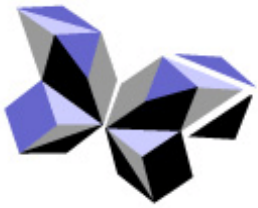

Digital Business Ecosystem

Contract n° 507953

WP 18: Socio-economic perspectives of sustainability and dynamic specification of behaviour in Digital Business Ecosystems

D18.5: Implementation of the SM Editor

Information Society Technologies

Project funded by the European Community under the “Information Society Technology” Programme

Contract Number: 507953

Project Acronym: DBE

Title: Digital Business Ecosystem

Deliverable N°: D18.5

Due dates: 07/2006

Delivery Date: 08/2006

Short Description: This Deliverable describes the SM Editor, a component of the DBE Studio, introduced in the project to enable users to easily create, edit and publish Service Manifests (SM).

Author: Soluta.net

Partners contributed: Soluta.net

Made available to: Public

Versioning		
Version	**Date**	**Author, Organisation**
00.01	31/07/06	Giulio Montanari, Matteo Spagnolo – Soluta.net
01.00	30/08/06	Giulio Montanari, Matteo Spagnolo – Soluta.net

Quality check

1st Internal Reviewer: John Kennedy – Intel

2nd Internal Reviewer: Dominik Dahlem - TCD

Table of Contents

Table of Figures

Index of Tables

1. Executive Summary

This document describes the Service Manifest Editor of the DBE project. It includes both a description of the characteristics and functionality offered by the Service Manifest Editor and a tutorial explaining, in detail, how to use this functionality. Derived from the Service Manifest Creator, this editor is an Eclipse plug-in which will allow DBE users to create, edit and deploy, in a user-friendly manner, Service Manifests. A Service Manifest is the complete representation of a DBE service and it contains all the models and information required to describe, retrieve, and execute a service.

Any DBE service must be represented through a Service Manifest. Based on the MDA approach, the Service Manifest Editor allows the creation of Service Manifests implementing two different service viewpoints: the CIM[1] viewpoint - which describes the service in non computational terms - and the PIM[2] viewpoint - which describes the methods for code development without referring to any technology platform.

The new Service Manifest Editor release described in this deliverable solves several issues of the previous version and addresses the new characteristics implemented in the latest version of the Service Manifest model. As a result the name of the tool has been changed from Service Manifest Creator to Service Manifest Editor as it is now able not only to create services but also to modify them. Other relevant changes are the removal of the SDNA concept, the possibility to manage the Interaction Form Service, the restyling of the graphical user interfaces and a better SM search mechanism which introduces the Query Formulator and Semantic Discovery Tool in the Service Manifest Editor [QF SDT].

1 Computational Independent Model (CIM) [MDA].
2 Platform Independent Model (PIM) [MDA].

2.Introduction

This document describes the Service Manifest Editor Version 0.2.1 (from now on referred to as the “SM Editor”) as a result of the task C20 of the project, introduced with the purpose of allowing DBE users to create and edit, in a user-friendly manner, Service Manifests.
Based on the detailed specification of the Service Manifest (see [SMFD]), this document is concerned with describing the functionality of the SM Editor and presenting a comprehensive User Guide. Moreover, the SM Editor is installed with the DBE Studio distribution[3] which is described in http://dbestudio.sourceforge.net/wiki/index.php/Install and therefore does not require an explicit section in this document.

3 The DBE Studio is available at http://dbestudio.sourceforge.net

3.SM Editor functionality

The SM Editor functionality offered to DBE users can be grouped into three phases, as follows:

- ***Creation***
  - it creates completely new Service Manifests from scratch
  - it uses exiting Service Manifests as a base for creating other Service Manifests
  - it stores Service Manifests locally in the File System

- ***Modification***
  - it modifies Service Manifests saved locally
  - it modifies Service Manifests saved remotely

- ***Publication***
  - it distributes Service Manifests in the Semantic Registry (SR)[4]

The Service Manifest created may represent either Distributed Business Services or Interaction Form Services. These two types of services are created using different functionality offered by the SM Editor.
Essentially, the SM Editor is a tool that enables service "assembly". The Service Manifest is a container of models and data, therefore its main task consists of enabling the user to choose among the models contained in the Knowledge Base (KB) and File System and insert them into the Service Manifest.

3.1 Creation

In this phase a new service can be created in two different ways: by creating the new Service Manifest from scratch or by using attributes inherited from another Service Manifest as a base for creating a new service.

3.2 Modification

This phase allows changes to the service parameters. It must be specified that changes in some parameters will affect the nature of the related services and thus this will force users to create a new Service Manifest not just a new version of the same SM. For this reason two ways of modification have been defined, one forcing the creation of a new service and one where a new service is not mandatory. A modification which requires the creation of a new Service Manifest can be considered itself as the creation of a new Service Manifest from an exiting one [see chapter 4.2 - How to create a Service Manifest from an existing one].

4 Further information on how the SR stores the SM can be found in [SMFD]

3.3 Publication

This phase includes the functionality that makes the service visible to users. Once the SM is published, it is saved into the Semantic Registry and is hence accessible to users.

4.SM Editor Tutorial

This section provides a detailed tutorial to explain how to use the functionality offered by the SM Editor.

4.1 How to modify a Service Manifest

The SM Editor allows the modification of Service Manifests imported from both the File System and the Model repository (KB):

4.1.1 Modification from File System

From the “Explorer Package” the user must double-click the Service Manifest to be modified and automatically a new SM Editor will be launched with the Service Manifest ready for modifications.

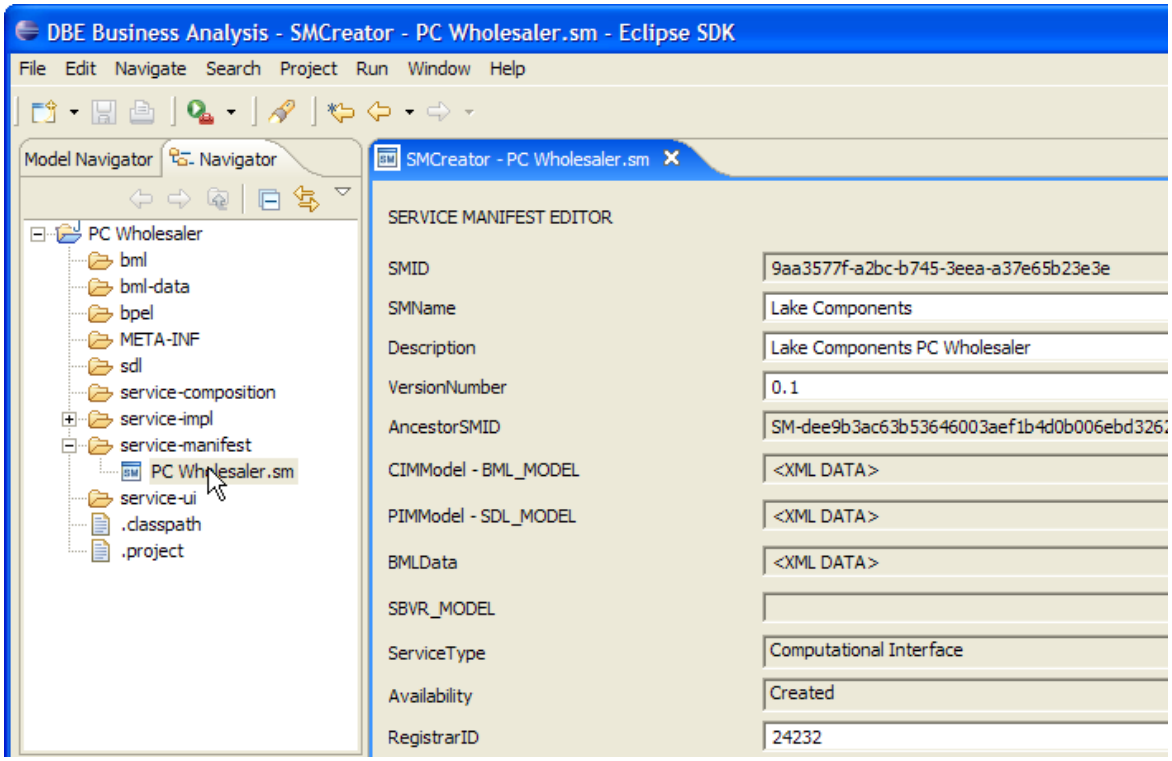

Figure 1: Modify Service Manifest from the Local File System

Once modified by selecting **File -> Save** from the toolbar menu, the Service Manifest will be overwritten with the new settings.

4.1.2 Modification from the Model Repository

To modify a Service Manifest published in the Service Registry the user must select **Windows -> Show View -> Other** and visualize the list of views available, as shown in Figure 2: Open View.

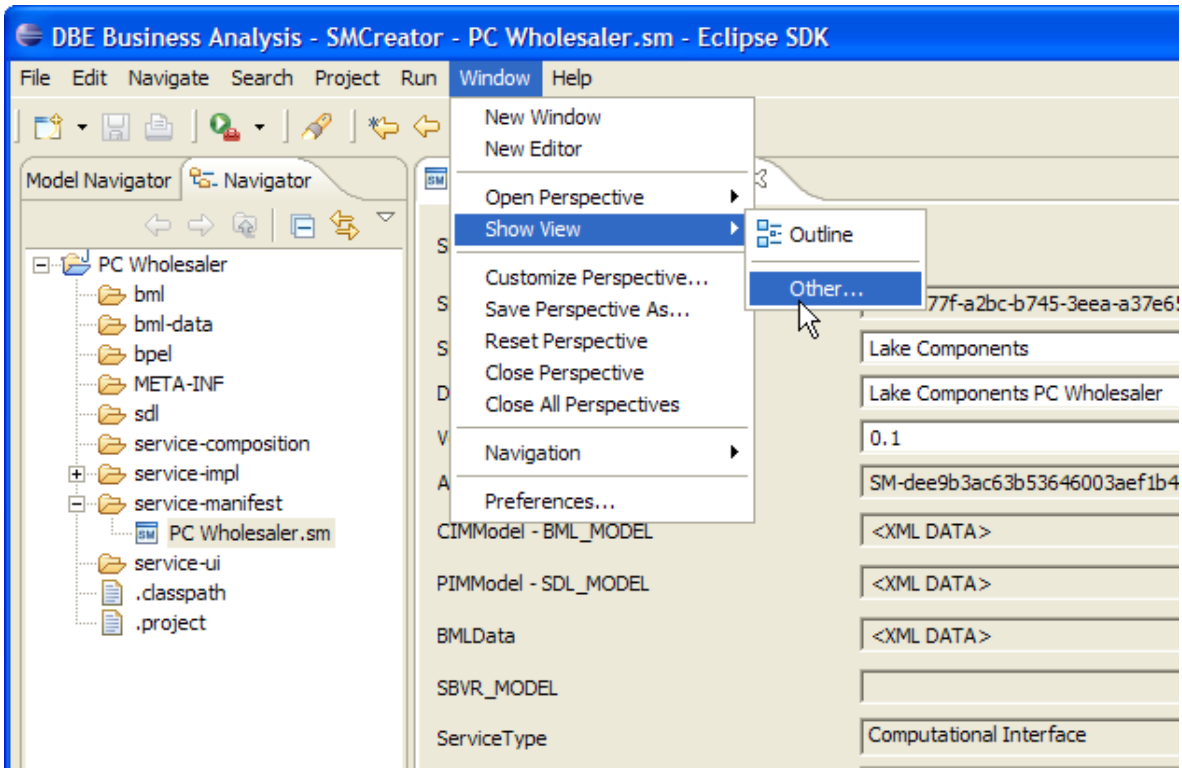

Figure 2: Open View

Then the user has to select **DBE -> Keyword Search** to open the view for managing the remote Service Manifests. In the "Keyword Search" view, the user has to set the search string and click on "Search". A list of Service Manifests contained in the KB will be presented, as shown in Figure 3: Keyword Search.

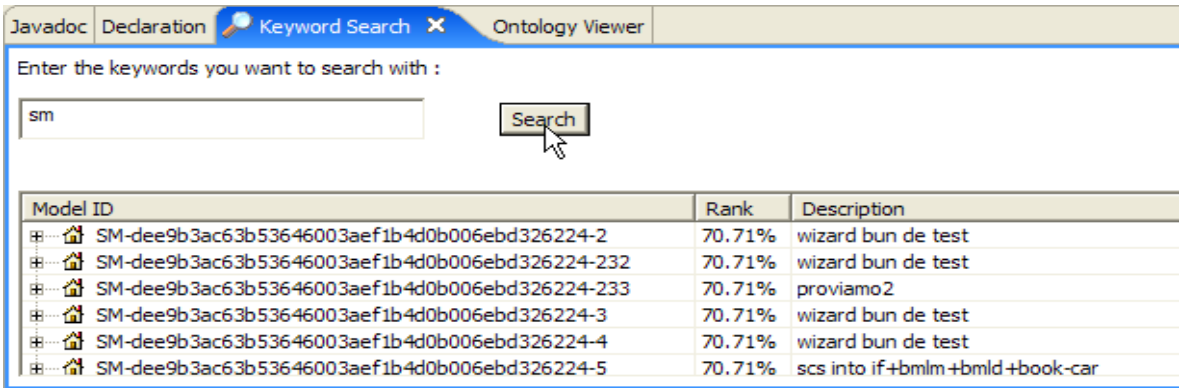

Figure 3: Keyword Search

By right clicking on the desired model, a context-menu with several options will be displayed. The user must select “Modify SM”. At the request to save the Service Manifest locally, the user must specify the name and the desired path; and the Service Manifest will be automatically saved and loaded into the SM Editor. Once the modifications are executed, by pressing the button “Save to SR”, the modified Service Manifest will overwrite the previous one. In case of modifications that affect the nature of the service, it will not be possible to overwrite the existing Service Manifest, but a new Service Manifest will be saved.
To use the view **Keyword Search**, the connection with the KB must be properly configured; see the discovery tool manual for details [QF SDT]..
Furthermore, to execute the Service Manifest search, from the toolbar menu, the user must open **Windows -> Preference** and select **DBEstudio -> Semantic discovery tool** and then select “Search for service”.
To use the feature “Save to SR” the connection with the KB must be properly configured, further details on that can be found in chapter 4.4 - SM Editor settings.

4.2 How to create a Service Manifest from an existing one

To create a new Service Manifest from an existing one the user has to select **Windows -> Show View -> Other** and visualize the list of available views, as shown in Figure 2: Open View.

To open the view to remote Service Manifest management, the user must select **DBE -> Keyword Search**, set the search string and click on “Search” as showed in Figure 3: Keyword Search. A list of Services Manifests contained in the KB will be presented. By right clicking on the desired model, a context-menu with several options will be displayed. The user has to select “Copy SM”. At the request to save the Service Manifest locally, the user must set the name and the desired path; the Service Manifest will be automatically saved and loaded into the SM Editor. Once the modifications are executed, by pressing the button “Save to SR” the new service will be deployed.

To use the view **Keyword Search**, the connection with the KB must be properly configured; see the discovery tool manual for details.

Furthermore, to execute the Service Manifest search, from the toolbar menu, the user must open **Windows -> Preference** and select **DBEstudio -> Semantic discovery tool** and then select “Search for service”.

To use the feature “Save to SR” the connection with the KB must be properly configured, further details on that can be found in chapter 4.4 - SM Editor settings.

4.3 How to create a new Service Manifest

To create a new Service Manifest the user must firstly create a project.

4.3.1 Creation of simple project

To create a simple project, from the Eclipse toolbar menu the user has to select **File -> New -> Other**, as showed in Figure 4: Open Wizard.

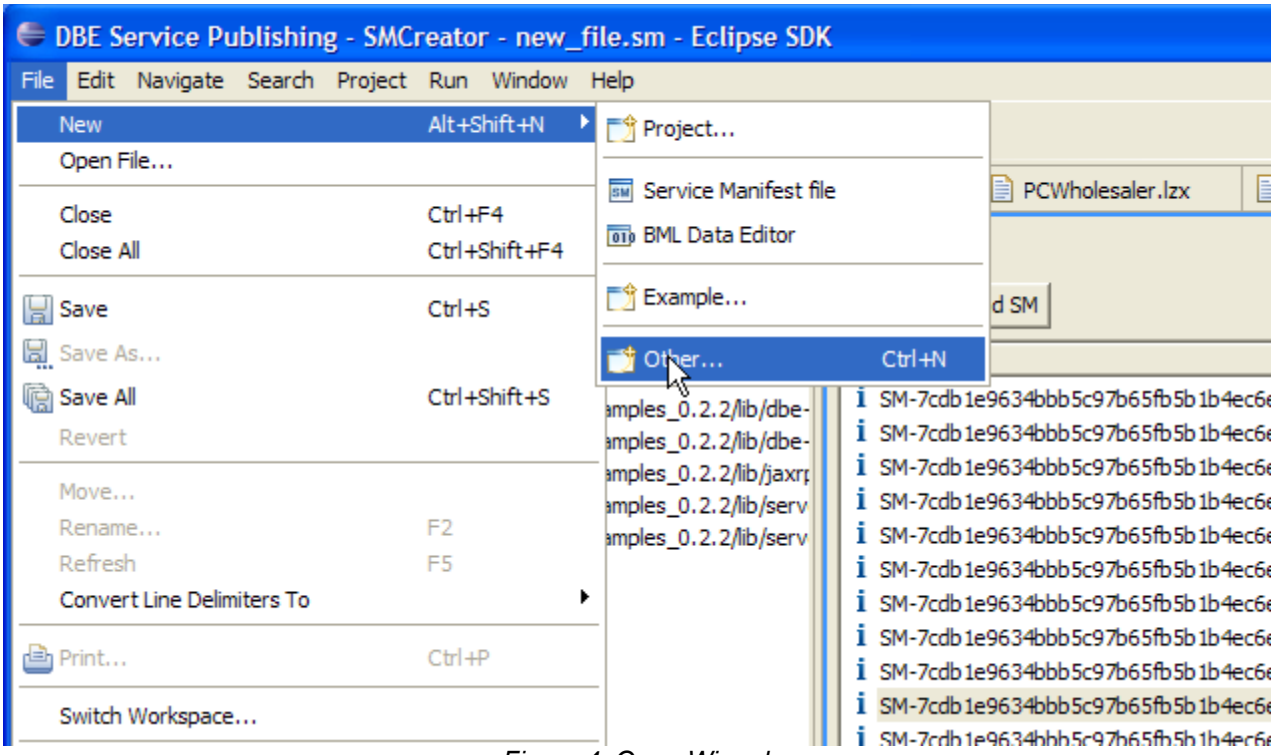

Figure 4: Open Wizard

A list of "NewProject" wizards will be presented to the user as shown in the Figure 5: New Project here below.

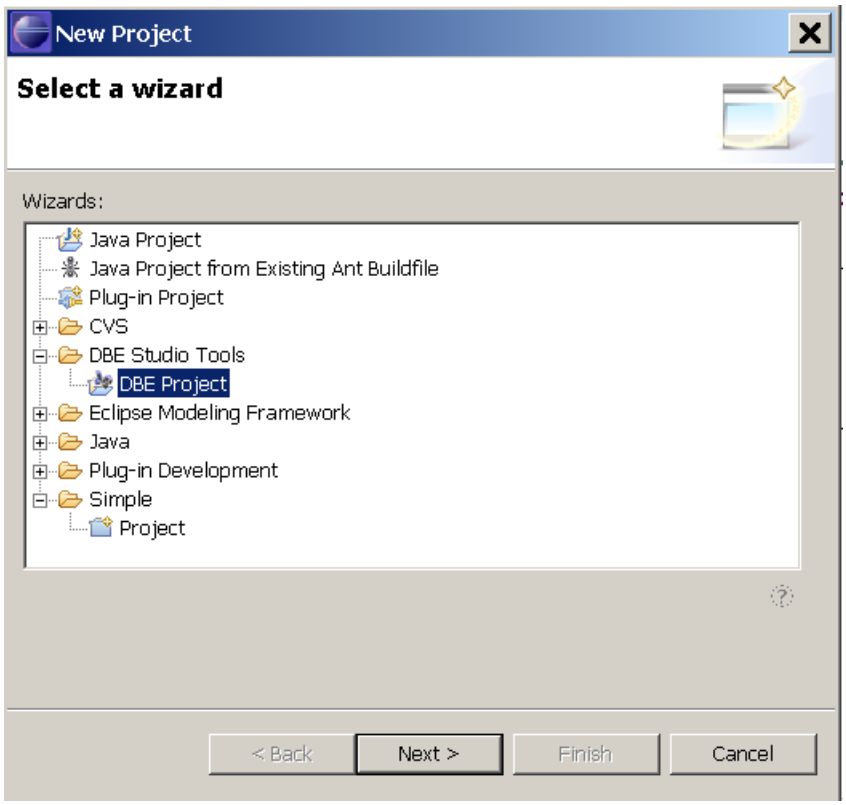

Figure 5: New Project

The user has to expand the folder “DBE Studio Tools” and select “DBE Project”, click on “Next”, then insert the project name. The Project folder will be created in order to store the Service Manifests.

4.3.2 Creation of a Service Manifest

In order to create a Service Manifest, from the Eclipse toolbar menu the user has to select **File -> New -> Other** [Figure 4: Open Wizard], then he has to expand the folder "DBE Studio Tools" and select "Service Manifest File". A windows displays two fieds, as shown in the Figure 6: New Service Manifest: in the field "Container" the user has to select the project in which he intends to insert the Service Manifest and in the fied "File name" insert the name of the Service Manifest.

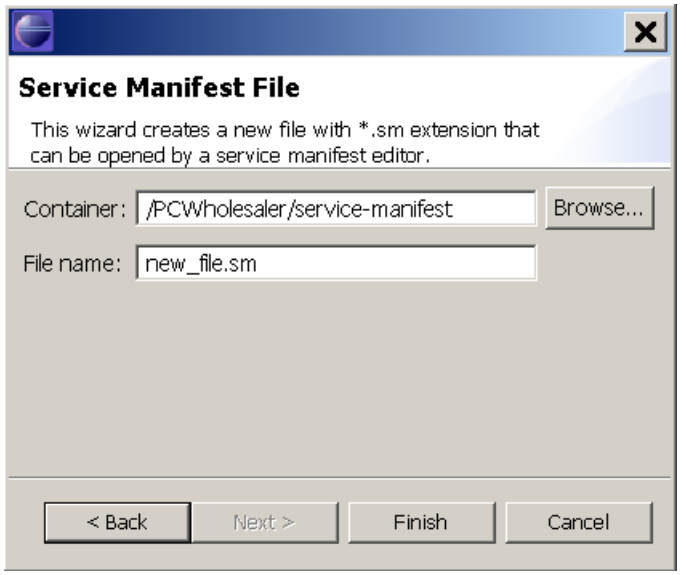

Figure 6: New Service Manifest

By pressing on the button “Finish”, the user will automatically open an empty Service Manifest which is also visible from the Package Explorer view.

4.4 SM Editor settings

To deploy or to load a service, the SM Editor uses the KB. In order to access the KB, the settings preferences must include its IP address. To set this preference, from the Eclipse toolbar menu the user has to press on **Windows -> Preference**, and select “DBE Studio” as shown in Figure 7: Setting Preferences .

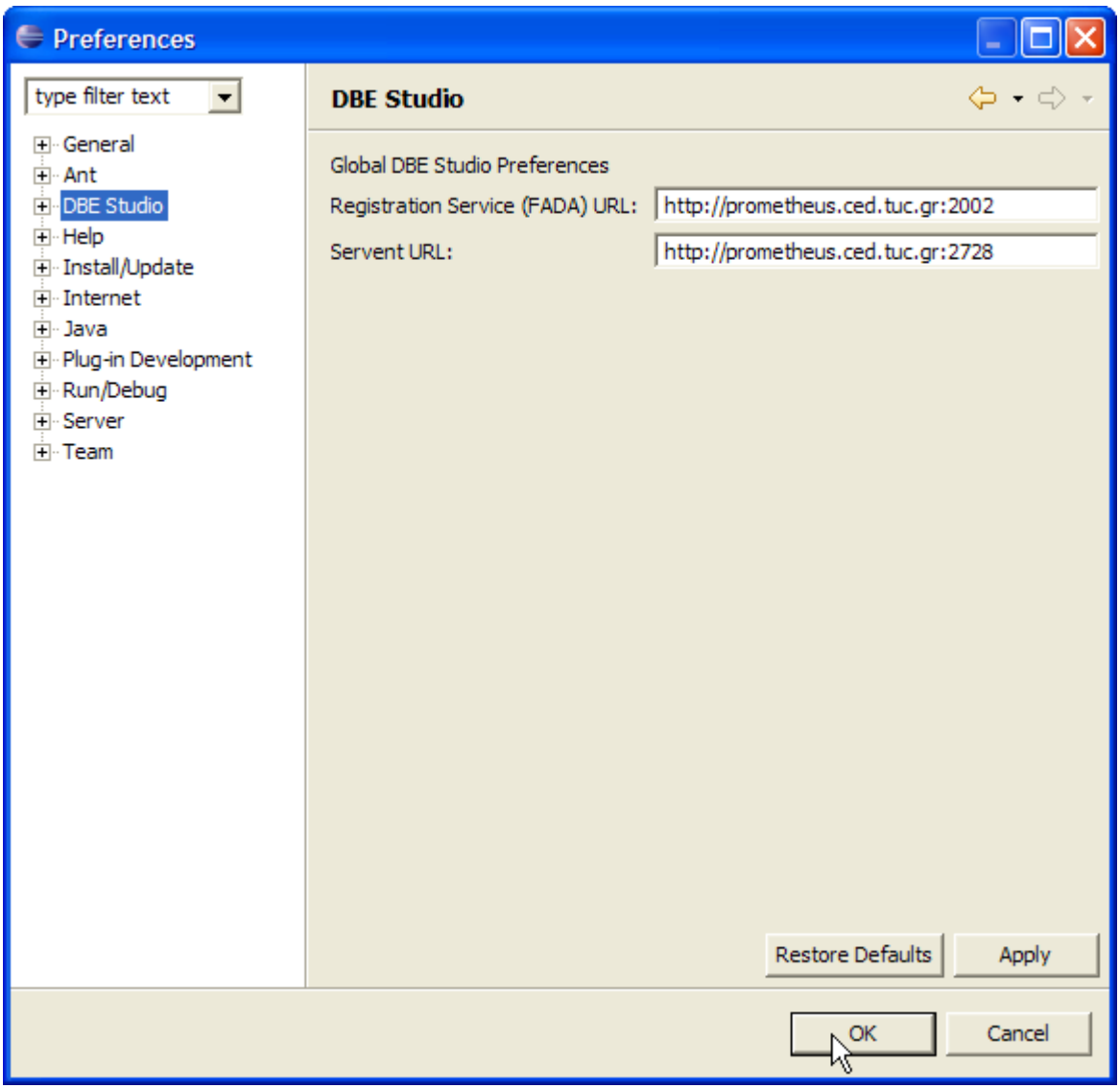

Figure 7: Setting Preferences

The window includes two fields where the user has to set the string to connect to KB. By default, the SM Editor tries to get connected to the address http://localhost:2002 for the FADA URL and http://localhost:2728 for the Servent URL. However, these addresses can change according to different needs. A list of nodes currently active is available at the URL: http://swallow.sourceforge.net/nodes.html

4.5 Storing the Service Manifest

The Service Manifest can be stored in two different locations: either in the local file system or in the KB.

4.5.1 Saving in the local File System

From the toolbar the user has to click on **File -> Save** or use the keyboard shortcut command CTRL+S.

With this functionality, the SM Editor offers the possibility to save the Service Manifest without deploying it on the KB. Furthermore, the SM Editor does not check that all the mandatory fields have been completed, allowing the user the option of inserting them afterwards.

4.5.2 Saving in the KB

The publication of Service Manifests on the KB makes them visible to all users. To publish a Service Manifest, the user has to press “Save to SR”. This will automatically save/deploy the Service Manifest and the Service Manifest list will get updated and it will display the latest inserted Service Manifest. To save/deploy the Service Manifest it is necessary to insert the mandatory fields indicated in (Table 1: Service Manifest Required Fields). The Service Manifest will not be saved if any mandatory field is not filled with data.

Attribute	***Mandatory***
SMID	yes
SMName	yes
Description	yes
VersionNumber	yes
RootSMID	yes
AncestorSMID	no
PIMModel	no
CIMModel	yes
ServiceType	yes
Availability	yes
RegistrarID	yes
SìPublicationDate	yes
LastChangeDate	yes
IconURL	no
InteractionForm	no
BPEL	no

Table 1: Service Manifest Required Fields

4.6 How to import PIM and CIM Models

There are two ways to import models, from the network and from from File System:

4.6.1 Importing from the File System

The user must press the related “FS” button (File System) which will open a window for navigating the File System. At this step the user has to select the proper file and press “Open”.

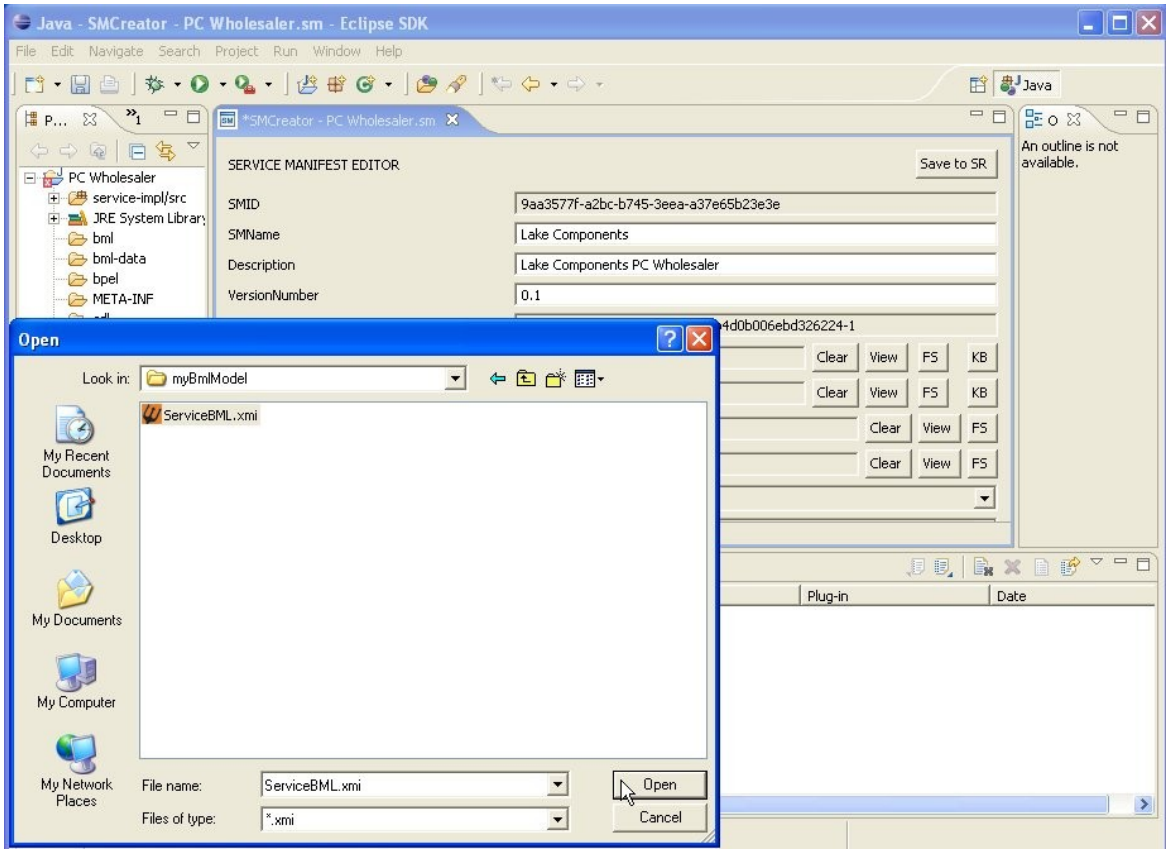

Figure 8: Model from FS

This will automatically import the file into the Service Manifest and make it visible to the user by pressing the button “View”. It should be noted that different models can be contained in files with different extensions. The following table [Table 2: Allowed Type] shows a list of models and the corresponding extensions accepted.

Model	***Extension***
BML Model	.xmi
SDL Model	.xml
BML Data	.xmi
SBVR Model	.xmi

Table 2: Allowed Type

4.6.2 Importing from the KB

To import a model from the KB, the user has to press the related “KB” button as showed in the Figure 9: Browse KB here below.

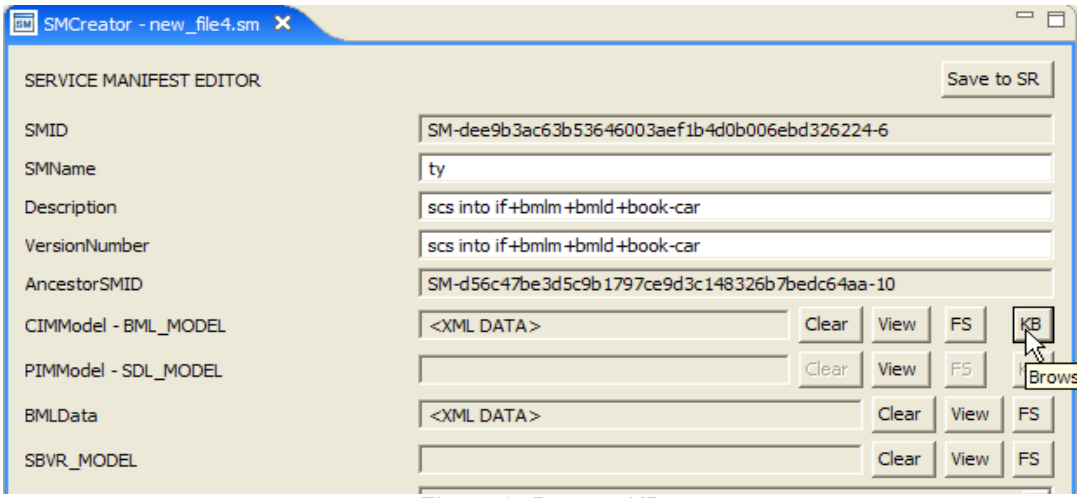

Figure 9: Browse KB

This will open a window with the list of BML models present into the KB (Model Repository), as shown in the Figure 10: Import Model from KB here below. The user has to select the desired model and press the “OK” button.

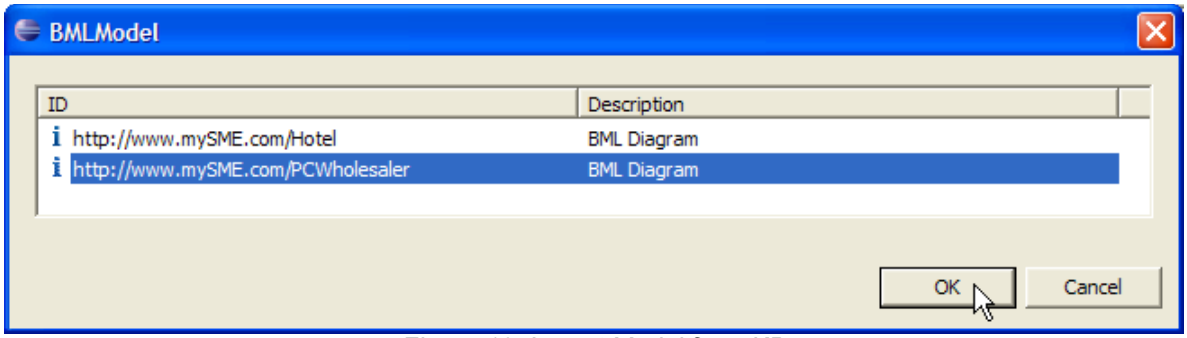

Figure 10: Import Model from KB

The model will automatically be imported and inserted into the Service Manifest. By pressing on the “View” button, as shown in [Figure 9: Browse KB], the user can visualize the model and make it visible to the user by pressing on the “View” button.

4.7 Methods of Service Manifest

4.7.1 Choosing the type of service

The ServiceType field defines the service type which can be: “Interaction Form” or “Computational Interface”.

4.7.1.1 Interaction Form

This has been introduced to facilitate the creation of simple services. It does not require as much IT expertise as standard DBE service, and, moreover, it does not require any Hardware infrastructure to provide the service [SMFD].

4.7.1.2 Computational Interface

To create this type of service, it is not necessary to have any Java expertise to implement the code starting from the PIM. Different to the Interaction Form, this type of service is executed in the machine of the service owner, allowing a more complete and customised service for the client [DBEArchReq].

4.7.2 Setting the service state

The “availability” is a flag which describes the service status, and can have the following values:

- **Created**: the Service Manifest is private to the owner and is not shared with the DBE community. It is stored in their private File System. The owner can modify the Service Manifest without restriction.
- **Available**: the Service Manifest is shared with the DBE community or groups of users.
- **Deprecated**: the Service Manifest still remains in the SR but it is not possible to have new agreements or to compose it in a new service chain. The Service Manifest Owner must guarantee the availability of the related proxy until the end of all the ongoing agreements. In any case the Service Manifest can be used as an ancestor for the creation of new Service Manifest. The registrar may re-publish the service.
- **Dead**: the Service Manifest has no proxies available. The SME provider has decided not to support the service any more. There are no ongoing agreements. The Service Manifest remains in the SR for historical purpose or it can be used as an ancestor for other services [SMFD].

5.Glossary

Term	*Acronym*	*Description*
Business Modelling Language	BML	BML is a language through which SMEs can describe their business model in order to enhance mechanism of discovery and selection of e-business partner within the DBE.
Business Service		The actual service supported by the DBE that is not either structural or basic
Computational Independent Model	CIM	The Computational Independent Model is a viewpoint of a system which focuses on the environment and the requirements for the system, but does not show details of the structure of the system [MDA Guide].
Interaction Form	IF	DBE services created and distributed by using a Wizard. Usually are simple services targeted at SMEs without IT staff or financial resources to invest in realizing a service. They include Self- Contained Services and Distributed Services.
Knowledge Base	KB	Is the part of the DBE system where the DBE knowledge is stored and managed. Such knowledge refers to ontologies, business and service description [KBDI].
Platform Independent Model	PIM	The Platform Independent Model is a viewpoint of a system which focuses on the information about the specific technology that is used in the realization of a particular platform. A PIM combines the specifications in the PIM with details that specify how that system uses a particular type of platform [MDA Guide].
Semantic Registry	SR	It is the component of the DBE Knowledge Base that hosts the service description published in the DBE environment and available for discovery and consumption.
Service Manifest	SM	All the specifications that describe a service from the computing and business point of view.
Model Driven Architecture	MDA	An IT system specification approach (proposed by OMG) that separates the specification of system functionality from the specification of the implementation of that functionality on a specific technology.
Platform Specific Model	PSM	The Platform Specific Model is a viewpoint of a system which focuses on the operation a system while hiding the details necessary for a particular platform.

6.References

DBEArchReq: Pierfranco Ferronato, DBE Architecture Requirements Ver 2.5, October 2004
KBDI: TUC/MUSIC, Knowledge Base Design and Implementation Status Ver 0.1, October 15, 2004
MDA: David S. Frankel, Model Driven Architecture: Applying MDA to Enterprise Computing, January 10, 2003
MDA Guide: OMG, MDA Guide Version 1.0.1, June 2003
QF SDT: TUC, D24.6 Query Formulator and Semantic Discovery Tool Final Release, April 2006
SMFD: Soluta, Service Manifest Full Definition, July 2006

- end of document -