalogue Transmission

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	TradeItemCatalogueTransmissionConfirmationMessage_async_Out
Namespace in ESR	http://sap.com/xi/EA-RETAIL/VP
Software Component Version in ESR	EA_RETAIL 700
Category	Outbound
Mode	Asynchronous
Related Web Service definition	

A TradeItemCatalogueTransmissionConfirmation is a confirmation from TradeItemCatalogueAuthoring of the correct transmission of requested TradeItemCatalogues. The structure of the message type TradeItemCatalogueTransmissionConfirmation is specified by the message data type TradeItemCatalogue.

Features

This outbound operation processes the following message type: TradeItemCatalogueTransmissionConfirmationMessage_async.

Prerequisites

Integration

This outbound operation belongs to the outbound service interface Item Catalogue Transmission Receiving Out.

Constraints

Notes for SAP Back-End System Administration

More Information

For more informations, see SAP Library

SAP Retail ▸ Distributed Data Processing ▸ Master Data Exchange Using GDSN and EANCOM-PRICAT Messages ▸ Sending and Receiving GDSN Retail Messages

Trade Item Catalogue Transmission Request Out

Technical Data

Entity Type	Service-Interface
Software Component	EA-RETAIL 700
Category	Outbound
Mode	Asynchronous

This outbound service interface contains an operation that is a request from Trade Item Catalogue Authoring for a Trade Item Catalogue Transmission Request.

Prerequisites

Integration

This outbound service interface belongs to process component Trade Item Catalogue Authoring and business object Trade Item Catalogue.

This outbound service interface encompasses the following outbound operation: Request for Trade Item Catalogue Transmission Request.

Constraints

Notes for SAP Back-End System Administration

More Information

For more informations, see SAP Library

SAP Retail ▸ Distributed Data Processing ▸ Master Data Exchange Using GDSN and EANCOM-PRICAT Messages ▸ Sending and Receiving GDSN Retail Messages

Request for Trade Item Catalogue Transmission Request

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	RequestForTradeItemCatalogueTransmissionRequest_Out
Namespace in ESR	http://sap.com/xi/EA-RETAIL/VP
Software Component Version in ESR	EA_RETAIL 700

Category	Outbound
Mode	Asynchronous
Related Web Service definition	

This message is used to request already received catalogue data from the data pool. A TradeItemCatalogueTransmissionRequestRequest is a request from TradeItemCatalogueAuthoring for a TradeItemCatalogueTransmissionRequest. The structure of the message type TradeItemCatalogueTransmissionRequestRequest is specified by the message data type TradeItemCatalogue.

Features

This outbound operation processes the following message type: TradeItemCatalogueTransmissionRequestRequestMessage_async.

Prerequisites

Integration

This outbound operation belongs to the outbound service interface Trade Item Catalogue Transmission Request Out.

Constraints

Notes for SAP Back-End System Administration

More Information

For more informations, see SAP Library

SAP Retail ▸ Distributed Data Processing ▸ Master Data Exchange Using GDSN and EANCOM-PRICAT Messages ▸ Sending and Receiving GDSN Retail Messages

Trade Item Catalogue Publishing

Technical Data

Entity Type	Process component
Software Component	EA-RETAIL 700

The process trade item catalogue publishing describes the synchronization steps when a retailer publish the received catalog data to the ERP back end.

This process component (together with Trade Item Catalogue Authoring) is part of the Global Data Synchronization ES bundle.

This new ES bundle allows retailers to connect to many manufacturers easily and receive only the data they need in a single easy-to-use format. It eliminates the retail headache of

manually tracking and updating information on hundreds or even thousands of products. The Global Data Synchronization ES bundle allows a retailer to subscribe to a particular product or group of products from a manufacturer through a data pool or on a peer-to-peer basis, directly with a manufacturer. Whenever the product size, availability, or other feature changes, the latest information is automatically sent to the retailer. The Global Data Synchronization ES bundle leverages enterprise SOA by connecting a retailer's ERP data to that of his manufacturers and suppliers using XML messaging through GDSN. The Global Data Synchronization ES bundle can be used by retail companies of any size and type. It is especially useful for retailers with ever-changing product assortments. The Global Data Synchronization ES bundle is typically operated out of a company's headquarters, purchasing departments, and so on. The Global Data Synchronization ES bundle is useful for any retailer seeking to streamline and automate its data management systems.

Constraints

Notes for SAP Back-End System Administration

More Information

For more informations, see SAP Library

SAP Retail Distributed Data Processing Master Data Exchange Using GDSN and EANCOM-PRICAT Messages Sending and Receiving GDSN Retail Messages

Published Trade Item Catalogue

Technical Data

Entity Type	Business object
Software Component	EA-RETAIL 700

The published trade item catalogue is the business object which is used to take over the vendor data to the internal retailer data. This publishing process is needed as only a selection of items is overtaken into the operative data of the retailer.

Related Business Object: Trade Item Catalogue

Constraints

Notes for SAP Back-End System Administration

More Information

For more informations, see SAP Library

SAP Retail Distributed Data Processing Master Data Exchange Using GDSN and EANCOM-PRICAT Messages Sending and Receiving GDSN Retail Messages

Accepting Trade Item Catalogue Out

Technical Data

Entity Type	Service-Interface
Software Component	EA-RETAIL 700
Category	Outbound
Mode	Asynchronous

This outbound service interface contains an operation that is the notification from Trade Item Catalogue Publishing of the acceptance of a single Trade Item Catalogue item.

Prerequisites

Integration

This outbound service interface belongs to process component Trade Item Catalogue Publishing and business object Published Trade Item Catalogue.

This outbound service interface encompasses the following outbound operation: Notify of Trade Item Catalogue Acceptance.

Constraints

Notes for SAP Back-End System Administration

More Information

For more informations, see SAP Library

SAP Retail ▸ Distributed Data Processing ▸ Master Data Exchange Using GDSN and EANCOM-PRICAT Messages ▸ Sending and Receiving GDSN Retail Messages

Notify of Trade Item Catalogue Acceptance

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	TradeItemCatalogueAcceptanceNotificationMessage_async_Out
Namespace in ESR	http://sap.com/xi/EA-RETAIL/VP
Software Component version in ESR	EA-RETAIL 700
Category	Outbound

Mode	Asynchronous
Related Web Service definition	

A TradeItemCatalogueAcceptanceNotification is the notification from TradeItemCataloguePublishing of the acceptance of TradeItemCatalogue. The structure of the message type TradeItemCatalogueAcceptanceNotification is specified by the message data type TradeItemCatalogue.

Features

This outbound operation processes the following message type: TradeItemCatalogueAcceptanceNotificationMessage_async.

Prerequisites

Integration

This outbound operation belongs to the outbound service interface Accepting Trade Item Catalogue Out.

Constraints

Notes for SAP Back-End System Administration

More Information

For more informations, see SAP Library

SAP Retail ▸ Distributed Data Processing ▸ Master Data Exchange Using GDSN and EANCOM-PRICAT Messages ▸ Sending and Receiving GDSN Retail Messages

Enterprise Services for SAP ERP Logistics

Outbound Delivery Processing

Technical Data

Entity Type	Process component
Software Component	*SAP ERP enterprise services package* (as of ECC-SE 600) SAP APPL (as of SAP_APPL 602)

Definition

Logistical control process that involves the outbound delivery of goods to a product recipient.

Use

The process combines all tasks concerning the outbound delivery process (preparation and triggering execution) and enables communication with the product recipient and billing.

Notes for SAP Back-End System Administration

Prerequisites

To use the operations contained in the process component, you must run the *Shipping* (LE-SHP) application component.

Processing

Consider the following responses of the change and update operations for this process component, if you use them:

The data that is transferred using the request messages overwrites only the respective objects in the back-end system databases that are also filled in the message structure. This also applies to the individual items of list-based elements.

The result of this is as follows:If optional elements in a request message are not filled, then the back-end system leaves the values of the respective fields in the database. The only exception is if this value is automatically updated by changes to other data.

Outbound Delivery

Technical Data

Entity Type	Business object
Software Component	*SAP ERP enterprise services package* (as of ECC-SE 600) SAP APPL (as of SAP_APPL 602)

Definition

Grouping together of goods that a vendor provides for shipping. The outbound delivery is the basis for the shipping notification and delivery note that are sent to the product recipient.

Use

Previous methods with which you could search for outbound deliveries use proprietary technologies.

To simplify communication between the shipping clerk and the *Outbound Delivery Processing* process component, new operations were defined that use commonly available technologies.

For example, a shipping clerk can perform analyses for his or her line manager. To obtain an overview of pending outbound deliveries, the shipping clerk can perform and display queries.

Related Business Objects

- *InboundDelivery*
- *CustomerReturnsDelivery*
- *VendorReturnsDelivery*

Manage Outbound Delivery

Technical Data

Entity Type	Service interface
Software Component	*SAP ERP enterprise services package* (as of ECC-SE 602) SAP APPL (as of SAP_APPL 602)
Category	Outbound
Mode	Synchronous

Use

This outbound service interface groups together operations with which the shipping clerk can create, change, read or delete outbound deliveries.

Related Service Interfaces

- *Outbound Delivery Event Out*
- *Query Outbound Delivery*
- *Outbound Delivery Splitting In*
- *Outbound Delivery Splitting Out*
- *Site Logistics Confirmation In*

Integration

This outbound service interface belongs to the Outbound Delivery Processing [Page 122] process component and the Outbound Delivery [Page 123] business object.

Change Outbound Delivery

Technical Data

Entity Type	Service operation
Technical Name in Enterprise Services Repository (ESR)	OutboundDeliveryERPChangeRequestConfirmation_In
Namespace in ESR	http://sap.com/xi/APPL/ /Global2
Software Component Version in ESR	As of SAP_APPL 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	ECC_OUTBOUNDDELIVERYCHGRC

Use

A shipping clerk uses this inbound operation to change an outbound delivery.

Features

The inbound operation processes the following message types:

- OutboundDeliveryERPChangeRequest_sync
- OutboundDeliveryERPChangeConfirmation_sync

The following value must be delivered:

- Delivery number

The following values can be changed:

- Delivery header
  - Delivery note external identification
  - Bill of lading
  - Partner
  - Address
  - Planned delivery date
  - Planned goods issue date
  - Picking date
  - Loading date
  - Actual goods issue date
  - Transportation scheduling date
  - Gross weight
  - Net weight
  - Volume
- Handling unit

  In the HandlingUnitID, you can specify the external identification for a handling unit that is to be created or a handling unit that already exists.

  The packed quantity for a delivery item can be changed by entering a delta value.

  If the HandlingUnitListCompleteTransmissionIndicator is set, all existing handling units in the delivery are deleted.

  You can repack to another handling unit by entering a DestinationID.

  In the case of storage locations with a handling unit requirement, neither unpacking nor deletion of the affected handling unit is possible. You can only repack from one handling unit to another.
- Delivery item
  - Action code

    Action code for creating, changing or deleting.

    Only batch split subitems can be created. In this case, the number of the batch split main item must be specified. The item number cannot be predetermined. Instead, it contains only a temporary number that can be used, for example, to pack the delivery item.

    Only the item number must be specified when deleting.
  - Batch split main item

This field only needs to be specified when creating a new batch split subitem.

- Material number
- Batch number
- Date of production
- Shelf life expiration date
- Serial number
- Delivery quantity in sales unit
- Delivery quantity in stock keeping unit
- Pick quantity in stock keeping unit or sales unit

  The pick quantity refers to the pick order with the number of the delivery.
- handlingUnitListCompleteTransmissionIndicator

  If this indicator is set, all handling units are deleted before the new handling units in *HandlingUnit* are created.
- PickQuantitiesApplyIndicator

  If this indicator is set, the pick quantities of all delivery items are copied as delivery quantities.

Prerequisites

For more information, see Outbound Delivery Processing [Page 122].

Integration

This service operation belongs to the Manage Outbound Delivery [Page 124] service interface.

Production

Technical Data

Entity Type	Process component
Software Component Version	*SAP ERP enterprise services package* (as of SAP_APPL 602)

Definition

The *Production* process component combines all the activities required to manage the production execution process on the manufacturing shop floor.

Use

You can use this process component to retrieve production orders based on the search criteria, read and confirm production orders and for repetitive manufacturing confirmation. This process component is used for all business objects related to *Production*.

Notes for SAP Back-End System Administration

The *Production* process component contains the business object, Production Order.

Production Order

Technical Data

Entity Type	Business object
Software Component Version	*SAP ERP enterprise services package* (as of SAP_APPL 602)

Definition

An order to produce a specific quantity of materials within a pre-defined time. It contains all the information required for the actual execution of the production process.

Use

The *Production Order* business object contains operations with which you can provide user interfaces and input help for actual execution of a production. In a production area, there is a constant need for material requirements and used components. These materials should be available within a certain time frame. As a follow up, production schedulers and material requirement planners need to know the following details:

- Production orders to be executed today
- Work centers involved
- Requirements from in-house, projects or from sales orders that can be fulfilled

  In such cases, a production order is used to fulfill the need for material requirement.

Notes for SAP Back-End System Administration

Prerequisites

You have to use the SAP ERP system in order to use the operations contained in this process component.

Manage Production Order In

Technical Data

Entity Type	Service interface
Software Component Version	*SAP ERP enterprise services package* (as of SAP_APPL 602)
Message Direction	Inbound
Transmission Type	Synchronous

Definition

The *Manage Production Order In* service interface enables you to change and update the details of production orders.

Use

The *Manage Production Order In* service interface contains operations to change and update the values of production orders in a production line.

Integration

The *Manage Production Order In* inbound service interface belongs to the Production [Page 126] process component and the Production Order [Page 127] business object.

Structure

This inbound service interface contains the following operations:

- `ProductionOrderChangeRequestConfirmation_In` (Change Production Order [Page 128])
- `ProductionOrderUpdateRequestConfirmation_In` (Update Production Order [Page 129])

Change Production Order

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	ProductionOrderChangeRequestConfirmation_In
Namespace in ESR	http://sap.com/xi/SPPL/Global2
Software Component Version in ESR	As of SAP_APPL 602
Category	Inbound
Mode	Synchronous

Definition

The *Change Production Order* operation enables you to change an existing production order.

Use

The *Change Production Order* operation changes a production order based on the parameters sent as part of *ProductionOrderChangeRequestConfirmation_In*. The operation returns the `ProductionOrderID` and `MaterialInternalID` parameters.

Features

This inbound operation processes the following message types:

- *Production Order Change Request*

  In this message, you have to select the parameter `ProductionOrderID` and the optional parameters are as follows:

  - Production Order Header data
  - Production Order Item data
  - Production Order Sequence data
  - Production Order Operation data

- *Production Order Change Confirmationt*

  This message returns the `ProductionOrderID` and `MaterialInternalID` of the changed production order.

This service supports idempotent behavior.

Notes for SAP Back-End System Administration

Prerequisites

- You have executed the SAP ECC component PP-SFC.
- Only the existing production orders can be changed.

Processing

The Business Add-In (BAdI) PORD_SE_PRODUCTIONORDERCHGRC (*Definition to Change Production Order*) is available for this operation.

Update Production Order

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	ProductionOrderUpdateRequestConfirmation_In
Namespace in ESR	http://sap.com/xi/APP/Global2
Software Component Version in ESR	As of SAP_APPL 602
Category	Inbound
Mode	Synchronous

Definition

The *Update Production Order* operation enables you to update a production order.

Use

The *Update Production Order* operation updates a production order based on the selected parameters sent as part of *ProductionOrderUpdateRequestConfirmation_In*. The operation returns `ProductionorderID` and `MaterialInternalID` as parameters.

Features

This inbound operation processes the following message types:

- *Production Order Update Request*

  In this message, you have to select the parameter `ProductionOrderID` and the optional parameters are as follows:

  - Production Order Header data
  - Production Order Item data
  - Production Order Sequence data
  - Production Order Operation data

- *Production Order Update Confirmation*

  This message returns the `ProductionOrderID` of the updated production order.

This service supports idempotent behavior.

Notes for SAP Back-End System Administration

Prerequisites

You need to execute the SAP ECC component PP-SFC.

Processing

The Business Add-In (BAdI) PORD_SE_PRODUCTIONORDERUPDRC (*Definition to Update Production Order*) is available for this operation.

Sales Contract Processing

Technical Data

Entity Type	Process component
Software Component	EA-GLTRADE 602

Sales Contract Processing enables a company to manage all processes related to handling sales contracts for customers.

Trade Price Specification Contract

Technical Data

Entity Type	Business object
Software Component	EA-GLTRADE 602

An outline sales agreement that contains special pricing conditions granted by a manufacturer to a distributor. A trade price specification contract is valid for a specified period of time and should contain all products, prices and end-customers eligible to the granted contract.

Trade Price Specification Contracting In

Technical Data

Entity Type	Service interface
Software Component	EA-GLTRADE 602
Category	Inbound
Mode	Asynchronous

This inbound service interface groups operations that exchange special price agreements between a manufacturer and a distributor.

Cancel Trade Price Specification Contract

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	TradePriceSpecificationContractCancelRequest_In
Software Component Version in ESR	EA-GLTRADE 602
Category	Inbound
Mode	Asynchronous
Related Web Service Definition	Not applicable

This inbound operation requests the cancellation of a price specification contract sent by the supplier.

Maintain Trade Price Specification Contract

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	TradePriceSpecificationContractRequest_In
Software Component Version in ESR	EA-GLTRADE 602
Category	Inbound
Mode	Asynchronous
Related Web Service Definition	Not applicable

This inbound operation requests the creation of a condition contract or the modification of an existing contract.

Trade Price Specification Contracting Out

Technical Data

Entity Type	Service interface
Software Component	EA-GLTRADE 602

Category	Outbound
Mode	Asynchronous

This outbound service interface contains an operation that exchanges special price agreements between a manufacturer and a distributor.

Confirm Trade Price Specification Contract

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	TradePriceSpecificationContractConfirmation_Out
Software Component Version in ESR	EA-GLTRADE 602
Category	Outbound
Mode	Asynchronous
Related Web Service Definition	Not applicable

This inbound operation generates a confirmation that either a TradePriceSpecificationContractRequest or a TradePriceSpecificationContractCancelRequest has been processed.

Sales Order Processing

Technical Data

Entity Type	Process component
Software Component	*SAP ERP enterprise services package* (as of ESA ECC-SE 600) SAP APPL (as of SAP_APPL 602)

Use

The process component *Sales Order Processing* enables the provision of applications that a seller can use to create, display, and change sales orders. The seller can also run trend analyses for customers and get an overview of sales orders that are blocked for delivery.

Notes for SAP Back-End System Administration

Prerequisites

If you want to use the operations contained in this process component, you must have implemented the *Sales and Distribution (SD)* application component.

Processing

If you use this process component, you should be aware of the behavior of the change and update operations:

The data transferred by the request messages completely overwrites the corresponding objects in the database of the back-end system. The back-end system expects a value for all elements in the message types structure. This also applies to elements that are designated as optional in the message structure.

This means that if elements are not filled in a request message, the back-end system interprets this information as a deletion entry and initializes the corresponding fields in the database.

For list-type elements that can contain several items, the items that are not filled are interpreted as a deletion entry. You can recognize the list-type elements by the fact that **unbounded** is entered for *maxOccurs*.

An order contains the items 10, 20, 30, 40, 50. With the request message for a change operation, item 10 is to be changed and item 30 deleted.

So that you do not delete items 20, 40, and 50 with the change operation, you must specify items 20, 40, and 50 along with items 10 and 30, even if the former have not been changed. Item 30 is deleted if you do not specify it.

Sales Order

Technical Data

Entity Type	Business object
Software Component	*SAP ERP enterprise services package* (as of ESA ECC-SE 600) SAP APPL (as of SAP_APPL 602)

Definition

An agreement between a seller and a customer concerning the sale and delivery of goods, as well as any services that are associated with these processes, on a specific date, for a specific quantity, and for a specific price.

Query Sales Order In

Technical Data

Entity Type	Service interface
Software Component	*SAP ERP enterprise services package* (as of ESA ECC-SE 600) SAP APPL (as of SAP_APPL 602)
Category	Inbound
Mode	Synchronous

Use

This inbound service interface contains the operations that the seller can use to search for sales orders and sales order items according to various criteria.

Integration

This inbound service interface belongs to the process component Sales Order Processing [Page 132] and the business object Sales Order [Page 133].

Find Sales Order Basic Data by Elements

Technical Data

Entity Type	Service operation
Technical Name in Enterprise Services Repository (ESR)	SalesOrderERPBasicDataByElementsQueryResponse_In
Namespace in ESR	http://sap.com/xi/APPL/Global2
Software Component Version in ESR	As of SAP_APPL 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	ECC_SALESORDER009QR

Use

This is an inbound operation that the seller can use to find basic data from sales orders by restricting the search according to specific criteria.

Sales orders can be filtered according to the following criteria: Document number range, purchase order number, sold-to party, sales area, sales office, sales group, material number, and creation date.

Features

The inbound operation processes the following message types:

- SalesOrderERPBasicDataByElementsQuery_sync
- SalesOrderERPBasicDataByElementsResponse_sync

No values must be supplied to execute the operations.

For performance reasons, SAP recommends that you restrict the selection. The following values are optional selection criteria:

- SelectionByID – Order number
- SelectionByBuyerPartyInternalID – Customer number
- SelectionByBuyerDocumentID – Purchase order number of customer
- SelectionBySalesOrganisationID – Sales organization
- SelectionByDistributionChannelCode – Distribution channel
- SelectionByDivisionCode – Division
- SelectionBySalesOfficeID – Sales office

- SelectionBySalesGroupID – Sales group
- SelectionByProcessingTypeCode – Order type
- SelectionByCreationUserAccountID – User who created the order
- SelectionByCreationUserAccountID – Date on which the order was created
- SelectionByItemProductInternalID – Material
- InvoicingBlockeIndicator – Billing block
- CreditBlockedIndicator – Credit block
- NotCompleteIndicator – Incompletion
- LifeCycleStatusNotComplete – Overall processing status is open

Integration

This inbound operation belongs to the inbound service interface Query Sales Order In [Page 133].

Notes for SAP Back-End System Administration

Prerequisites

For more information, see Sales Order Processing [Page 132].

Processing

You can use the Business Add-In (BAdI) SLS_APPL_SE_SALESORDER009QR for this operation.

Manage Sales Order In

Technical Data

Entity Type	Service interface
Software Component	*SAP ERP enterprise services package* (as of ESA ECC-SE 600) SAP APPL (as of SAP_APPL 602)
Category	Inbound
Mode	Synchronous

Use

This inbound service interface contains the operations that the sales clerk can use to process sales orders in sales and distribution.

Integration

This inbound service interface belongs to the process component Sales Order Processing [Page 132] and the business object Sales Order [Page 133].

Create Sales Order V1

Technical Data

Entity Type	Service operation
Technical Name in Enterprise Services Repository (ESR)	SalesOrderERPCreateRequestConfirmation_In_V1
Namespace in ESR	http://sap.com/xi/APPL/Global2
Software Component Version in ESR	As of SAP_APPL 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	ECC_SalesOrderCRTRC1

Use

This is an inbound operation that the seller can use to create a sales order for a customer.

Features

The inbound operation processes the following message types:

- SalesOrderERPCreateRequest_sync
- SalesOrderERPCreateConfirmation_sync

The following values must be supplied:

- ProcessingTypeCode – Order type in the back-end system (such as SO for standard order)
- BuyerParty-InternalID – customer number (sold-to party in the back-end system)
  - Value
- SalesOrganisationParty-InternalID – sales organization of customer in the back-end system
  - Value
- SalesTerms-DistributionChannelCode – distribution channel of customer in the back-end system
  - Value
- SalesTerms-DivisionCode – division of customer in the back-end system
  - Value
- Item-ProductSellerID – material number of product in the back-end system
- Item-TotalValues-RequestedQuantity-UnitCode – sales unit of product in the back-end system
- Item-TotalValues-RequestedQuantity – desired delivery quantity for product in the back-end system

The following values must not be used:

- BuyerID
  - schemeAgencySchemeAgencyID
  - schemeAgencyID
  - schemeID
- BuyerParty-InternalID
  - schemeAgencyID
  - schemeID
- SalesOrganisationParty-InternalID
  - schemeAgencyID
  - schemeID
- SalesTerms-DistributionChannelCode
  - listAgencyID
  - listAgencySchemeID
  - listVersionID
  - listAgencySchemeAgencyID
  - listID
- SalesTerms-DivisionCode
  - listAgencyID
  - listAgencySchemeID
  - listVersionID
  - listAgencySchemeAgencyID
  - listID

Integration

This inbound operation belongs to the inbound service interface Manage Sales Order In [Page 135].

Notes for SAP Back-End System Administration

Prerequisites

For more information, see Sales Order Processing [Page 132].

Processing

You can use the Business Add-In (BAdI) SLS_APPL_SE_SALESORDERCRTRC1 for this operation.

Read Sales Order V2

Technical Data

Entity Type	Service operation
Technical Name in Enterprise Services Repository (ESR)	SalesOrderERPByIDQueryResponse_In_V2
Namespace in ESR	http://sap.com/xi/APPL/Global2
Software Component Version in ESR	As of SAP_APPL 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	ECC_SALESORDER010QR

Use

This is an inbound operation that the seller can use to display a sales order.

Features

The inbound operation processes the following message types:

- SalesOrderERPByIDQuery_sync_V2
- SalesOrderERPByIDResponse_sync_V2

The following value must be supplied:

- SalesOrderID Number of sales order
  - Content

The following values must not be used:

- SalesOrderID
  - schemeAgencyID

The following fields must be filled:

- ID (GDT SalesOrderID) for SalesOrderERPByIDQuery_sync_V2

Integration

This inbound operation belongs to the inbound service interface Manage Sales Order In [Page 135].

Notes for SAP Back-End System Administration

Prerequisites

You are using the *Sales and Distribution (SD)* application component.

Processing

ProcessingTypeCode stands for *Sales Document Type.*

LifeCycleNotCompleteIndicator is a Boolean value. If the value is TRUE, it means that the overall processing status of the document is not "Completely processed" (**C**).

The operation only processes sales orders (VBTYP is **C** and cannot be changed).

You can use the Business Add-In (BAdI) SLS_SE_SALESORDERER010QR2 for this operation.

Change Sales Order

Technical Data

Entity Type	Service operation
Technical Name in Enterprise Services Repository (ESR)	SalesOrderERPChangeRequestConfirmation_In
Namespace in ESR	http://sap.com/xi/APPL/Global2
Software Component Version in ESR	As of SAP_APPL 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	ECC_SALESORDERERPCHGRC

Use

This is an inbound operation that the seller can use to change a sales order.

Features

The inbound operation processes the following message types:

- SalesOrderERPChangeRequest_sync
- SalesOrderERPChangeConfirmation_sync

The following value must be supplied:

- SalesOrderID Number of sales order
  - Content

The following values must not be used:

SalesOrder

- SalesOrderID

    - schemeAgencyID
- DataOriginCategoryCode
    - LIST_ID
    - LIST_VERSION_ID
    - LIST_AGENCY_ID
    - LIST_AGENCY_SCHEME_ID
    - LIST_AGENCY_SCHEME_AGENCY_ID
- BuyerDocumentID
    - SCHEME_ID
    - SCHEME_AGENCY_ID
    - SCHEME_AGENCY_SCHEME_AGENCY_ID
- StatusObject
    - UserStatusProfileCode
        - LIST_ID
        - LIST_VERSION_ID
        - LIST_AGENCY_ID
        - LIST_AGENCY_SCHEME_ID
        - LIST_AGENCY_SCHEME_AGENCY_ID
    - UserStatusCode
        - LIST_ID
        - LIST_VERSION_ID
        - LIST_AGENCY_ID
        - LIST_AGENCY_SCHEME_ID
        - LIST_AGENCY_SCHEME_AGENCY_ID
- BuyerPartyInternalID
    - SCHEME_ID
    - SCHEME_AGENCY_ID
- GoodsReceipeintPartyInternalID
    - SCHEME_ID
    - SCHEME_AGENCY_ID
- ReceivingPointPartyInternalID
    - SCHEME_ID
    - SCHEME_AGENCY_ID

- PartyInternalID
  - SCHEME_ID
  - SCHEME_AGENCY_ID
- SalesTermsReasonCode
  - LIST_ID
  - LIST_VERSION_ID
  - LIST_AGENCY_ID
  - LIST_AGENCY_SCHEME_ID
  - LIST_AGENCY_SCHEME_AGENCY_ID
- PricingTermsPriceSpecificationCustomer
  - LIST_ID
  - LIST_VERSION_ID
  - LIST_AGENCY_ID
  - LIST_AGENCY_SCHEME_ID
  - LIST_AGENCY_SCHEME_AGENCY_ID
- PricingTermsCustomerGroupCode
  - LIST_ID
  - LIST_VERSION_ID
  - LIST_AGENCY_ID
  - LIST_AGENCY_SCHEME_ID
  - LIST_AGENCY_SCHEME_AGENCY_ID
- PricingTermsSalesDistrictCode
  - LIST_ID
  - LIST_VERSION_ID
  - LIST_AGENCY_ID
  - LIST_AGENCY_SCHEME_ID
  - LIST_AGENCY_SCHEME_AGENCY_ID
- PriceComponentPriceSpecificationElement_T
  - LIST_ID
  - LIST_VERSION_ID
  - LIST_AGENCY_ID
  - LIST_AGENCY_SCHEME_ID
  - LIST_AGENCY_SCHEME_AGENCY_ID

- DeliveryTermsDeliveryPlantID
  - SCHEME_AGENCY_ID
- DeliveryTermsDeliveryBlockingReasonCode
  - LIST_ID
  - LIST_VERSION_ID
  - LIST_AGENCY_ID
  - LIST_AGENCY_SCHEME_ID
  - LIST_AGENCY_SCHEME_AGENCY_ID
- CashDiscountTermsCode
  - LIST_ID
  - LIST_VERSION_ID
  - LIST_AGENCY_ID
  - LIST_AGENCY_SCHEME_ID
  - LIST_AGENCY_SCHEME_AGENCY_ID
- TextCollectionTextTypeCode
  - LIST_ID
  - LIST_VERSION_ID
  - LIST_AGENCY_ID
  - LIST_AGENCY_SCHEME_ID
  - LIST_AGENCY_SCHEME_AGENCY_ID

SalesOrderItem

- DataOriginCategoryCode
  - LIST_ID
  - LIST_VERSION_ID
  - LIST_AGENCY_ID
  - LIST_AGENCY_SCHEME_ID
  - LIST_AGENCY_SCHEME_AGENCY_ID
- BuyerDocumentID
  - SCHEME_ID
  - SCHEME_AGENCY_ID
  - SCHEME_AGENCY_SCHEME_AGENCY_ID
- StatusObject
  - UserStatusProfileCode

        - LIST_ID
        - LIST_VERSION_ID
        - LIST_AGENCY_ID
        - LIST_AGENCY_SCHEME_ID
        - LIST_AGENCY_SCHEME_AGENCY_ID
    - UserStatusCode
        - LIST_ID
        - LIST_VERSION_ID
        - LIST_AGENCY_ID
        - LIST_AGENCY_SCHEME_ID
        - LIST_AGENCY_SCHEME_AGENCY_ID
- SalesAndServiceBusinessArea
    - DivisionCode
        - LIST_ID
        - LIST_VERSION_ID
        - LIST_AGENCY_ID
        - LIST_AGENCY_SCHEME_ID
        - LIST_AGENCY_SCHEME_AGENCY_ID
- ProductInternalID
    - SCHEME_ID
    - SCHEME_AGENCY_ID
- ShippingPointPartyInternalID
    - SCHEME_ID
    - SCHEME_AGENCY_ID
- ReceivingPointPartyInternalID
    - SCHEME_ID
    - SCHEME_AGENCY_ID
- PartyInternalID
    - SCHEME_ID
    - SCHEME_AGENCY_ID
- InventoryManagedLocationInternalID
    - SCHEME_ID
    - SCHEME_AGENCY_ID

- SalesTermsRejectionReasonCode
  - LIST_ID
  - LIST_VERSION_ID
  - LIST_AGENCY_ID
  - LIST_AGENCY_SCHEME_ID
  - LIST_AGENCY_SCHEME_AGENCY_ID
- PricingTermsPriceSpecificationCustomer
  - LIST_ID
  - LIST_VERSION_ID
  - LIST_AGENCY_ID
  - LIST_AGENCY_SCHEME_ID
  - LIST_AGENCY_SCHEME_AGENCY_ID
- PricingTermsCustomerGroupCode
  - LIST_ID
  - LIST_VERSION_ID
  - LIST_AGENCY_ID
  - LIST_AGENCY_SCHEME_ID
  - LIST_AGENCY_SCHEME_AGENCY_ID
- PricingTermsSalesDistrictCode
  - LIST_ID
  - LIST_VERSION_ID
  - LIST_AGENCY_ID
  - LIST_AGENCY_SCHEME_ID
  - LIST_AGENCY_SCHEME_AGENCY_ID
- PriceComponentPriceSpecificationElement_T
  - LIST_ID
  - LIST_VERSION_ID
  - LIST_AGENCY_ID
  - LIST_AGENCY_SCHEME_ID
  - LIST_AGENCY_SCHEME_AGENCY_ID
- DeliveryTermsDeliveryPlantID
  - SCHEME_AGENCY_ID
- DeliveryTermsDeliveryBlockingReasonCode

    - LIST_ID
    - LIST_VERSION_ID
    - LIST_AGENCY_ID
    - LIST_AGENCY_SCHEME_ID
    - LIST_AGENCY_SCHEME_AGENCY_ID
- CashDiscountTermsCode
    - LIST_ID
    - LIST_VERSION_ID
    - LIST_AGENCY_ID
    - LIST_AGENCY_SCHEME_ID
    - LIST_AGENCY_SCHEME_AGENCY_ID
- TextCollectionTextTypeCode
    - LIST_ID
    - LIST_VERSION_ID
    - LIST_AGENCY_ID
    - LIST_AGENCY_SCHEME_ID
    - LIST_AGENCY_SCHEME_AGENCY_ID
- ScheduleLineDeliveryBlockingReason
    - LIST_ID
    - LIST_VERSION_ID
    - LIST_AGENCY_ID
    - LIST_AGENCY_SCHEME_ID
    - LIST_AGENCY_SCHEME_AGENCY_ID

Integration

This inbound operation belongs to the inbound service interface Manage Sales Order In [Page 135].

Notes for SAP Back-End System Administration

Prerequisites

The order to be changed must exist in the back-end system.

If you use this process component, you should be aware of the behavior of the change and update operations:

The data transferred by the request messages completely overwrites the corresponding objects in the database of the back-end system. The back-end system expects a value for all elements in the message types structure. This also applies to elements that are designated as optional in the message structure.

For more information, see Sales Order Processing [Page 132].

Processing

You can use the Business Add-In (BAdI) SLS_APPL_SE_SALESORDERERPCHGRC for this operation.

Change Sales Order Item

Technical Data

Entity Type	Service operation
Technical Name in Enterprise Services Repository (ESR)	SalesOrderERPItemChangeRequestConfirmation_In
Namespace in ESR	http://sap.com/xi/APPL/Global2
Software Component Version in ESR	As of SAP_APPL 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	ECC_SALESORDERITEMCHGRC

Use

This is an inbound operation that the seller can use to change a sales item in a sales order.

Features

The inbound operation processes the following message types:

- SalesOrderERPItemChangeRequest_sync
- SalesOrderERPItemChangeConfirmation_sync

The following values must be supplied:

- SalesOrderID - Number of sales order
  - Content
- SalesOrderItemID

The following values must not be used:

SalesOrderID

- schemeAgencyID

SalesOrderItem

- DataOriginCategoryCode
  - LIST_ID
  - LIST_VERSION_ID
  - LIST_AGENCY_ID
  - LIST_AGENCY_SCHEME_ID

    - LIST_AGENCY_SCHEME_AGENCY_ID
- BuyerDocumentID
    - SCHEME_ID
    - SCHEME_AGENCY_ID
    - SCHEME_AGENCY_SCHEME_AGENCY_ID
- StatusObject
    - UserStatusProfileCode
        - LIST_ID
        - LIST_VERSION_ID
        - LIST_AGENCY_ID
        - LIST_AGENCY_SCHEME_ID
        - LIST_AGENCY_SCHEME_AGENCY_ID
    - UserStatusCode
        - LIST_ID
        - LIST_VERSION_ID
        - LIST_AGENCY_ID
        - LIST_AGENCY_SCHEME_ID
        - LIST_AGENCY_SCHEME_AGENCY_ID
- SalesAndServiceBusinessArea
    - DivisionCode
        - LIST_ID
        - LIST_VERSION_ID
        - LIST_AGENCY_ID
        - LIST_AGENCY_SCHEME_ID
        - LIST_AGENCY_SCHEME_AGENCY_ID
- ProductInternalID
    - SCHEME_ID
    - SCHEME_AGENCY_ID
- ShippingPointPartyInternalID
    - SCHEME_ID
    - SCHEME_AGENCY_ID
- ReceivingPointPartyInternalID
    - SCHEME_ID

    - SCHEME_AGENCY_ID
- PartyInternalID
    - SCHEME_ID
    - SCHEME_AGENCY_ID
- InventoryManagedLocationInternalID
    - SCHEME_ID
    - SCHEME_AGENCY_ID
- SalesTermsRejectionReasonCode
    - LIST_ID
    - LIST_VERSION_ID
    - LIST_AGENCY_ID
    - LIST_AGENCY_SCHEME_ID
    - LIST_AGENCY_SCHEME_AGENCY_ID
- PricingTermsPriceSpecificationCustomer
    - LIST_ID
    - LIST_VERSION_ID
    - LIST_AGENCY_ID
    - LIST_AGENCY_SCHEME_ID
    - LIST_AGENCY_SCHEME_AGENCY_ID
- PricingTermsCustomerGroupCode
    - LIST_ID
    - LIST_VERSION_ID
    - LIST_AGENCY_ID
    - LIST_AGENCY_SCHEME_ID
    - LIST_AGENCY_SCHEME_AGENCY_ID
- PricingTermsSalesDistrictCode
    - LIST_ID
    - LIST_VERSION_ID
    - LIST_AGENCY_ID
    - LIST_AGENCY_SCHEME_ID
    - LIST_AGENCY_SCHEME_AGENCY_ID
- PriceComponentPriceSpecificationElement_T
    - LIST_ID

    - LIST_VERSION_ID
    - LIST_AGENCY_ID
    - LIST_AGENCY_SCHEME_ID
    - LIST_AGENCY_SCHEME_AGENCY_ID
- DeliveryTermsDeliveryPlantID
    - SCHEME_AGENCY_ID
- DeliveryTermsDeliveryBlockingReasonCode
    - LIST_ID
    - LIST_VERSION_ID
    - LIST_AGENCY_ID
    - LIST_AGENCY_SCHEME_ID
    - LIST_AGENCY_SCHEME_AGENCY_ID
- CashDiscountTermsCode
    - LIST_ID
    - LIST_VERSION_ID
    - LIST_AGENCY_ID
    - LIST_AGENCY_SCHEME_ID
    - LIST_AGENCY_SCHEME_AGENCY_ID
- TextCollectionTextTypeCode
    - LIST_ID
    - LIST_VERSION_ID
    - LIST_AGENCY_ID
    - LIST_AGENCY_SCHEME_ID
    - LIST_AGENCY_SCHEME_AGENCY_ID
- ScheduleLineDeliveryBlockingReason
    - LIST_ID
    - LIST_VERSION_ID
    - LIST_AGENCY_ID
    - LIST_AGENCY_SCHEME_ID
    - LIST_AGENCY_SCHEME_AGENCY_ID

Integration

This inbound operation belongs to the inbound service interface Manage Sales Order In [Page 135].

Notes for SAP Back-End System Administration

Prerequisites

The order to be changed must exist in the back-end system.

If you use this process component, you should be aware of the behavior of the change and update operations:

The data transferred by the request messages completely overwrites the corresponding objects in the database of the back-end system. The back-end system expects a value for all elements in the message types structure. This also applies to elements that are designated as optional in the message structure.

For more information, see Sales Order Processing [Page 132].

Processing

You can use the Business Add-In (BAdI) SLS_APPL_SE_SOITEMCHGRC for this operation.

Enterprise Services for SAP ERP Financials

CashPointProcessing

Technical Data

Entity Type	Process component
Software Component	FI-CA (as of FI-CA 6.02)

Definition

Reflects payments by your customers at internal and external payment collectors.

Use

Using the *CashPointProcessing* process component, you can integrate external cash desk applications with open item management. You can use the following functions:

- Transfer of payment information from external cash desks
- Display of information about open items at external cash desks
- Reconciliation between external cash desks and open item management

Notes for SAP Back-End System Administration

Prerequisites

In order to use the operations in the process component, you must use one of the following industry components (that use *Contract Accounts Receivable and Payable* (FI-CA)):

- *Utilities Industry*
- *Telecommunications*
- *Public Sector Contract Accounts Receivable and Payable*

- *Media*
- *Insurance (Collections/Disbursements)*
- *Contract Accounts Receivable and Payable*

More information

For more information about the features of the operations of the *CashPointProcessing* process component, see the documentation for FI-CA under *Business Transactions* → *Payments* → *Processing Incoming and Outgoing Payments* → Processing Payments at External Cash Desks [Extern].

CashPointPayment

Technical Data

Entity Type	Business object
Software Component	FI-CA (as of FI-CA 6.02)

Definition

Transfer of payment information for a business partner from *CashPointProcessing* to open item management.

Use

This business object enables the transfer of payments at external cash desks.

The operations of this business object offer the following functions:

- Transfer of payments at external cash desks
- Reversal of payments at external cash desks

Notes for SAP Back-End System Administration

Prerequisites

You are using software component FI-CA (as of version FI-CA 6.02).

You have configured FI-CA for the connection of external cash desk services (see Implementation Guide for FI-CA → *Business Transactions* → *Payments* → *Processing of Incoming/Outgoing Payments* → *External Cash Desk Services*).

Processing

You can use the operations of this business object to forward information about payments at external cash desks to an open item management for posting of the payment.

Cash Point Payment Notifying In

Technical Data

Entity Type	Service interface
Software Component	FI-CA (as of FI-CA 6.02)

Category	Inbound
Mode	Synchronous

Definition

Transfer of payment information for a business partner from *CashPointProcessing* to open item management.

Use

The service interface enables the transfer of payments at external cash desks to Contract Accounts Receivable and Payable (FI-CA).

Create Cash Point Payment

Technical Data

Entity Type	Service operation (corresponds to the message interface)
Technical Name in Enterprise Services Repository (ESR)	CashPointPamyentCreateNotification_In
Namespace in ESR	http://sap.com/xi/FICA/Global
Software Component Version in ESR	As of FI-CA 602
Category	Inbound
Mode	Asynchronous
Related Web Service Definition	Not available

Definition

Transfer of information about payments at external payment collectors.

Use

The inbound operation *CashPointPaymentCreateNotification_In* accepts the data from external payment collectors in *Contract Accounts Receivable and Payable* (FI-CA), clears the open items, and posts the payments.

Notes for SAP Back-End System Administration

Processing

The system checks the data transferred to the SAP back end system and determines whether the data is stored in FI-CA. In FI-CA, a clerk can follow the further processing status of the payments in the *Monitor for External Cash Desk Services*.

The system processes the set of payment information transferred to the service operation as a whole. If the back end cannot save one of the records, the total set of payment information is rejected and transferred to the local XI monitor.

> If the data can be saved in the back end system, and an error occurs in the further processing (for example, during the clearing of open items), the system posts the successful payments. It displays the incorrect records individually in the *Monitor for External Cash Desk Services*. You can schedule these for a further posting as necessary. In this case, there is no error in the XI monitor.

Cancel Cash Point Payment

Technical Data

Entity Type	Service operation (corresponds to the message interface)
Technical Name in Enterprise Services Repository (ESR)	CashPointPamyentReverseNotification_In
Namespace in ESR	http://sap.com/xi/FICA/Global
Software Component Version in ESR	As of FI-CA 602
Category	Inbound
Mode	Asynchronous
Related Web Service Definition	Not available

Definition

Reversal of a payment to an external payment collector.

Use

The inbound operation *CashPointPaymentReverseNotification_In* accepts the data of a payment that is to be reversed at an external payment collector in *Contract Accounts Receivable and Payable* (FI-CA) and executes a reversal.

Notes for SAP Back-End System Administration

Processing

The system checks the data transferred to the SAP back end system and determines whether the data is stored in FI-CA. In FI-CA, a clerk can follow the further processing status of the reversal in the *Monitor for External Cash Desk Services*.

The system processes the set of reversal information transferred to the service operation as a whole. If the back end cannot save one of the records in the back end system, the total set of reversal information is rejected and transferred to the local XI monitor.

> If the data can be saved in the back end system, and an error occurs in the further processing (for example, during the reversal of the original payment), the system posts the successful reversal. It displays the incorrect records individually in the *Monitor for External Cash Desk Services*. You can schedule a repeat of the reversal for these records if necessary. There is no error in the XI monitor.

CashPointOpenItemSummary

Technical Data

Entity Type	Business object
Software Component	FI-CA (as of FI-CA 6.02)

Definition

CashPointOpenItemSummary is an aggregated business object and contains the total of *CashPointOpenItems* for a business partner.

CashPointOpenItem reflects an open item of *Contract Accounts Receivable and Payable* (FI-CA) that can be paid at an external payment collector.

Use

This business object enables you to display open items at external cash desks.

The operations of this business object enable you to read open items.

Notes for SAP Back-End System Administration

Prerequisites

You are using software component FI-CA (as of version FI-CA 6.02).

You have configured FI-CA for the connection of external cash desk services (see Implementation Guide for FI-CA → *Business Transactions* → *Payments* → *Processing of Incoming/Outgoing Payments* → *External Cash Desk Services*).

Query Cash Point Open Item Summary In

Technical Data

Entity Type	Service interface
Software Component	FI-CA (as of FI-CA 6.02)
Category	Inbound
Mode	Synchronous

Definition

Query of open items from Contract Accounts Receivable and Payable (FI-CA) for payment at external cash desks.

Use

This service interface is called by external cash desks to read and display open items from FI-CA.

Find Cash Point Open Item Summary by Elements

Technical Data

Entity Type	Service operation (corresponds to the message interface)
Technical Name in Enterprise Services Repository (ESR)	CashPointOpenItemSummaryByElementsQueryResponse_In
Namespace in ESR	http://sap.com/xi/FICA/Global
Software Component Version in ESR	As of FI-CA 602

Category	Inbound
Mode	Synchronous
Related Web Service Definition	FKK_CASHPOINTOPENITEMGETQUERY

Definition

Reading of open items for display at an external cash desk (CashPoint)

Use

The inbound operation *CashPointOpenItemSummaryByElementsQueryResponse_In* reads the open items for a business partner and returns them in a suitable form for external cash desks.

CashPointClosingDocument

Technical Data

Entity Type	Business object
Software Component	FI-CA (as of FI-CA 6.02)

Definition

Transfer of messages for day-end closing of an external cash desk (CashPoint).

Use

This business object enables the transfer of messages for day-end closing of external cash desks to *Contract Accounts Receivable and Payable* (FI-CA) and is used to reconcile the payment notification with the payments actually processed.

The operations of this business object enable you to transfer day-end closing reports from external cash desks.

Notes for SAP Back-End System Administration

Prerequisites

You are using software component FI-CA (as of version FI-CA 6.02).

You have configured FI-CA for the connection of external cash desk services (see Implementation Guide for FI-CA → *Business Transactions* → *Payments* → *Processing of Incoming/Outgoing Payments* → *External Cash Desk Services*).

Cash Point Document Notifying In

Technical Data

Entity Type	Service interface
Software Component	FI-CA (as of FI-CA 6.02)
Category	Inbound
Mode	Synchronous

Definition

Transfer of messages for day-end closing from an external cash desk (CashPoint).

Use

The service interface enables the transfer of day-end closing messages from external cash desks to Contract Accounts Receivable and Payable (FI-CA).

Create Closing Document

Technical Data

Entity Type	Service operation (corresponds to the message interface)
Technical Name in Enterprise Services Repository (ESR)	CashPointClosingDocumentNotification_In
Namespace in ESR	http://sap.com/xi/FICA/Global
Software Component Version in ESR	As of FI-CA 602
Category	Inbound
Mode	Asynchronous
Related Web Service Definition	Not available

Definition

Transfer of messages for day-end closing of external cash desks to *Contract Accounts Receivable and Payable* (FI-CA).

Use

The inbound operation *CashPointClosingDocumentNotification_In* accepts the data from external payment collectors in *Contract Accounts Receivable and Payable* (FI-CA) and reconciles it with the corresponding payments.

Notes for SAP Back-End System Administration

Processing

The system checks the data transferred to the SAP back end system and determines whether the data is stored in FI-CA. There a clerk can follow the further processing status of the messages for day-end closing in the *Monitor for External Cash Desk Services*.

The system processes the set of day-end closing information transferred to the service operation as a whole. If the back end cannot save one of the records, the total set of information for day-end closing is rejected and transferred to the local XI monitor.

> If the data can be saved in the back end system, and an error occurs in the further processing (for example, during clearing of the payments), the system posts the successful day-end closing messages. It displays the incorrect records individually in the *Monitor for External Cash Desk Services*.

Contract Accounts Receivables Payables

Processing

Technical Data

Entity Type	Process component
Software Component	FI-CA (as of FI-CA 6.02)

Definition

Subledger accounting for processing large volumes of documents with the typical functions of accounts receivable accounting.

Use

The *Contracts Accounts Receivables Payables Processing* process component enables you to create, export, and process contract accounts and to create cash security deposits. With the Enterprise Bill Presentment and Payment services (EBPP), you can select the items of a business partner and release them for payment (for example, bank collection or credit card payment) completely or with a partial amount. A payment specification is created for the selected items, and the payment run makes the payment.

Notes for SAP Back-End System Administration

Prerequisites

To use the operations contained in the process component, you must configure *Contract Accounts Receivable and Payable* (FI-CA). For more information about the configuration of FI-CA, see the Implementation Guide and documentation [Extern] of FI-CA.

Since subfunctions of the process component are based on existing functions of *SAP Biller Direct*, you should also consider the section "Configuration of Contract Accounts Receivable and Payable" under "Business Customizing" in the Configuration Guide of SAP Biller Direct for FI-CA [Extern].

Processing

The Contract Accounts Receivables Payables Processing process component enables communication with invoicing in the industry components *Utilities* (IS-U-IN) and *Telecommunication*.

More information

For more information about the features of the operations, see the application documentation of *SAP Biller Direct* and FI-CA.

ContractAccount

Technical Data

Entity Type	Business object
Software Component	FI-CA (as of FI-CA 6.02)

Definition

Structure element of *ContractAccountsReceivableandPayableProcessing* for entering and reporting all receivables and payables of one or more companies to one or more business partners. Contains guidelines and agreements with one or more business partners with regard to payments, dunning, interest calculation, and tax calculation for receivables and payables.

Use

This business object enables you to create and manage contract account master data. In *Contract Accounts Receivable and Payable* (FI-CA), each business partner-relevant posting is assigned to one business partner and to one contract account.

The contract account master record stores business partner-specific procedures that are used for posting and processing the line items of a given contract account. This includes specifications for payments and dunning, for example.

This business object enables you to query the items of a contract account.

Notes for SAP Back-End System Administration

Prerequisites

You have executed the back end configuration of SAP Biller Direct for FI-CA (see the Implementation Guide for FI-CA under *Integration* → *Financial Supply Chain Management* → *Biller Direct* → *Basic Setting for Customer Portal in Internet*).

You have maintained all necessary authorizations (in particular, for the authorization object F_KK_EBPP).

Processing

You can use the operations of this business object to query items of one or more contract accounts of a business partner.

Query Contract Account In

Technical Data

Entity type	Service interface
Software component	*mySAP ERP enterprise services package* (as of ESA ECC-SE 600)
Category	Inbound
Mode	Synchronous

Definition

Request to contract account for provision of contract account data.

Usage

Query Contract Account In groups together operations that read information that is relevant for the contract account from the system.

Structure

The inbound service interface *Query Contract Account In* contains the inbound operation `ContractAccountAccountBalanceByCustomerQueryResponse_In` (*Find Contract Account Balance*).

Find Contract Account Balance

Technical Data

Entity Type	Service operation (corresponds to the message interface)
Technical Name in Enterprise Services Repository (ESR)	ContractAccountERP ReceivablesPayablesRegisterItemByIDQueryResponse_In
Namespace in ESR	http://sap.com/xi/FICA/Global
Software Component Version in ESR	As of FI-CA 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	ECC_CONTACC_ITEM_BYID_QR

Definition

Query of items in a contract account of a business partner with or without restriction of the selection by earliest clearing date, latest due date, and/or clearing status.

Use

The inbound operation *ContractAccountERPReceivablesPayablesRegisterItemByIDQueryResponse_In* returns the items of a contract account of a business partner in accordance with the selection criteria.

In contrast to the service ECC_CONTACC_SP_ITEM_GRP_BYBPQR, the items are not grouped.

You can determine the following as selection criteria; the business partner to be considered, the clearing status, and the earliest clearing date and the latest due date that a subitem of a document may have in order to be returned in the selection.

Restrictions of the Services

If you specify an external contract account number, the internal contract account number must be determined by the definition of event 6600.

Since this service is based on the functions of *SAP Biller Direct* (subject to licence), the use of this service is also subject to the same licence conditions. See SAP Note 1063631 (collective note) that provides information about corrections and frequently asked questions with regard to the service.

Notes for SAP Back-End System Administration

Prerequisites

You are using software component FI-CA (as of version FI-CA 6.02).

You have executed the back end configuration of *SAP Biller Direct* in FI-CA (see the Implementation Guide for FI-CA under *Integration* → *Financial Supply Chain Management* → *Biller Direct* → *Basic Setting for Customer Portal in Internet*).

The user has sufficient authorizations, in particular with regard to the authorization object F_KK_EBPP.

Find Contract Account Split Item Group by Business Partner ID

Technical Data

Entity Type	Service operation (corresponds to the message interface)
Technical Name in Enterprise Services Repository (ESR)	ContractAccountERPSplitItemGroupByBusinessParnterIDQueryResponse_In
Namespace in ESR	http://sap.com/xi/FICA/Global
Software Component Version in ESR	As of FI-CA 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	ECC_CONTACC_SP_ITEM_GRP_BYBPQR

Definition

Query and summarization of items of one or more contract accounts of a business partner with or without restriction of the selection by earliest clearing date, latest due date, and/or clearing status.

Use

The inbound operation *ContractAccountERPSplitItemGroupByBusinessParnterIDQueryResponse_In* returns the items of one or more contract accounts of a business partner in accordance with the selection criteria.

The items are grouped with reference to the original invoice.

You can use the following as selection criteria; the contract accounts to be considered for the business partner to be considered, the clearing status, and the earliest clearing date and the latest due date that a subitem of a document may have in order to be returned in the selection.

Restrictions of the Service

In the rare cases in which invoices are partially paid by payments in other currencies, errors can occur when you display the partial amounts of the invoices.

Since this service is based on the functions of *SAP Biller Direct* (subject to licence), the use of this service is also subject to the same licence conditions.

See SAP Note 1063500 (collective note) that provides information about corrections and frequently asked questions with regard to the service.

Notes for SAP Back-End System Administration

Prerequisites

You are using software component FI-CA (as of version FI-CA 6.02).

You have executed the back end configuration of *SAP Biller Direct* in FI-CA (see the Implementation Guide for FI-CA under *Integration* → *Financial Supply Chain Management* → *Biller Direct* → *Basic Setting for Customer Portal in Internet*).

The user has sufficient authorizations, in particular with regard to the authorization object F_KK_EBPP.

PaymentDirective

Technical Data

Entity Type	Business object
Software Component	FI-CA (as of FI-CA 6.02)

Definition

Used to define the following parameters for the planned payment of a selected group of receivables and payables of one or more business partners:

- Payment date
- Amount
- Payment method
- Bank details (for bank collection and transfer)

The payment run uses *PaymentDirective* for the automatic processing of incoming and outgoing payments.

In the processing of payment lots, *PaymentDirective* identifies an item group with payment advice note.

Use

This business object enables the specification of open items for payment. The payment date, payment method, and cash discount are saved. The payment run pays the payment specification.

Notes for SAP Back-End System Administration

Prerequisites

You have executed the back end configuration of SAP Biller Direct for FI-CA (see the Implementation Guide for FI-CA under *Integration* → *Financial Supply Chain Management* → *Biller Direct* → *Basic Setting for Customer Portal in Internet*).

You have maintained all necessary authorizations in particular, for the authorization object F_KK_EBPP.

Processing

With the operations of this business object you can define items for payment. You can also select payment specifications.

Query PaymentDirective In

Technical Data

Entity Type	Service interface
Software Component	FI-CA (as of FI-CA 6.02)
Category	Inbound

Mode	Synchronous

Definition

Query to the payment specification for provision of data for a contract account.

Use

Query PaymentDirective In groups operations that read information for payment specifications from the system.

Structure

The inbound service interface *Query PaymentDirective In* contains the inbound operations *Find Payment Directive by Business Partner ID* and *Find Payment Directive by Receivables Payables Register Item Split Item ID*.

Find Payment Directive by Business Partner ID

Technical Data

Entity Type	Service operation (corresponds to the message interface)
Technical Name in Enterprise Services Repository (ESR)	PaymentDirectiveERPByBusinessPartnerIDQueryResponse_In
Namespace in ESR	http://sap.com/xi/FICA/Global
Software Component Version in ESR	As of FI-CA 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	ECC_PAYMENT_DIR_BY_BP_QR

Definition

Query of payment specifications of a business partner

Use

The inbound operation *PaymentDirectiveERPByBusinessPartnerIDQueryResponse_In* returns all payment specifications of a business partner.

Using selection criteria, you can restrict the contract accounts and company codes to be considered. You can also filter the results according to the status of the payment specifications.

Restrictions of the Services

See SAP Note 1063634 (collective note) that provides information about corrections and frequently asked questions with regard to the service.

Notes for SAP Back-End System Administration

Prerequisites

You are using software component FI-CA (as of version FI-CA 6.02).

Find Payment Directive by Receivables Payables Register Item Split Item ID

Technical Data

Entity Type	Service operation (corresponds to the message interface)
Technical Name in Enterprise Services Repository (ESR)	PaymentDirectiveERPByReceivablesPayablesRegisterItem SplitItemIDQueryResponse_In
Namespace in ESR	http://sap.com/xi/FICA/Global
Software Component Version in ESR	As of FI-CA 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	ECC_PAYMENT_DIR_BY_SPLIT_ID_QR

Definition

Query of the payment specification that a specific subitem pays or has paid.

Use

The inbound operation *PaymentDirectiveERPByReceivablesPayablesRegisterItemSplitItemIDQueryResponse_In* returns the payment specification that has paid or is to pay the subitem specified.

Restrictions of the Services

See SAP Note 1063635 (collective note) that provides information about corrections and frequently asked questions with regard to the service.

Notes for SAP Back-End System Administration

Prerequisites

You are using software component FI-CA (as of version FI-CA 6.02).

Manage PaymentDirective In

Technical Data

Entity Type	Service interface
Software Component	FI-CA (as of FI-CA 6.02)
Category	Inbound
Mode	Synchronous

Definition

The inbound service interface *Manage PaymentDirective In* contains operations to manage (for example, create) *Payment Directives*.

Use

For example, the service interface contains the service operation *Create PaymentDirective* that you can use to create a payment specification.

Structure

The service interface contains the service operation *Create PaymentDirective*.

Create Payment Directive

Technical Data

Entity Type	Service operation (corresponds to the message interface)
Technical Name in Enterprise Services Repository (ESR)	PaymentDirectiveERPCreateRequestConfirmation_In
Namespace in ESR	http://sap.com/xi/FICA/Global
Software Component Version in ESR	As of FI-CA 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	ECC_PAYMENT_DIR_CREATE_RC

Definition

Creation of a payment specification

Use

The inbound operation *PaymentDirectiveERPCreateRequestConfirmation_In* creates a payment specification and assigns items to it that are to be paid by the payment specification.

You can distribute the payment amount to subitems by specifying a payment amount for each subitem.

Alternatively, you can group several subitems and specify the payment amount per group. The internal clearing control distributes the amount to the subitems belonging to the group.

You can also specify a payment amount for all subitems at header level of the payment specification. The internal clearing control then distributes the amount to all subitems.

To ensure that the status of the items has not changed since they were called (for example, by the ECC_CONTACC_SP_ITEM_GRP_BYBPQR service), when they are called a *ChangeStateID* (hash value) must be transferred. This corresponds to the *ChangeStateID* that is returned by the services ECC_CONTACC_SP_ITEM_GRP_BYBPQR and ECC_CONTACC_ITEM_BYID_QR.

Event 1249 decides whether the execution date of the payment specification set by the caller is suitable as payment date for all items. The tolerance for late or early payment can be adjusted in the implementation of the event.
If the times are too far apart, the request is rejected.

Restrictions of the Services

- The service has the *ExternallyCreatedIndicator* field that defines whether the payment specification was created "externally" by the business partner or "internally" by the

clerk.
The current implementation in Electronic Bill Presentment and Payment (EBPP) only supports the external creation of payment specifications.

- If the service is to return the payment specification created as confirmation, the ECC_PAYMENT_DIR_BY_SPLIT_ID_QR service must be called.
- You can only assign items of the same business partner and contract account to a payment specification.
- The payment currency and original currency of the item must be the same.
- If no paying company code is transferred to the service, event 1249 must be implemented.
- As described under "Use" above, payment amounts of the payment specification may be specified at the level of the subitems, a group of subitems, or the header. You cannot specify the amount for selected items at the level of the subitems, and for others, at group level.
- Since this service is based on the functions of *SAP Biller Direct* (subject to licence), the use of this service is also subject to the same licence conditions.
  See SAP Note 1063632 (collective note) that provides information about corrections and frequently asked questions with regard to the service.

Notes for SAP Back-End System Administration

Prerequisites

You are using software component FI-CA (as of version FI-CA 6.02).

You have executed the back end configuration of *SAP Biller Direct* in FI-CA (see the Implementation Guide for FI-CA under *Integration* → *Financial Supply Chain Management* → *Biller Direct* → *Basic Setting for Customer Portal in Internet*).

The user has sufficient authorizations, in particular with regard to the authorization object F_KK_EBPP.

Due Item Processing

Technical Data

Entity Type	Process component
Software Component	*SAP ERP enterprise services package* (as of ESA ECC SE 600) SAP_APPL (as of SAP_APPL 602)

Definition

The process component *Due Item Processing* is used for managing payables and receivables from goods and services, and also the resulting sales tax and withholding tax.

Use

Using the process component *Due Item Processing* you can manage, clear, or dun payables and receivables from goods, services or taxes (open items), and also initiate payment processes for them. Moreover, you meet the requirements for creating reports, such as tax returns.

With Enterprise Bill Presentment and Payment Services (EBPP Services), you can select the items of a customer, and release for payment (for example by bank collection or credit card payment) either in whole or in part. Following the release, these items are taken into account by the payment program.

Notes for SAP Back-End System Administration

Requirements

To be able to use the operations contained in the process component, you have to configure *Accounts Receivable Accounting* (see the Implementation Guide *Financial Accounting → Accounts Receivable and Accounts Payable*).

As the release of the items is based on the existing functions of the application *SAP Biller Direct*, you should also make the settings detailed in the section 'Business Customizing' under 'Configuration of Accounts Receivable Accounting' in the Configuration Guide for SAP Biller Direct for Accounts Receivable Accounting [Extern]. In particular, you need to include the settings for creating a payment method for Biller Direct, and for the assignment of a document type for transfers, that are required for partial payments.

Further Information

For additional information on the functions of these operations, see the application documentation for *SAP Biller Direct* and *Accounts Receivable Accounting* (FI-AR).

TradeReceivablesPayablesAccount

Technical Data

Entity Type	Business Object
Software Component	*SAP ERP enterprise services package* (as of ESA ECC SE 600) SAP_APPL (as of SAP_APPL 602)

Definition

Grouping element of the *Due Item Processing* process component for creating and reporting of all receivables or payables from goods and services of a company with respect to a business partner. The business object contains guidelines for and agreements with a business partner regarding the payment and dunning of receivables and payables.

Use

This business object enables the querying of items of a customer in one or more company codes, and also their release for payment.

For a *TradeReceivablesPayablesAccountEnterprise Services for SAP ERP Enhancement Packages*, the effects of business transactions on the inventory of receivables and payables from goods and services are continuously recorded in the business object *TradeReceivablesPayables*.

The business object does not itself contain individual flows or totals.

Notes for SAP Back-End System Administration

Requirements

You have configured the back end for *SAP Biller Direct* for FI-AR (see the Implementation Guide under *Financial Accounting → Accounts Receivable and Accounts Payable Accounting*).

Processing

Using the operations of this business object, you can query the items of a customer and release them for payment.

Query Trade Receivables Payables Account In

Technical Data

Entity Type	Service interface
Software Component	*SAP ERP enterprise services package* (as of ESA ECC SE 600) SAP_APPL (as of SAP_APPL 602)
Category	Inbound
Mode	Synchronous

Definition

Query of open and paid items of a customer.

Use

The inbound service interface *Query Trade Receivables Payables Account In* encompasses operations that receive inquiries from the bill recipient regarding his open and paid receivables and payables. They then return the corresponding receivables and payables from the biller to the bill recipient.

Integration

The inbound service interface *Query Trade Receivables Payables Account In* belongs to the process component Due Item Processing [Extern] and to the business object Trade Receivables Payables Account [Extern].

Structure

The inbound service interface encompasses the following inbound operations:

- `TradeReceivablesPayablesAccountCustomerOpenItemByCustomerQueryResponse_In` (Find Open Item by Customer [Extern])
- `TradeReceivablesPayablesAccountCustomerClearedItemByCustomerQueryResponse_In` (Find Cleared Item by Customer [Extern])

Find Trade Receivables Payables Account Split Item Group by Customer ID

Technical Data

Entity Type	Service Operation (corresponds to the message interface)
Technical Name in Enterprise Services Repository (ESR)	TradeReceivablesPayablesAccountERPSplitItemGroup ByCustomerIDQueryResponse_In
Namespace in ESR	http://sap.com/xi/SAP_APPL/Global2
Software Component Version in ESR	As of SAP_APPL 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	ECC_TRPACC_SP_ITEM_GRP_BYCUSQR

Definition

Query and summarization for items of a customer in one of more company codes for bills with or without the selection being restricted by earliest clearing date, latest due date, or clearing status.

Use

The inbound operation *TradeReceivablesPayablesAccountERPSplitItemGroupByCustomerIDQueryResponse_In* returns the items of a customer in accordance with the selection criteria.

The items are grouped and summarized with reference to the original bill.

For selection criteria you can use the company codes to be read, clearing status, and also the earliest clearing date and latest due date, which must have a document in order to be included in the selection.

Constraints for the Service

In rare cases where bills have been partially paid by payments in other currencies, errors may occur when displaying the partial amounts of the bills.

As this service is based on the functions of *SAP Biller Direct* (license required), the use of this service is also subject to these license terms.

You should consult SAP Note 1061624 (which is a composite SAP Note) which gives information on corrections and frequently asked questions relating to this service.

Trade Receivables Payables Account Action In

Technical Data

Entity Type	Service interface
Software Component	SAP_APPL (as of SAP_APPL 602)
Category	Inbound
Mode	Synchronous

Definition

The inbound service interface *TradeReceivablesPayablesAccount Action In* contains operations for processing actions on a *TradeReceivablesPayablesAccount*.

Use

The service interface contains, for example, the service operation *Release Trade Receivables Payables Account Split Item Group for Payment*, which you can use to release open items for payment.

Release Trade Receivables Payables Account Split Item Group for Payment

Technical Data

Entity Type	Service Operation (corresponds to the message interface)
Technical Name in Enterprise Services Repository (ESR)	TradeReceivablesPayablesAccountERPSplitItemGroup PaymentReleaseRequestConfirmation_In
Namespace in ESR	http://sap.com/xi/SAP_APPL/Global2
Software Component Version in ESR	As of SAP_APPL 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	ECC_TRPACC_SP_GRP_PYRLEASE_RC

Definition

Release of bills for payment by bank collection or credit card.

Use

The inbound operation *TradeReceivablesPayablesAccountERPSplitItemGroupByCustomerIDQueryResponse_In* accepts the items and subitems (grouped for a bill), and releases them for the payment amount entered for payment by the payment run.

The customer has the option of entering either a payment amount for each bill or a payment amount for all of the bills together.

To make sure that no changes have been made to the items (for example, payment) since last calling up the bills, the hash value for the split items has to be entered in the *ChangeStateID* field.

The payment amount can be entered at group level. However, you cannot use any grouping for this. The grouping must correspond exactly to the one that has been returned by the service ECC_TRPACC_SP_ITEM_GRP_BYCUSQR. This corresponds to a grouping by due date and clearing status.

Constraints for the Service

Bills can only be released for payment in the currency which you use to manage them in the system.

When you call the service, you can only release bills in the same currency for payment. If you also want to release a bill that is in a different currency, you have to call the service for this bill again.

You can make either one payment of all bills together, or you can pay bills in amounts shown separately for each bill. You cannot mix the two methods.

As this service is based on the functions of *SAP Biller Direct* (license required), the use of this service is also subject to these license terms.

You should consult SAP Note 1061625 (which is a composite SAP Note) which gives information on corrections and frequently asked questions relating to this service.

Notes for SAP Back-End System Administration

Processing

The system enters a payment method in the released items. In the case of a partial payment, the back-end system also posts documents for the partial amounts.

The payment run then pays the released items.

The service uses existing business logic for the application *SAP Biller Direct* for this. Consequently you have to make settings in the back-end system of *SAP Biller Direct*. Configure at least one Biller Direct payment method there. If you also want to enable partial payments, this requires additional Customizing. See the section 'Business Customizing' under 'Configuration of Accounts Receivable Accounting' in the Configuration Guide for SAP Biller Direct for Accounts Receivable Accounting [Extern].

Expense and Reimbursement Management

Technical Data

Entity Type	Process component
Software Component	*SAP ERP enterprise services package* (as of ESA ECC-SE 600) *EA-HR* (as of EA-HR 602)

Definition

Supports companies in controlling expenses and reimbursing them to employees.

Use

The *Expense and Reimbursement Management* process component has the following main features:

- Review of expense reports of internal or external employees
- Consideration of legal requirements and corporate policies
- Payment of reimbursement amounts to employees
- Transfer of expense reports to the Financial Accounting system
- Transfer of taxable amounts to the Payroll system (for internal employees only)
- Receipt of employee-specific master data from an external personnel administration system

Expense Arrangement External Input

Technical Data

Entity Type	Business object
Software Component	*EA-HR* (as of EA-HR 602)

Definition

Provides a summary of all employee-specific input for *Expense and Reimbursement Management*.

Use

The *Expense Arrangement External Input* business object is used for transmission of employee master data between enterprise systems.

This business object is used for input from external systems only.

You can use this business object to supply your *SAP Travel Management* system with a mini-master set of personnel data from an external personnel administration system. The data enables you to settle the expenses of your employees.

Related Business Objects

Travel Planning Arrangement [Page 180]

Expense Arrangement External Input Transmission Receiving In

Technical Data

Entity Type	Service interface
Software Component	*EA-HR* (as of EA-HR 602)
Category	Inbound
Mode	Asynchronous

Definition

Provides a summary of all employee-specific input for *Expense and Reimbursement Management*.

Use

The *Expense Arrangement External Input Transmission Receiving In* inbound service interface includes an operation that you can use to supply your *SAP Travel Management* system with a mini-master set of personnel data from an external personnel administration system. In your *SAP Travel Management* system, the data is saved in the following infotypes:

- Infotype Actions (0000)
- Infotype *Organizational Assignment* (0001)
- Infotype *Personal Data* (0002)
- Infotype *Addresses* (0006), subtype *Permanent residence* (1)

- Infotype *Bank Details* (0009) with the following subtypes:
  - Subtype *Main bank* (0)
  - Subtype *Travel expenses* (02)
- Infotype *Travel Privileges* (0017)
- Infotype *Cost Distribution* (0027), subtype *Travel expenses* (02)
- Infotype *Communication* (0105) with the following subtypes:
  - Subtype *E-mail* (0010)
  - Subtype *Credit card number(s)* (0011)
  - Subtype *First telephone number at work* (0020)
  - Subtype *Cell Phone* (CELL)

Related Service Interfaces

- Expense Arrangement External Input Transmission Receiving Out [Page 174]
- Travel Planning Arrangement Out [Page 189]

Integration

The *Expense Arrangement External Input Transmission Receiving In* inbound service interface belongs to the Expense and Reimbursement Management [Page 170] process component and the Expense Arrangement External Input [Page 171] business object.

Structure

The *Expense Arrangement External Input Transmission Receiving In* inbound service interface provides the `ExpenseArrangementExternalInputTransmitRequest_In` (Maintain Expense Arrangement External Input [Page 172]) inbound operation.

Maintain Expense Arrangement External Input

Technical Data

Entity Type	Service operation
Technical Name in Enterprise Services Repository (ESR)	ExpenseArrangementExternalInput-TransmitRequest_In
Namespace in ESR	http://sap.com/xi/Travel/Global2
Software Component Version in ESR	As of EA-HR 602
Category	Inbound
Mode	Asynchronous

Definition

A request to transmit an *Expense Arrangement External Input*.

Use

The *Maintain Expense Arrangement External Input* inbound operation is used to supply your *SAP Travel Management* system with a mini-master set of personnel data from an external personnel administration system.

Features

This inbound operation processes the following message types:

- `ExpenseArrangementExternalInputTransmitRequest`
- `StandardFaultMessage`

This inbound operation processes one EmployeeID at a time. It transmits the complete *Expense Arrangement External Input* data of an employee. A delta-logic is **not** available.

An EmployeeID can have multiple WorkAgreementIDs. The SAP back-end system identifies employees and work agreements as follows:

- An **employee** is uniquely identified by an **external person ID** (corresponding to the EmployeeID).
- A **work agreement** is uniquely identified by a **personnel number** (corresponding to the WorkAgreementID).

A **non**-SAP system provides its version of EmployeeID and WorkAgreementID through EmployeeUUID and WorkAgreementUUID. The system maps these structures using the PAOCF_UUID_PERNR table.

The data replication at employee level can be distinguished as follows:

- External person ID (EmployeeID) and related personnel numbers (WorkAgreementIds) do **not** exist in the SAP back-end system:

  In this case, this inbound operation creates a new external person ID by creating a personnel number in the SAP back-end system. If the EmployeeID includes more than one WorkAgreementID, the operation creates a personnel number for each WorkAgreementID.

- External person ID (EmployeeID) exists, but a related personnel number (WorkAgreementId) does **not** exist in the SAP back-end system:

  In this case, this inbound operation creates a new personnel number in the SAP back-end system.

- External person ID (EmployeeID) and personnel number (WorkAgreementId) already exist in the SAP back-end system:

  In this case, this inbound operation only updates the changes in the SAP back-end system. Such changes might be:

  - A change in the address
  - A new bank account
  - A new event. e. g. a work agreement is terminated

- Mixed run (Concurrent Employment (CE) scenarios only):

  This scenario is used, for example, when an employee has a new work agreement and, at the same time, his or her e-mail address of an existing work agreement changes. This calls for a mixed run. This inbound operation creates a new work agreement and changes the existing work agreement.

Integration

This inbound operation belongs to the Expense Arrangement External Input Transmission Receiving In [Page 171] inbound service interface.

Notes for SAP Back-End System Administration

Prerequisites

Your XI user is an administrator in the personnel area you are dealing with. You can use transaction **SM30** to assign the *Administrator* role to a personnel area in the *Administrators* table (T526).

The SAP system assigns the personnel numbers in the range you specify (internal number assignment). You can define number ranges in Customizing for *Personnel Management* by choosing *Personnel Administration* → *Basic Settings* → *Maintain number range intervals for personnel numbers*.

If you want to process credit card details, the credit card companies must be defined in the *Names of Credit Card Companies* table (V_T706_CCOMP). You can use transaction **SM30** to assign credit card companies.

You have activated the *Travel Management* (FIN_TRAVEL_1) business function to use the associated Business Add-In (BAdI) in Customizing.

Processing

The BAdI PTRM_SE_EXPARR_CONV (*BAdI: Mapping for ExpenseArrangementExternalInputTransmitRequest*) is available for this operation.

Error Handling

Error handling is done at two processing stages:

- Incorrect incoming data

  When incorrect incoming data causes problems, the back-end system returns an error message to the sending system using the Confirm Expense Arrangement External Input Transmission [Page 175] outbound operation. The back-end system extracts the error message from the CX_PAD_SE_PROXY exception.

- Error during processing

  When an error occurs during the processing of the incoming data (infotype updates), the back-end system raises an exception to XI and terminates the processing. Before raising the exception, the back-end system performs a rollback.

Expense Arrangement External Input Transmission Receiving Out

Technical Data

Entity Type	Service interface
Software Component	*EA-HR* (as of EA-HR 602)
Category	Outbound
Mode	Asynchronous

Definition

Confirms an *Expense Arrangement External Input* transmission request.

Use

The *Expense Arrangement External Input Transmission Receiving Out* outbound service interface includes an operation that confirms a transmission request from an external personnel administration system using the Maintain Expense Arrangement External Input [Page 172] inbound operation.

Related Service Interfaces

Expense Arrangement External Input Transmission Receiving In [Page 171]

Integration

The *Expense Arrangement External Input Transmission Receiving Out* outbound service interface belongs to the Expense and Reimbursement Management [Page 170] process component and the Expense Arrangement External Input [Page 171] business object.

Structure

The *Expense Arrangement External Input Transmission Receiving Out* outbound service interface provides the `ExpenseArrangementExternalInputTransmitConfirmation_Out` (Confirm Expense Arrangement External Input Transmission [Page 175]) outbound operation.

Confirm Expense Arrangement External Input Transmission

Technical Data

Entity Type	Service operation
Technical Name in Enterprise Services Repository (ESR)	ExpenseArrangementExternalInput-TransmitConfirmation_Out
Namespace in ESR	http://sap.com/xi/Travel/Global2
Software Component Version in ESR	As of EA-HR 602
Category	Outbound
Mode	Asynchronous

Definition

Confirmation to an *Expense Arrangement External Input* transmission request.

Use

The *Confirm Expense Arrangement External Input Transmission* outbound operation is used to confirm a transmission request from an external personnel administration system using the Maintain Expense Arrangement External Input [Page 172] inbound operation.

Features

This outbound operation processes the `ExpenseArrangementExternalInputTransmitConfirmation` message type.

Integration

This outbound operation belongs to the Expense Arrangement External Input Transmission Receiving Out [Page 174] outbound service interface.

Payment Processing

Technical Data

Entity Type	Process component
Software Component	*SAP ERP enterprise services package* (as of ECC-SE 600) *SAP APPL* (as of SAP-APPL 602)

Definition

Handles all incoming and outgoing payments and represents the main data base for the liquidity status.

Use

The *Payment Processing* process component is the central component for the standard processing of payment transactions for all processes in a company or a group of companies. *Payment Processing* supports the communication with house banks and business partners that is required for processing.

Notes for SAP Backend System Administration

Processing

Take into account the following change and update operations of this process component when you use it:

The data transferred by request-messages overwrites only those objects affected in the databases of the back-end system that are also filled in the message structure. This also applies to the individual items of list-like elements.

This means that if optional elements are **not** filled in a request message, the back-end system leaves the values as they are in the affected fields in the database. The only exception to this behavior is when these values are automatically updated due to changes to other data.

Payment Order

Technical Data

Entity Type	Business object
Software Component	*SAP ERP enterprise services package* (as of ECC-SE 602) *SAP APPL* (as of SAP_APPL 602)

Definition

An order to make a payment to a business partner. A *Payment Order* can be a collective instruction that contains several separate instructions.

Use

A payment order is either executed by the company itself (for example, check printing) or by a third party (for example, a bank transfer by the house bank).

Process flow

Once a company wants to execute an incoming or outgoing payment, the company sends an order with the payment data using a `CollectivePaymentOrderRequest` message to a house bank or another agent (for example, to another company if payment is made on its behalf).

Related Business Objects

Bank Statement [Extern]

Collective Payment Order In

Technical Data

Entity Type	Service interface
Software Component	*SAP APPL* (as of SAP-APPL 602)
Category	Inbound
Mode	Asynchronous

Definition

Notifies processing status for single or multiple payment orders.

Use

The *Collective Payment Order In* inbound service interface comprises operations for tracking the status of one or multiple payment orders.

Related Service Interfaces

- Collective Payment Order Out (delivered with ECC-SE as of 602)
- Bank Statement In (delivered with ECC-SE as of 602)

Integration

The *Collective Payment Order In* inbound service interface belongs to the Payment Processing [Page 176] process component and the Payment Order [Page 176] business object.

Structure

The *Collective Payment Order In* inbound service interface provides the `CollectivePaymentOrderNotification_In` (Collective Payment Order Notification [Page 177]) inbound operation.

Collective Payment Order Notification

Technical Data

Entity Type	Service operation
Technical Name in Enterprise Service Repository (ESR)	CollectivePaymentOrderNotification_In
Namespace in ESR	http://sap.com/xi/APPL/Global2
Software Component Version in ESR	As of SAP APPL 602

Category	Inbound
Mode	Asynchronous
Related Web Service Definition	-

Definition

Transfers information on the processing status for one or multiple payment orders.

Use

The *Collective Payment Order Notification* inbound operation reports the processing status for single or multiple payment orders.

Features

This inbound operation processes the `CollectivePaymentOrderNotification` message type.

Integration

This outbound operation belongs to the Collective Payment Order In [Page 177] inbound service interface.

PaymentRegister

Technical Data

Entity Type	Business Object
Software Component	SAP_APPL (as of SAP_APPL 602)

Definition

Payment register of a company that contains the individual payments of each payment transaction (*PaymentRegisterItem*), and also the totals for each company and each *CashLocation*.

Use

This business object manages the incoming and outgoing payments in a register.

Notes for SAP Back-End System Administration

Processing

Using the operations of this business object, you can display the data for a payment from the *PaymentRegister* for a particular ID number.

Manage Payment Register In

Technical Data

Entity Type	Service Interface
Software Component	SAP_APPL (as of SAP_APPL 602)

Category	Inbound
Mode	Synchronous

Definition

The *Payment Register In* inbound service interface contains operations for processing (for example, reading) entries for a *Payment Register*.

Use

The service interface contains, for example, the *Read PaymentRegister* service operation, with which you can read payments from a payment register.

Structure

The service interface contains the *Read PaymentRegister* service operation.

Read Payment Register Item

Technical Data

Entity Type	Service Operation (corresponds to the message interface)
Technical Name in Enterprise Services Repository (ESR)	PaymentRegisterERPItemByPaymentReferenceQueryResponse_In
Namespace in ESR	http://sap.com/xi/SAP_APPL/Global2
Software Component Version in ESR	As of SAP_APPL 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	ECC_PAYREG_ITEM_BY_PAYREF_QR

Definition

Query of payments (*PaymentRegisterItems*) from a register (*PaymentRegister*) for an identification number (*PaymentReference*).

Use

The inbound operation *PaymentRegisterERPItemByPaymentReferenceQueryResponse_In* returns payments for a payment reference number.

In doing so, the system can access payment documents from *AccountsReceivables* and payment documents from *ContractAccounting*.

You can select the payments by entering a payment reference. To select payments from *AccountsReceivables*, you have to use a payment reference (*PaymentReference*) in the form **AR** + company code + fiscal year + document number of the payment document.

You have two options for reading payments in *ContractAccounting*. If the payment involves an item in the payment lot, you can enter the payment lot number with the prefix 'PL' added as the ID of the *PaymentReference*. You can choose the payment item in the payment lot as the *ItemID*.

You can also select payments by entering the prefix **PD** + the document number of the payment document from *ContractAccounting*.

Constraints for the Service

The service currently only deals with incoming payments.

Although the service is able to select documents from both contract accounts and accounts receivable accounting, a mixed selection is not possible. This means that when you call up the system you can select payments from either accounts receivable or from contract accounts. Within contract accounts you can only ever select by payment documents (prefix **PD**) or payment lot items (prefix **PL**).

You should consult SAP Note 106167 (which is a composite SAP Note) which gives information on corrections and frequently asked questions relating to this service.

Travel Management

Technical Data

Entity Type	Process component
Software Component	*EA-HR* (as of EA-HR 602)

Definition

Enables companies and their employees to request, plan, and book business trips, including travel services.

Use

The *Travel Management* process component supports the integration to *Expense and Reimbursement Management* and online booking of the following travel services through connected reservation systems:

- Flight
- Hotel
- Rental car
- Train
- Other services

Travel Planning Arrangement

Technical Data

Entity Type	Business object
Software Component	*EA-HR* (as of EA-HR 602)

Definition

An arrangement for an employee related to travel planning.

Use

The *Travel Planning Arrangement* business object is used for exchanging travel planning arrangements, including personal preferences, between the *SAP Travel Management* system and travel service providers (for example, a global reservation system or a profile database of

your travel agency). These travel planning arrangements are taken into consideration while planning or booking trips.

In a Concurrent Employment (CE) scenario, an employee (EmployeeID) can have multiple work agreements (WorkAgreementIDs). The SAP back-end system identifies employees and work agreements as follows:

- An **employee** is uniquely identified by an **external person ID** (corresponding to the EmployeeID).
- A **work agreement** is uniquely identified by a **personnel number** (corresponding to the WorkAgreementID).

Related Business Objects

Expense Arrangement External Input [Page 171]

Notes for SAP Back-End System Administration

Processing

A **non**-SAP system provides its version of EmployeeID and WorkAgreementID through EmployeeUUID and WorkAgreementUUID. The system maps these structures using the PAOCF_UUID_PERNR table.

Travel Planning Arrangement In

Technical Data

Entity Type	Service interface
Software Component	*EA-HR* (as of EA-HR 602)
Category	Inbound
Mode	Asynchronous

Definition

Provides *Travel Planning Arrangements* for *SAP Travel Management*.

Use

The *Travel Planning Arrangement In* inbound service interface combines operations that transfer certain travel planning arrangements from travel service providers to your *SAP Travel Management* system. The *SAP Travel Management* system saves the data in the following infotypes:

- Infotype *Flight Preference* (0471)
- Infotype *Hotel Preference* (0472)
- Infotype *Rental Car Preference* (0473)
- Infotype *Train Preference* (0474)
- Infotype *Customer Program* (0475)

If, up to now, your employees booked travel services using conventional channels, such as telephone or e-mail, the travel service providers usually stored the employees' travel preferences in their profile databases. Using this inbound service interface, you can use the existing data of travel service providers for an initial set-up of your employees' travel preferences in your *SAP Travel Management* system.

After the initial set-up of an employee's preferences, your *SAP Travel Management* system will be the primary master system for these preferences (the operations do **not** overwrite existing preferences).

You can use the *Travel Planning Arrangement Out* outbound service interface to update the profile databases of travel service providers.

Related Service Interfaces

Travel Planning Arrangement Out [Page 189]

Integration

The *Travel Planning Arrangement In* inbound service interface belongs to the Travel Management [Page 180] process component and the Travel Planning Arrangement [Page 180] business object.

Structure

The *Travel Planning Arrangement In* inbound service interface groups the following inbound operation together:

- `TravelPlanningArrangementFlightArrangementCreatedNotification_In` (Change Arrangement Based on Flight Arrangement Created Notification [Page 182])
- `TravelPlanningArrangementHotelArrangementCreatedNotification_In` (Change Arrangement Based on Hotel Arrangement Created Notification [Page 184])
- `TravelPlanningArrangementCarRentalArrangementCreatedNotification_In` (Change Arrangement Based on Car Rental Arrangement Created Notification [Page 185])
- `TravelPlanningArrangementTrainArrangementCreatedNotification_In` (Change Arrangement Based on Train Arrangement Created Notification [Page 186])
- `TravelPlanningArrangementCustomerLoyaltyProgrammeArrangement-CreatedNotification_In` (Change Arrangement Based on Customer Loyalty Programme Arrangement Created Notification [Page 188])

Notes for SAP Back-End System Administration

Prerequisites

You have activated the *Travel Management* (FIN_TRAVEL_1) business function to use the associated Business Add-Ins (BAdI) in Customizing.

Change Arrangement Based on Flight Arrangement Created Notification

Technical Data

Entity Type	Service operation
Technical Name in Enterprise Services Repository (ESR)	TravelPlanningArrangementFlightArrangement-CreatedNotification_In
Namespace in ESR	http://sap.com/xi/Travel/Global2
Software Component Version in ESR	As of EA-HR 602
Category	Inbound

Mode	Asynchronous

Definition

A notification to *SAP Travel Management* for the creation of flight arrangements for an employee.

Use

The *Change Arrangement Based on Flight Arrangement Created Notification* inbound operation is used to create flight arrangements for an employee. The SAP back-end system stores the details in infotype *Flight Preference* (0471).

You can use this inbound operation to transfer flight arrangements from a travel service provider (for example, a global reservation system or a profile database of your travel agency) to your *SAP Travel Management* system.

Features

This inbound operation processes the following message types:

- `TravelPlanningArrangementFlightArrangementCreatedNotification`
- `StandardFaultMessage`

If this inbound operation does **not** provide a validity timeframe for the flight arrangements, the SAP back-end system sets the following values:

- Start date = system date
- End date = 12/31/9999

This inbound operation creates exactly one set of flight arrangements for an employee.

In a Concurrent Employment (CE) scenario, this inbound operation creates the same set of flight arrangements for all work agreements of an employee.

Prerequisites

In a CE scenario, your XI user and the employees' SAP users in the SAP back-end system must be CE-enabled.

Integration

This inbound operation belongs to the Travel Planning Arrangement In [Page 181] inbound service interface.

Constraints

This inbound operation does **not** overwrite existing flight preferences in the SAP back-end system. This inbound operation can be executed successfully, only if **no** flight arrangements exist in the employee's infotype 0471.

In a CE scenario, if flight arrangements exist in only one work agreement of the employee, then this inbound operation is terminated with an exception.

Notes for SAP Back-End System Administration

Processing

The Business Add-In (BAdI) PTRM_SE_TPAFLTARR_CONV (*BAdI: Mapping for TravelPlanningArrangementFlightArrangement*) is available for this operation. For an enhanced inbound message interface, you can use the INBOUND_MAPPING method to perform the inbound mapping.

Error Handling

When an error occurs during the create operation, the back-end system raises an exception to XI and terminates processing. Before raising the exception, the back-end system performs a rollback (in case of a CE scenario, the back-end system performs a rollback for the work agreements of the employee that were already processed successfully).

Change Arrangement Based on Hotel Arrangement Created Notification

Technical Data

Entity Type	Service operation
Technical Name in Enterprise Services Repository (ESR)	TravelPlanningArrangementHotelArrangement-CreatedNotification_In
Namespace in ESR	http://sap.com/xi/Travel/Global2
Software Component Version in ESR	As of EA-HR 602
Category	Inbound
Mode	Asynchronous

Definition

A notification to *SAP Travel Management* for the creation of hotel arrangements for an employee.

Use

The *Change Arrangement Based on Hotel Arrangement Created Notification* inbound operation is used to create hotel arrangements for an employee. The SAP back-end system stores the details in infotype *Hotel Preference* (0472).

You can use this inbound operation to transfer hotel arrangements from a travel service provider (for example, a global reservation system or a profile database of your travel agency) to your *SAP Travel Management* system.

Features

This inbound operation processes the following message types:

- `TravelPlanningArrangementHotelArrangementCreatedNotification`
- `StandardFaultMessage`

If this inbound operation does **not** provide a validity timeframe for the hotel arrangements, the SAP back-end system sets the following values:

- Start date = system date
- End date = 12/31/9999

This inbound operation creates exactly one set of hotel arrangements for an employee.

In a Concurrent Employment (CE) scenario, this inbound operation creates the same set of hotel arrangements for all work agreements of an employee.

Prerequisites

In a CE scenario, your XI user and the employees' SAP users in the SAP back-end system must be CE-enabled.

Integration

This inbound operation belongs to the Travel Planning Arrangement In [Page 181] inbound service interface.

Constraints

This inbound operation does **not** overwrite existing hotel arrangements in the SAP back-end system. This inbound operation can be executed successfully, only if **no** hotel arrangements exist in the employee's infotype 0472.

In a CE scenario, if hotel arrangements exist in only one work agreement of the employee, then this inbound operation is terminated with an exception.

Notes for SAP Back-End System Administration

Processing

The Business Add-In (BAdI) PTRM_SE_TPAHOTELARR_CONV (*BAdI: Mapping for TravelPlanningArrangementHotelArrangement*) is available for this operation. For an enhanced inbound message interface, you can use the INBOUND_MAPPING method to perform the inbound mapping.

Error Handling

When an error occurs during the create operation, the back-end system raises an exception to XI and terminates processing. Before raising the exception, the back-end system performs a rollback (in case of a CE scenario, the back-end system performs a rollback for the work agreements of the employee that were already processed successfully).

Change Arrangement Based on Car Rental Arrangement Created Notification

Technical Data

Entity Type	Service operation
Technical Name in Enterprise Services Repository (ESR)	TravelPlanningArrangementCarRental-ArrangementCreatedNotification_In
Namespace in ESR	http://sap.com/xi/Travel/Global2
Software Component Version in ESR	As of EA-HR 602
Category	Inbound
Mode	Asynchronous

Definition

A notification to *SAP Travel Management* for the creation of car rental arrangements for an employee.

Use

The *Change Arrangement Based on Car Rental Arrangement Created Notification* inbound operation is used to create car rental arrangements for an employee. The SAP back-end system stores the details in infotype *Rental Car Preference* (0473).

You can use this inbound operation to transfer car rental arrangements from a travel service provider (for example, a global reservation system or a profile database of your travel agency) to your *SAP Travel Management* system.

Features

This inbound operation processes the following message types:

- `TravelPlanningArrangementCarRentalArrangementCreatedNotification`
- `StandardFaultMessage`

If this inbound operation does **not** provide a validity timeframe for the car rental arrangements, the SAP back-end system sets the following values:

- Start date = system date
- End date = 12/31/9999

This inbound operation creates exactly one set of car rental arrangements for an employee.

In a Concurrent Employment (CE) scenario, this inbound operation creates the same set of car rental arrangements for all work agreements of an employee.

Prerequisites

In a CE scenario, your XI user and the employees' SAP users in the SAP back-end system must be CE-enabled.

Integration

This inbound operation belongs to the Travel Planning Arrangement In [Page 181] inbound service interface.

Constraints

This inbound operation does **not** overwrite existing car rental arrangements in the SAP back-end system. This inbound operation can be executed successfully, only if **no** car rental arrangements exist in the employee's infotype 0473.

In a CE scenario, if car rental arrangements exist in only one work agreement of the employee, then this inbound operation is terminated with an exception.

Notes for SAP Back-End System Administration

Processing

The Business Add-In (BAdI) PTRM_SE_TPACARRNARR_CONV (*BAdI: Mapping for TravelPlanningArrangementCarRentalArrangement*) is available for this operation. For an enhanced inbound message interface, you can use the INBOUND_MAPPING method to perform the inbound mapping.

Error Handling

When an error occurs during the create operation, the back-end system raises an exception to XI and terminates processing. Before raising the exception, the back-end system performs a rollback (in case of a CE scenario, the back-end system performs a rollback for the work agreements of the employee that were already processed successfully).

Change Arrangement Based on Train Arrangement Created Notification

Technical Data

Entity Type	Service operation

Technical Name in Enterprise Services Repository (ESR)	TravelPlanningArrangementTrainArrangement-CreatedNotification_In
Namespace in ESR	http://sap.com/xi/Travel/Global2
Software Component Version in ESR	As of EA-HR 602
Category	Inbound
Mode	Asynchronous

Definition

A notification to *SAP Travel Management* for the creation of train arrangements for an employee.

Use

The *Change Arrangement Based on Train Arrangement Created Notification* inbound operation is used to create train arrangements for an employee. The SAP back-end system stores the details in infotype *Train Preference* (0474).

You can use this inbound operation to transfer train arrangements from a travel service provider (for example, a global reservation system or a profile database of your travel agency) to your *SAP Travel Management* system.

Features

This inbound operation processes the following message types:

- `TravelPlanningArrangementTrainArrangementCreatedNotification`
- `StandardFaultMessage`

If this inbound operation does **not** provide a validity timeframe for the train arrangements, the SAP back-end system sets the following values:

- Start date = system date
- End date = 12/31/9999

This inbound operation creates exactly one set of train arrangements for an employee.

In a Concurrent Employment (CE) scenario, this inbound operation creates the same set of train arrangements for all work agreements of an employee.

Prerequisites

In a CE scenario, your XI user and the employees' SAP users in the SAP back-end system must be CE-enabled.

Integration

This inbound operation belongs to the Travel Planning Arrangement In [Page 181] inbound service interface.

Constraints

This inbound operation does **not** overwrite existing train arrangements in the SAP back-end system. This inbound operation can be executed successfully, only if **no** train arrangements exist in the employee's infotype 0474.

In a CE scenario, if train arrangements exist in only one work agreement of the employee, then this inbound operation is terminated with an exception.

This inbound operation does **not** provide the identification fields for the Bahn Internet Booking Engine (BIBE) of the Deutsche Bahn Corporate Portal.

Notes for SAP Back-End System Administration

Processing

The Business Add-In (BAdI) PTRM_SE_TPATRAINARR_CONV (*BAdI: Mapping for TravelPlanningArrangementTrainArrangement*) is available for this operation. For an enhanced inbound message interface, you can use the INBOUND_MAPPING method to perform the inbound mapping.

Error Handling

When an error occurs during the create operation, the back-end system raises an exception to XI and terminates processing. Before raising the exception, the back-end system performs a rollback (in case of a CE scenario, the back-end system performs a rollback for the work agreements of the employee that were already processed successfully).

Change Arrangement Based on Customer Loyalty Programme Created Notification

Technical Data

Entity Type	Service operation
Technical Name in Enterprise Services Repository (ESR)	TravelPlanningArrangementCustomerLoyalty-ProgrammeCreatedNotification_In
Namespace in ESR	http://sap.com/xi/Travel/Global2
Software Component Version in ESR	As of EA-HR 602
Category	Inbound
Mode	Asynchronous

Definition

A notification to *SAP Travel Management* for the creation of customer loyalty programs for an employee.

Use

The *Change Arrangement Based on Customer Loyalty Programme Created Notification* inbound operation is used to create customer loyalty programs for an employee who has subscribed to these programs. The SAP back-end system stores the details in infotype *Customer Program* (0475).

You can use this inbound operation to transfer customer loyalty programs from a travel service provider (for example, a global reservation system or a profile database of your travel agency) to your *SAP Travel Management* system.

Features

This inbound operation processes the following message types:

- `TravelPlanningArrangementCustomerLoyaltyProgrammeCreatedNotification`
- `StandardFaultMessage`

If this inbound operation does **not** provide a validity timeframe for a customer loyalty program, the SAP back-end system sets the following values:

- Start date = system date

- End date = 12/31/9999

This inbound operation creates exactly one set of customer loyalty programs for an employee.

In a Concurrent Employment (CE) scenario, this inbound operation creates the same set of customer loyalty programs for all work agreements of an employee.

Prerequisites

In a CE scenario, your XI user and the employees' SAP users in the SAP back-end system must be CE-enabled.

Integration

This inbound operation belongs to the Travel Planning Arrangement In [Page 181] inbound service interface.

Constraints

This inbound operation does **not** overwrite existing customer loyalty programs in the SAP back-end system. This inbound operation can be executed successfully, only if **no** customer loyalty programs exist in the employee's infotype 0475.

In a CE scenario, if customer loyalty programs exist in only one work agreement of the employee, then this inbound operation is terminated with an exception.

Notes for SAP Back-End System Administration

Processing

The Business Add-In (BAdI) PTRM_SE_TPACLP_CONV (*BAdI: Mapping for TravelPlanningArrangementCustomerLoyaltyProgramme*) is available for this operation. For an enhanced inbound message interface, you can use the INBOUND_MAPPING method to perform the inbound mapping.

Error Handling

When an error occurs during the create operation, the back-end system raises an exception to XI and terminates processing. Before raising the exception, the back-end system performs a rollback (in case of a CE scenario, the back-end system performs a rollback for the work agreements of the employee that were already processed successfully).

Travel Planning Arrangement Out

Technical Data

Entity Type	Service interface
Software Component	*EA-HR* (as of EA-HR 602)
Category	Outbound
Mode	Asynchronous

Definition

Notifies the *Travel Planning Arrangement* related to an employee.

Use

The *Travel Planning Arrangement Out* outbound service interface includes an operation that you can use to supply external travel service providers (for example, a global reservation system or a profile database of your travel agency) with the *Travel Planning Arrangement*

data from your *SAP Travel Management* system. This operation also transfers additional employee data.

The *Travel Planning Arrangement* data is stored in the following infotypes:

- Infotype *Flight Preference* (0471)
- Infotype *Hotel Preference* (0472)
- Infotype *Rental Car Preference* (0473)
- Infotype *Train Preference* (0474)
- Infotype *Customer Program* (0475)

The additional employee data is stored in the following infotypes:

- Infotype *Organizational Assignment* (0001)
- Infotype *Personal Data* (0002)
- Infotype *Addresses* (0006), subtype *Permanent residence* (1)
- Infotype *Cost Distribution* (0027), subtype *Travel expenses* (02)
- Infotype *Communication* (0105) with the following subtypes:
  - Subtype *E-mail* (0010)
  - Subtype *Credit card number(s)* (0011)
  - Subtype *First telephone number at work* (0020)
  - Subtype *Cell Phone* (CELL)

Related Service Interfaces

- Travel Planning Arrangement In [Page 181]
- Expense Arrangement External Input Transmission Receiving In [Page 171]

Integration

The *Travel Planning Arrangement Out* outbound service interface belongs to the Travel Management [Page 180] process component and the Travel Planning Arrangement [Page 180] business object.

Structure

The *Travel Planning Arrangement Out* outbound service interface provides the `TravelPlanningArrangementEmployeeNotification_Out` (Notify of Travel Planning Arrangement Employee [Page 190]) outbound operation.

Notes for SAP Back-End System Administration

Prerequisites

You have activated the *Travel Management* (FIN_TRAVEL_1) business function to use the associated programs and Business Add-Ins (BAdI) in Customizing.

Notify of Travel Planning Arrangement Employee

Technical Data

Entity Type	Service operation
Technical Name in Enterprise Services Repository (ESR)	TravelPlanningArrangementEmployee-Notification_Out
Namespace in ESR	http://sap.com/xi/Travel/Global2
Software Component Version in ESR	As of EA-HR 602
Category	Outbound
Mode	Asynchronous

Definition

Notification about the *Travel Planning Arrangement* related to an employee.

Use

The *Notify of Travel Planning Arrangement Employee* outbound operation is used to supply external travel service providers (for example, a global reservation system or a profile database of your travel agency) with the *Travel Planning Arrangement* data from your *SAP Travel Management* system. For more information about the data, see Travel Planning Arrangement Out [Page 189].

Features

This outbound operation processes the `TravelPlanningArrangementEmployeeNotification` message type.

You trigger this outbound operation by executing the following programs in the SAP back-end system:

- PTRM_SE_PLAN_ARRANGMNT_INITIAL (*Initial Upload for TravelPlanningArrangementEmployeeNotification*)

  This program runs an initial upload. On the selection screen of the program, you can select the employees for whom this outbound operation supplies external service providers with *Travel Planning Arrangements*. For more information, see the program documentation in the SAP back-end system.

- PTRM_SE_PLAN_ARRANGMNT_DELTA (*Delta Upload for TravelPlanningArrangementEmployeeNotification*)

  After successful execution of the initial upload, you can use this program to run delta uploads. On the selection screen of the program, you can select the employees for whom this outbound operation supplies external service providers with changes. The program provides the following operation modes:

  - A *Normal Run* determines the employees, whose data changed since the last run, and sends their data.
  - An *Error Run* sends only the employee data that failed during the last run.

  For more information, see the program documentation in the SAP back-end system.

For delta loads, this outbound operation requires information about which employee data has changed since the last run. The PTRM_SE_PLAN_ARRANGMNT_SETUP (*Initial Settings for TravelPlanningArrangementEmployeeNotification*) program provides the settings needed to capture the delta information. For more information, see the program documentation in the SAP back-end system.

In a Concurrent Employment (CE) scenario, even if a change occurs in only one personnel number of an employee, the operation transfers the data for all personnel numbers of the employee.

Prerequisites

The distribution scenario must be activated. You can activate the distribution scenario by executing the PTRM_SE_PLAN_ARRANGMNT_SETUP program in the SAP back-end system.

Integration

This outbound operation belongs to the Travel Planning Arrangement Out [Page 189] outbound service interface.

Constraints

This outbound operation does **not** provide the identification fields for the Bahn Internet Booking Engine (BIBE) of the Deutsche Bahn Corporate Portal.

This outbound operation transfers only those fields of the infotypes (see Travel Planning Arrangement Out [Page 189]) that are relevant for travel service providers.

Notes for SAP Back-End System Administration

Processing

The Business Add-In (BAdI) PTRM_SE_TPAEE_CONV (*BAdI: Mapping for TravelPlanningArrangementEmployeeNotification*) is available for this operation. For an enhanced outbound message interface, you can use the OUTBOUND_MAPPING method to perform the outbound mapping.

Error Handling

When an error occurs while reading the data of an employee, the operation does **not** send the employee data and sets the error flag in the PAD_SE_DELTA_LOG log table to ‘X’.

In a CE scenario, when an error occurs for only one of the personnel numbers related to an employee, the operation does **not** send any employee data and sets the error flags for all personnel numbers in the PAD_SE_DELTA_LOG log table to ‘X’.

Enterprise Services for SAP ERP Human Capital Management

Personnel Administration

Technical Data

Entity Type	Process component
Software Component	*SAP ERP enterprise services package* (as of EA-HR 602)

Definition

Manages all employee-related data in the company and includes all fundamental Personnel Management administrative processes that take place within a company.

Use

An administrator’s primary task within a company is handling information requests. These information requests usually occur ad hoc and outside of the system, for example, by

telephone. Information from the process component is also used to supply downstream processes, such as payroll.

You can use the *Personnel Administration* process component to provide services for queries relating to information on employee data.

Notes for SAP Back-End System Administration

Prerequisites

To use the operations contained in the process component, you must implement *SAP ECC Enterprise Extension Human Capital Management* (EA-HR).

Personnel Leaving

Technical Data

Entity Type	Business object
Software Component	*SAP ERP enterprise services package* (as of EA-HR 602)

Definition

Ending of an employee's *Work Agreement*.

The *Work Agreement* can be ended by either the employee or the employer.

Use

The *Personnel Leaving* business object contains all relevant data for ending an employee's work agreement.

If a company accesses employee-related data from external systems, a message about the ending of the work agreement must be sent to these external systems.

Notes for SAP Back-End System Administration

Prerequisites

You use *SAP ECC Enterprise Extension Human Capital Management* (EA-HR).

Personnel Leaving Out

Technical Data

Entity Type	Service interface
Software Component	*SAP ERP enterprise services package* (as of EA-HR 602)
Category	Outbound
Mode	Asynchronous

Use

This outbound service interface contains an operation with which the information about the ending of an employee's work agreement can be sent to external systems.

Integration

The outbound service interface is part of the Personnel Administration [Page 192] process component and the Personnel Leaving [Page 193] business object.

Notify of Personnel Leaving

Technical Data

Entity Type	Operation
Technical Name in Enterprise Services Repository (ESR)	PersonnelLeavingNotification_Out
Namespace in ESR	http://sap.com/xi/EA-HR/Global2
Software Component Version in ESR	As of EA-HR 602
Category	Outbound
Mode	Asynchronous

Use

A company saves data for their employees and accesses employee-related data from external systems. For example, this includes the employee's e-mail address or credit card information. When an employee leaves the company, data consistency between the systems must be ensured.

This outbound operation sends an appropriate message to these external systems.

Features

The outbound operation processes the following message type:

`PersonnelLeavingNotificationMessage`

The following operation fields are available:

- *LeavingDate*

  Contains the date on which the work agreement was ended.

- *PersonnelEventTypeCode*

  The *PersonnelEventTypeCode* provides information about the type of ending of the work agreement.

- *Employee_ID*

- *PersonFormattedName*

  Contains the employee's name.

- *CompanyID*

- *CompanyName*

- *WorkAgreement_ID*

Prerequisites

You must activate the business function HCM_SRV_CI_1 *HCM, Enterprise Services 01* to have this outbound operation available.

Integration

This outbound operation belongs to the Personnel Leaving Out [Page 193] outbound service interface.

Notes for SAP Back-End System Administration

Prerequisites

To make the required settings for this outbound operation, you must start the report PAD_SE_LEAVING_NOTIFIC_SETUP. This report makes entries in the following Customizing tables to log the ending of a *Work Agreement* in cluster PCL4:

- T585A
- T585B
- T585C

Processing

Use the report PAD_SE_LEAVING_NOTIFICATION to trigger the outbound operation. The report can be started in two ways:

- For all *Work Agreements* that meet the selection criteria.
- For all *Work Agreements* that meet the selection criteria and that were incorrect in the last run.

A message about the ending of a work agreement is only sent for the *Work Agreements* that were ended on the current date (sy-datum) and for which no message has already been sent.

If the processing was successful, the outbound operation is triggered. A success message is written to the application log, and an entry is written in the ERROR = ‘’ field of the PAD_SE_DELTA_LOG log table.

If the processing of a *WorkAgreementID* was incorrect, an appropriate error message is included in the application log and an entry is written in the ERROR = ‘X’ field of the PAD_SE_DELTA_LOG log table. All other *WorkAgreementIDs* are processed further.

The report PAD_SE_LEAVING_NOTFIC_SETUP (*Initial Settings for PersonnelLeavingNotification*) is available for making the initial settings.

For more information about the reports, see the Implementation Guide for *Personnel Administration* under *Enterprise Services* → *Personnel Administration*.

The Business Add-In (BAdI) PAD_SE_LEAVING_NOTIF_CONV (*Mapping for PersonnelLeavingNotification*) for the Enhancement Spot PAOC_PAD_SE_PROXY is available for this outbound operation. This allows you to map customer fields from the back-end system to the corresponding fields of the outbound operation, or to make changes to the standard fields of the operation.

For more information about the BAdI, see the Implementation Guide for *Personnel Administration* under *Enterprise Services* → *Personnel Administration* → *BAdI: Mapping for PersonnelLeavingNotification*.

Enterprise Services for Trading Industries

Trading Order Processing

Technical Data

Entity Type	Process component
Software Component	EA-GLTRADE 602

Commodity management refers to the purchasing, selling, trading, logistic, and financial planning and execution of commodities. Commodities are exchange-traded materials as oil, wheat, aluminum, and electricity. Trading Order Processing provides pure physical goods trading solutions, without the necessity of price risk management, hedge pricing, or market risk of the physical deal.

Trading Order

Technical Data

Entity Type	Business object
Software Component	EA-GLTRADE 602

The trading order is the main document in Trading Order Management. You can process both purchase and sales data either simultaneously or separately, using one document. This forms the basis of all purchasing, sales and expense processes and provides you with a clear overview and process control of all business-related transactions. The trading order can cast profitability simulations immediately, taking into account purchasing prices, sales prices, and expenses. The trading contract can support back-to-back transactions as well as one-sided purchasing and sales transactions.

Manage Trading Order

Technical Data

Entity Type	Service interface
Software Component	EA-GLTRADE 602
Category	Synchronous
Mode	Inbound

This inbound service interface contains an operation that extracts data from a target trading order.

Read Trading Order

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	TradingOrderByIDQueryResponse_In
Software Component Version in ESR	EA-GLTRADE 602
Category	Inbound
Mode	Synchronous
Related Web Service Definition	Not applicable

This inbound operation is used to get the data of a target trading order.

Query Trading Order In

Technical Data

Entity Type	Service interface
Software Component	EA-GLTRADE 602
Category	Synchronous
Mode	Inbound

This inbound service interface contains an operation that extracts data from a target trading order.

Find Trading Order By Elements

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	TradingOrderByElementsQueryResponse_In
Software Component Version in ESR	EA-GLTRADE 602
Category	Inbound

Mode	Synchronous
Related Web Service Definition	Not applicable

This inbound operation is used to get the data of a target trading order.

Trading Ordering In

Technical Data

Entity Type	Service interface
Software Component	EA-GLTRADE 602
Category	Asynchronous
Mode	Inbound

This inbound service interface contains a group of operations that create or modify a trading order. This includes adding new items, changing existing items, and canceling items.

Cancel Trading Order

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	TradingOrderCancelRequest_In
Software Component Version in ESR	EA-GLTRADE 602
Category	Inbound
Mode	Asynchronous
Related Web Service Definition	Not applicable

This inbound operation is a request to cancel a trading order.

Maintain Trading Order

Technical Data

Entity Type	Operation

Technical Name in Enterprise Service Repository (ESR)	TradingOrderRequest_In
Software Component Version in ESR	EA-GLTRADE 602
Category	Inbound
Mode	Asynchronous
Related Web Service Definition	Not applicable

This inbound operation is a request to create or change a trading order during a commodity trading process.

Release Trading Order

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	TradingOrderReleaseRequest_In
Software Component Version in ESR	EA-GLTRADE 602
Category	Inbound
Mode	Asynchronous
Related Web Service Definition	Not applicable

This inbound operation is a request to release a trading order and create the required follow-up documents (such as a sales order, purchase order, or both) in one step.

Trading Ordering Out

Technical Data

Entity Type	Service interface
Software Component	EA-GLTRADE 602
Category	Asynchronous
Mode	Outbound

This outbound service interface contains a group of operations that create or modify a trading order. This includes adding new items, changing existing items, and canceling items.

Confirm Trading Order

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	TradingOrderConfirmation_Out
Software Component Version in ESR	EA-GLTRADE 602
Category	Outbound
Mode	Asynchronous
Related Web Service Definition	Not applicable

This outbound operation generates a confirmation to tell the result of each request to the requester after the requested execution.

Notify of Trading Order

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	TradingOrderNotification_Out
Software Component Version in ESR	EA-GLTRADE 602
Category	Outbound
Mode	Asynchronous
Related Web Service Definition	Not applicable

This outbound operation generates a notification when the trading order is modified.

Notify of Trading Order Cancellation

Technical Data

Entity Type	Operation

Technical Name in Enterprise Service Repository (ESR)	TradingOrderCancelledNotification_Out
Software Component Version in ESR	EA-GLTRADE 602
Category	Outbound
Mode	Asynchronous
Related Web Service Definition	Not applicable

This outbound operation generates a notification when the trading order is cancelled.

Notify of Trading Order Release

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	TradingOrderReleasedNotification_Out
Software Component Version in ESR	EA-GLTRADE 602
Category	Outbound
Mode	Asynchronous
Related Web Service Definition	Not applicable

This outbound operation generates a notification when the trading order is released.

Analytical View of Trading Order

Technical Data

Entity Type	Business object
Software Component	EA-GLTRADE 602

An analytical representation of a trading order and its structure. This business object analyzes the mark-to-market evaluation of the trading order as well as its products. It identifies changes in logistics, such as quantities, values and deadlines.

Analytical View of Trading Ordering Out

Technical Data

Entity Type	Service interface
Software Component	EA-GLTRADE 602
Category	Asynchronous
Mode	Outbound

This outbound service interface contains an operation that publishes analytical information for a trading order.

Notify of Analytical View of Trading Order

Technical Data

Entity Type	Operation
Technical Name in Enterprise Service Repository (ESR)	AnalyticalViewOfTradingOrderERPNotification_Out
Software Component Version in ESR	EA-GLTRADE 602
Category	Outbound
Mode	Asynchronous
Related Web Service Definition	Not applicable

This outbound operation generates an AnalyticalViewOfTradingOrder notification when an underlying document of the trading order is processed. .

Copyright

© Copyright 2007 SAP AG. All rights reserved.

SAP Copyrights and Trademarks

SAP Library document classification: PUBLIC

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

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

Microsoft, Windows, Outlook, Excel, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, System i, System i5, System p, System p5, System x, System z, System z9, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, Informix, i5/OS, POWER, POWER5, POWER5+, OpenPower and PowerPC are trademarks or registered trademarks of IBM Corporation.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

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

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML, and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

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

MaxDB is a trademark of MySQL AB, Sweden.

SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves information purposes only. National product specifications may vary.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

SAP - Important Disclaimers and Legal Information

SAP Library document classification: PUBLIC

This document is for informational purposes only. Its content is subject to change without notice, and SAP does not warrant that it is error-free. SAP MAKES NO WARRANTIES, EXPRESS OR IMPLIED, OR OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.

Coding samples

Any software coding and/or code lines / strings ("Code") included in this documentation are only examples and are not intended to be used in a productive system environment. The Code is only intended better explain and visualize the syntax and phrasing rules of certain coding. SAP does not warrant the correctness and completeness of the Code given herein, and SAP shall not be liable for errors or damages caused by the usage of the Code, except if such damages were caused by SAP intentionally or grossly negligent.

Internet hyperlinks

The SAP documentation may contain hyperlinks to the Internet. These hyperlinks are intended to serve as a hint where to find supplementary documentation. SAP does not warrant the availability and correctness of such supplementary documentation or the ability to serve for a particular purpose. SAP shall not be liable for any damages caused by the use of such documentation unless such damages have been caused by SAP's gross negligence or willful misconduct.

Accessibility

The information contained in the SAP Library documentation represents SAP's current view of accessibility criteria as of the date of publication; it is in no way intended to be a binding guideline on how to ensure accessibility of software products. SAP specifically disclaims any liability with respect to this document and no contractual obligations or commitments are formed either directly or indirectly by this document. This document is for internal use only and may not be circulated or distributed outside your organization without SAP's prior written authorization.

Non-Gender Specific Language

As far as possible, SAP documentation is non-gender specific. Depending on the context, the reader is addressed directly with "you", or a non-gender specific substantive (such as “sales person” or “working days”) is used. If when referring to members of both sexes, however, the third person singular cannot be avoided or a non-gender specific substantive does not exist, SAP reserves the right to use the masculine form of the substantive and pronoun. This is to ensure that the documentation remains comprehensible.