SAP NetWeaver Process Integration 7.1
Mapping Enhancements

SAP NetWeaver Regional Implementation Group
SAP NetWeaver Product Management
December 2007
Benefits

After reading the document, you will be able to:

- Know the new features of the graphical mapping tool in PI 7.1
- Work with variables
- Parameterize the mappings during configuration
- Use the function libraries for User Defined Functions
- Create database lookups with help of the graphical support
1. Miscellaneous Features
2. Graphical Variables
3. Parameterized Mappings
4. Development of User-Defined Functions
5. Functional Libraries
6. Graphical Support for Lookups
# Miscellaneous (I)

<table>
<thead>
<tr>
<th>Feature</th>
<th>Details</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface Mappings are now called Operation Mappings</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Choice between SAP XML toolkit and jdk 1.5 toolkit</td>
<td>For compatibility reasons SAP XML toolkit is still available&lt;br&gt;Only relevant for Java and XSLT mappings</td>
<td></td>
</tr>
<tr>
<td>Set failure behavior in functions FixValues and Value mapping</td>
<td>Functions may:&lt;br&gt;- Return the initial value&lt;br&gt;- Return a default&lt;br&gt;- Throw an exception</td>
<td></td>
</tr>
<tr>
<td>Output of fields and functions can be used for multiple target fields</td>
<td>reuse of complex combinations of functions&lt;br&gt;Simplify refactoring of mappings with repeating combinations of functions&lt;br&gt;Better runtime performance</td>
<td></td>
</tr>
</tbody>
</table>

- Functions may:
  - Return the initial value
  - Return a default
  - Throw an exception

- Output of fields and functions can be used for multiple target fields

- Set failure behavior in functions FixValues and Value mapping

- Choice between SAP XML toolkit and jdk 1.5 toolkit for compatibility reasons SAP XML toolkit is still available. Only relevant for Java and XSLT mappings.

- Interface Mappings are now called Operation Mappings.

- Functions may:
  - Return the initial value
  - Return a default
  - Throw an exception

- Output of fields and functions can be used for multiple target fields.

- Set failure behavior in functions FixValues and Value mapping.
### Miscellaneous (II)

<table>
<thead>
<tr>
<th>Feature</th>
<th>Details</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Refactoring capabilities of Message Mappings</td>
<td>Tool support to adjust mappings after structure changes (e.g., changes of the root node name, changes of the namespace in Message Types)</td>
<td>Tool support to adjust mappings after structure changes (e.g., changes of the root node name, changes of the namespace in Message Types)</td>
</tr>
<tr>
<td></td>
<td>Applies to changes of the source and target structure</td>
<td>Applies to changes of the source and target structure</td>
</tr>
<tr>
<td></td>
<td>„Map“ original node to new node</td>
<td>„Map“ original node to new node</td>
</tr>
</tbody>
</table>

![Diagram showing refactoring capabilities of Message Mappings](image)
Graphical Variables

- Store intermediate mapping results (including context information)
- Reuse in multiple target field mappings
- Enhanced runtime performance

1. Right-click on element of your target structure and select Add Variable
2. Enter a name for the variable
3. Define a mapping to set the intermediate variable
4. The variable can be used in all subsequent mappings. It will not be written to the final document
Copy entire XML Subtree

- Complete copy of XML subtrees
- Target structure not mandatory
- Occurrence of the target node has to correspond with the source node

1. Set up a target field mapping that links the relevant nodes
2. Right-click on the node of your source structure and select Return as XML
3. The entire node (including the underlying XML branch) will be copied to the target node. The target node will take over the occurrence of the source node.
Parameterized Mappings

- Function Properties and Parameters in User-defined Functions (denoted by the icon) can be set in Interface Determinations and Transformation Steps of Integration Processes and Monitoring Processes (import parameters)
- Payload information can be written to simple typed containers of an Integration Process and Monitoring Processes (export parameters)
- Provide simple typed constants or channels (for lookup functions)
- Parameters can be used in
  - Message Mappings
  - Java Mappings
  - XSLT Mappings
- Examples:
  - Reuse of mappings in multiple Interface Determinations with varying parameterizations
  - Set Channel for lookups at configuration time
  - Transfer content of simple typed containers to the message payload
Parameterized Mappings – Binding

**Interface Determination / Transformation**

OMpar1 = “value”

OMpar2 = Channel

**OP Mapping 1**

OMpar1

OMpar2

**Binding**

**Mapping Program**

MPpar1

MPpar2

IB Constant

CONST (Constant)

BIC Leo
# Parameterized Mappings in Message Mappings – Doing

1. Add list of parameters in the Signature tab of your Message Mapping
2. Set a Name, Category, Type and the direction for the relevant parameter

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Add list of parameters in the Signature tab of your Message Mapping.</td>
</tr>
<tr>
<td>2</td>
<td>Set a Name, Category, Type and the direction for the relevant parameter.</td>
</tr>
<tr>
<td>3</td>
<td>Double click on the function which provides a property / parameter.</td>
</tr>
<tr>
<td>4</td>
<td>Select the parameter defined in step 2.</td>
</tr>
</tbody>
</table>

![Diagram](image-url)
Parameterized Mappings in Operation Mappings – Doing

1. Select “Parameters” in your Operation Mapping
2. Set a Name, Category, Type and the direction for the relevant parameter
3. Select „Binding“ to set the dependencies between Message Mapping and Operation Mapping parameters
4. Set the binding
Parameterized Mappings in Interface Determination – Doing

1. Create your Interface Determination and add the relevant Operations Mapping
2. Assign constants or channels to your parameter
Parameterized Mappings in Integration Process – Doing

1. Create a Transformation Step in your Integration Process or Monitoring Process.
2. Assign constant values or simple typed containers to import parameters
3. Assign simple typed containers to export parameters

![Diagram of Parameterized Mappings]

- **Import**: Parameter name is `OMPart1_In`, value is `YnValue`
- **Export**: Parameter name is `OMPart1_out`, value is `SimpleContainer`
New Development of User-Defined Functions (UDF)

To create or change a UDF, either:

1. Select „Create New Function“

or

2. Select the „Functions“ tab
UDF Development - Using “Create New Function“

1. Set a (technical) Name and a (functional) Title of your UDF
2. Set the Execution Type for the required context handling
3. Specify a Category
4. Define Arguments (input variables) and Parameters (function properties). For Parameters you may set a Title in addition
5. Select „Create Function“
6. Add import statements (if necessary)
7. Develop your coding
1. Specify a (technical) **Name**
2. Specify a (functional) **Title**, the **Execution Type** for context handling and a **Category**
3. Define **Arguments** (input variables) and **Parameters** (function properties). For Parameters you may set a **Title** in addition
4. Adapt the **import** statement (if necessary)
5. Develop your coding
Use-cases
- Reuse of user-defined functions
- Enhanced portability of user-defined functions

To set up a Library:
1. Provide Function Library name
2. Set Import Instruction / add reference to Imported Archive
3. Add a new (technical) function name and provide (functional) Title (may differ)
4. Specify a function Category
5. Set signature variables (Arguments and Parameters)
6. Develop Java code
7. Use Import/Export buttons to transfer Java source code from/to Java IDE (e.g. the NWDS)
Use a Function Library

1. Add the relevant Function Library to the library list. You may select all libraries of the actual and underlying SCVs.
2. The Categories of the Function Libraries show up in the drop down menu.
3. Select the relevant Category and the relevant Function.
Function Library Entities

- Class Name
- Package Name
- Function Name

are technical (Java) constructs
### Graphical Support for RFC Lookups

1. Select function RFC Lookup from category Conversions. Double click on function.

2. Select a Communication Channel from your parameter list (see Parameterized Mappings for details)

3. Select the relevant RFC structure (has to be imported beforehand)

4. Specify request parameters for the RFC

5. Select the required response parameters of the RFC

6. Finish your target field mapping
### Graphical Support for JDBC Lookups

1. Select function **JDBC Lookup** from category Conversions. Double click on function.

2. Select a **Communication Channel** from your parameter list.

3. Select the relevant table structure (has to be defined as External Definition beforehand).

4. Specify select parameters for the SQL statement.

5. Specify the resultset of your SQL statement.

6. Finish your target field mapping.

---

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Select function <strong>JDBC Lookup</strong> from category Conversions. Double click on function.</td>
</tr>
<tr>
<td>2</td>
<td>Select a <strong>Communication Channel</strong> from your parameter list.</td>
</tr>
<tr>
<td>3</td>
<td>Select the relevant table structure (has to be defined as External Definition beforehand).</td>
</tr>
<tr>
<td>4</td>
<td>Specify select parameters for the SQL statement.</td>
</tr>
<tr>
<td>5</td>
<td>Specify the resultset of your SQL statement.</td>
</tr>
<tr>
<td>6</td>
<td>Finish your target field mapping.</td>
</tr>
</tbody>
</table>
Summary

- You have learned the new features of the graphical mapping tool in PI 7.1
- You can work with variables
- You can parameterize the mappings during configuration
- You can use the function libraries for User Defined Functions
- You can create database lookups with help of the graphical support