Service-Enabling a Legacy Application with SAP NetWeaver PI 7.1

Applies to:

SAP NetWeaver Process Integration 7.1 (PI 7.1)

Summary

This article describes the tasks that have to be performed by a customer who wants to service-enable a legacy application using SAP NetWeaver Process Integration.

The scenario covers all steps necessary to make legacy functionality available as a standardized service, starting with process modeling in the Enterprise Services Repository and the design of the necessary service interfaces, data types, as well as mappings and communication channel templates. Furthermore, all steps for how to configure the communication between service consumer and provider are described.

Authors: Udo Paltzer, Peter Gutsche

Company: SAP AG

Created on: 15 May 2008

Author Bio

Udo Paltzer joined SAP AG in 2001 with a university degree in physics. Udo currently applies his multiple skills as a product manager for SAP NetWeaver SOA Middleware with immediate responsibility for the rollout of SAP NetWeaver Process Integration. Udo is responsible for the Service Bus related topics of SAP NetWeaver SOA Middleware in particular. Furthermore, Udo acts as a contact person for partners and ISVs who are interested in certifying solutions, such as ESR content or adapters, for SAP NetWeaver Process Integration.

Peter Gutsche joined SAP in 1999 with a university degree in physics. He is a member of the SOA Middleware product management team. As an expert in the topic area of SAP NetWeaver Process Integration he is responsible for writing product documentation. Additionally, he acts as SDN content area manager for SOA Middleware and is responsible for general information rollout.

Table of Contents

Introduction

About This Article

This article describes the tasks that have to be performed by a customer who wants to service-enable a legacy application that does not support standards-based access using SAP NetWeaver Process Integration (SAP NW PI).

The article is kept short in order to provide an end-to-end view of the overall use case. However, at the end of each section, hyperlinks to more detailed information are provided. You can therefore use this article as the point of entry to all the relevant information that you will need when you want to set up a specific scenario that corresponds to this use case.

Note that this article guides you through all steps required to set up a typical use case as outlined in the next section. However, the article is kept as general as possible in order that it might be adapted to each possible customer scenario.

Use Case

The use case is that you service-enable a legacy application using SAP NetWeaver PI (release: SAP NetWeaver PI 7.1). In particular, you set up a scenario based on message-broker-based communication between a service consumer and a service provider. The message broker can either be the Integration Server or the Advanced Adapter Engine. Using a message broker, you can interconnect various end point types with each other, such as file, database, Web services, Java Message Service (JMS), Remote Function Call (RFC) documents, intermediate documents (IDoc) for electronic data interchange (EDI), and RosettaNet Implementation Framework (RNIF) messaging.

The use case description covers all those steps necessary to make the legacy functionality available as a standardized service, starting with process modeling in the Enterprise Services (ES) Repository and the design of the necessary service interfaces, data types, as well as mappings and communication channel templates (containing pre-configuration of the connectivity). Furthermore, all steps on how to configure the communication between service consumer and provider and how to publish the corresponding service description to the Services Registry are described. To round off the use case, links to detailed information on process execution and monitoring are provided.

To keep the article readable, and the use case description as simple as possible, the following assumptions are made about the use case:

- The business functionality of the customer is at the *provider* side (legacy application).
- The *consumer* application side is already made available out of the box by SAP and can be connected to the provider side using SAP NetWeaver PI.

  For the consumer, the necessary service interfaces, message types, and data types are already specified in detail and shipped with the ESR content by SAP.

Overview of Tasks

Tasks at Provider Side

To set up the use case, the following tasks have to be performed at the **provider side**:

1. Installing and configuring SAP NetWeaver Process Integration
2. Designing/enhancing the ESR Content
   a) Downloading the relevant ESR (Enterprise Services Repository) content from SAP Service Marketplace and importing it into the ES Repository

      The pre-defined ESR content covers the service interfaces, message types, and data types of the consumer side.

   b) Defining software component versions and namespaces
   c) Defining the service provided by the legacy application as an interface in the ES Repository

      This includes defining the service interface for the provider side.

   d) Defining the mapping between consumer and provider interface
3. Configuring the business process

   In this use case, the consumer calls the provider using a message broker interconnected between both (also referred to as “brokered communication”). The message broker can either be the Integration Server or the Advanced Adapter Engine. You configure broker-based message exchange in the Integration Directory.
4. Publishing the configured inbound processing

   Publish the corresponding service to the Services Registry to enable the consumer to call the message broker (either the Integration Server or the Advanced Adapter Engine). This service is basically determined by the configured inbound processing and the URL of the message broker. If you use the Integration Server, then this is a sender agreement, if you use the Advanced Adapter Engine, then it is an integrated configuration.

Tasks at Consumer Side

To run the scenario, the following tasks have to be performed at the **consumer side**:

1. Discovering the service to call the provider (message broker)

   In the broker-based scenario, the consumer calls the provider using a message broker interconnected between both. Essentially, the message broker acts as service provider for the consumer. To be able to call the message broker, the consumer finds the service description in the Services Registry.
2. Implementing the consumer application
3. Executing and monitoring the process

Installing and Configuring SAP NetWeaver Process Integration

As a prerequisite for being able to perform all subsequent tasks, SAP NetWeaver Process Integration has to be installed and configured within your system landscape.

More information on the installation of SAP NetWeaver Process Integration 7.1:

Implementation Documentation Center

SAP NetWeaver Process Integration 7.1 - Installation Documentation.

More information on the configuration of SAP NetWeaver Process Integration 7.1:

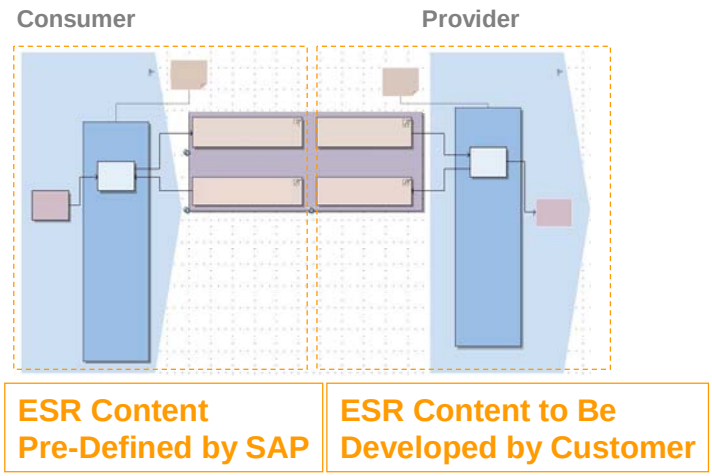
Configuring SAP NetWeaver Process Integration

Designing/Enhancing the ESR Content

Overview

According to the assumptions made above, the consumer service and part of the content delivery is already specified by SAP.

The graphic below illustrates this further. In the graphic, the process is represented by a process component interaction model from the ES Repository. A process component interaction model shows the interaction between two process components (a consumer and a provider process component). The interface objects belonging to the consumer process component are already made available by SAP, whereas the interface objects that belong to the provider process component as well as the mappings (“in between” both process components) have to be defined by the customer.



Process Model Showing Which Parts of the Content are Pre-Defined by SAP

To define a complete process, you have to do the following:

1. Import relevant ESR content

   You download the pre-defined ESR content from SAP Service Marketplace and import it into the ES Repository.

2. Define software component versions and namespaces

   As prerequisite for the development of ESR content, you need to define software components, software component versions, and namespaces.

3. Model the process in ES Repository

You outline the graphical representation of the process flow and of the interaction of the process components involved.

4. Assign the pre-defined service interfaces of the consumer part to the model

   You assign the already-specified interface objects of the consumer side to the process model.

5. Define the service provided by the legacy application as an interface in the ES Repository

   You service-enable the application on the provider side by defining its interfaces as services in the ES Repository.

   Subsequently, you assign the new defined ESR objects of the provider to the process model.

6. Define mapping of consumer and provider interface operation.
7. Define communication channel templates

Importing the Relevant ESR Content

All ESR content that is shipped by SAP can be downloaded from SAP Software Distribution Center on SAP Service Marketplace at http://service.sap.com/swdc under *Download → Support Packages and Patches Entry by Application Group → SAP Content → ESR Content (XI Content)*.

To import the ESR content, you have to do the following:

1. Download the ESR content you are looking for from the corresponding page to your hard disc.
2. Open the ES Repository.
3. Import the ESR content from your hard disc to the ES Repository.

More information: Importing ESR Content

Defining Software Component Versions and Namespaces

As prerequisite for the development of ESR content, you need to define software components, software component versions, and namespaces. A software component version is a shipment unit of ESR objects. You need software component versions and namespaces to group your ESR objects.

How you design the software components depends on the business process and the typical properties of the system landscape your process is later to be deployed in.

However, as a starting point consider the following recommendations:

- Define a separate software component for each component involved.

  Each of these software components will contain objects that “belong” to a particular process component (basically, the service interfaces, message types, data types, and communication channel templates).

- Define a separate software component for the mappings.

  Mappings are defined for pairs of interface operations belonging to *different* process components. Therefore, it is reasonable to assign the mappings to separate software components that are decoupled from the process-component-specific software components mentioned above.

- If you would like to enhance objects delivered by SAP, define an own software component that is based on the software component containing the SAP-objects to be enhanced.

  More information:

  Defining and Removing Software Dependencies

  Underlying Software Component Versions

To define software components and software component versions, you have the following options:

- Using an SLD-based software component version

  To be able to develop within a software component version that is relevant for productive use, it must be maintained in the software catalog of the System Landscape Directory (SLD). SAP products and the corresponding software component versions are already maintained in the SLD. When you need a new software component version (for own developments), you first need to define a product version and software component version for your development in the SLD.

  More information: Software Catalog

- Using local software component versions

  In addition to the shipment-relevant software component versions, you can also initially work with a *local* software component version in the ES Repository; this does not have to be maintained in the SLD. In this case, you do not need to perform any tasks in the SLD.

  Note that objects developed within local software component versions cannot be used in productive scenarios.

To make use of a software component version for developing objects in the ES Repository, you either have to import the software component version defined in SLD to the ES Repository (if you are using an SLD-based software component version) or you create a local software component version.

You perform these steps in the ES Repository.

1. In your back-end system, use transaction `SXMB_IFR` to open the SAP NetWeaver Process Integration menu.
2. Choose *Enterprise Services Builder* and log on.
3. Perform the appropriate steps described under Create a Software Component Version.
4. You need at least one namespace in the software component version (either SLD-based or local). To do this, perform the appropriate steps described under Creating a Namespace.

Modeling the Process

The ES Repository provides different modeling environments for process modeling. Different model types can be created depending on the use case.

For a description of all modeling environments and model types, select the following chapter in SAP Library: Defining Design Objects for Modeling Applications.

In this article we assume that you start with a *process component interaction model*. Using this model type, you can describe the interaction of two process components and of the involved ESR objects, such as service interfaces, operations, message types, and mappings.

In this use case, create a process component interaction model and start with creating a process component for the consumer and a process component for the provider. Then, add a graphical representation for each relevant business object, service interface, service interface operation, message type, and operation mapping. If you would also like to specify parts of the connectivity configuration, you can also add graphical representations for communication channel templates.

Follow the guidelines described under: Modeling the Interaction Between Two Process Components.

Note that so far you have only created a model including *graphical representations* of the involved ESR objects. In subsequent steps, you can do the following:

- If an ESR object is already specified for a certain part of the model, you can assign the ESR object its graphical representation in the model.
- If no ESR object is specified for a certain part of the model, define the ESR object first and then assign it to the model.

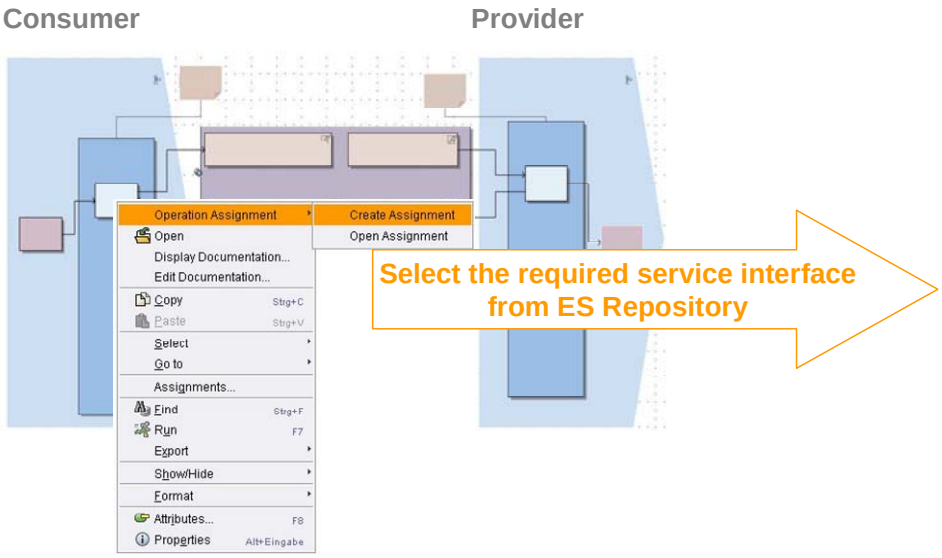
How you do this is described in the following two sections.

Assigning the Pre-Defined ESR Content to the Model

Since the ESR content for the consumer part of the communication is already available in the ES Repository, you can simply assign the relevant objects to the model.

For example, to assign a service interface operation, do the following:

In the model, position the cursor on the graphical representation of the operation (a light blue rectangle in the model) and select *Operation Assignment* → *Create Assignment* in the context menu.



Creating Assignment to a Service Interface Operation

Assign the other relevant objects analog to this procedure.

Defining the Interface Objects for the Legacy Application in the ES Repository

The provider service is already implemented in the provider back-end system. However, it is not yet available as a standardized service. Service-enabling the legacy application means: Defining the corresponding interface objects in the ES Repository. This is called *inside-out* development of interfaces.

To develop a service inside-out, you can either import an XML description for an existing function to the ES Repository, or you can create the interface objects manually within the ES Repository. The interface description can be used by any number of callers for communication using the Integration Broker.

Note that you can also link external documentation from the ESR object.

More information:

Developing Service Interfaces (contains information on how to develop service interfaces in the ES Repository)

Developing Enhanced Services Inside-Out (contains information on the inside-out development approach for services in particular)

Subsequently, assign the relevant interface objects to the process model.

Defining the Mapping of Consumer and Provider Interface

In general, the structures of the consumer and provider interface differ from each other. Therefore, you have to define the transformation between both structures, also referred to as a *mapping*.

You can define one or more mappings for an interface operation pair in the ES Repository. When configuring the process later in the Integration Directory, you can select the mapping that corresponds to a given collaborative scenario.

More information: Mapping Messages to Each Other Using Mapping Objects

Once you have finished the definition of the mapping object, assign the mappings to the process model.

Defining Communication Channel Templates

When you specify the business process in the ES Repository, parts of the connectivity configuration might already be known, for example, the adapter types that are used. Within a communication channel template, you can already specify these parts of the configuration in the ES Repository. Later, when you configure the business process for a specific system landscape, you select the communication channel template and create the relevant communication channel in the Integration Directory based on that template.

More information: Communication Channel Template

Once you have finished the definition of the communication channel template, assign the communication channel template to the process model.

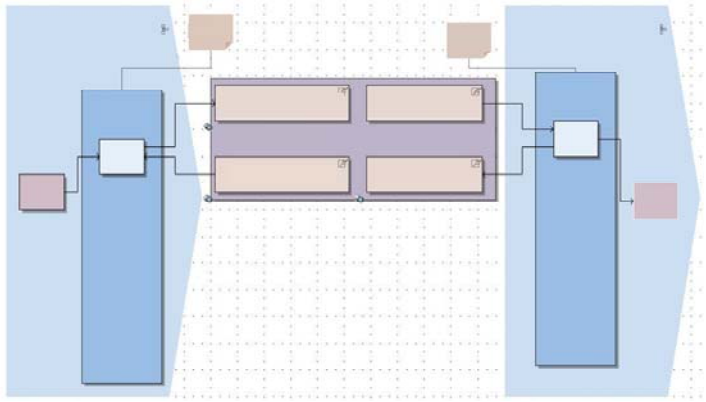
Configuring the Business Process

Overview

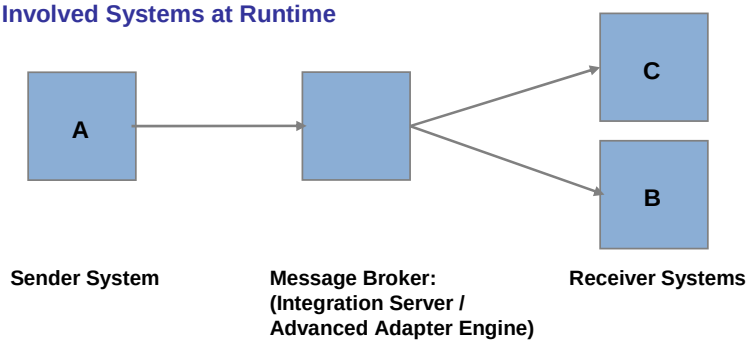
To configure the business process defined in the ES Repository means: Specify the message exchange in a specific system landscape for the process.

The figure below shows the process model from the ES Repository. To keep it simple, let us assume that in a specific customer implementation, the consumer (or sender) application runs on one system whereas the provider (or receiver) application runs on two different systems.

Process Components Interaction Model in ES Repository



Involved Systems at Runtime



Configuring a Process for a Specific System Landscape

There are two options to set up brokered communication:

- Using the Integration Server as message broker

  Using this option, you can make use of the complete set of functions provided by SAP NetWeaver Process Integration with regard to routing and mapping.

- Using the Advanced Adapter Engine as message broker

  Using this option, the consumer calls the provider using the Advanced Adapter Engine thus bypassing the Integration Server.

  However, using the Advanced Adapter Engine allows for exchanging messages with higher performance than using the Integration Server.

General Configuration Steps (for Both Communication Options)

You have to perform the following general configuration tasks for either option:

- Defining the system landscape in the System Landscape Directory

  In the System Landscape Directory (SLD), define business systems for those parts of the system landscape that are known by the configuration expert at the customer side.

  More information: Tasks in the System Landscape Directory

- Specifying the available senders and receivers of messages

  Based on the description of the system landscape in the SLD, for each system that acts as sender or receiver of messages, a *communication component* of the type *business system* has to be defined in the Integration Directory.

  For those parts of the system landscape that are not known by the configuration expert, and which are therefore not defined in the SLD, define business components.

  More information: Defining Communication Components

- Defining communication channels (configuring the connectivity)

  To configure the connectivity to a sender or receiver back-end system, you define a *communication channel* and assign the communication channel to the corresponding communication component. For each supported adapter, a specific communication channel type is available that contains all the relevant adapter attributes. To configure connectivity to a sender back-end system, you define a sender channel; to configure connectivity to a receiver back-end system, you define a receiver channel. From the message broker point of view, a sender channel contains the configuration of the inbound processing (how an incoming message is handled at the Integration Broker side) whilst a receiver channel contains the configuration of the outbound processing (how the outgoing message is handled at the message broker side).

  More information: Defining Communication Channels

  If parts of the configuration are already defined as communication channel templates in the ES Repository, you can create the corresponding communication channels based on the available templates.

  More information: Communication Channel Template

Configuring Message Exchange Using the Integration Server

When you use the Integration Server as message broker, perform the following additional steps to configure the process:

- Configuring routing

  To define to which receivers the Integration Server has to route a message, define a *receiver determination*.

  If within your business process the routing of the message depends on the business content of the message (that is, the payload), then you can specify *content-based routing*. To do this, add a condition to the receiver determination that depends on the values of certain elements in the payload.

  Using the system landscape illustrated in the figure above, the scenario can be configured in a way that the message is sent to either of the two available receiver systems – or to both – depending on the value of a specific field in the payload.

  For example, you can configure the routing dependent on the value of a field OrderNumber in the message so that the following routing rule applies: “If the OrderNumber exceeds a defined value x, send the message to receiver system B, in all other cases, send the message to receiver system C”.

  To further refine the configuration of the routing, you have to specify to which inbound interface the message has to be sent at the receiver side, as well as the mapping that should be performed to transform the data structures on the sender and the receiver side. Note that in the ES Repository you have the option to define more than one mapping for a connection between consumer and provider. When you configure the routing, you have to specify which of these mappings should be applied. You specify these settings in an object called *interface determination*.

More information: Defining Logical Routing

- Configuring the inbound and outbound message processing

  To specify which connectivity should be used for a specific sender-receiver pair, define a collaboration agreement. For inbound processing at the Integration Server specify a sender agreement, for outbound processing specify a receiver agreement. Assign to the sender or receiver agreement the communication channel that should be used.

  More information: Defining Collaboration Agreements

Configuring Message Exchange Using the Advanced Adapter Engine

When you use the Advanced Adapter Engine as message broker, do the following additional steps to configure the process:

Create an integrated configuration in the Integration Directory. Use this option to specify all configuration settings for routing and for the outbound and inbound processing in a single integrated configuration object.

More information: Executing Integrated Configuration

Note that when using the Advanced Adapter Engine, only a sub-set of the available adapters are supported.

More information: Advanced Adapter Engine

Configuring Security Settings (for B2B Scenarios)

If your process involves business partners belonging to different companies or organizations, you can make use of different concepts for the configuration of business-to-business (B2B) scenarios.

For example, you can configure security settings on the message level (signing and encrypting messages) as well as on transport layer.

More information:

- Configuring Security Settings (contains an overview of supported security settings)
- Configuring B2B Processes (contains information on general concepts for the configuration of B2B scenarios)
- XML Validation (contains information on a specific concept supported by SAP NetWeaver PI that enables you to check the structure of a message payload)

Publishing the Configured Inbound Processing

Publish the corresponding service to the Services Registry to enable the consumer to call the message broker (either the Integration Server or the Advanced Adapter Engine) This service is basically determined by the configured inbound processing and the URL of the message broker.

- If you have configured message processing using the Integration Server, publish the sender agreement to the Services Registry.

  To do this, open the sender agreement and in the menu of the object editor select *Sender Agreement → Publish in SR*.

  More information: Publishing Sender Agreements in the Services Registry

- If you have configured message processing using the Advanced Adapter Engine, publish the integrated configuration to the Services Registry.

  To do this, open the integrated configuration and in the menu of the object editor select *Integrated Configuration → Publish in SR*.

Tasks at the Consumer Side

Discovering the Service to Call the Provider

Since the use case is based on brokered communication, the consumer calls the provider using either the Integration Server or the Advanced Adapter Engine.

To discover the service to call the message broker, do the following:

1. Open the Services Registry.
2. Select the Advanced Search.
3. Under *Select Classification* select *SoftwareComponentVersion*.
4. Select the software component version of the inbound interface that fits to the key of the sender agreement/integrated configuration in the Integration Directory.

More information: Services Registry

Implementing the Consumer Application

Use the relevant development environment to implement the consumer application that invokes the service call.

Executing and Monitoring the Process

When executing the process invoked by the service call, you have the option to monitor the process. For example, you can monitor the exchanged messages, the performance of the process, or the involved communication channels.

To monitor the process, you can use the SAP NetWeaver Administrator or the Runtime Workbench.

More information:

PI Monitoring with SAP NetWeaver Administrator

Tools for Monitoring

Related Content

SAP NetWeaver Process Integration Library

Service Enabling with SAP NetWeaver Process Integration 7.1

SAP NetWeaver Process Integration 7.1 Demo Examples

Copyright