PI Best Practices: Naming Conventions

Applicable Releases:
SAP NetWeaver 7.1x

Topic Area:
SOA Middleware

Capability:
SOA Management

Version 1.0
July 2009
## Document History

<table>
<thead>
<tr>
<th>Document Version</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00</td>
<td>First official release of this guide</td>
</tr>
</tbody>
</table>
## Typographic Conventions

<table>
<thead>
<tr>
<th>Type Style</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example Text</td>
<td>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.</td>
</tr>
<tr>
<td>Example text</td>
<td>Emphasized words or phrases in body text, graphic titles, and table titles</td>
</tr>
<tr>
<td>Example text</td>
<td>File and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.</td>
</tr>
<tr>
<td>Example text</td>
<td>User entry texts. These are words or characters that you enter in the system exactly as they appear in the documentation.</td>
</tr>
<tr>
<td>&lt;Example text&gt;</td>
<td>Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.</td>
</tr>
<tr>
<td>EXAMPLE TEXT</td>
<td>Keys on the keyboard, for example, F2 or ENTER.</td>
</tr>
</tbody>
</table>

## Icons

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>🔄</td>
<td>Caution</td>
</tr>
<tr>
<td>📋</td>
<td>Note or Important</td>
</tr>
<tr>
<td>🍀</td>
<td>Example</td>
</tr>
<tr>
<td>🏞</td>
<td>Recommendation or Tip</td>
</tr>
</tbody>
</table>
# Table of Contents

1. **Introduction** .......................................................................................................................... 1

2. **Notation** ................................................................................................................................ 4

3. **Naming Conventions** ........................................................................................................... 5
   3.1 System Landscape Directory ........................................................................................ 5
   3.1.1 Product ............................................................................................................. 5
   3.1.2 Software Component Version .......................................................................... 6
   3.1.3 Technical System ............................................................................................. 8
   3.1.4 Business System ............................................................................................. 9
   3.1.5 Business System Group ................................................................................ 10
   3.2 Enterprise Service Repository Objects ....................................................................... 10
   3.2.1 Folder ............................................................................................................. 10
   3.2.2 Local SWCV ................................................................................................... 11
   3.2.3 Namespace .................................................................................................... 11
   3.2.4 Process Integration Scenario & Action ...................................................................... 12
   3.2.5 Model ............................................................................................................. 13
   3.2.6 Integration Process Related Objects .................................................................... 13
   3.2.7 Alert Category ................................................................................................ 17
   3.2.8 Data Type ...................................................................................................... 17
   3.2.9 Data Type Enhancement ............................................................................... 18
   3.2.10 Message Type ............................................................................................... 19
   3.2.11 Fault Message Type ...................................................................................... 20
   3.2.12 Service Interface .......................................................................................... 20
   3.2.13 Service Operation .......................................................................................... 21
   3.2.14 External Definition .......................................................................................... 22
   3.2.15 Context Object ............................................................................................... 23
   3.2.16 Message Mapping .......................................................................................... 23
   3.2.17 Mapping Template ......................................................................................... 24
   3.2.18 Operation Mapping ........................................................................................ 24
   3.2.19 Imported Archive ............................................................................................ 25
   3.2.20 Function Library ............................................................................................. 26
   3.2.21 Configurable Parameter ................................................................................. 27
   3.2.22 Communication Channel Template ............................................................... 27
   3.2.23 Adapter Metadata .......................................................................................... 28
   3.3 Integration Directory Objects ...................................................................................... 29
   3.3.1 Configuration Scenario .................................................................................. 29
   3.3.2 Folder ............................................................................................................. 29
   3.3.3 Party ............................................................................................................... 29
3.3.4 Communication Component ................................................................. 30
3.3.5 Communication Channel ................................................................. 32
3.3.6 Value Mapping ................................................................................. 32

4. References ............................................................................................... 34
1. Introduction

This document provides best practices for defining naming conventions for objects in the System Landscape Directory (SLD), Enterprise Services Repository (ESR), and Integration Directory (ID).

The naming conventions described here are not compulsory. Our intention is to provide general recommendations and principles that should help to define your own company wide naming conventions. As every company is different in structure and size it is important to gauge what level of granularity is required when naming objects in your landscape.

Find a balance between what is workable and what allows for sufficient growth as you develop additional objects. Although this document concentrates on design time naming conventions, consideration should also be given to runtime usability. E.g., ensure conventions applied to design-time objects do not impair administrators when monitoring thousands of messages in the Runtime Workbench or Integration Engine.

Well defined naming conventions should help to standardize and facilitate the implementation of your integration scenarios. The recommendations given here apply for both mediated scenarios via SAP NetWeaver Process Integration (PI) and applications following the Service Oriented Architecture (SOA) approach.

Scope

Where applicable, we follow the same recommendations given in the ‘SAP Exchange Infrastructure 3.0: Best Practices for Naming Conventions’ guide, accessible on SDN https://www.sdn.sap.com/irj/scn/go/portal/prtroot/docs/library/uuid/90b213c2-d311-2a10-89bf-956dbb63aa7f. This guide applies to XI 3.0 and PI 7.0. In the current paper, we deal with the new objects and changes which have been introduced with PI 7.1.

The table below summarizes the scope of the current paper. It shows at a glance which objects have been covered in the previous guide, and which objects need to be revised.

<table>
<thead>
<tr>
<th>Area</th>
<th>Object</th>
<th>Previously covered</th>
<th>Needs revision</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLD</td>
<td>Product</td>
<td>✓</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Software Component</td>
<td>✓</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Technical System</td>
<td>✓</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Business System</td>
<td>✓</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Business System Group</td>
<td>×</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ESR</td>
<td>Folder</td>
<td>×</td>
<td></td>
<td>New in PI 7.1</td>
</tr>
<tr>
<td></td>
<td>Local Software Component</td>
<td>×</td>
<td></td>
<td>New in PI 7.1</td>
</tr>
<tr>
<td></td>
<td>Namespace</td>
<td>✓</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Process Integration Scenario</td>
<td>✓</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Model</td>
<td>×</td>
<td></td>
<td>New in PI 7.1</td>
</tr>
<tr>
<td></td>
<td>Integration Process</td>
<td>✓</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Monitoring Process</td>
<td>×</td>
<td></td>
<td>New in PI 7.1</td>
</tr>
<tr>
<td>Feature</td>
<td>Status</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>--------</td>
<td>------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Step Group</td>
<td>✗</td>
<td>New in PI 7.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alert Category</td>
<td>✗</td>
<td>New in PI 7.1, at least new in ESR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Type</td>
<td>✓/✓</td>
<td>New classifications: free style versus Global Data Type (GDT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Type Enhancement</td>
<td>✓/✗</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Message Type</td>
<td>✓/✗</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fault Message Type</td>
<td>✗</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Service Interface</td>
<td>✓/✓</td>
<td>previously Message Interface</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Service Operation</td>
<td>✗</td>
<td>previously Message Interface</td>
<td></td>
<td></td>
</tr>
<tr>
<td>External Definition</td>
<td>✓/✗</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Context Object</td>
<td>✓/✗</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Message Mapping</td>
<td>✓/✗</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mapping Template</td>
<td>✓/✗</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operation Mapping</td>
<td>✓/✓</td>
<td>Previously Interface Mapping</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Imported Archive</td>
<td>✓/✗</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Function Library</td>
<td>✗</td>
<td>New in PI 7.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Configurable Parameter</td>
<td>✗</td>
<td>New in PI 7.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Communication Channel</td>
<td>✗</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adapter Metadata</td>
<td>✗</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID Configuration Scenario</td>
<td>✓/✗</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Folder</td>
<td>✗</td>
<td>New in PI 7.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Party</td>
<td>✓/✗</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Business Component</td>
<td>✓/✗</td>
<td>previously Business Service</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Integration Process Service</td>
<td>✓/✗</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Communication Channel</td>
<td>✓/✗</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Value Mapping</td>
<td>✓/✗</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Changes in Naming with SAP NetWeaver PI 7.1

For previous releases, the Integration Repository was used as part of the SAP Exchange Infrastructure. In SAP NetWeaver Process Integration 7.1, this repository has new functions that support a SOA approach. For this reason, numerous name changes have been introduced with SAP NetWeaver Process Integration 7.1, as listed in the table below.

<table>
<thead>
<tr>
<th>Previous Releases</th>
<th>As of SAP NetWeaver PI 7.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adapter Engine</td>
<td>Advanced Adapter Engine</td>
</tr>
<tr>
<td>Integration Builder (Design)</td>
<td>Enterprise Services Builder (ES Builder)</td>
</tr>
<tr>
<td>Integration Builder (Configuration)</td>
<td>Integration Builder</td>
</tr>
<tr>
<td>Integration Directory</td>
<td>Integration Directory</td>
</tr>
<tr>
<td>Integration Repository</td>
<td>Enterprise Services Repository (ES Repository)</td>
</tr>
<tr>
<td>Interface Mapping</td>
<td>Operation Mapping</td>
</tr>
<tr>
<td>Message Interface</td>
<td>Service Interface</td>
</tr>
<tr>
<td>Process integration content (XI content)</td>
<td>Enterprise Services Repository content (ESR content)</td>
</tr>
<tr>
<td>SAP NetWeaver Exchange Infrastructure</td>
<td>Process Integration usage type (SAP NetWeaver PI)</td>
</tr>
<tr>
<td>Service (in the Integration Directory)</td>
<td>Communication Component</td>
</tr>
</tbody>
</table>
2. Notation

General Recommendations

- For general guidelines when creating ESR objects, please refer to the SAP Help Portal http://help.sap.com/saphelp_nwpi71/helpdata/EN/3f/9ddca372cbf146bdcdf54db7737623/frameset.htm
- Use Upper Camel Case: Words of a name are joined without spaces, and are capitalized within the compound
- For better readability, use shorter names, abbreviations, or codes, see also general abbreviation rules below

Naming Rule Notation

We follow the notation as stated in table below.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>x is a fixed term</td>
</tr>
<tr>
<td>&lt;x&gt; (brackets)</td>
<td>x is a variable term according to a specified rule</td>
</tr>
<tr>
<td>* (star)</td>
<td>The previous item occurs zero, one or many times</td>
</tr>
<tr>
<td>+ (plus)</td>
<td>The previous item occurs one or many times</td>
</tr>
<tr>
<td>? (question mark)</td>
<td>The previous item is optional (occurs zero or one times)</td>
</tr>
<tr>
<td></td>
<td>(pipe)</td>
</tr>
<tr>
<td>(expression)</td>
<td>Parenthesis group the expression between them</td>
</tr>
</tbody>
</table>

General Abbreviation Rules

- Always guarantee that resulting abbreviations remain understandable and distinguishable
- Try not to create abbreviations that are themselves English words
- An abbreviation should be associated with only one unabbreviated word
- Do not use abbreviations that will conflict with commonly known abbreviations
- Your abbreviation should begin with the same letter as the word being abbreviated
- Words that are four or less characters in length should not be abbreviated
- Distinguish between
  - Abbreviations for Individual terms (e.g., Arrgmt - Arrangement)
  - Acronyms (e.g., RFQ - Request for Quote)
3. Naming Conventions

3.1 System Landscape Directory

The System Landscape Directory (SLD) serves as a central information repository for your system landscape. For PI, it keeps information about all your technical and business systems within your landscape as well as all installable and installed software components. For more details, please refer to SAP help portal [http://help.sap.com/saphelp_nwpi71/helpdata/en/31/f0ff69551e4f259fdad799a229363e/frameset.htm](http://help.sap.com/saphelp_nwpi71/helpdata/en/31/f0ff69551e4f259fdad799a229363e/frameset.htm).

3.1.1 Product

A product is a unit that is delivered and visible to the customer, it is installable and renewable. In an SAP environment, a product corresponds to an SAP technical component.

A product can have several versions. Each product consists of various software components or versions of software components.

Standard SAP products and software components are shipped with the SLD and accessible from the SLD’s software catalog. For your own customer specific developments, create your own product and software component in SLD.


General Rules

- The product is the relevant business application and not the technical environment
- Do not use SAP standard products for custom PI development
- Add your customer name or abbreviation to the product name to distinguish from SAP standard content. If you place the customer name at the beginning of the product name, you can naturally group together all applications belonging to the same customer
- If your product is an enhancement of a SAP standard product, this should be reflected in the name
- The product version does not necessarily have to be aligned with the versioning of related SAP standard products
- Group your products vertically rather than horizontally. Horizontally means that a product corresponds to an integration scenario, i.e., you keep all your objects of a specific integration scenario in one product or software component. For the vertical approach, you define a product or software component for each application, see also below
- Optionally, specify a product group which categorize your products, e.g., indicating that it is an SAP standard enhancement, or a canonical definition, etc. Please refer to XI 3.0 Naming conventions for detailed definition of product group
  - Interface Products (Group I): Represent interfaces and extensions to your applications
  - PI Application Products (A): Represents objects in the ESR other than interface objects which are jointly used by sending and receiving application (e.g., mapping objects, integration processes, etc)
PI Best Practices: Naming Conventions

- Basis Products (B): Ideal for templates, generic structures, shared Java programs, etc
- Canonical Definitions (C): For generic business objects, intended for reuse

### Syntax

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product Name</td>
<td><code>&lt;Company&gt;_&lt;Group&gt;_Business Application&gt;</code></td>
</tr>
<tr>
<td>Version</td>
<td><code>&lt;Generic Version Number starting with 1.0&gt;</code></td>
</tr>
<tr>
<td>Vendor</td>
<td><code>&lt;Your Company Domain Name&gt;</code></td>
</tr>
</tbody>
</table>

### Glossary

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Company&gt;</td>
<td>Specify a string naming your company</td>
</tr>
<tr>
<td>&lt;Group&gt;</td>
<td>I</td>
</tr>
<tr>
<td>&lt;Business Application&gt;</td>
<td>Specify your actual business application or technical component, e.g., Sales, Financial, CRM, etc</td>
</tr>
<tr>
<td>&lt;Generic Version Number&gt;</td>
<td>1.0 → 1.1 → 2.0 etc</td>
</tr>
<tr>
<td>&lt;Company Domain Name&gt;</td>
<td>This could be <a href="http://www.mycompany.com">www.mycompany.com</a></td>
</tr>
</tbody>
</table>

### Example

MyCompany_I_ERP, Version 1.0 of MyCompany.com
MyCompany_A_ERP, Version 1.0 of MyCompany.com
MyCompany_Financial, Version 1.2 of MyCompany.com
MyCompany_MDM, Version 1.0 of MyCompany.com

### 3.1.2 Software Component Version

A software component version (SWCV) is a shipment unit for design objects in the ESR. A SWCV can be used within several products.

In order to increase reusability from existing objects from other software components, you can define usage dependencies within the SLD which is then reflected as basis software components within the ESR.

For testing purposes, you can define so called local software component versions. This is however directly done in the ESR, see below.


### Software Component Strategy

When you begin using SAP Process Integration you are faced with the issue of how many software components should be used to represent a particular integration scenario. As with naming conventions it is very important to lay a solid foundation before you begin creating integration scenarios. Once you start developing objects or have scenarios in production it is very hard to change an approach or strategy. The following table introduces the three common approaches to software component
strategy. For more information please refer to “XI 3.0 Best Practices for Naming Conventions” and the numerous articles and reference material found on SDN.

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>One (horizontal approach)</td>
<td>Using one software component to hold all your design objects may be possible but does not promote re-use and is devoid of clear conceptual separation. Would you put every line of code into one program? The same rule applies here: you need some method of demarcation.</td>
</tr>
<tr>
<td>Two (vertical approach)</td>
<td>As step up from the single software component strategy is to have two, one representing the sender and the other representing the receiver. Whilst this gives clear separation and an understanding of what belongs where, it does not cater for the objects that cross sender and receiver systems. Such objects include mappings, integration scenarios and integration processes. Where do you put these? If you chose this strategy you have to create clear rules which determine where these objects are placed, i.e., mappings are always in the sender or receiver. It is your responsibility to enforce such rules amongst your development team as with the naming conventions otherwise your ESR will become an SOA artifact library that is not organized and difficult to use.</td>
</tr>
<tr>
<td>Three</td>
<td>An extension on the previous strategy is to use three software components to represent an integration scenario. One for the sender, another for the receiver and a final software component for the objects which cross both. This allows for clear conceptual separation and a high level of re-use. Please see “XI 3.0 Best Practices for Naming Conventions” for more information.</td>
</tr>
</tbody>
</table>

**General Rules**

- Use a meaningful description of your software component which is common for a number of repository objects and should be grouped together
- The name should reflect the main application from your backend application
- Do not use SAP standard software components for custom PI development
- Add your customer name or abbreviation to the software component name to distinguish from SAP standard content.
- If your software component is an enhancement of an SAP standard software component, this should be reflected in the name
- The software component version does not necessarily have to be aligned with the versioning of related SAP standard software components
- As seen for products, you can categorize your software components in the same way using a group
- Although you can use lower case in SLD, the name of the SWCV is displayed in capital letters in ESR. So, use underscore to separate the name sections in order to make the name better readable
## Syntax

<table>
<thead>
<tr>
<th>SWCV Name</th>
<th>&lt;Company&gt;(<em>&lt;Group&gt;)?</em>&lt;Product&gt;_&lt;Application&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vendor</td>
<td>&lt;Your Company Domain Name&gt;</td>
</tr>
<tr>
<td>Version</td>
<td>&lt;Generic Version Number starting with 1.0&gt;</td>
</tr>
</tbody>
</table>

## Glossary

<table>
<thead>
<tr>
<th>&lt;Company&gt;</th>
<th>Specify a string naming your company</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Group&gt;</td>
<td>I</td>
</tr>
<tr>
<td>&lt;Product&gt;</td>
<td>Specify the product the software component belongs to</td>
</tr>
<tr>
<td>&lt;Application&gt;</td>
<td>Specify your actual application on a functional area level, e.g. Global Spend Analysis in SRM, Supply Network Planning in Manufacturing, etc</td>
</tr>
<tr>
<td>&lt;Generic Version Number&gt;</td>
<td>1.0 → 1.1 → 2.0 etc</td>
</tr>
<tr>
<td>&lt;Company Domain Name&gt;</td>
<td>This could be <a href="http://www.mycompany.com">www.mycompany.com</a></td>
</tr>
</tbody>
</table>

**Example**

MyCompany_I_ERP_Procurement, Version 1.0 of MyCompany.com
MyCompany_Financial_Accounting, Version 1.2 of MyCompany.com
MyCompany_MDM_BusinessPartner, Version 1.0 of MyCompany.com
MyCompany_A_MDM_ValueMapping, Version 1.0 of MyCompany.com

## 3.1.3 Technical System

Technical systems are tangible application systems that are installed in your system landscape.  

### General Rules

- Maintain technical systems in SLD for A2A scenarios only. For B2B scenarios, parties and business services are maintained in the Integration Directory
- Naming conventions only apply to non-SAP/3rd party systems since SAP systems are registered at SLD automatically via SLD data supplier, hence maintain technical systems only if not registered automatically
- The name of the technical system should usually refer to its relevant business application or usage and not the name of a specific server
- Specify the landscape/environment stage, i.e., distinguish between development, test, and production environment, and other. Place the stage indicator either at the beginning or as suffix, depending on the way you like them to be ordered
- Optionally, group your technical systems according to location or organizational structure, e.g., America, Europe, Asia
### Syntax

| Technical System | <Location | Organizational Area>?<Business Application>_<Stage> |
|------------------|-----------------------------------------------|

### Glossary

<table>
<thead>
<tr>
<th>&lt;Location&gt;</th>
<th>Specify the geographical area the system is located in</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Organizational Area&gt;</td>
<td>Specify the business unit the system is assigned to</td>
</tr>
<tr>
<td>&lt;Business Application&gt;</td>
<td>Specify your actual business application or usage of the system</td>
</tr>
<tr>
<td>&lt;Stage&gt;</td>
<td>D</td>
</tr>
</tbody>
</table>

#### Example

AsiaPacificProcurementSystem_Q  
GlobalHumanResources_P  
MDMServer_P

### 3.1.4 Business System

Business systems are logical systems, which function as senders or receivers within PI.


#### General Rules

- The name of the business system should refer to its relevant business application or usage
- If the business system is the only business system of the related technical system, you may use the same name like for the technical system
- Specify the stage
- Group your business systems according to location or organizational structure

### Syntax

<table>
<thead>
<tr>
<th>SAP System</th>
<th>&lt;System ID&gt;&lt;Client&gt;_&lt;Stage&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Other System</td>
<td>&lt;Location</td>
</tr>
</tbody>
</table>

### Glossary

| <System> | Specify the three digit number of your SAP system |
| <Client> | Specify the client number of your SAP system |
| <Stage> | D | Q | P | T | <Other> |
| <Location> | Specify the geographical area the system is located in |
| <Organizational Area> | Specify the business unit the system is assigned to |
| <Business Application> | Specify your actual business application or usage of the system |
3.1.5 Business System Group

All systems that are related to the same integration server domain can be assigned to a business system group. Business system groups are a prerequisite for defining transport targets in SLD. Usually you divide the systems into groups depending on the stage or technical tier. When defining transport targets between two groups, you ensure that during transport business system components in the Integration Directory are properly assigned to each other.

http://help.sap.com/saphelp_nwpi71/helpdata/EN/ef/a21e3e0987760be10000000a114084/frameset.htm

General Rules

- Use a meaningful description of your group
- Specify the stage

<table>
<thead>
<tr>
<th>Syntax</th>
<th>System Group</th>
<th>&lt;Group Description&gt;_&lt;Stage&gt;</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Glossary</th>
<th>&lt;Description&gt;</th>
<th>Provide a meaningful description of your group</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&lt;Stage&gt;</td>
<td>D</td>
</tr>
</tbody>
</table>
Recommendations

- It does not make sense to replace process integration scenarios with folders since as mentioned above each object can be assigned to one folder only
- Use folders to separate different object types, e.g., one folder ‘mapping’ consisting of subfolders ‘message mapping’, ‘operation mapping’, ‘libraries’, ‘configurable parameters’, etc
- Use folders to separate your different model types, e.g., folders for model types ‘Process Component Model’, ‘Process Component Interaction Model’, ‘Integration Scenario Model’, etc

3.2.2 Local SWCV

You can define local software component versions directly in the Enterprise Services Repository. They are intended for temporary development of local objects in ESR or for testing purposes only however you cannot carry out any runtime tests. Local objects cannot be used in any configuration in the Integration Directory nor can they be transported. Once you decide to make the local objects general available, you can transfer them via release transfer.

Recommendations

- For local SWCV same conventions apply like for SWCV as specified above since it is intended to transfer the objects underneath to a productive SWCV

3.2.3 Namespace

Repository namespaces are used to avoid naming conflicts within object types in the ES Repository


General Rules

- Define namespace either
  - based on business processes (namespace is global and independent from the organizational unit inside the company) or
  - based on organizational units or geographical area (namespace implements different solutions for the company’s organizational units)
  - or a mixture of both
- use urn instead of http in order not to confuse that this URL actually cannot be called

Syntax

<table>
<thead>
<tr>
<th>Based on processes:</th>
<th>Based on organization unit:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Namespace</td>
<td></td>
</tr>
<tr>
<td>urn:&lt;Company URN&gt;:&lt;Process Level&gt;:(&lt;Further Levels&gt;:)*&lt;Application&gt;</td>
<td>urn:&lt;Company URN&gt;:&lt;Application&gt;:&lt;Org Unit Level&gt;:(&lt;Further Levels&gt;)*</td>
</tr>
</tbody>
</table>
3.2.4 Process Integration Scenario & Action

A Process Integration Scenario models the complete exchange of messages for a collaborative process and provides an overview of the process flow. The Process Integration Scenario combines all objects that are involved in this process: interface objects, mapping objects, executable integration processes from the Enterprise Services Repository, and product versions from the System Landscape Directory. The Process Integration Scenario contains all design time information about the process that is required for its configuration in the Integration Directory.

Actions are functions within an application component subdividing the process flow of a Process Integration Scenario.

http://help.sap.com/saphelp_nwpi71/helpdata/EN/88/7adb7a030b424b8ef29b99461e52a8/frameset.htm

General Rules

- Focus more on the process rather than the integration requirement. E.g., show the process that requires replications instead of creating a ‘replication scenario’.
- Add a meaningful description of the business scenario, or the general function of the same
- Specify your application. Either use the software component or a meaningful application component description
- For actions, specify business object, and related operation or action such as send, receive, create, request, change, etc

Syntax

Integration Scenario <Application><Business Process>

Action <Business Object><Operation>

Glossary

<table>
<thead>
<tr>
<th>&lt;Application&gt;</th>
<th>Specify your actual business application</th>
</tr>
</thead>
</table>

Example

urn:MyCompany.com:ERP:Finance:Accounting
urn:MyCompany.com:Procurement:Europe:France
urn:MyPartner.com:MDM:Vendor:ValueMapping
3.2.5  Model

The models in the ES Repository support a model-driven development of an Enterprise Service Oriented Architecture. The ESR offers a modeling environment where you can create various models.

http://help.sap.com/saphelp_nwpi71/helpdata/EN/e5/9ad967721045cb8cb038fb61267e2e/frameset.htm

Naming conventions regarding models in ESR are described in detail in the ‘PI Best Practices: Modeling’ how to guide, accessible on SDN

3.2.6  Integration Process Related Objects

An integration process is an executable cross-system process for processing messages, also known as cross-component Business Process Management (ccBPM). In an integration process you define all the process steps to be executed, containers, and correlations in order to control the process. You can distinguish between process steps that operate on messages and such that control the process flow. Containers are variables which store the process data such as messages or simple data types for keeping a counter for instance. Correlations are used to assign messages that belong together to the same process instance.

http://help.sap.com/saphelp_nwpi71/helpdata/EN/3c/831620a4f1044dba38b370f77835cc/frameset.htm

3.2.6.1  Integration Process

General Rules

- If the integration process implements a specific integration pattern, add the same in the name, such as sync/async bridge, collect, multicast, serialize, etc
- If the integration process processes a specific business object, add the same in the name, e.g., Purchase Order, Invoice, etc

### Syntax

| Integration Process | <Business Object><General Action | Pattern> |
|---------------------|----------------------------------|

### Glossary

<table>
<thead>
<tr>
<th>&lt;Business Object&gt;</th>
<th>Specify the underlying business object</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;General Action&gt;</td>
<td>Specify an action which is carried out by the integration process, e.g., replicate, distribute, etc</td>
</tr>
<tr>
<td>&lt;Pattern&gt;</td>
<td>SyncAsyncBridge</td>
</tr>
</tbody>
</table>

**Example**

- PurchaseOrderCollect
- AccountingPostingsReplicate
- QuotationMerge

### 3.2.6.2 Monitoring Process

A monitoring process is a special kind of integration process that you use as part of Business Activity Monitoring (BAM). You use a monitoring process to monitor the milestones in a business process. The business process can be distributed across multiple applications. When a milestone is reached, the applications each publish events, to which a central monitoring process is subscribed.

http://help.sap.com/saphelp_nwpi71/helpdata/EN/43/d57fb5c9ed3ab3e10000000a422035/frameset.htm

**General Rules**

- In general, same naming conventions apply like for integration processes since monitoring processes usually refer to actual integration processes that are monitored by them

### Syntax

<table>
<thead>
<tr>
<th>Monitoring Process</th>
<th>&lt;Business Object&gt;&lt;General Action&gt;</th>
</tr>
</thead>
</table>

### Glossary

<table>
<thead>
<tr>
<th>&lt;Business Object&gt;</th>
<th>Specify the underlying business object</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;General Action&gt;</td>
<td>Specify an action which is carried out by the process</td>
</tr>
</tbody>
</table>

**Example**

- GoodIssuesDistribute
3.2.6.3 Process Step

General Rules

- Use either business object or container element upon where an operation is carried out
- Use either a more general description of an action or the respective step type such as send, receive, transform, switch, fork, etc.
- We distinguish between process steps that relate to the actual messages (receive, send, transformation, etc) and steps that control the message flow (switch, control, block, fork, loop, wait, etc).
  - For message relevant steps, first specify the object and then its action
  - For process flow relevant steps, first specify the action and then the object together with an appropriate preposition

Syntax

For message relevant steps:

<table>
<thead>
<tr>
<th>Step Name</th>
<th>&lt;Business Object</th>
<th>Container&gt;</th>
<th>&lt;Step Type</th>
<th>Action&gt;</th>
</tr>
</thead>
</table>

For process flow relevant steps:

<table>
<thead>
<tr>
<th>Step Name</th>
<th>&lt;Step Type</th>
<th>Action&gt;</th>
<th>&lt;Preposition&gt;</th>
<th>&lt;Business Object</th>
<th>Container&gt;</th>
</tr>
</thead>
</table>

Glossary

- **<Business Object>** Specify the underlying business object representing business content
- **<Container>** Use the container name if the step runs on a container element
- **<Step type>** Send | Receive | Transform | Switch | Fork | Wait | Loop | <Other>
- **<Action>** Use a general action if step types are not applicable
- **<Preposition>** Use an appropriate preposition for the action

Example

- InvoiceReceive
- WaitForAcknowledgment
- LoopAtPurchaseOrderItems
- QuotationRequestReceive
- MapToQuotationResponses

3.2.6.4 Step Group

You can merge a frequently-used sequence of steps into a step group.


July 2009 15
General Rules

- The name should reflect the purpose and use of the step group
- Naming convention can be defined similar to process steps, however step groups do not necessarily refer to only one variable or only one step, so those options are omitted here

### Syntax

<table>
<thead>
<tr>
<th>Step Group Name</th>
<th>&lt;Business Object&gt;&lt;General Operation&gt;</th>
</tr>
</thead>
</table>

### Glossary

<table>
<thead>
<tr>
<th>&lt;Business Object&gt;</th>
<th>Specify the underlying business object representing business content</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;General Operation&gt;</td>
<td>Specify a general operation, purpose or use of the step group</td>
</tr>
</tbody>
</table>

#### Example

- PuchaseOrderSendWithExceptionHandling
- GenericObjectCollect

3.2.6.5 Container Element

General Rules

- The name of the container element should describe the related message
- Use message type if the container element is based on an interface
- Distinguish between global and local container elements. Former can be used in the complete integration process whereas latter only for a specific process block. Use a respective prefix, e.g., indicate that the container is global whereas for local containers the prefix can be omitted
- Use plurals to denote multiple line elements

### Syntax

| Container Name | <Prefix>?<Message Type | Business Object> |
|----------------|------------------------|

### Glossary

<table>
<thead>
<tr>
<th>&lt;Prefix&gt;</th>
<th>Global</th>
<th>Local</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Message Type&gt;</td>
<td>In case of an interface, use the message type name</td>
<td></td>
</tr>
<tr>
<td>&lt;Business Object&gt;</td>
<td>Specify the underlying business object representing business content</td>
<td></td>
</tr>
</tbody>
</table>

#### Example

- SalesOrderRequests
- GlobalCounter
- GlobalQuotationResponses
3.2.6.6 Correlation

General Rules

- Use a meaningful description for the common key the correlation is based on
- Use business names, not technical names

<table>
<thead>
<tr>
<th>Syntax</th>
<th>&lt;Meaningful identifier&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correlation</td>
<td>Name</td>
</tr>
</tbody>
</table>

**Example**

- PurchaseOrderID
- BusinessTransactionID
- QuotationID

3.2.7 Alert Category

An alert category contains various properties and other specifications that define the alert within that category. The category defines the conditions when a specific alert is sent to whom. During alert category definition, you specify the alert text, expiry time, escalation, and all other conditions related to the sending of this kind of alert.

http://help.sap.com/saphelp\%5Fnwpi71/helpdata/en/43/1b9259fb002be8e10000000a11466f/frameset.htm

**General Rules**

- The name should reflect the purpose and use of the alert category
- The name should reflect the kind of recipient of the alert category

**Example**

- AlertOnSend

3.2.8 Data Type

You use data types to describe the data structures of messages. You can either develop data types that comply with CCTS (Core Component Technical Specification) or create free-style (classical) data types. In case of new developments, SAP recommends that you develop data types that comply with CCTS. You can also reference core data types or aggregated data types from freely-modeled data types.


In the following, we refer to free-style data types only.

**General Rules**

- Inbound SAP RFC and IDoc names (external definitions) must not be renamed or copied to new objects.
- The data type name should rather refer to the related business object (e.g., order, customer, accounting document, etc) than to object methods or events (e.g., create order, update customer, change date, etc). Methods or events are represented by operations.
- Avoid technical names such as record, file, table, etc.
- Keep the data type name short since it will be used in other object names.
- Use plural form for multiple instance data types.

**Syntax**

<table>
<thead>
<tr>
<th>Data Type Name</th>
<th>&lt;Business Object&gt;</th>
</tr>
</thead>
</table>

**Glossary**

| <Business Object> | Specify the underlying business object representing business content |

**Example**

BusinessPartner
Address
Quotation

### 3.2.9 Data Type Enhancement

SAP applications can allow customers to enhance application source code without making modifications to meet customer-specific requirements that are not supported in the SAP standard shipment. The applications can use Business Add-Ins (BAdIs) for this purpose. In the ESR, data types can be enhanced. If the service interfaces referring to the enhanced data types are used for exchanging messages, customers can use these enhancements to transfer additional data with a message.


**General Rules**

- The name should refer to the related data type.
- Optionally, add a description indicating the kind of modification or the area that requires the extension.
### 3.2.10 Message Type

A message type comprises a data type that describes the structure of a message. The message type defines the root element of a message. It describes an instance of a message to be exchanged.


**General Rules**

- The name should refer to the name of its root data type
- Add a category such as notification, request, confirmation, query, response, etc
- Optionally, add an action which should be executed on the business object such as create, change, update, cancel, maintain, modify, delete, read, migrate, acknowledge, etc

#### Syntax

<table>
<thead>
<tr>
<th>Message Type Name</th>
<th>&lt;Business Object&gt;&lt;Action&gt;?&lt;Message Type Category&gt;</th>
</tr>
</thead>
</table>

#### Glossary

<table>
<thead>
<tr>
<th>&lt;Business Object&gt;</th>
<th>Specify the underlying business object representing business content</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Action&gt;</td>
<td>Create</td>
</tr>
<tr>
<td>&lt;Category&gt;</td>
<td>Notification</td>
</tr>
</tbody>
</table>

**Example**

- PurchaseOrderCancellationRequest
- PurchaseOrderChangeRequest
- PurchaseOrderMigrateRequest
- SalesOrderConfirmation
- QuotationResponse
3.2.11 Fault Message Type

Fault message types are used to catch application-specific errors.


General Rules

- The name should refer to the corresponding message type
- Optionally, add an error type to categorize the error

| Syntax | Fault Message Type | <Business Object><Error Type>?

| Glossary |<Business Object>| Specify the underlying business object representing business content
|<Error Type>| Specify an error category, e.g., authorization error, authentication error, error during update, etc

Example

PurchaseOrderAuthorizationError
BankAccountStatementQueryError

3.2.12 Service Interface

With SAP NetWeaver Process Integration 7.1, Service Interfaces have been introduced. They have been derived from Message Interfaces that are used in SAP NetWeaver PI 7.0 and XI 3.0. A Service Interface is a platform and language independent design time representation of a service. It describes one or more operations to be implemented in an application system, and is used in mediated scenarios via the Integration Server or Point-to-Point scenarios via the Web Service Runtime.

An Interface Pattern is a new attribute of a service interface that describes the type of communication that is to be executed on the message.


The Interface Pattern determines the naming convention. In this paper, we focus on the pattern stateless, and stateless XI 3.0 compatible.

General Rules

- A Service Interface name should be based on the business object on which the actions take place
- Add a more generic interaction activity such as Processing, Ordering, Invoicing, etc
- For Interface Pattern stateless XI 3.0 compatible, the name of the Service Interface has to be identical to the operation name
• The name should indicate the direction, i.e., whether it is inbound, outbound, or abstract. Put the direction at the beginning of the name if you like to group your service interfaces according to the same. Otherwise, if you prefer to keep different service interfaces of the same object together in the ESR, specify the direction as a suffix.

### Syntax

**For Interface Pattern stateless:**

<table>
<thead>
<tr>
<th>Service Interface</th>
<th>&lt;Business Object&gt;&lt;Interaction Activity&gt;_&lt;Direction&gt;</th>
</tr>
</thead>
</table>

*Note: For Interface Pattern stateless XI 3.0 compatible, please refer to Service Operation*

### Glossary

<table>
<thead>
<tr>
<th>&lt;Business Object&gt;</th>
<th>Specify the underlying business object representing business content</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Interaction Activity&gt;</td>
<td>Specify a more generic interaction activity</td>
</tr>
<tr>
<td>&lt;Direction&gt;</td>
<td>Inb</td>
</tr>
</tbody>
</table>

#### Example

- PurchaseOrderProcessing_Out
- BankAccountStatementProcessing_In
- SupplierQuery_Abs
- SalesOrderConfirmation_In

### 3.2.13 Service Operation

Service Operations are entities that perform specific tasks on a business object, e.g., creating, updating, or deleting a business object. The operation is related to an action applied to an object (or a method), or an event triggered or received.

#### General Rules

- A Service Operation name should be based on the business object on which the operation is carried out.
- Specify an action either requested by the operation (in outbound case) or offered by the operation (in inbound case) such as Create, Change, Update, Cancel, Maintain, Modify, Delete, List, Save, etc.
- Optionally, add a transaction activity such as Request, Query, Notify, Confirm, Respond, etc.
- A service operation name is unique within a service interface.
- Each service operation within the same service interface must refer to a different message type.
- The name should indicate the mode of the operation, i.e., whether it is a synchronous or an asynchronous communication.
- As mentioned above, For stateless XI 3.0 compatible Interface Pattern, the name of the service interface and operation must be identical.
- For stateless XI 3.0 compatible Interface Pattern, the name should indicate the direction.
### Syntax

#### For Interface Pattern stateless:

| Operation Name | <Business Object><Action><Transaction Activity>_<Mode> |

#### For Interface Pattern stateless XI 3.0 compatible:

| Operation Name | <Business Object><Action><Transaction Activity>_<Mode>_<Direction> |

### Glossary

| <Business Object> | Specify the underlying business object representing business content |
| <Action> | Create | Change | Update | Cancel | Maintain | Modify | Delete | Read | <Other> |
| <Transaction Activity> | Request | Query | Notification | Confirmation | Response | <other> |
| <Mode> | Sync | Async |
| <Direction> | Inb | Out | Abs |

#### Example

- SalesOrderCreateRequest_Async
- BusinessPartnerRead_Sync
- QuotationCreateResponse_Async

- SalesOrderCreateRequest_Async_Out
- BusinessPartnerRead_Sync_In
- SalesOrderCreateConfirmation_Async_Abs

### 3.2.14 External Definition

An external definition enables you to import a local WSDL, XSD, or DTD file to the ES Repository.

http://help.sap.com/saphelp_nwpi71/helpdata/EN/26/9e97b0f525d743882936c2d6f375c7/frameset.htm

### General Rules

- The name should reflect its message type or related business object

### Syntax

| External Definition | <Business Object | Message Type> |

### Glossary

| <Business Object> | Specify the underlying business object representing business content |
| <Message Type> | Use the message type as specified in the XSD for instance |
### 3.2.15 Context Object

As an alternative to xpath expressions, context objects enable you to access the content of messages during configuration in the Integration Directory.

http://help.sap.com/saphelp_nwpi71/helpdata/EN/d6/e44fcf98baa24a9686a7643a33f26f/frameset.htm

**General Rules**

- Use business names, no technical names
- The name should reflect the target element of the xpath expression

**Syntax**

<table>
<thead>
<tr>
<th>Context Object</th>
<th>&lt;Meaningful Description</th>
<th>Respective Element in xpath Expression&gt;</th>
</tr>
</thead>
</table>

**Example**

- PartnerNumber
- CountryOfShipment

### 3.2.16 Message Mapping

A message mapping refers to a mapping of messages which is supported by the graphical mapping editor in ESR. The editor enables you to design a structure mapping between any two XML structures.


**General Rules**

- The name should be based on the source and target message types
- If possible, try to abbreviate the name of the message type
- For message mappings including IDocs, the IDoc extension type should be included
- For IDocs, replace the dot by an underscore
- Source and target messages can be separated by a delimiter such as ‘to’ or ‘2’
- In case of multiple source or target messages, separate the same by a delimiter such as ‘and’
- Optionally, in case of multiple source or target messages, you can add a suffix such as ‘multi’ to indicate that it is about a multi message mapping. This is especially relevant when you map one
source message type containing several source messages to one target message type containing several messages where it is not obvious that it refers to a multi message mapping

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Message Mapping</th>
<th>&lt;Source Message Type&gt;(and_&lt;Further Source Message Type&gt;)<em>to</em>&lt;Target Message Type&gt;(and_&lt;Further Target Message Type&gt;)*(multi)?</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Glossary</th>
<th>&lt;Message Type&gt;</th>
<th>Specify the message type name</th>
</tr>
</thead>
</table>

**Example**
- MaterialMasterCreateRequest_to_MATMAS_MATMAS03
- PurchaseOrderRequest_and_InvoiceResponse_to_InvoiceConfirmation
- InvoiceNotification_to_InvoiceNotification_multi

### 3.2.17 Mapping Template

Just like you can reuse data types in different message types, you can also save parts of message mappings as Mapping Templates and reuse them elsewhere.


**General Rules**
- The name should be based on the source and target element/node of the message type
- Use a suffix indicating that it refers to a template such as Template

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Mapping Template</th>
<th>&lt;Source Element&gt;<em>to</em>&lt;Target Element&gt;_Template</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Glossary</th>
<th>&lt;Element&gt;</th>
<th>Specify the element or node name</th>
</tr>
</thead>
</table>

**Example**
- VendorAddress_to_BusinessPartnerAddress_Template
- OrderItem_to_OrderDetails_Template

### 3.2.18 Operation Mapping

Operation mappings register your mapping program for a pair of operations in the ESR.

**General Rules**

- The name should be based on the source and target service operations
- Optionally, specify service interfaces
- If possible, try to abbreviate the name of the operations and service interfaces consistently
- Source and target terms can be separated by a delimiter such as ‘to’ or ‘2’
- In case of multiple source or target messages, separate the same by a delimiter such as ‘and’
- Optionally, in case of multiple source or target messages, you can add a suffix such as ‘multi’
- Omit prefixes indicating direction and communication mode

**Syntax**

<table>
<thead>
<tr>
<th>Operation Mapping</th>
</tr>
</thead>
</table>

**Glossary**

<table>
<thead>
<tr>
<th>&lt;Service Interface&gt;</th>
<th>Specify the service interface name omitting direction</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Service Operation&gt;</td>
<td>Specify the service operation name omitting mode</td>
</tr>
</tbody>
</table>

**Example**

- PurchaseOrderCreateRequest_to_SalesOrderCreateRequest
- MaterialMasterCreateRequest_to_MATMAS_MATMAS03
- QuotationResponse_to_QuotationResponses_multi

### 3.2.19 Imported Archive

In order to reuse existing mapping programs, you can import XSLT and Java mappings as archives in the ESR.


**General Rules**

- For archives that contain one single mapping only, specify source and target message type
- For archives that contain several mappings, specify related application or a meaningful name to categorize the mappings
- Specify the Transformation Technology such as Java or XSLT. For archives that contain several mappings, this is optional since the archive can contain mappings of different technologies
### Syntax

**For archives containing one single mapping only:**

<table>
<thead>
<tr>
<th>Archive Name</th>
<th>&lt;Source Message Type&gt;<em>to</em>&lt;Target Message Type&gt;_&lt;Technology&gt;</th>
</tr>
</thead>
</table>

**For archives containing several mappings:**

| Archive Name | <Application | Mapping Category>_(<Technology>)? |
|--------------|--------------|

### Glossary

<table>
<thead>
<tr>
<th>&lt;Message Type&gt;</th>
<th>Specify the message type of the source and target message</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Technology&gt;</td>
<td>XSLT</td>
</tr>
<tr>
<td>&lt;Application&gt;</td>
<td>Specify an application the archives are related to</td>
</tr>
<tr>
<td>&lt;Mapping Category&gt;</td>
<td>Use a group name to categorize your archives</td>
</tr>
</tbody>
</table>

#### Example

- SalesOrderItemRequest_to_SalesOrderRequest_XSLT
- ContractManagement_Java
- PurchaseOrderProcessing_XSLT

### 3.2.20 Function Library

A function library enables mapping developers to use user-defined functions across message mappings. You can use user-defined functions from a function library in message mappings and in mapping templates.


#### General Rules

- Function libraries group together various user defined functions. So, its name should reflect a category of functions

#### Syntax

<table>
<thead>
<tr>
<th>Library Name</th>
<th>&lt;User Defined Function Category&gt;</th>
</tr>
</thead>
</table>

#### Glossary

| <Function Category> | Specify a group name to categorize your user defined functions |

#### Example

- CodePageConversion
- AccessPayloadHeader
### 3.2.21 Configurable Parameter

A configurable parameter is a variable that you can use during design time in ESR, whereas its value is specified during configuration time in the Integration Directory. This increases the reusability of the respective objects since you do not need to change the same if you need to alter the value. Parameters can be used either in integration processes or in message mappings.

You can create multiple configurations for one process and define different values for a configurable parameter in each one.

http://help.sap.com/saphelp_nwpi71/helpdata/en/44/1f1a5c932d0d19e10000000a114a6b/frameset.htm

You can use parameters to transfer values to a mapping program at configuration time thereby increasing the number of possible applications for a mapping program.


**General Rules**

- Use a meaningful description
- Specify the parameter type such as Agent, Channel, Simple type

<table>
<thead>
<tr>
<th>Syntax</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter Name</td>
<td><code>&lt;Description&gt;_&lt;Type&gt;</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Glossary</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;Type&gt;</code></td>
<td>Agent</td>
</tr>
<tr>
<td><code>&lt;Simple Type&gt;</code></td>
<td>Text</td>
</tr>
</tbody>
</table>

**Example**

- DBLookup_Channel
- Administrator_Agent
- ErrorDescription_Text

### 3.2.22 Communication Channel Template

You can use the communication channel template as a template to define a communication channel at configuration time.


**General Rules**

- Naming convention should be similar to the one for communication channel
- The name should reflect the adapter type such as IDoc, RFC, File, JDBC, JMS, Proxy, SOAP, Http, Mail, RNIF, etc
- The name should reflect the direction, i.e., Sender or Receiver
If the channel is used to transport a specific business object or message type, add the same to the channel name.

Optionally, add a meaningful description of its usage.

Use a suffix indicating that it refers to a template such as Template.

**Syntax**

| Channel Template | <Adapter Type><Direction><Object | Description>_<Template |

**Glossary**

- `<Adapter Type>`: IDoc | RFC | File | JDBC | JMS | Proxy | SOAP | Http | Mail | RNIF | <Other>
- `<Direction>`: Sender | Receiver
- `<Object>`: Specify an underlying business object or message type
- `<Description>`: Describe the usage of the channel

**Example**

- JDBCReceiverMDMDataBase_Template
- SOAPS SenderSAML_Template

### 3.2.23 Adapter Metadata

With adapter metadata you can define configuration data needed for a certain type of adapter at design time. Adapter metadata defines the part of a channel that is specific to the adapter type.


**General Rules**

- Add the adapter metadata to your own software component
- The namespace below should reflect that it contains adapter metadata
- The adapter metadata name should refer to the name of the adapter itself or the adapter type

**Syntax**

| Adapter Metadata | <Adapter Name | Adapter Type>

**Glossary**

- `<Adapter Name>`: Specify the name of your own developed adapter
- `<Adapter Type>`: Adapter type may refer to a communication protocol

**Example**

- SFTP
- Swift
3.3 Integration Directory Objects

In the Integration Directory, you configure your business scenarios, i.e., you specify the parties and systems that exchange messages, routing rules, and collaboration agreements.

http://help.sap.com/saphelp%5Fnwpi71/helpdata/en/b9/64663c7a6c2545ae292fb26d8c33c4/frameset.htm

3.3.1 Configuration Scenario

Using configuration scenarios, you can group configuration objects to structure the content of the Integration Directory more clearly.

http://help.sap.com/saphelp%5Fnwpi71/helpdata/en/06/18ba91813cee4c8eacea4fcf0b46c6/frameset.htm

General Rules

• It is recommended to create the Configuration Scenario based on a Process Integration Scenario from ESR via the Model Configurator

• The name of the Configuration Scenario should refer to the Process Integration Scenario

3.3.2 Folder

You can define folders for grouping your objects. You can create any number of sub folders for each folder, see also above.

General Rules

• For grouping business scenarios rather use Configuration Scenarios

• Grouping according to object type is supported in the Integration Directory anyway

• Use folders to group multiple scenarios belonging to the same business partner or company

• Use folders to restrict authority on specific objects

3.3.3 Party

A communication party (party for short) represents a larger unit, which is involved in a collaborative process. Using a communication party, you generally address a company within a business-to-business process (B2B).

http://help.sap.com/saphelp%5Fnwpi71/helpdata/en/45/b5cb4677d547f3e10000000a114a6b/frameset.htm

General Rules

• Use parties for communicating with external partners

• Specify a party for your own company. This is required in a B2B communication

• Rather use a descriptive name for your partner than a technical name such as the customer number
Optionally, add a location suffix (region or country) when you partner with several external partners of the same company within different locations. On the other hand, if you have a large number of external partners, the geographical area of the external partner may be included as a prefix to enable better sorting in the Integration Directory.

The name has to be unique within the Integration Directory.

Parties should be independent of the stage. So, the stage information should not be added to the name.

### Syntax

<table>
<thead>
<tr>
<th>For several partners of same company in different locations:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Party Name</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>For a large number of external partners:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Party Name</strong></td>
</tr>
</tbody>
</table>

### Glossary

<table>
<thead>
<tr>
<th>&lt;Description&gt;</th>
<th>Specify the name of your partner</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Location&gt;</td>
<td>Specify a geographical area</td>
</tr>
</tbody>
</table>

### Example

- MyPartnerUS
- MyPartnerEMEA
- Buyer
- Seller

### 3.3.4 Communication Component

A communication component enables you to address a sender or receiver of messages. This can be a business system, an integration process, or a business component.

A business system is maintained in the SLD, see above.

### 3.3.4.1 Business Component

A business component is usually used in business-to-business processes, in combination with a party for the communication with external partners. The term Business Component has been introduced in SAP NetWeaver PI 7.1. In previous releases, this was called Business Service.

[http://help.sap.com/saphelp%5Fnwpi71/helpdata/en/7d/6b82cd0d1aef48ab5953524c9cc5b2/frameset.htm](http://help.sap.com/saphelp%5Fnwpi71/helpdata/en/7d/6b82cd0d1aef48ab5953524c9cc5b2/frameset.htm)

### General Rules

- The name should reflect the respective transaction or service. This itself can be related to a business object.
• Optionally, the name can contain the specific role within the external communication such as Buyer, Seller, etc
• Business Components can also be used in case of technical adapters such as file, JDBC, etc

<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>In case of B2B:</strong></td>
</tr>
<tr>
<td>Component Name</td>
</tr>
<tr>
<td><strong>In case of technical adapters:</strong></td>
</tr>
<tr>
<td>Component Name</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Glossary</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Service&gt; Specify a business function or transaction</td>
</tr>
<tr>
<td>&lt;Business Object&gt; Specify the underlying business object representing business content</td>
</tr>
<tr>
<td>&lt;Role&gt; Buyer</td>
</tr>
<tr>
<td>&lt;Adapter Type&gt; IDoc</td>
</tr>
</tbody>
</table>

**Example**
- PurchasingSeller
- PurchasingBuyer
- MasterDataReplicationJDBC

**Note**
For RNIF and CIDX communication standard, you have to meet specific naming conventions as described in the SAP Help Portal

### 3.3.4.2 Integration Process Service
The Integration Process Service is a runtime representative of an integration process

**General Rules**
- Use same name like in ESR

**Example**
- QuotationCollect
3.3.5 Communication Channel

You define the details for the inbound or outbound processing of a message in a communication channel. Here, you configure adapters in particular.

http://help.sap.com/saphelp%5Fnwpi71/helpdata/en/2b/d5653fd1d3b81ae10000000a114084/frameset.htm

General Rules

- The name should reflect the adapter type such as IDoc, RFC, File, JDBC, JMS, Proxy, SOAP, Http, Mail, RNIF, etc
- The name should reflect the direction, i.e., Sender or Receiver
- If the channel is used to transport a specific business object or message type, add the same to the channel name
- Optionally, add a meaningful description of its usage
- Use Ack prefix for IDoc acknowledgements

| Syntax | Channel Name (Ack_)?<Adapter Type><Direction><Object | Description>? |
|--------|---------------------------------------------------------------|
| Glossary | <Adapter Type> | IDoc | RFC | File | JDBC | JMS | Proxy | SOAP | Http | Mail | RNIF | <Other> |
| <Direction> | Sender | Receiver |
| <Object> | Specify an underlying business object or message type |
| <Description> | Describe the usage of the channel |

Example

IDocReceiver
FileSenderProduct
SOAPReceiverQuotation
MailReceiverQuotation

3.3.6 Value Mapping

You use the value mapping function to map different representations of an object to each other.

http://help.sap.com/saphelp%5Fnwpi71/helpdata/en/13/ba20dd7beb14438bc7b04b5b6ca300/frameset.htm

General Rules

- A Group displays the different representations of an object
- In PI, an Agency can represent one of the following:
- An organization or company that produces and manages an identification scheme, e.g., Dun & Bradstreet which provides D-U-N-S number to identify companies
- A technical unit or a system within which an object is identified uniquely by an identifier or value, e.g., a business system
  - A Scheme represents a frame of reference within which it is possible to identify an object uniquely, e.g., the object customer is an identification scheme, within which a customer is identified uniquely by a value
  - A value identifies the object within the given identification scheme

### Syntax

<table>
<thead>
<tr>
<th>Group</th>
<th>&lt;Business Object&gt;&lt;Business Object Instance&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business Object</td>
<td>&lt;Agency Name</td>
</tr>
<tr>
<td>Scheme</td>
<td>&lt;XML Field Element Name</td>
</tr>
</tbody>
</table>

### Glossary

<table>
<thead>
<tr>
<th>Business Object</th>
<th>Specify the underlying business object representing business content</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object Instance</td>
<td>Specify a characteristic value</td>
</tr>
<tr>
<td>Agency Name</td>
<td>An agency is an organization or company that produces and manages an identification scheme, e.g., Dun &amp; Bradstreet</td>
</tr>
<tr>
<td>Business System</td>
<td>Specify a key representing your business system</td>
</tr>
<tr>
<td>Element</td>
<td>Specify an element representing business content</td>
</tr>
</tbody>
</table>

### Example

For the business object Country, different country codes are used within your several backend systems.

Each country refers to a group, e.g., CountryGermany, CountryUS, etc.

The agencies refer to the backend systems, e.g., ERP, CRM, SRM.

The scheme is the business object Country.

Within the group CountryUS, you maintain following values for instance:

- Agency ERP – Scheme Country: US
- Agency CRM – Scheme Country: USA
4. References

- SAP Exchange Infrastructure 3.0: Best Practices for Naming Conventions  

- PI Best Practices: Modeling  

- Naming Conventions for XI Content Packages Used for Certification  

- SAP Partners Developing Content using XI3.0  

- SOA 300 training course: Enterprise SOA design, modeling, and governance