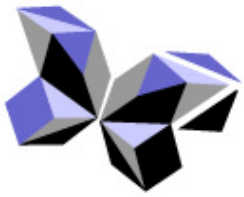


Digital Business Ecosystem

Contract n°507953

Workpackage WP26: DBE Portal

Deliverable D26.7: DBE Portal Specification Version 2

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

Contract Number: 507953

Project Acronym: DBE

Title: Digital Business Ecosystem

Deliverable N°: D26.7

Due dates: 30/11/2006

Delivery Date: 30/11/2006

Short Description:

A Digital Business Ecosystem Portal is a user friendly entry point to the DBE that provides the means to search, browse and execute DBE Services over the DBE Peer-To-Peer network using nothing more than a web browser. This document sets out the design of the DBE Portal and the requirements to satisfy this design. It also details the implementation of the supporting DBE Portal toolkit and any issues encountered in its implementation.

Author: Intel Ireland Ltd.

Partners contributed:

Made available to: Public

Versioning

Version	Date	Author, Organisation
0.1	24/05/2005	Andy Edmonds, TCD. Initial Draft
0.2	28/11/2005	Andy Edmonds, Intel Ireland Ltd. Major revision to original initial draft.
0.3	17/02/2005	Andy Edmonds, Intel Ireland. Final modifications and correction before submittal.
0.4	28/02/2006	Andy Edmonds, Intel Ireland Ltd. Suggestions from internal reviewer added.
0.5	09/03/2006	Andy Edmonds, Intel Ireland Ltd. Further suggestions from internal reviewers added.
0.6	12/11/2006	Andy Edmonds, Intel Ireland Ltd. Additions inserted to reflect 2.0 version of the Portal
0.7	30/11/2006	Andy Edmonds, Intel Ireland Ltd. Additions inserted based on internal reviewers comments

Quality check:

1st Internal Reviewer : Pierfranco Ferronato, Soluta.

2nd Internal Reviewer: Juanjo Aparicio, Techideas.

Table of Contents

Index of Figures

Index of Tables

1 Executive Summary

A Digital Business Ecosystem Portal is a user friendly entry point to the DBE that provides the means to search, browse and execute DBE Services over the DBE Peer-To-Peer network using nothing more than a web browser. This document sets out the design of the DBE Portal and the requirements to satisfy this design. It also details the implementation of the supporting DBE Portal toolkit and any issues encountered in its implementation.

2 Introduction

This is a report based on design of a means to search and access DBE services in a way inspired by current trends in web applications It is written as a part of the deliverable "D26.7 DBE Portal Specification Version 2" within the Digital Business Ecosystem project. This deliverable is composed of both this report and supplementing code. The code and implementation that this document refers to can be freely accessed at http://swallow.sf.net and is distributed as part of the ExE [14].

2.1 Background information

From the DBE web site (http://www.digital-ecosystem.org):

> *What is DBE?*
>
> The Digital Business Ecosystem (DBE) is an Internet-based software environment in which business applications can be developed and used. The unique feature of the DBE is that applications within the ecosystem are able to perform new functions that were, up to now, undreamed of by users.
>
> The DBE is an open, free environment where even the smallest specialist software developer can participate competitively in the massive global marketplace for business applications. It will enable end users to easily access and use those applications as services, and to have the benefits of intelligence, interaction and adaptation as the software evolves in response to their own usage and that of others. The initial target of the DBE is those complex commercial transactions and processes that are not easily or economically served by current even state-of-the-art software technologies.

2.2 The DBE Portal

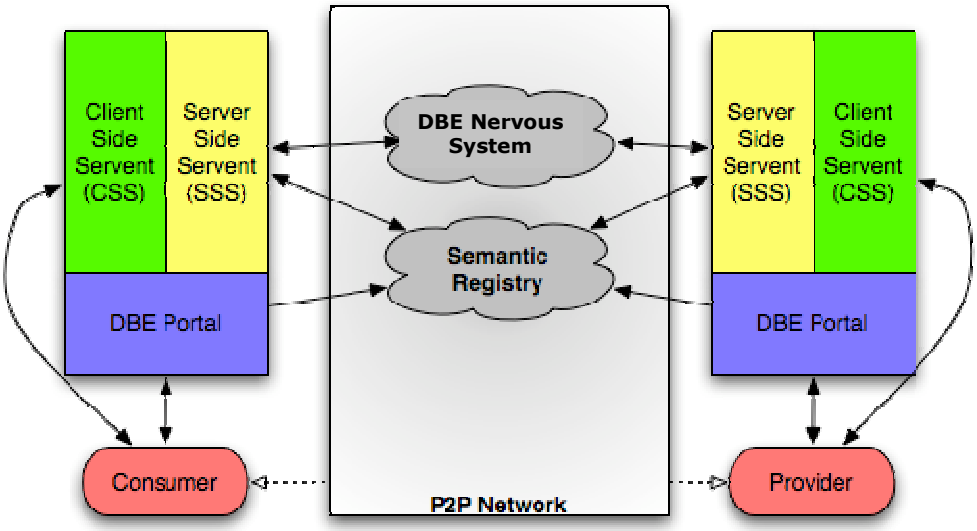


Figure 1 An Overview of the DBE Portal

A Digital Business Ecosystem [15] (DBE) Portal is a user friendly entry point to the DBE that provides the means to search, browse and execute DBE Services over the DBE Peer-To-Peer (P2P) network using nothing more than a web browser. Its relationship with the Execution Environment (ExE) is shown in Figure 1. A DBE Portal enables any user of the DBE with a minimum of technical know-how, and without any installation of DBE applications, to access, consume and use the DBE and the services it hosts.

A DBE Portal is an evolved and enhanced DBE Desktop [1]. The initial release of the DBE Portal seeks to match the functionality already contained in the DBE Desktop. The main functionality presented in the DBE Portal, just as in the DBE Desktop, is the ability to search, browse and execute services. The development of the DBE Desktop has halted except for critical bug fixes. Future feature additions and functionality enhancements will be folded into the DBE Portal development effort.

The method of interfacing with the DBE Portal is by using any web browser (e.g. Firefox, Internet Explorer, Mozilla, and Opera). The DBE Portal does not require any heavy client side plug-ins such as Java run-time environments and as such the web browser is the thin client. All computation is performed on the servent, through the DBE Portal, with the results of computation displayed on the client (web browser). The computation on the servent [8] is performed by interfacing with the DBE via the toolkits and frameworks provided by the Java development kit [16] (JDK) and those already developed by DBE partners. These include the servent Application Programming Interfaces (API) and the DBE client toolkit. The framework that allows DBE services to display user interfaces within the client (web browser) is specified in deliverable 20.1 [2].

A programmer's toolkit, the Portal Toolkit, has been developed to allow the execution of services through user interfaces (UI) displayed by the end-users' web browsers.

Taking this approach allows any user of the DBE, without an installation of any DBE applications, to access the DBE with just the bare minimum of a web browser.

The DBE Portal is dependent on other DBE components as shown in the diagram below (Figure 2). Where a user does not have the facility to run their own servent and consequently a DBE Portal, it could be possible, where a Small-to-Medium Enterprise (SME) allows it, for that user to use another SME's DBE Portal to search, browse and execute DBE services. For this to be an attractive proposition for a SME to consider, incentives must be offered to encourage them to allow lone users access their DBE Portal.

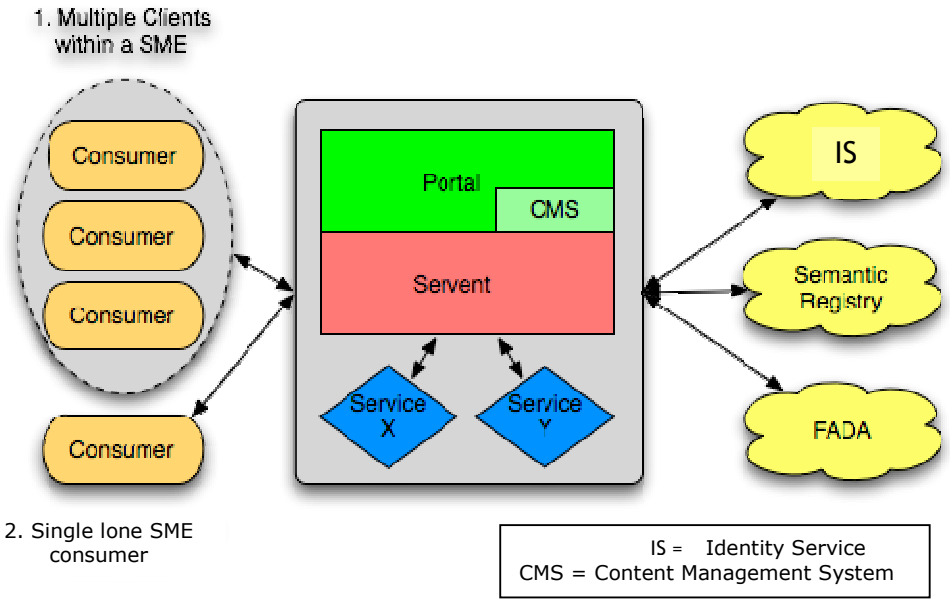


Figure 2 DBE Portal and Related Components

3 DBE Portal Feature Requirements

The basic premise of the DBE Portal is to allow any user of the DBE, be they consumers[1] or providers[2], to interact with services advertised in the DBE using the following three basic functionalities:

- *Search* – using the DBE Portal a DBE user can provide search terms that describe the service that the user is looking for. The user can also be presented with the option to formulate a query using advanced query operators. The returned results are services that best match the entered search terms.

- *Browse Service Information* – After submitting a search request and retrieving back the results, the user can view additional information about a service that the user wants to use.

- *Execute* – using the results from the action of *searching* or *browsing*, once a suitable service is selected, the user can invoke and execute that service. From the point of view of the DBE Portal, the execution of a service involves the retrieval of the user interface of that service, and displaying it to the requesting user. On entering data, the user can then *execute* the UI's functionality.

It is this set of functionality that has been implemented in the first version of the DBE Portal. The above features fall into the category of application/service functionality and are features related to direct manipulation of DBE services. With the necessary features (listed above) implemented, it is envisioned that new and extra functionalities will be added to the DBE Portal, especially those encapsulated by the general notion of account management, which includes tasks such as service bookmarking, profile management, workflow management etc.

Although the DBE Portal has been designed to satisfy these basic and mandatory use cases, additional functionality has also been considered. Although considered important, it should be noted that the additional functionality will only be implemented when the core functionality of the DBE Portal is complete and released. The secondary functionalities (use cases) fall into the categories of infrastructural or informational and include:

- Facilities to provide a free "web-presence" of the SME not only to users within the DBE network but also to the wider population of users that use the World Wide Web (WWW). This will provide

[1] A consumer can be viewed as a classic client; an entity that uses the functionality of a service to achieve an end-goal.
[2] A provider, in the sense of DBE, is an entity that provides a service that can be consumed. The service that the provider offers may utilise external services and in this provider specific context, the provider is also a consumer too.

a means for the SME to disseminate information about itself and its services. To this end, a Content Management System (CMS) for the organisation of SME information may be used. A CMS is a system used to organize and facilitate collaborative content creation. It provides a simple way to create, maintain and update content hosted on a DBE Portal and display it using the familiar mechanisms of a web server via a "web site".

- Using a DBE Portal, a DBE user could, starting from a particular taxonomy, drill down until a service meeting the user's requirements is discovered. The desired functionality of browsing DBE service could emulate that of Universal Description, Discovery and Integration [13] (UDDI) service listing, for example:
    - White directory: based on address and contact information. This information can be found within a service's Business Modelling Language (BML) data model (M0), contained in the service's service manifest.
    - Yellow directory: based on service categorised by industry/business. This information can be found within a service's BML model (M1), contained in the service's service manifest
    - Green directory: based on technical information about services. This type of information can be found within a service's Service Definition Language (SDL) model, contained in the service's service manifest.

- A means of user registration. This will allow registration of new identities with the DBE using the Identity Service (IS) as it is anticipated that DBE Portals will be entry points (i.e. registration agents) for an SME wishing to join the DBE. It will be here where the SME will register with the DBE in order to provide its services.

- A way to allow user profile creation and management. This should facilitate the modification and management of a registered user's profile [3] identified by the user's IS identity. Using the user's profile, it is hoped that preferred services can be bookmarked and stored by saving a reference to the service's Service Manifest Identifier (SMID). It is foreseen that the Distributed Storage System (DSS) could be used as a mechanism to allow the storage of such information and the user's profile.

- A DBE Portal should also be registered with a directory-type component (see Ch. 5) when the SME starts its servent and comes online. Conversely, it should be unregistered when the SMEs servent is shutdown and goes offline. This task may also involve updating of the DBE Portal's endpoint information with the directory-type component especially if the SME's Internet Service

Provider (ISP) mandates the use of Dynamic Host Control Protocol (DHCP) [10] allocated Internet Protocol (IP) [11] addresses and or the DBE Portal is located behind a Network Address Translator (NAT) [12].

- A distributed DBE service development environment where DBE developers can create service compositions and deploy those compositions. User interfaces could, potentially, also be created within a DBE Portal to visually represent those service compositions to consumers.

- A means and mechanism to access the DBE applications. The DBE suite of applications enables the development of DBE services, the consumption of services and the provisioning of DBE services. These applications include the DBE Studio and related plugins, the DBE Servent and the DBE Evolutionary Environment (EvE). Also any frameworks that a developer may be interested in using e.g. the APA should be accessible.

- A place where DBE related statistics can be viewed, e.g., number of Knowledge Base (KB) nodes, number of currently “live” SMEs, total number of “live” DBE services, network topology and it size. This task could be performed by another DBE core service that DBE Portal could use, which could perhaps supply greater information than the partial view of the network available to the DBE Portal. This service could track various P2P system nodes and topologies formed between servent and P2P system nodes. Another, but simpler way to get some statistics would be to issue a query to the Semantic Registry (SR) requesting how many entities of the DBE Portal type are currently registered.

- Support for the resumption of DBE user sessions could also be included in the portal so that DBE users could resume work from where they last left after logging out of DBE. By supporting this, it would enable the retrieval of results from long running transactions as it is infeasible to expect DBE users to remain logged in to the DBE over period of days. The user’s state could be saved in DSS.

4 DBE Portal Use Cases

What follows is a listing of possible use cases related to the DBE Portal. Those listed below may not be implemented in the initial release of the DBE Portal but will start to form a development road map for the DBE Portal. From the DBE technical annex, it is stated that there will be two releases, both deliverables, of the DBE Portal corresponding to the versions of DBE Portal Version 1 and DBE Portal Version 2. Below is a table summarising the use cases collected so far for the DBE Portal (those use cases in *italics* are suggested future work).

Use Case Reference	Version Implemented	Use Case Description
UUC1	0.1	Find Services Use-case
UUC2	0.1	Browse Service Information Use-case
UUC3	0.1	Invoke Service UI Use-case
UUC4	0.2	Manage Profile Use-case
UUC5	0.1	Access DBE Components Use-case
UUC6	*n/a*	*Register with Identity Service Use-case*
UUC7	*n/a*	*Login/Restore Session Use-case*
UUC8	*n/a*	*Browse DBE Services Use-case*
PoUC1	0.1	Disseminate SME Information Use-case
PoUC2	0.2	Register With DBE Use-case
PrUC1	0.2	Administer Service Use-case
PrUC2	0.2	Deploy Service Use-case
PrUC3	*n/a*	*Author Service UI Use-case*
PrUC4	*n/a*	*Compose Service Use-case*
CUC1	*n/a*	*Bookmark Service Use-case*

Table 1 Reference Table of Use Cases

4.1 DBE User Use-cases

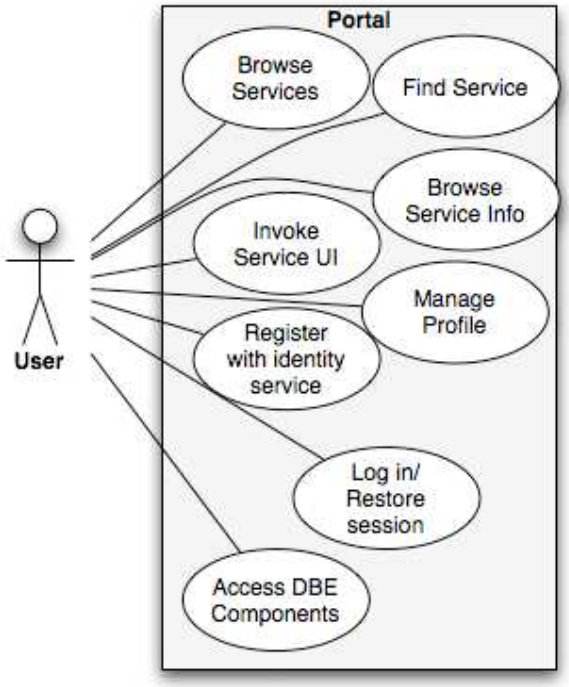


Figure 3 DBE User Use-cases

Use Case UUC1	**Find Service**
Area	DBE Portal
Objective	Find existing DBE Service through a Web Browser
Actors	User
Pre-Conditions	DBE Services must be available
Post-Conditions	
Description	The user will issue search queries in a Web Browser in order to find services he/she is interested in.
Dependencies	User Interface (web-based), Semantic Registry
Release	Version 1
Relevant Partner	TUC, SUN, Intel

Table 2 Find Services Use-case

Use Case UUC2	**Browse Service Information**
Area	DBE Portal
Objective	Browse DBE Service Information through a Web Browser
Actors	User
Pre-Conditions	DBE Services must be available
Post-Conditions	
Dependencies	Semantic Registry, Portal Toolkit
Release	Version 1
Relevant Partner	TUC, Intel

Table 3 Browse Service Information Use-case

Use Case UUC3	Invoke Service UI
Area	DBE Portal
Objective	Use an existing DBE Service's UI through a Web Browser
Actors	User
Pre-Conditions	DBE Services must be available The search query must return at least two DBE Services
Post-Conditions	
Description	The user will use an existing DBE Service through a Web Browser. The service will display an Openlaszlo user interface with input fields to fill in by the user.
Dependencies	Web-based GUI Development Tool, Portal Toolkit
Release	Version 1
Relevant Partner	Intel, TUC, SUN

Table 4 Invoke Service UI Use-case

Use Case UUC4	Manage Profile
Area	DBE Portal
Objective	Edit, update and manage a user's DBE preferences.
Actors	User
Pre-Conditions	User is logged in. User-profile service is available
Post-Conditions	
Description	The user can manage his/her preferences and persist these to suitable storage
Dependencies	User profiling service
Release	Version 2
Relevant Partner	FZI, Intel

Table 5 Manage Profile Use-case

Use Case UUC5	Access DBE Components
Area	DBE Portal
Objective	Download DBE components to develop and access DBE services
Actors	User
Pre-Conditions	
Post-Conditions	
Description	A user wanting to develop DBE services can access the necessary resources to accomplish this.
Dependencies	Sourceforge project site
Release	Version 1
Relevant Partner	Intel

Table 6 Access DBE Components Use-case

4.1.1 Use Cases for Further Work

Use Case UUC6	Register with Identity Service
Area	DBE Portal
Objective	Issue a registration request for a DBE Identity and receive corresponding credentials
Actors	User

Pre-Conditions	User is not registered with the identity service
Post-Conditions	User has a DBE identity
Description	A user can register for a DBE Identity though the DBE Portal
Dependencies	Identity service.
Relevant Partner	TCD, Intel

Table 7 Register with Identity Service Use-case

Use Case UUC7	**Login/Restore Session**
Area	DBE Portal
Objective	Return to a long-running session through the DBE Portal
Actors	User
Pre-Conditions	An existing session running
Post-Conditions	
Description	When a user executes a long-running process e.g. a workflow, the user will be able to log out and at a later stage log back in and check the status of the process
Dependencies	Identity service, user-profiling service
Relevant Partner	University of Surrey, Intel

Table 8 Login/Restore Session Use-case

Use Case UUC8	**Browse DBE Services**
Area	DBE Portal
Objective	Browse DBE Service Information through a Web Browser in a UDDI fashion
Actors	User
Pre-Conditions	DBE Services must be available
Post-Conditions	
Description	
Dependencies	Semantic Registry, Portal Toolkit
Relevant Partner	TUC, Intel

Table 9 Browse DBE Services Use-case

4.2 DBE Portal Use-cases

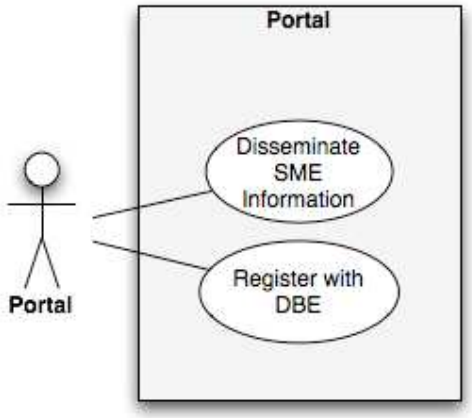


Figure 4 DBE Portal Use-cases

Use Case PoUC1	Disseminate SME Information
Area	DBE Portal
Objective	Provide an “out of the box” web site.
Actors	Portal
Pre-Conditions	
Post-Conditions	
Description	This will provide a free web presence that will allow SMEs provide more information about its business.
Dependencies	Content Management System (Possibly)
Release	Version 1
Relevant Partner	Intel

Table 10 Disseminate SME Information Use-case

Use Case PoUC2	Register With DBE
Area	DBE Portal
Objective	Register the Portal with DBE infrastructure
Actors	Portal
Pre-Conditions	
Post-Conditions	
Description	By registering the Portal with DBE infrastructure, it will be possible to search for Portals within a particular business domain.
Dependencies	KB, SR
Release	Version 2
Relevant Partner	Intel, SUN

Table 11 Register With DBE Use-case

4.3 DBE Provider Use-cases

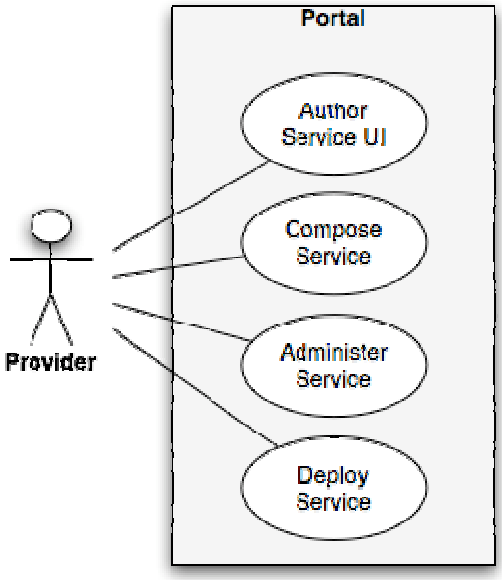


Figure 5 DBE Provider Use-case

Use Case PrUC1	**Administer Services**
Area	Portal
Objective	Administer various services of the ExE
Actors	Provider
Pre-Conditions	
Post-Conditions	
Description	This will allow the administration of the various components of the ExE through the portal
Dependencies	P2P system, Servent, KB, SR
Release	Version 2
Relevant Partner	Intel, SUN

Table 12 Administer Service Use-case

Use Case PrUC2	**Deploy Service**
Area	Portal
Objective	Deploy an implemented service
Actors	Provider
Pre-Conditions	
Post-Conditions	
Description	This will allow the deployment of an implemented service through the Portal
Dependencies	Servent
Release	Version 2
Relevant Partner	Intel, SUN

Table 13 Deploy Service Use-case

4.3.1 Use Cases for Further Work

Use Case PrUC3	**Author Service UI**
Area	DBE Portal
Objective	Allow users create declarative UIs within the Portal
Actors	Provider
Pre-Conditions	
Post-Conditions	
Description	A user can use the Portal to author declarative UIs and associating it with a service on the portal's servent
Dependencies	UI Technology
Relevant Partner	UCE, SUN, Intel

Table 14 Author Service UI Use-case

Use Case PrUC4	**Compose Service**
Area	Portal
Objective	Allows users to create service compositions
Actors	Provider
Pre-Conditions	
Post-Conditions	
Description	A user will be able to create service compositions of services found within the DBE within the Portal
Dependencies	Service composition engine
Relevant Partner	TCD, Intel

Table 15 Compose Service Use-case

4.4 DBE Consumer Use-cases

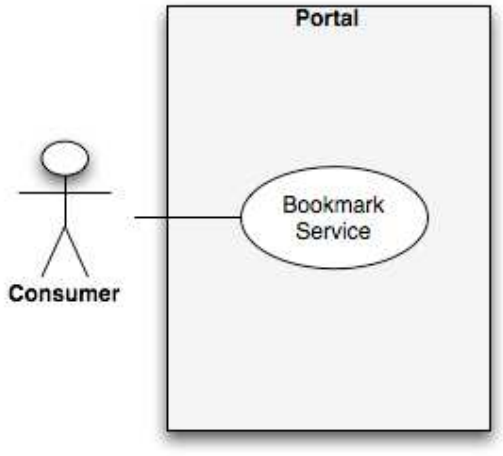


Figure 6 DBE Consumer Use-case

4.4.1 Use Cases for Further Work

Use Case CUC1	**Bookmark Service**
Area	DBE Portal
Objective	Store pointers to used DBE service instances
Actors	Consumer
Pre-Conditions	Existing services
Post-Conditions	
Description	This will allow users store pointers to used DBE services much like a user would bookmark websites
Dependencies	
Relevant Partner	Intel, TUC

Table 16 Bookmark Service Use-case

5 DBE Portal Considerations

5.1 DBE Portal Registration

Originally in the initial stages of the design and specification of the DBE Portal, it was conceived that there would be a DBE Directory, a type of meta-portal named as the DBE Gateway Portal. It would be here where all DBE Portals hosted on servents would register themselves along with their endpoint information. It would be a directory that aggregated all SMEs currently online and organised them according to their ontology.

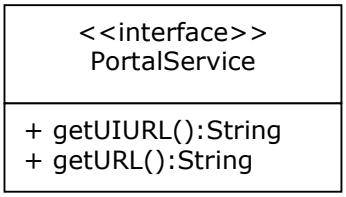
Although this approach would suffice for searching for DBE Portals and the services hosted within them, it would unfortunately be deficient. The primary reason to reject such a design is that in essence the DBE Gateway Portal would be a single point of failure in an otherwise fully decentralised design. Another major point of rejection is that parallel tracks of development would be necessitated to develop such a Gateway Portal. From exploratory work carried out by the author, it was viewed that a new lightweight directory service would be required to enable the Gateway Portal within DBE. Obviously this would circumvent established frameworks and core services already available within the DBE ecosystem and from a managerial point of view with regards to resource allocation would be inefficient and wasteful. Posed with these problems, it was necessary to find a solution using the current frameworks and DBE core services so that DBE Portals could register themselves within the DBE networks in order to be searchable.

To this end, the notion of modelling a DBE Portal as a DBE service was conceived. What is required for this solution is to describe the DBE Portal as a BML model, a model that should not change from SME to SME. Such a BML model is a model template. The BML template provides a means to search for a group of DBE Portals that fall under a particular type of business domain in a distributed P2P fashion. All that will change is the BML data associated with that model. The BML data required for a DBE Portal will be supplied when it is installed along with the servent and will be a one-time process.

By using features of BML, it is possible to organise SME DBE Portals into categories much in the same way as organisation is performed in folksonomies [4] by using the data modelled in the BML. By following such an approach it is feasible to imagine self-organising structure and categorisation within the DBE where more accurate terms describing a particular business domain are popularised. Those popularised are the terms that are the “fittest” for describing such a domain. It is important to note that by following this approach it does not require that an administrator look after a particular ontology and it is possible to organise services in a navigable, automatically created structure.

Although, initially it made sense to store endpoint information in the BML data, this after further thought was considered infeasible. It would only work where it can be guaranteed that for all SMEs their

endpoint information was immutable and never changing, but for many SMEs this is not the case (e.g. where there IP address is allocated by DHCP). As BML data is considered to be slowly changing over time, it is not a suitable place to store endpoint information, which could be potentially updated every hour. To resolve this, a simple SDL model of a DBE Portal is required. It will be this SDL model and its corresponding interface and methods that will return back endpoint information to locate a SME's DBE Portal. The interface of the SDL model, shown as a Java interface for brevity, is modelled as follows:



Snippet 1 DBE Portal SDL Interface

With the DBE Portal models created and implemented the DBE Portal service would be exported to the servent where it will be published (with its service manifest) to the Semantic Registry (SR) and have its service proxy registered with the P2P network. With this performed the DBE Portals contained within the DBE network are then searchable using the existing DBE infrastructure.

Typically, when a SME first wants to log onto the DBE, the SME starts the servent which in turn starts the DBE Portal. As the servent starts, it registers all services that are currently deployed on it including the DBE Portal service. As it registers the DBE Portal service, the servent should also update the DBE Portal's endpoint information by checking for:

- Renewed DHCP IP address – whether or not the SMEs ISP has requested that the SME's machine through which it accesses the network should renew its IP address.
- NAT Status – whether or not the servent is behind a NAT and if so its type and the external IP address the servent is contactable at. This NAT information could possibly be obtained by using a STUN [5] server.

Once the DBE Portal service has started up, the DBE Portal to which it corresponds is available through the established mechanisms of discovery and consumption in the DBE.

- *Discovery* is performed by the SR and consumption of the DBE Portal service is executed by the servent.

- *Consumption* happens when the DBE Portal service's proxy returns the endpoint of the DBE Portal where it can be located and displayed within a standard web browser.

When the SME shuts down the servent, its DBE Portal goes off-line and the service is shutdown thus removing it from the P2P system and making it unavailable. This validates the metaphor "the SME is closed for business" and although closed for business the SME still has a visibility within the DBE, just like a closed shop has on a street.

5.2 Users and the DBE Portal

There are a number of scenarios in which users of a service can interact with the DBE Portal and its related services. The simplest case is that of a user, acting as a single entity, directly accessing a DBE Portal that is externally accessible and allows both anonymous and authenticated users interact with it. This is shown in Figure 7.

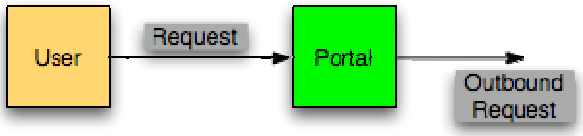


Figure 7 A Single User and a DBE Portal

A second case is where multiple users belonging to the same organisation or SME interact with their instance of a DBE Portal. In this case, users within the SME's domain are the only users interacting with it. This DBE Portal in this case is not accessible from domains outside of the SME's domain. This is shown in Figure 8.

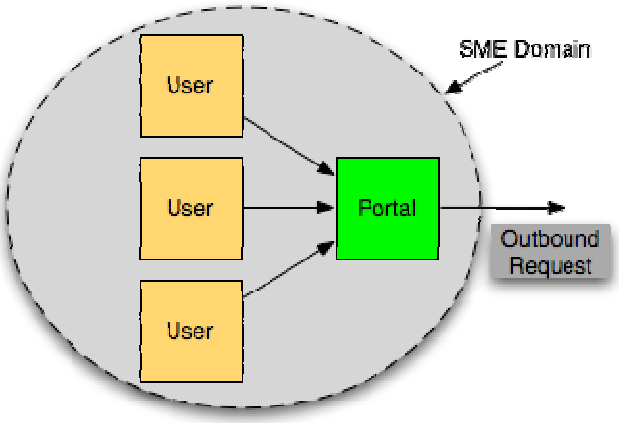


Figure 8 Multiple Users with the Same Domain and a DBE Portal

5.3 Security and the DBE Portal

This is related to the previous section on the topic of users and the DBE Portal. What is in question here is where and at what level is authentication and authorisation performed?

In the first case, the single user entity scenario, the single user remains anonymous, or authenticates against the DBE Portal. With the authentication information, it can then, on the behalf of the user, authenticate against the Identity Service (IS). If the DBE Portal allows for anonymous access, then the policies of what an anonymous user can do at that DBE Portal could be set by using the administration interface of it. Below, an example of a single user invoking a service with authentication is shown:

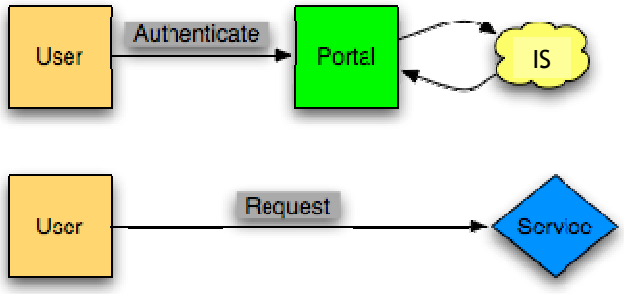


Figure 9 Single User Authentication

The next scenario is when there are multiple users within the same SME domain that require the use of services that reside outside of the SME's domain. The DBE Portal in this case is not accessible from domains outside of the SME's domain. There are two ways authentication can be performed in this scenario. The first method in which authentication can be performed is shown in Figure 10. In this scenario, the DBE Portal holds the SME's DBE credentials that allow users invoke services through it. It authenticates for the SME domain that it is part of. Any users within this domain can then use the services advertised within the DBE as it is assumed that they are part of the SME. Taking this approach allows for transparent authentication, makes the implementation of clients easier as they need not implement authentication mechanisms and allows the SME itself manage each of its users independently without having to go to a third party to modify a users permissions or access rights.

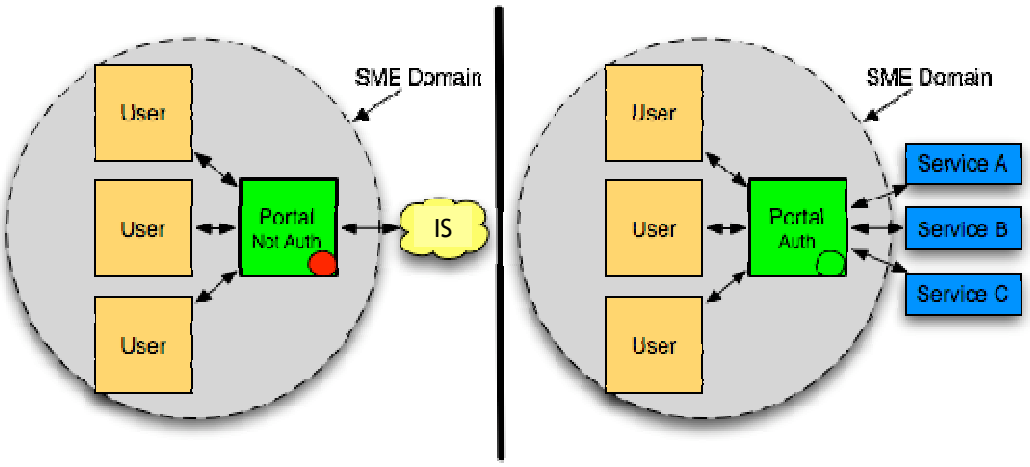


Figure 10 Multiple User Authentications: Scenario 1

The second way authentication can be performed with multiple users within a SME's domain is shown in Figure 11. In this scenario, every user within the SME's domain must authenticate against the SME's DBE Portal. It then authenticates on behalf of the user with the IS. Once authenticated, a user then can use the services within the DBE. Using this scenario allows for each user within the SME to have a stronger identity as each user will have an identity within the IS. This disables a masquerading attack where a hostile user initiates an attack of some sort. This attack would be possible if authentication is performed as in the previous scenario.

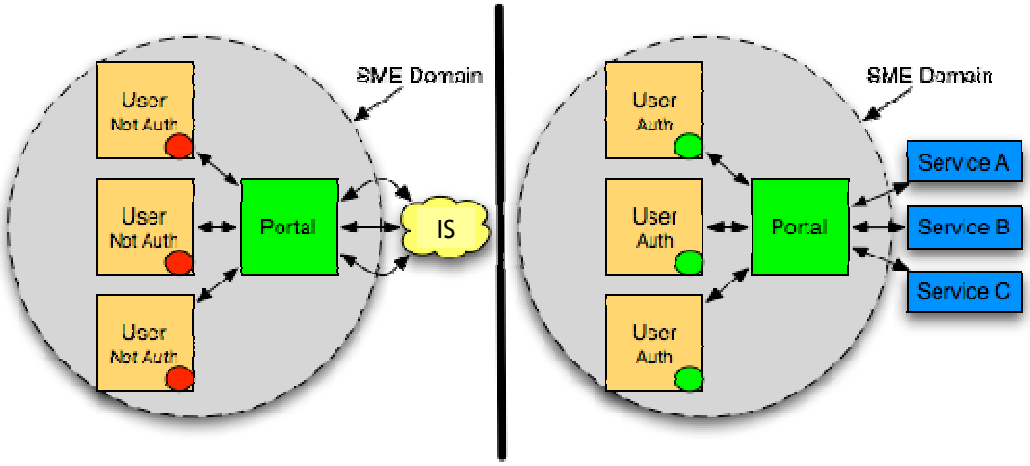


Figure 11 Multiple User Authentication: Scenario 2

6 DBE Portal Toolkit

The DBE Portal Toolkit has been designed to support the:

- Retrieval of a remote UI on a remote servent and display of it using the local servent
- Asynchronous[3] remote execution of a service through a locally hosted user interface

Both the above tasks are carried out by the DBE Portal Toolkit that comprises of:

- DBE Portal Client – This component is responsible for the retrieval of UIs and execution of services represented by those UIs. This is a component available within the Openlaszlo engine, which runs on the Servent.
- DBE Portal Service. – This component is responsible for the provisioning of UIs when a request is received for a service's UI. This is a service written as a DBE Service that runs inside the Servent.

To retrieve the remote UI, the method, `IDBEPortal::getUI()`, contained in the DBE Portal Client is used to retrieve the requested UI. This method returns back the URL of where the UI can be retrieved.

The remote execution of the service through the UI is not UI-technology specific. However, as a *defacto* UI technology the computing domain partners have chosen Openlaszlo [6] as the UI technology on the recommendation of [2]. There are two main phases related to remote execution of a service when Openlaszlo is considered.

- Execution between the UI and servent: this accomplished using Openlaszlo-specific JavaRPC [7] that provides an execution binding between the Openlaszlo engine and the servent.
- Asynchronous execution between the servent and service: this is performed using standard servent and APA API's.

These two tasks of remote UI retrieval and remote service execution are detailed further in the following sections (6.1, 6.2).

[3] The invocation is asynchronous between client UI and servent. However execution is not fully asynchronous due to the current P2P system implementation between servents and services.

6.1 Remote UI Retrieval

For the retrieval of a user interface to occur the following steps (illustrated in **Figure 12** and **Figure 13**) take place:

1. The user searches for a service using the Query Formulator-Semantic Discovery Tool (QF-SDT)[4] that is displayed in the user's web browser. The service could be hosted on a remote servent.
2. The QF-SDT served by portal and displayed by the user's web browser takes the request and searches the Semantic Registry (SR) for matching services using the Portal Toolkit (PT)
3. A list of matching service offerings (Service Manifests (SM)) are displayed via the QF-SDT interface
4. The user selects a service interesting him/her.
5. The PT downloads the service proxy on the local servent corresponding to the user selected service.
6. The PT retrieves the service's UI using the service's proxy.
7. This UI is mirrored at the local portal using the PT.
8. The QF-SDT is returned a URL to the mirrored UI on the local portal and displays that UI in the user's web browser using the QF-SDT.

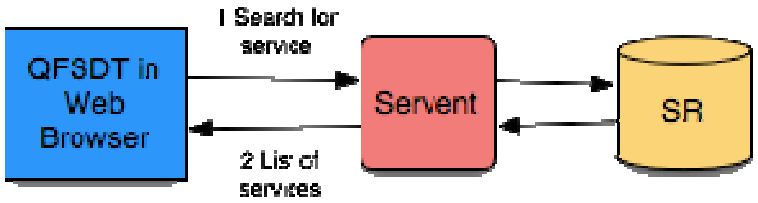


Figure 12 Remote UI Retrieval

[4] The Query Formulator/ Semantic Discovery Tool mentioned here is an Openlaszlo port of the DBE Studio QF-SDT eclipse plugin. Both have been implemented by TUC.

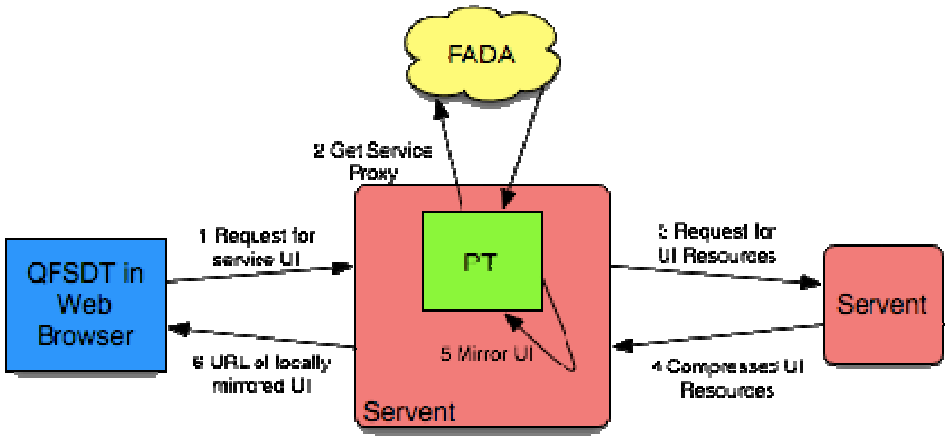


Figure 13 Remote UI Retrieval

6.2 Remote Service Execution

From what was shown in 5.2, it can be shown that either the user authenticates themselves to the IS and then directly invokes a particular service or the DBE Portal authenticates on behalf of the user within a SME domain and relays service invocations to and from the user. The first case is typically where a user of the DBE does not need or cannot run the DBE applications. In this case, another means of interacting with the DBE needs to be provided. Figure 14 is an overview of how this process of a single user invoking a service may operate:

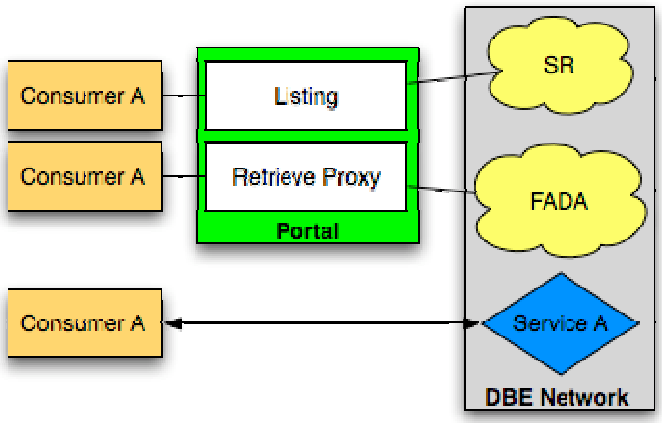


Figure 14 DBE Portal Single User Service Invocation

The task of executing a service becomes more complicated when authentication is performed on behalf of a user by the DBE Portal. In this case, it is necessary for the DBE Portal to maintain a state table of all invocations made from the authenticated users. The DBE Portal then becomes a service relay. This

functionality can then be likened to that of a NAT. Within the Portal Toolkit, service execution through a UI is performed in the following fashion and illustrated in Figure 15:

1. The user has a UI to interact with. This UI was acquired from the remote UI retrieval process (6.1).
2. The user presses a button on the UI which calls the PT with:
    a. SMID
    b. Target method name
    c. Method parameters
3. This invocation is passed on to the Openlaszlo engine on the local servent via JavaRPC to a PT component within the Openlaszlo engine.
4. To create the remote call to the service by the PT, the PT needs to discover what are the parameter types – this is done by retrieving and reading the SDL types contained in the SM.
5. The PT invokes the Client Side Servent (CSS)
6. The CSS then invokes the remote service using servent infrastructure and the corresponding values returned to the Openlaszlo client UI via the same JavaRPC mechanism.

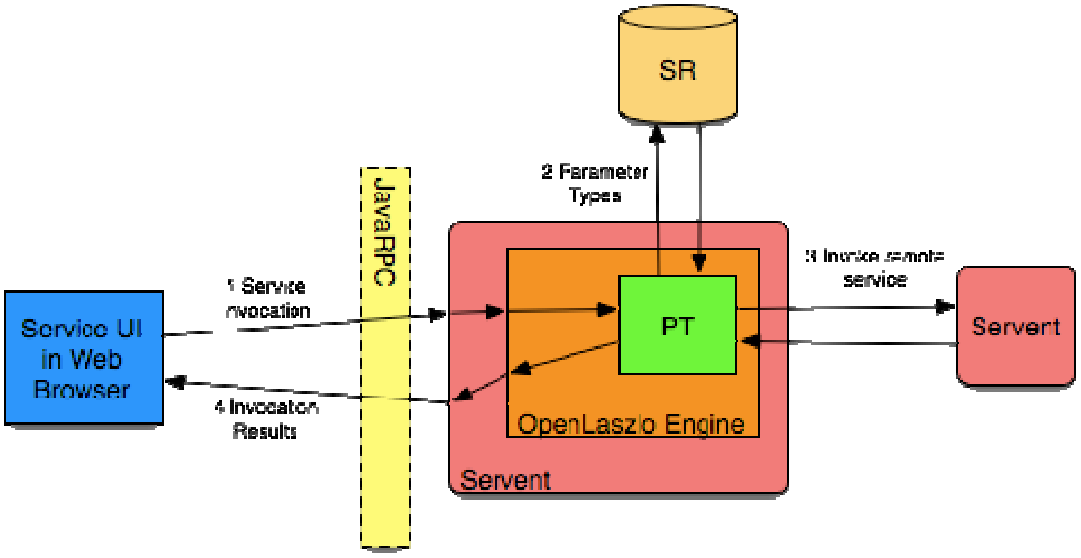


Figure 15 Remote Service Execution

6.3 Portal Toolkit Logical Structure

The portal toolkit is split into two logical parts. The first component is one which sits inside the OpenLaszlo engine, the DBE Portal Client. This component is shown below in the following Unified Modelling Language (UML) class diagram:

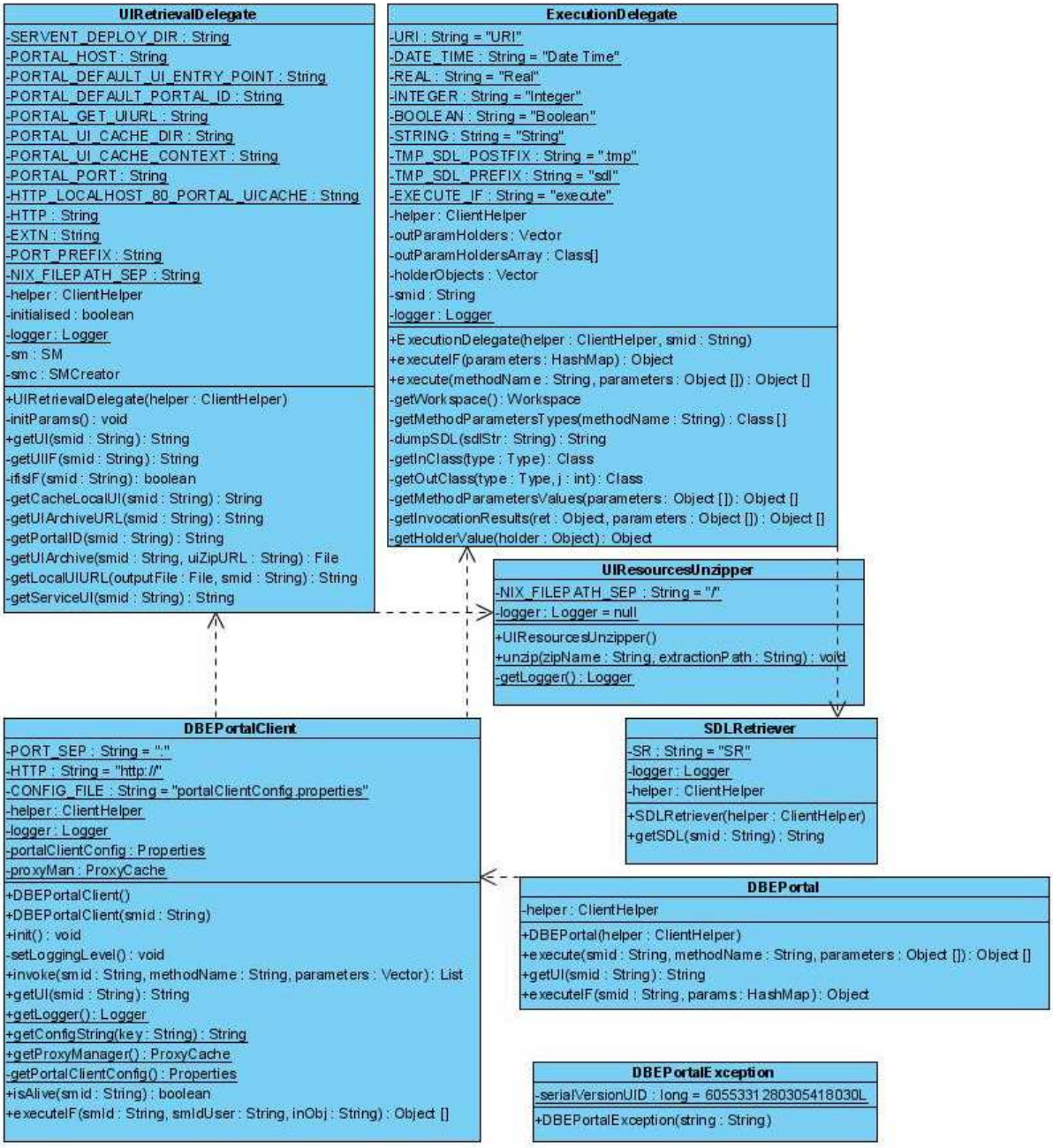


Figure 16 DBE Portal Client Class Diagram

The second logical part of the DBE Portal Toolkit is the DBE Portal service that runs as a service on the Servent. Its UML class diagram is shown as follows:

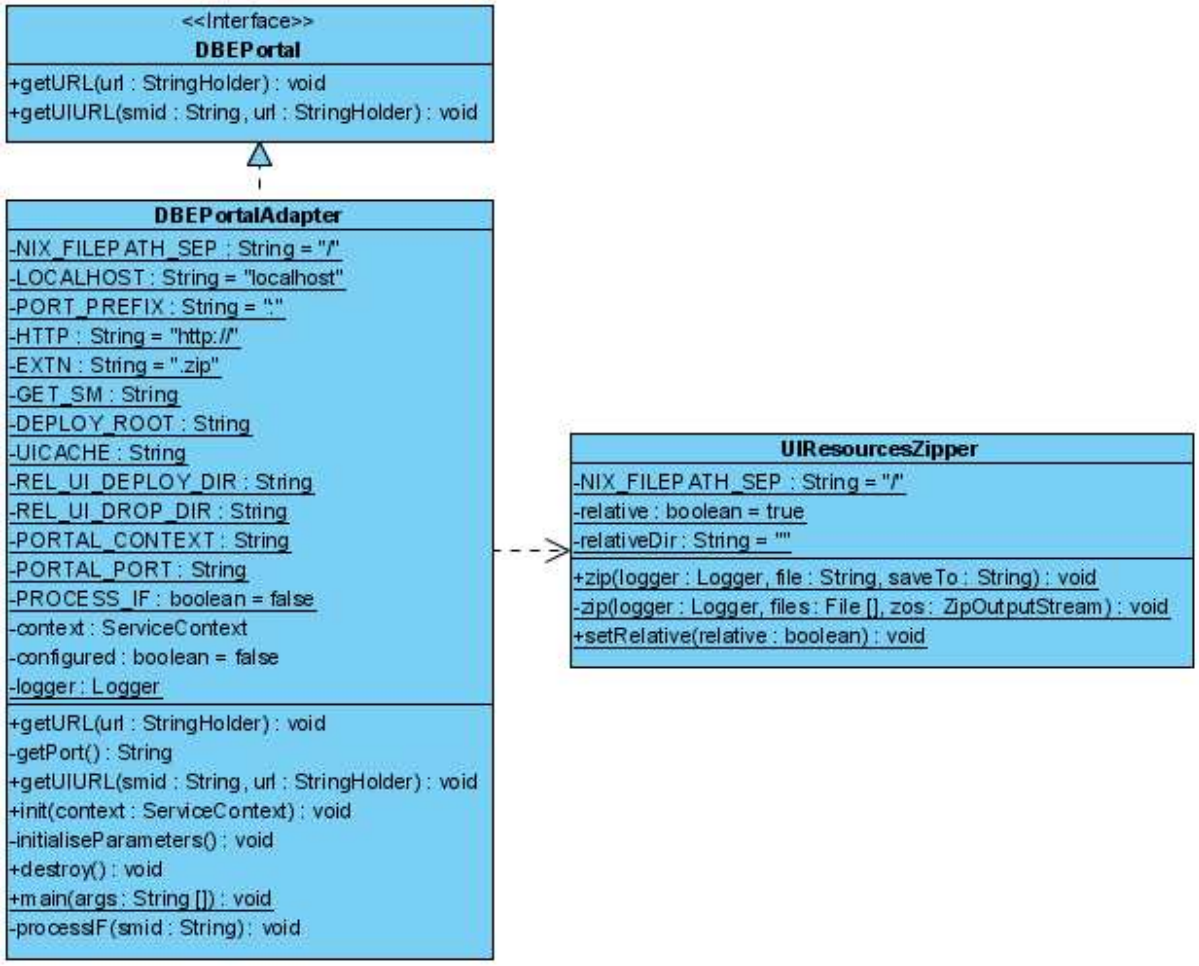


Figure 17 DBE Portal Service Class Diagram

6.4 Portal Toolkit Pending Issues

During the implementation and testing of the DBE Portal Toolkit an issue [9] of poor latency and performance was experienced. The symptom noted initially was service UI's timing out when RPC calls were executed to the backend service associated with the UI. On further investigation, it was found that a synchronous call to the P2P system (Federated Advanced Directory Architecture, FADA) API was causing a bottleneck and was impacted by the propagation delay when P2P system nodes propagate search requests for services. This issue has been solved by the inclusion of asynchronous searches in the implementation of FADA 5.3.5. Service proxy caching was also introduced into the implementation of the Portal Toolkit and this reduces the number of searches on the FADA network and in doing so reduces latency.

7 DBE Portal Dependencies

There are a number of components external to the DBE Portal on which it is dependent on. Synchronisation is required between the DBE Portal and these to ensure an approach that is well integrated and uses each component to its maximum potential and hopefully beyond. The components on which the DBE Portal is dependent (illustrated in Figure 18) on are as follows:

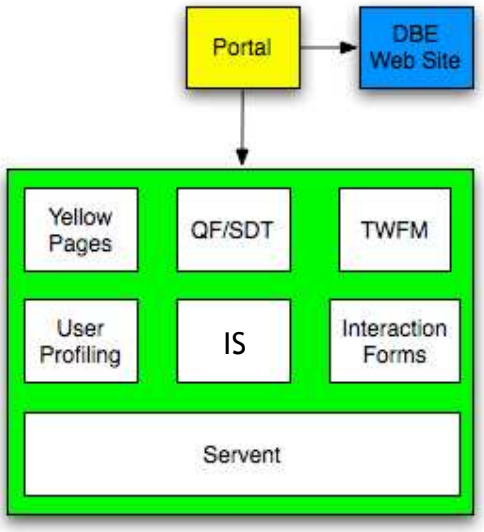


Figure 18 DBE Portal Interdependencies

- IS – In order for the DBE Portal to provide identity registration, validation and management, the Portal will require interfaces to the IS. The partner responsible for this component is TCD.
- Servent – This component will be required in order to provide to the DBE Portal information regarding its IP address visibility. The DBE Portal may also need the activation of the servent's underlying servlet engine that provides for servlets and Java Server Pages (JSP). The partner responsible for this component is SUN.
- User profiling – If supported by this component, DBE user preferences pertaining to the DBE Portal can be saved to this service. The partner responsible for this component is FZI.
- "Yellow pages" – This is intended to provide a business directory type interface for DBE service. The partner responsible for this component is Soluta.

- Interaction forms – Interaction forms will provide template UIs that a SME can customise if the SME does have the required skills to author a UI on their own behalf. The partner responsible for this component is Soluta.
- Query Formulator/Semantic Discovery Tool – This component will provide the means to search and browse the contents of the DBE network for services. An Openlaszlo version of it has been integrated into the portal. The partner responsible for this component is TUC.
- Transactional Workflow Manager – This component's interfaces will be required if long running transactions and notification of their completion is to be supported in the DBE Portal. The partner responsible for this component is University of Surrey.

8 Case Studies

In this section two common case studies are presented that SME's looking to either customise or integrate the default and generic portal may find useful. The first looks at creating a web site for a SME using the generic portal html interface that is supplied with a default installation of the portal. The second case looks at the case where an SME with an existing web site wishes to integrate the Portal search interface into their website.

8.1 Creating a Website Using the Portal

When a user first browses to the portal web site, web site that they will see is one as shown in the figure below.

Figure 19 Default Portal Website

To customise the default generic web site template (as shown in **Figure 19**) that comes with the DBE Portal a user needs to have basic knowledge of HTML. The DBE Portal website's content is held within the directory:

"<SERVENT_ROOT>/deploy/core-webapps/webapps/portal/content".

The content of each HTML file is described in the following table:

Filename	Purpose
aboutus.html	This file can be edited to provide a brief description of your business
admin.html	This file provides a means to administrate your portal
config.jsp	This file allows you to edit your portal configuration. Users will need some experience with Java Servlet Pages in order to modify this file
deploy.html	This file allows you to upload a DBE service to the servent and have it deployed
download.html	This file points to locations where the DBE software can be downloaded from
home.html	This is the first page that a user will see when the Portal is loaded for the first time in a web browser
search.html	This page loads the OpenLaszlo interface to allow the searching for and execution of DBE services

These files are styled by a CSS file that is located in the directory:

```
<SERVENT_ROOT>/deploy/core-webapps/webapps/portal/content/style
```

Users with CSS experience can then adjust the style and presentation of the content to suit their needs. Although these files are supplied by default, this fact does not preclude a SME from supplying additional ones.

With the content of the SME's portal created and edited, the final step in customising the Portal is to edit and modify the index file, located at:

```
<SERVENT_ROOT>/deploy/core-webapps/webapps/portal/index.html
```

This file binds the content HTML files together using frames. An example of a customised portal can be seen in **Figure 20** and is included in the Portal distribution.

Figure 20 Customised Portal Website

8.2 Integrating the Portal into an Existing Website

To integrate the DBE Portal into your existing website is a simple case of linking to a HTML file. The HTML file in question is the one located at:

```
<SERVENT_ROOT>/deploy/core-webapps/webapps/portal/content/search.html
```

This HTML file loads up the OpenLaszlo search interface where users can execute services returned in the search result page.

9 References

[1] DBE Desktop: http://sourceforge.net/project/showfiles.php?group_id=143906&package_id=167564

[2] SUN Microsystems; M20.1: “Draft User Interface Specification”. Available from http://www.digital-ecosystem.org/.

[3] FZI; Del 7.2: “Initial Description of Profiling mechanism design and rationale with respect to one or two use cases”. Available from http://www.digital-ecosystem.org/.

[4] Folksonomy; Vanderwal, T. (2005). "Off the Top: Folksonomy Entries"; http://www.vanderwal.net/random/category.php?cat=153

[5] STUN; “Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs)”; http://www.ietf.org/rfc/rfc3489.txt

[6] Openlaszlo; http://www.openlaszlo.org

[7] JavaRPC; http://www.laszlosystems.com/lps-3.1.1/docs/guide/rpc-javarpc.html

[8] DBE Servent; http://swallow.sf.net

[9] FADA latency issue; http://sourceforge.net/tracker/index.php?func=detail&aid=1415453&group_id=138511&atid=740885

[10] Dynamic Host Configuration Protocol (DHCP); http://www.ietf.org/rfc/rfc2131.txt

[11] Internet Protocol (IP); http://www.ietf.org/rfc/rfc791.txt

[12] The IP Network Address Translator (NAT); http://www.ietf.org/rfc/rfc1631.txt

[13] Universal Description, Discovery and Integration (UDDI); http://www.uddi.org

[14] Digital Business Ecosystem Execution Environment (ExE); http://swallow.sf.net

[15] Digital Business Ecosystem Website; http://www.digital-ecosystem.org

[16] SUN Microsystems Java Development Kit (JDK); http://www.java.com

10 Glossary

Acronym	Definition
APA	Abstract Protocol Adapter.
API	Application Programming Interface.
BML	Business Modelling Language.
CMS	Content Management System.
CSS	Client Side Servent.
DBE	Digital Business Ecosystem.
DIS	Distributed Identity Service.
DSS	Distributed Storage Service.
EvE	Evolutionary Environment.
ExE	Execution Environment.
FADA	Federated Advanced Directory Architecture.
IP	Internet Protocol.
ISP	Internet Service Provider.
JDK	Java Development Kit.
JSP	Java Server Pages.
KB	Knowledge Base.
NAT	Network Address Translator.
P2P	Peer-To-Peer.
PT	Portal Toolkit.
QF-SDT	Query Formulator/ Service Discovery Tool.
SDL	Service Definition Language.
SM	Service Manifest.
SME	Small-to-Medium Enterprise.
SR	Semantic Registry.
SSS	Server Side Servent.
TWFM	Transactional Workflow Manager.
UDDI	Universal Description, Discovery and Integration.
UI	User Interface.
UML	Unified Modelling Language.
WWW	World Wide Web.

11 Appendix

11.1 Configuring the DBE Portal

The portal can be configured in two main fashions:

- Through it's web administration interface
- Directly editing its configuration file

For those unfamiliar with configuring the portal, it maybe best to perform the configuration of the portal using the web interface. However, it is recommended that the portal be configured by directly editing the portal's configuration file.

The web interface to configuring the portal can be accessed at:

http://<YOUR_HOSTNAME><YOUR_PORT>/content/config.jsp

Here you can select a parameter to change and supply the required value for it. Do note that in order for the changes to occur you will need to restart the servent.

To directly edit the portal's configuration file you open the file located at:

<SERVENT_ROOT>/bin/portalClientConfiguration.properties

Then change the required parameters and corresponding values. Those parameters are documented within the configuration file. Again, for the changes to be reflected, the servent needs to be restarted.

JavaDoc for the Portal Toolkit

Package Summary		Page
org.dbe.kb.qi.utils		42
org.dbe.kb.toolkit.proxyutils		44
org.dbe.toolkit.portal		111
org.dbe.toolkit.portal.client		116
org.dbe.toolkit.portal.execution		124
org.dbe.toolkit.portal.interactionform		129
org.dbe.toolkit.portal.network		131
org.dbe.toolkit.portal.qfsdt		132
org.dbe.toolkit.portal.service		137
org.dbe.toolkit.portal.ui		152
org.dbe.toolkit.portal.ui.tools		160

Package org.dbe.kb.qi.utils

Class Summary		Page
ServiceDiscoverer		42

Class ServiceDiscoverer

org.dbe.kb.qi.utils

```
java.lang.Object
  └org.dbe.kb.qi.utils.ServiceDiscoverer
```

public class **ServiceDiscoverer**
extends Object

Field Summary		Page
private static long	clientCounter	43

Constructor Summary	Page
ServiceDiscoverer()	43

Method Summary		Page
static void	closeSession(String sessionId)	44
static String	getAvailabilityOfService(String id)	44
static synchronized Vector	getBMLModelFromId(String id)	43
static Vector	getModelElements(String modelElementParentPath)	44
static Vector	getModelIds(String keywordsAndId)	43
static Vector	getMoreResults(String sessionId)	43
static synchronized String	getQuerySessionId()	44
static Vector	getServices(String keywordsAndId)	43
static Vector	getSMInfo(String id)	43
static synchronized Vector	getSSLModelFromId(String id)	43
static String	getURLToExecuteService(String id)	43
static Vector	performAdvancedSearch(String arguments)	43

Field Detail

clientCounter

```
private static long clientCounter
```

Constructor Detail

ServiceDiscoverer

```
public ServiceDiscoverer()
```

Method Detail

getServices

```
public static Vector getServices(String keywordsAndId)
```

---

getModelIds

```
public static Vector getModelIds(String keywordsAndId)
```

---

getSSLModelFromId

```
public static synchronized Vector getSSLModelFromId(String id)
```

---

getBMLModelFromId

```
public static synchronized Vector getBMLModelFromId(String id)
```

---

getSMInfo

```
public static Vector getSMInfo(String id)
```

---

getMoreResults

```
public static Vector getMoreResults(String sessionId)
```

---

performAdvancedSearch

```
public static Vector performAdvancedSearch(String arguments)
```

---

getURLToExecuteService

```
public static String getURLToExecuteService(String id)
```

---

getQuerySessionId

```
public static synchronized String getQuerySessionId()
```

---

closeSession

```
public static void closeSession(String sessionId)
```

---

getModelElements

```
public static Vector getModelElements(String modelElementParentPath)
```

---

getAvailabilityOfService

```
public static String getAvailabilityOfService(String id)
```

Package org.dbe.kb.toolkit.proxyutils

Interface Summary		Page
IRootNode		64
NameAdapter		73

Class Summary		Page
AbstractNode	The AbstractNode class is the ancestor of all nodes providing a set of methods that must all implement.	45
AttributeNode	The AttributeNode class is the node for the Mof's Attribute classes of the tree control of FormulateQueryWizardPage class.	47
BmlInstanceNameAdapter		51
ContextProvider		52
ContextRoot	The ContextRoot class is the root of all nodes.	54
DefaultNameAdapter		62
ImmInstanceNameAdapter		63
Messages		68
MofHelper	The MofHelper class provides methods for accessing in a unified manner Mof Classes.	69
NavigationNode	The NavigationNode class contains a navigable object that has children.	73
OdmInstanceNameAdapter		76
Proxy		77
ProxyFinding		89
ProxyHandling	The ProxyHandler class is responsible to connect to a FADA proxy and handle the whole connection and the communication.	90

QueryNode		*100*
SslInstanceNameAdapter		*106*
ValueNode	The ValueNode class represents a constraint (either hard or soft) on it's parent AttributeNode.	*107*

Class AbstractNode

org.dbe.kb.toolkit.proxyutils

```
java.lang.Object
  └org.dbe.kb.toolkit.proxyutils.AbstractNode
```

Direct Known Subclasses:

AttributeNode, ContextRoot, NavigationNode, QueryNode, ValueNode

---

abstract public class **AbstractNode**
extends Object

The AbstractNode class is the ancestor of all nodes providing a set of methods that must all implement. It is a common interface of them in order to be easily explorable. They are nodes of the tree control of FormulateQueryWizardPage class.

Author:

George Kotopoulos

Version:

1.0

Field Summary		Page
protected NameAdapter	**nameAdapter** The name Adapter of the object	*46*
protected AbstractNode	**parent** The parent of this node	*46*

Constructor Summary	Page
AbstractNode()	*46*

Method Summary		Page
abstract Object[]	**getChildren**() Returns an array with this nodes children.	*46*
String	**getId**()	*47*
String	**getName**() Gets a String representation of the name of the node.	*46*
abstract Object	**getObject**() Gets the Object of the current node.	*47*
AbstractNode	**getParent**() Gets the parent of the current node.	*47*

abstract boolean	**hasChildren**() Returns true if this node has children.	*46*

Field Detail

nameAdapter

```
protected NameAdapter nameAdapter
```

The name Adapter of the object

---

parent

```
protected AbstractNode parent
```

The parent of this node

Constructor Detail

AbstractNode

```
public AbstractNode()
```

Method Detail

hasChildren

```
public abstract boolean hasChildren()
```

Returns true if this node has children. False if it is a leaf node.

Returns:

True if this node has children. False if it is a leaf node.

---

getChildren

```
public abstract Object[] getChildren()
```

Returns an array with this nodes children.

Returns:

An array with this nodes children.

---

getName

```
public String getName()
```

Gets a String representation of the name of the node.

Returns:

a String representation of the name of the node.

getId

```
public String getId()
```

getObject

```
public abstract Object getObject()
```

Gets the Object of the current node. Each node is holding an object. Usally a MOF object

Returns:

The Object of the current node

getParent

```
public AbstractNode getParent()
```

Gets the parent of the current node.

Returns:

The parent of the current node.

Class AttributeNode

org.dbe.kb.toolkit.proxyutils

```
java.lang.Object
  └org.dbe.kb.toolkit.proxyutils.AbstractNode
      └org.dbe.kb.toolkit.proxyutils.AttributeNode
```

public class **AttributeNode**
extends AbstractNode

The AttributeNode class is the node for the Mof's Attribute classes of the tree control of FormulateQueryWizardPage class.

Author:

George Kotopoulos

Version:

1.0

Field Summary		Page
private Vector	**children** The childer of this node	48
private RefObject	**object** The Mof Attribute object	48
private Vector	**pathToRoot** The path to root	49

Constructor Summary	Page
AttributeNode(RefObject node, AbstractNode parent, NameAdapter nameAdapter) Creates a new AttributeNode for a given Mof Attribute.	49

Method Summary		Page
ValueNode	**addValue**(String op, String val, int weight) Adds a child to this node.	50
private void	**constructPath**() Constructs the path to root.	51
Object[]	**getChildren**() Returns an array with this nodes children.	49
Object	**getObject**() Gets the Object of the current node.	49
String[]	**getOperations**() Gets the available operations for the Attribute object, depending on the type of the object.	50
Iterator	**getPathToRoot**() Gets an iterator containing all the nodes from this one to the root, starting from this one.	50
String	**getType**() Gets the a String representation of the type of the current object	50
boolean	**hasChildren**() Returns true if this node has children.	49
void	**removeValue**(ValueNode node) Removes a node from the children of this node.	50
static String	**xmlSchemaType2OclType**(String xmlSchemaType)	50

Methods inherited from class org.dbe.kb.toolkit.proxyutils.AbstractNode
getChildren, getId, getName, getObject, getParent, hasChildren

Field Detail

object

```
private RefObject object
```

The Mof Attribute object

---

children

```
private Vector children
```

The childer of this node

---

pathToRoot

```
private Vector pathToRoot
```

The path to root

Constructor Detail

AttributeNode

```
public AttributeNode(RefObject node,
                     AbstractNode parent,
                     NameAdapter nameAdapter)
```

Creates a new AttributeNode for a given Mof Attribute.

Method Detail

hasChildren

```
public boolean hasChildren()
```

Returns true if this node has children. False if it is a leaf node.

Overrides:

hasChildren in class AbstractNode

Returns:

True if this node has children. False if it is a leaf node.

See Also:

org.dbe.dtool.nodes.AbstractNode.hasChildren()

---

getChildren

```
public Object[] getChildren()
```

Returns an array with this nodes children.

Overrides:

getChildren in class AbstractNode

Returns:

An array with this nodes children.

See Also:

org.dbe.dtool.nodes.AbstractNode.getChildren()

---

getObject

```
public Object getObject()
```

Gets the Object of the current node. Each node is holding an Mof Attribute object

Overrides:

getObject in class AbstractNode

Returns:

The Mof Attribute object of the current node

See Also:
`org.dbe.dtool.nodes.AbstractNode.getObject()`

---

getType

```
public String getType()
```

Gets the a String representation of the type of the current object

---

xmlSchemaType2OclType

```
public static String xmlSchemaType2OclType(String xmlSchemaType)
```

---

getOperations

```
public String[] getOperations()
```

Gets the available operations for the Attribute object, depending on the type of the object.

Returns:
The available operations for the Attribute object, depending on the type of the object.

---

addValue

```
public ValueNode addValue(String op,
                          String val,
                          int weight)
```

Adds a child to this node. Constructs a ValueNode object.

Parameters:
`op` - A string representation of the operation
`val` - The literal value of the constraint
`weight` - true if it a hard constraint, false otherwise.

Returns:
The newely created child ValueNode.

---

removeValue

```
public void removeValue(ValueNode node)
```

Removes a node from the children of this node.

Parameters:
`node` - The ValueNode to be removed.

---

getPathToRoot

```
public Iterator getPathToRoot()
```

Gets an iterator containing all the nodes from this one to the root, starting from this one.

Returns:

An iterator containing all the nodes from this one to the root, starting from this one.

---

constructPath

```
private void constructPath()
```

Constructs the path to root.

Class BmlInstanceNameAdapter

org.dbe.kb.toolkit.proxyutils

```
java.lang.Object
  └org.dbe.kb.toolkit.proxyutils.BmlInstanceNameAdapter
```

All Implemented Interfaces:

NameAdapter

---

public class **BmlInstanceNameAdapter**
extends Object
implements NameAdapter

Author:

christos

Constructor Summary	Page
BmlInstanceNameAdapter()	51

Method Summary		Page
String	**getId**(RefBaseObject object)	52
String	**getName**(RefBaseObject object)	52

Methods inherited from interface org.dbe.kb.toolkit.proxyutils.NameAdapter
getId, getName

Constructor Detail

BmlInstanceNameAdapter

```
public BmlInstanceNameAdapter()
```

Method Detail

getName

```
public String getName(RefBaseObject object)
```

Specified by:

`getName` in interface `NameAdapter`

---

getId

```
public String getId(RefBaseObject object)
```

Specified by:

`getId` in interface `NameAdapter`

Class ContextProvider

org.dbe.kb.toolkit.proxyutils

```
java.lang.Object
  └org.dbe.kb.toolkit.proxyutils.ContextProvider
```

---

public class **ContextProvider**
extends Object

Field Summary		Page
private static Vector	**BMLContexts**	53
private static Vector	**IMMContexts**	53
private static Vector	**SDLContexts**	53
private static Vector	**SSLContexts**	53

Constructor Summary	Page
ContextProvider()	53

Method Summary		Page
static String	**getContextPathForClass**(String className, String metamodel) Returns the context path that the given class belongs	53
static Vector	**getContextVectorItems**(String metamodel) Returns a vector with the supported contexts for a metamodel	53
static Vector	**getContextVectorPaths**(String metamodel) Returns a vector with the supported contexts paths for a metamodel	53

Field Detail

BMLContexts

```
private static Vector BMLContexts
```

SSLContexts

```
private static Vector SSLContexts
```

SDLContexts

```
private static Vector SDLContexts
```

IMMContexts

```
private static Vector IMMContexts
```

Constructor Detail

ContextProvider

```
public ContextProvider()
```

Method Detail

getContextVectorItems

```
public static Vector getContextVectorItems(String metamodel)
```

Returns a vector with the supported contexts for a metamodel

Returns:

Vector with supported metamodel contexts

---

getContextVectorPaths

```
public static Vector getContextVectorPaths(String metamodel)
```

Returns a vector with the supported contexts paths for a metamodel

Returns:

Vector with supported metamodel contexts paths

---

getContextPathForClass

```
public static String getContextPathForClass(String className,
                                            String metamodel)
```

Returns the context path that the given class belongs

Parameters:
`className` - Context's last item name (RefClass name)
Returns:
String representing the context path

Class ContextRoot

org.dbe.kb.toolkit.proxyutils

```
java.lang.Object
  └org.dbe.kb.toolkit.proxyutils.AbstractNode
      └org.dbe.kb.toolkit.proxyutils.ContextRoot
```

All Implemented Interfaces:
IRootNode

---

public class **ContextRoot**
extends AbstractNode
implements IRootNode

The ContextRoot class is the root of all nodes. It reperesents the query and Mof Class hierarchy of the metamodel. It is the root node of the tree control of FormulateQueryWizardPage class.

Author:
George Kotopoulos
Version:
1.0

Field Summary		Page
private HashSet	**attributeNodes** The attribute nodes.	56
private Object[]	**children** The children of this node.	56
private RefObject[]	**context** The MofClass beeing the query context	56
private boolean	**isInstance**	57
private boolean	**isSelected**	57
private String	**metamodel** The metamodel name to which this query applies to	56
private String	**name** The query name	57
private Vector[]	**path** The package path to the MofClass beeing query context	56

Fields inherited from interface org.dbe.kb.toolkit.proxyutils.IRootNode
ALL, HARD, NONE, SOFT

Constructor Summary	*Page*
ContextRoot(Collection root, String name, String metamodel) Creates a new query, with a name and for a context.	*57*
ContextRoot(RefBaseObject root, String name, String metamodel) Creates a new query, with a name and for a context.	*57*
ContextRoot(RefBaseObject root, String name, QueryNode parent, String metamodel) Creates a new query, with a name and for a context.	*57*

	Method Summary	*Page*
protected boolean	**addNode**(ValueNode node) Adds a ValueNode to the query.	*59*
Object[]	**getChildren**() Returns an array with this nodes children.	*59*
RefObject	**getContext**(int index) Gets the context of the query.	*58*
String	**getMetamodel**() Gets a String representation of the query name.	*61*
String	**getName**() Gets a String representation of the name of the node.	*58*
Iterator	**getNodes**() Gets an iterator with the ValueNodes of this query.	*60*
Iterator	**getNodes**(boolean isHard) Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)	*61*
Collection	**getNodesAsCollection**() Gets an iterator with the ValueNodes of this query.	*61*
protected Vector	**getNodesAsVector**(boolean isHard) Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)	*61*
int	**getNodesFlag**() Gets a int flag for the type of the ValueNodes.	*60*
Object	**getObject**() Gets the Object of the current node.	*59*
Vector	**getPath**() Gets a Vector containg the package path to the context Mof class of this query.	*59*
Vector	**getPath**(int index)	*59*
String	**getQueryName**() Gets a String representation of the query name.	*58*
String	**getResultType**() Gets the result type of this context, i.e.	*58*
boolean	**hasChildren**() Returns true if this node has children.	*58*
boolean	**hasNodes**() Returns true is the query has at least one value node, false otherwise.	*60*
private void	**init**(Collection root, String name, String metamodel)	*57*
boolean	**isInstance**() Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)	*61*
boolean	**isSelected**()	*62*

protected boolean	**removeNode**(ValueNode node) Remove a ValueNode fr5om the query.	*60*
protected void	**setSelected**(boolean val)	*62*

Methods inherited from class org.dbe.kb.toolkit.proxyutils.AbstractNode
getChildren, getId, getName, getObject, getParent, hasChildren

Methods inherited from interface org.dbe.kb.toolkit.proxyutils.IRootNode
getChildren, getMetamodel, getNodes, getNodes, getNodesFlag, getPath, getQueryName, getResultType, hasNodes, isInstance

Field Detail

context

```
private RefObject[] context
```

The MofClass beeing the query context

---

children

```
private Object[] children
```

The children of this node. Has always only one, a NavigationNode

---

path

```
private Vector[] path
```

The package path to the MofClass beeing query context

---

attributeNodes

```
private HashSet attributeNodes
```

The attribute nodes. The actual constraints of this query.

---

metamodel

```
private String metamodel
```

The metamodel name to which this query applies to

---

name

```
private String name
```

The query name

---

isSelected

```
private boolean isSelected
```

---

isInstance

```
private boolean isInstance
```

Constructor Detail

ContextRoot

```
public ContextRoot(RefBaseObject root,
                   String name,
                   QueryNode parent,
                   String metamodel)
```

Creates a new query, with a name and for a context.

ContextRoot

```
public ContextRoot(RefBaseObject root,
                   String name,
                   String metamodel)
```

Creates a new query, with a name and for a context.

ContextRoot

```
public ContextRoot(Collection root,
                   String name,
                   String metamodel)
```

Creates a new query, with a name and for a context.

Method Detail

init

```
private void init(Collection root,
                  String name,
                  String metamodel)
```

---

getContext

```
public RefObject getContext(int index)
```

Gets the context of the query.

Returns:

The MofClass context of the query.

---

hasChildren

```
public boolean hasChildren()
```

Returns true if this node has children. False if it is a leaf node.

Overrides:

hasChildren in class AbstractNode

Returns:

True if this node has children. False if it is a leaf node.

See Also:

org.dbe.dtool.nodes.AbstractNode.hasChildren()

---

getName

```
public String getName()
```

Gets a String representation of the name of the node.

Overrides:

getName in class AbstractNode

Returns:

a String representation of the name of the node.

See Also:

org.dbe.dtool.nodes.AbstractNode.getName()

---

getQueryName

```
public String getQueryName()
```

Gets a String representation of the query name.

Specified by:

getQueryName in interface IRootNode

Returns:

a String representation of the query name.

---

getResultType

```
public String getResultType()
```

Gets the result type of this context, i.e. the metamodel.

Specified by:

getResultType in interface IRootNode

Returns:
The result type of this context, i.e. the metamodel.

---

getObject

```
public Object getObject()
```

Gets the Object of the current node. It is the context and its' type is MofClass
Overrides:
`getObject` in class `AbstractNode`
Returns:
The MofClass of the context
See Also:
`org.dbe.dtool.nodes.AbstractNode.getObject()`

---

getChildren

```
public Object[] getChildren()
```

Returns an array with this nodes children.
Specified by:
`getChildren` in interface `IRootNode`
Overrides:
`getChildren` in class `AbstractNode`
Returns:
An array with this nodes children.
See Also:
`org.dbe.dtool.nodes.AbstractNode.getChildren()`

---

getPath

```
public Vector getPath()
```

Gets a Vector containg the package path to the context Mof class of this query.
Specified by:
`getPath` in interface `IRootNode`
Returns:
a Vector containg the package path to the context Mof class of this query.

---

getPath

```
public Vector getPath(int index)
```

---

addNode

```
protected boolean addNode(ValueNode node)
```

Adds a ValueNode to the query.

Parameters:
`node` - The node to be added.
Returns:
true if added, false otherwise.

---

removeNode

```
protected boolean removeNode(ValueNode node)
```

Remove a ValueNode fr5om the query.
Parameters:
`node` - The ValueNode to be removed.
Returns:
true if removed successfully, false otherwise.

---

hasNodes

```
public boolean hasNodes()
```

Returns true is the query has at least one value node, false otherwise.
Specified by:
`hasNodes` in interface `IRootNode`
Returns:
true is the query has at least one value node, false otherwise.

---

getNodesFlag

```
public int getNodesFlag()
```

Gets a int flag for the type of the ValueNodes. Valid values are NONE (0), HARD (1), SOFT (2) and ALL (3)
Specified by:
`getNodesFlag` in interface `IRootNode`
Returns:
NONE (0), HARD (1), SOFT (2) and ALL (3).

---

getNodes

```
public Iterator getNodes()
```

Gets an iterator with the ValueNodes of this query.
Specified by:
`getNodes` in interface `IRootNode`
Returns:
The ValueNodes of this query.

---

getNodesAsCollection

```
public Collection getNodesAsCollection()
```

Gets an iterator with the ValueNodes of this query.

Returns:

The ValueNodes of this query.

---

getNodes

```
public Iterator getNodes(boolean isHard)
```

Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)

Specified by:

`getNodes` in interface `IRootNode`

Parameters:

`isHard` - if the ValueNodes to be returned are hard constrains or not.

Returns:

An Iterator with the ValueNodes of a certain type (hard or soft constrains)

---

getNodesAsVector

```
protected Vector getNodesAsVector(boolean isHard)
```

Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)

Parameters:

`isHard` - if the ValueNodes to be returned are hard constrains or not.

Returns:

An Iterator with the ValueNodes of a certain type (hard or soft constrains)

---

isInstance

```
public boolean isInstance()
```

Description copied from interface: `IRootNode`

Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)

Specified by:

`isInstance` in interface `IRootNode`

Returns:

An Iterator with the ValueNodes of a certain type (hard or soft constrains)

---

getMetamodel

```
public String getMetamodel()
```

Description copied from interface: `IRootNode`

Gets a String representation of the query name.

Specified by:

`getMetamodel` in interface `IRootNode`

Returns:
a String representation of the query name.

---

setSelected

```
protected void setSelected(boolean val)
```

---

isSelected

```
public boolean isSelected()
```

Class DefaultNameAdapter

org.dbe.kb.toolkit.proxyutils

```
java.lang.Object
  └ org.dbe.kb.toolkit.proxyutils.DefaultNameAdapter
```

All Implemented Interfaces:
NameAdapter

---

public class **DefaultNameAdapter**
extends Object
implements NameAdapter

Author:
gkoto

Constructor Summary	Page
DefaultNameAdapter()	62

Method Summary		Page
String	**getId**(RefBaseObject object)	63
String	**getName**(RefBaseObject object)	63

Methods inherited from interface org.dbe.kb.toolkit.proxyutils.NameAdapter
getId, getName

Constructor Detail

DefaultNameAdapter

```
public DefaultNameAdapter()
```

Method Detail

getName

```
public String getName(RefBaseObject object)
```

Specified by:
`getName` in interface `NameAdapter`

---

getId

```
public String getId(RefBaseObject object)
```

Specified by:
`getId` in interface `NameAdapter`

Class ImmInstanceNameAdapter

org.dbe.kb.toolkit.proxyutils

```
java.lang.Object
  └org.dbe.kb.toolkit.proxyutils.ImmInstanceNameAdapter
```

All Implemented Interfaces:
NameAdapter

---

public class **ImmInstanceNameAdapter**
extends Object
implements NameAdapter

Author:
christos

Constructor Summary	*Page*
ImmInstanceNameAdapter()	*64*

Method Summary		*Page*
String	**getId**(RefBaseObject object)	*64*
String	**getId**(RefBaseObject object, String metamodel)	*64*
String	**getName**(RefBaseObject object)	*64*
String	**getName**(RefObject object, String modelId, String metamodel, Hashtable hashTable)	*64*

Methods inherited from interface org.dbe.kb.toolkit.proxyutils.NameAdapter
getId, getName

Constructor Detail

ImmInstanceNameAdapter

```
public ImmInstanceNameAdapter()
```

Method Detail

getName

```
public String getName(RefBaseObject object)
```

Specified by:

getName in interface NameAdapter

---

getId

```
public String getId(RefBaseObject object)
```

Specified by:

getId in interface NameAdapter

---

getId

```
public String getId(RefBaseObject object,
                    String metamodel)
```

---

getName

```
public String getName(RefObject object,
                      String modelId,
                      String metamodel,
                      Hashtable hashTable)
```

Interface IRootNode

org.dbe.kb.toolkit.proxyutils

All Known Implementing Classes:

ContextRoot, QueryNode

---

public interface **IRootNode**

Author:

gkoto

Field Summary		Page
int	**ALL** a int flag for both hard and soft constrains.	66
int	**HARD** a int flag for hard constrains.	65
int	**NONE** a int flag for none constraint.	66
int	**SOFT** a int flag for soft constrains.	65

Method Summary		Page
Object[]	**getChildren**() Returns an array with this nodes children.	67
String	**getMetamodel**() Gets a String representation of the query name.	67
Iterator	**getNodes**() Gets an iterator with the ValueNodes of this query.	66
Iterator	**getNodes**(boolean isHard) Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)	67
int	**getNodesFlag**() Gets a int flag for the type of the ValueNodes.	66
Vector	**getPath**() Gets a Vector containg the package path to the context Mof class of this query.	67
String	**getQueryName**() Gets a String representation of the query name.	66
String	**getResultType**() Gets the result type of this context, i.e.	67
boolean	**hasNodes**() Returns true is the query has at least one value node, false otherwise.	66
boolean	**isInstance**() Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)	67

Field Detail

HARD

```
public static final int HARD
```

a int flag for hard constrains. The value is 1

SOFT

```
public static final int SOFT
```

a int flag for soft constrains. The value is 2

ALL

```
public static final int ALL
```

a int flag for both hard and soft constrains. The value is 3

NONE

```
public static final int NONE
```

a int flag for none constraint. The value is 0

Method Detail

getQueryName

```
public String getQueryName()
```

Gets a String representation of the query name.
Returns:
a String representation of the query name.

---

getNodesFlag

```
public int getNodesFlag()
```

Gets a int flag for the type of the ValueNodes. Valid values are NONE (0), HARD (1), SOFT (2) and ALL (3)
Returns:
NONE (0), HARD (1), SOFT (2) and ALL (3).

---

hasNodes

```
public boolean hasNodes()
```

Returns true is the query has at least one value node, false otherwise.
Returns:
true is the query has at least one value node, false otherwise.

---

getNodes

```
public Iterator getNodes()
```

Gets an iterator with the ValueNodes of this query.
Returns:
The ValueNodes of this query.

---

getNodes

```
public Iterator getNodes(boolean isHard)
```

Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)

Parameters:

`isHard` - if the ValueNodes to be returned are hard constrains or not.

Returns:

An Iterator with the ValueNodes of a certain type (hard or soft constrains)

---

isInstance

```
public boolean isInstance()
```

Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)

Returns:

An Iterator with the ValueNodes of a certain type (hard or soft constrains)

---

getMetamodel

```
public String getMetamodel()
```

Gets a String representation of the query name.

Returns:

a String representation of the query name.

---

getChildren

```
public Object[] getChildren()
```

Returns an array with this nodes children.

Returns:

An array with this nodes children.

See Also:

`org.dbe.dtool.nodes.AbstractNode.getChildren()`

---

getPath

```
public Vector getPath()
```

Gets a Vector containg the package path to the context Mof class of this query.

Returns:

a Vector containg the package path to the context Mof class of this query.

---

getResultType

```
public String getResultType()
```

Gets the result type of this context, i.e. the metamodel.

Returns:

The result type of this context, i.e. the metamodel.

Class Messages

org.dbe.kb.toolkit.proxyutils

```
java.lang.Object
  └org.dbe.kb.toolkit.proxyutils.Messages
```

---

public class **Messages**
extends Object

Field Summary		Page
private static final String	**BUNDLE_NAME**	68
private static final ResourceBundle	**RESOURCE_BUNDLE**	68

Constructor Summary		Page
private	**Messages**()	68

Method Summary		Page
static String	**getString**(String key)	68

Field Detail

BUNDLE_NAME

```
private static final String BUNDLE_NAME
```

RESOURCE_BUNDLE

```
private static final ResourceBundle RESOURCE_BUNDLE
```

Constructor Detail

Messages

```
private Messages()
```

Method Detail

getString

```
public static String getString(String key)
```

Class MofHelper

org.dbe.kb.toolkit.proxyutils

```
java.lang.Object
  └org.dbe.kb.toolkit.proxyutils.MofHelper
```

public class **MofHelper**
extends Object

The MofHelper class provides methods for accessing in a unified manner Mof Classes. It uses the JMI package for Mof model and the reflection package.

Author:
George Kotopoulos
Version:
1.0

Field Summary		Page
private static Hashtable	**hash** A hashtable containing for each type of data its hierarchy	70

Constructor Summary	Page
MofHelper()	70

Method Summary		Page
static Vector	**changeToNodes**(Vector input, AbstractNode parent, NameAdapter nameAdapter) Changes Mof classes into Nodes depending on the type of each class	72
private static Vector	**getAllContents**(MofClass metaObject) Gets the features of a MofClass.	72
static Vector	**getAllContentsAsNodes**(MofClass metaObject, AbstractNode parent, NameAdapter nameAdapter) Gets the features of a MofClass as Nodes	72
static Vector	**getFeatures**(Attribute metaObject, AbstractNode parent, NameAdapter nameAdapter) Retrieves all the features of a Mof Attribute (i.e References and Attributes).	71
static Vector	**getFeatures**(MofClass metaObject, AbstractNode parent, NameAdapter nameAdapter) Retrieves all the features of a MofClass (i.e References and Attributes)	70
static Vector	**getFeatures**(Reference metaObject, AbstractNode parent, NameAdapter nameAdapter) Retrieves all the features of a Mof Reference (i.e References and Attributes)	70
static Vector	**getFeatures**(RefObject object) Retrieves all the features of a RefObject (i.e M1 objects).	71
static Vector	**getFeatures**(RefObject object, AbstractNode parent, NameAdapter nameAdapter) Retrieves all the features of a RefObject (i.e M1 objects).	71
private static RefObject	**getObject**(RefObject object)	71

private static AbstractNode	**newNode**(RefObject object, AbstractNode parent, NameAdapter nameAdapter)	*71*
private static void	**retrieveHierarchy**(String metamodel) Constructs a hash and retrieves in it the class hierarchies of SSL's Core, Instantiation and Types packages.	*72*
private static void	**retrievePackageHierarchy**(RefPackage pack) Adds to a hash the class hierarchies of a package.	*72*

Field Detail

hash

```
private static Hashtable hash
```

A hashtable containing for each type of data its hierarchy

Constructor Detail

MofHelper

```
public MofHelper()
```

Method Detail

getFeatures

```
public static Vector getFeatures(MofClass metaObject,
                                 AbstractNode parent,
                                 NameAdapter nameAdapter)
```

Retrieves all the features of a MofClass (i.e References and Attributes)

Parameters:

`metaObject` - The MofClass whose features will be retrieved

`parent` - The parent node of this MofClass

Returns:

a Vector with the features as Nodes.

---

getFeatures

```
public static Vector getFeatures(Reference metaObject,
                                 AbstractNode parent,
                                 NameAdapter nameAdapter)
```

Retrieves all the features of a Mof Reference (i.e References and Attributes)

Parameters:

`metaObject` - The Reference whose features will be retrieved.

`parent` - The parent node of this class

Returns:

a vector with the features as Nodes.

---

getFeatures

```
public static Vector getFeatures(Attribute metaObject,
                                 AbstractNode parent,
                                 NameAdapter nameAdapter)
```

Retrieves all the features of a Mof Attribute (i.e References and Attributes). Attribute may be have as type a complex class whose features are retrieved.

Parameters:

`metaObject` - The Reference whose features will be retrieved.

`parent` - The parent node of this class

Returns:

a vector with the features as Nodes or null if Attribute is of primitive type.

---

getFeatures

```
public static Vector getFeatures(RefObject object,
                                 AbstractNode parent,
                                 NameAdapter nameAdapter)
```

Retrieves all the features of a RefObject (i.e M1 objects). Attributes may have as type a complex class whose features are retrieved.

Parameters:

`object` - The RefObject whose features will be retrieved.

`parent` - The parent node of this class

Returns:

a vector with the features as Nodes or null if Attribute is of primitive type.

---

getFeatures

```
public static Vector getFeatures(RefObject object)
```

Retrieves all the features of a RefObject (i.e M1 objects). Attributes may have as type a complex class whose features are retrieved.

Parameters:

`object` - The RefObject whose features will be retrieved.

Returns:

a vector with the features as Nodes or null if Attribute is of primitive type.

---

newNode

```
private static AbstractNode newNode(RefObject object,
                                    AbstractNode parent,
                                    NameAdapter nameAdapter)
```

---

getObject

```
private static RefObject getObject(RefObject object)
```

---

getAllContents

```
private static Vector getAllContents(MofClass metaObject)
```

Gets the features of a MofClass.

Parameters:

`metaObject` - The MofClass whose features will be returned

Returns:

a Vector of MofClasses with the features.

---

getAllContentsAsNodes

```
public static Vector getAllContentsAsNodes(MofClass metaObject,
                                           AbstractNode parent,
                                           NameAdapter nameAdapter)
```

Gets the features of a MofClass as Nodes

Parameters:

`metaObject` - The MofClass whose features will be returned

`parent` - The parent Node of the MofClass

Returns:

a vector with the features as Nodes.

---

changeToNodes

```
public static Vector changeToNodes(Vector input,
                                   AbstractNode parent,
                                   NameAdapter nameAdapter)
```

Changes Mof classes into Nodes depending on the type of each class

Parameters:

`input` - A Vector with the features found

`parent` - The parent Node of the features

Returns:

a vector with the features as Nodes.

---

retrieveHierarchy

```
private static void retrieveHierarchy(String metamodel)
```

Constructs a hash and retrieves in it the class hierarchies of SSL's Core, Instantiation and Types packages. These hierarchies are then used for retrieving the children of an abstract class

---

retrievePackageHierarchy

```
private static void retrievePackageHierarchy(RefPackage pack)
```

Adds to a hash the class hierarchies of a package. These hierarchies are then used for retrieving the children of an abstract class.

Parameters:

`pack` - The RefPackage whose class hierarchies will be retrieved.

Interface NameAdapter

org.dbe.kb.toolkit.proxyutils

All Known Implementing Classes:

BmlInstanceNameAdapter, DefaultNameAdapter, ImmInstanceNameAdapter, OdmInstanceNameAdapter, SslInstanceNameAdapter

---

public interface **NameAdapter**

Author:

gkoto

Method Summary		Page
`String`	**`getId`**`(RefBaseObject object)`	*73*
`String`	**`getName`**`(RefBaseObject object)`	*73*

Method Detail

getName

```
public String getName(RefBaseObject object)
```

---

getId

```
public String getId(RefBaseObject object)
```

Class NavigationNode

org.dbe.kb.toolkit.proxyutils

```
java.lang.Object
  └org.dbe.kb.toolkit.proxyutils.AbstractNode
      └org.dbe.kb.toolkit.proxyutils.NavigationNode
```

---

public class **NavigationNode**
extends AbstractNode

The NavigationNode class contains a navigable object that has children. Navigable object can be all the model elements that have any type of feature; i.e Attributes, Operations, AssotiationEnds etc. They are nodes of the tree control of FormulateQueryWizardPage class.

Author:

George Kotopoulos

Version:
1.0

Field Summary		Page
private Object[]	**children** This node's children	74
static Object[]	**EMPTY_ARRAY**	75
private RefObject	**object** The Mof object of this NavigationNode.	74

Constructor Summary	Page
NavigationNode(RefObject node, AbstractNode parent, NameAdapter nameAdapter) Create a NavigationNode for a navigable object having a certain parent.	75

Method Summary		Page
private void	**calcChildren**() Calculates the children of this node.	75
Object[]	**getChildren**() Returns an array with this nodes children.	76
String	**getName**() Gets a String representation of the name of the node.	75
Object	**getObject**() Gets the Object of the current node.	75
AbstractNode	**getParent**() Gets the parent of the current node.	76
boolean	**hasChildren**() Returns true if this node has children.	75

Methods inherited from class org.dbe.kb.toolkit.proxyutils.AbstractNode
getChildren, getId, getName, getObject, getParent, hasChildren

Field Detail

object

```
private RefObject object
```

The Mof object of this NavigationNode. This object is navigable.

---

children

```
private Object[] children
```

This node's children

EMPTY_ARRAY

```
public static Object[] EMPTY_ARRAY
```

Constructor Detail

NavigationNode

```
public NavigationNode(RefObject node,
                      AbstractNode parent,
                      NameAdapter nameAdapter)
```

Create a NavigationNode for a navigable object having a certain parent.

Method Detail

calcChildren

```
private void calcChildren()
```

Calculates the children of this node. Children are computed on first demand and not when constructing the node for computaional efficiency purposes.

---

hasChildren

```
public boolean hasChildren()
```

Returns true if this node has children. False if it is a leaf node.

Overrides:

hasChildren in class AbstractNode

Returns:

True if this node has children. False if it is a leaf node.

See Also:

`org.dbe.dtool.nodes.AbstractNode.hasChildren()`

---

getName

```
public String getName()
```

Gets a String representation of the name of the node.

Overrides:

getName in class AbstractNode

Returns:

a String representation of the name of the node.

See Also:

`org.dbe.dtool.nodes.AbstractNode.getName()`

---

getObject

```
public Object getObject()
```

Gets the Object of the current node. Each node is holding an object. It is a Mof ModelElement object.

Overrides:

`getObject` in class `AbstractNode`

Returns:

The Object of the current node

---

getChildren

```
public Object[] getChildren()
```

Returns an array with this nodes children.

Overrides:

`getChildren` in class `AbstractNode`

Returns:

An array with this nodes children.

See Also:

`org.dbe.dtool.nodes.AbstractNode.getChildren()`

---

getParent

```
public AbstractNode getParent()
```

Gets the parent of the current node.

Overrides:

`getParent` in class `AbstractNode`

Returns:

The parent of the current node.

See Also:

`AbstractNode.getParent()`

Class OdmInstanceNameAdapter

org.dbe.kb.toolkit.proxyutils

```
java.lang.Object
  └org.dbe.kb.toolkit.proxyutils.OdmInstanceNameAdapter
```

All Implemented Interfaces:

NameAdapter

---

public class **OdmInstanceNameAdapter**
extends Object
implements NameAdapter

Constructor Summary	Page
OdmInstanceNameAdapter()	77

Method Summary		Page
String	**getId**(RefBaseObject object)	*77*
String	**getName**(RefBaseObject object)	*77*

Methods inherited from interface org.dbe.kb.toolkit.proxyutils.NameAdapter
getId, getName

Constructor Detail

OdmInstanceNameAdapter

```
public OdmInstanceNameAdapter()
```

Method Detail

getName

```
public String getName(RefBaseObject object)
```

Specified by:
getName in interface NameAdapter

---

getId

```
public String getId(RefBaseObject object)
```

Specified by:
getId in interface NameAdapter

Class Proxy

org.dbe.kb.toolkit.proxyutils

```
java.lang.Object
  └org.dbe.kb.toolkit.proxyutils.Proxy
```

---

public class **Proxy**
extends Object

Author:
gkoto

Field Summary		Page
private org.dbe.kb.mdrman.JMItool	**bmlModelsTool**	*80*
private org.dbe.kb.mdrman.JMItool	**bmlTool**	*80*
private org.dbe.kb.mdrman.JMItool	**immModelsTool**	*80*

private org.dbe.kb.mdrman.JMItool	**immTool**	*80*
private org.dbe.kb.service.KBI	**kbService**	*79*
private org.dbe.kb.mdrman.JMItool	**odmModelsTool**	*81*
private org.dbe.kb.mdrman.JMItool	**odmTool**	*80*
private org.dbe.kb.mdrman.JMItool	**qmlTool**	*80*
private String	**querySessionId**	*81*
private org.dbe.kb.mdrman.JMItool	**sdlModelsTool**	*80*
private org.dbe.kb.mdrman.JMItool	**sdlTool**	*80*
private org.dbe.kb.sm.SMtool	**smTool**	*81*
private org.dbe.kb.service.SRI	**srService**	*80*
private org.dbe.kb.mdrman.JMItool	**sslModelsTool**	*80*
private org.dbe.kb.mdrman.JMItool	**sslTool**	*80*

Constructor Summary	***Page***
Proxy(Object service)	*81*

Method Summary		***Page***
private void	**addModel**(String metamodel, String id)	*87*
void	**clearTools**()	*88*
void	**closeSession**() Closes the current session	*88*
void	**closeSession**(String sessionId)	*88*
void	**createQuerySession**()	*88*
void	**exportModel**(ByteArrayOutputStream out)	*86*
Collection	**findDepSMs**(String sessionId, String metamodel, String modelId)	*87*
String	**getBML**()	*82*
String	**getBMLModel**(String id, String nodeId)	*88*
private org.dbe.kb.mdrman.JMItool	**getBmlTool**()	*84*
private org.dbe.kb.mdrman.JMItool	**getBmlTool**(String modelId, String bmlModelId, String nodeId, boolean isSM)	*85*
String	**getCurrentQuerySessionId**()	*89*
private String	**getIMM**()	*82*
private org.dbe.kb.mdrman.JMItool	**getImmTool**()	*85*
private org.dbe.kb.mdrman.JMItool	**getImmTool**(String modelId)	*85*
void	**getModel**(String metamodel, String id, boolean isSM, String nodeId)	*87*
void	**getModel**(String metamodel, String id, String nodeId, boolean isSM)	*86*
RefPackage	**getModelPackage**(String metamodel)	*84*
RefPackage	**getModelPackage**(String metamodel, String nodeId, String modelId, boolean isSM)	*84*
String	**getNewSession**()	*88*
private String	**getODM**()	*81*

private org.dbe.kb.mdrman.JMItool	**getOdmTool**()	*85*
private org.dbe.kb.mdrman.JMItool	**getOdmTool**(String modelId)	*85*
private String	**getQML**()	*82*
org.dbe.kb.metamodel.qml.QmlPackage	**getQMLpackage**()	*84*
private org.dbe.kb.mdrman.JMItool	**getQmlTool**()	*85*
Collection	**getResults**()	*83*
Collection	**getResults**(String sessionId)	*83*
private String	**getSDL**()	*82*
private org.dbe.kb.mdrman.JMItool	**getSdlTool**()	*86*
private org.dbe.kb.mdrman.JMItool	**getSdlTool**(String modelId)	*86*
String	**getSM**(String smId, String nodeId)	*83*
org.dbe.kb.sm.SMtool	**getSMInfo**(String smid, String nodeId)	*88*
private String	**getSSL**()	*81*
String	**getSSLModel**(String id, String nodeId)	*88*
private org.dbe.kb.mdrman.JMItool	**getSslTool**()	*86*
private org.dbe.kb.mdrman.JMItool	**getSslTool**(String modelId, String bmlModelId, String nodeId, boolean isSM)	*86*
private org.dbe.kb.mdrman.JMItool	**getTool**(String metamodel)	*84*
private org.dbe.kb.mdrman.JMItool	**getTool**(String metamodel, String modelId, String nodeId, boolean isSM)	*84*
private void	**importModel**(String metamodel, String modelId, String id, String nodeId, boolean isSM)	*87*
boolean	**isAlive**()	*88*
boolean	**isSR**()	*86*
void	**newQuery**()	*87*
String	**retrieveBMLdata**(String smId, String nodeId)	*83*
String	**retrieveBMLmodel**(String modelId, String nodeId, boolean isSM)	*83*
String	**retrieveIMMmodel**(String modelId)	*82*
void	**retrieveModel**(String metamodel, String id, boolean isSM, String nodeId)	*87*
void	**retrieveModel**(String metamodel, String id, String nodeId, boolean isSM)	*87*
String	**retrieveODMmodel**(String modelId)	*81*
String	**retrieveSDLmodel**(String modelId)	*82*
String	**retrieveSSLdata**(String smId, String nodeId)	*84*
String	**retrieveSSLmodel**(String modelId, String nodeId, boolean isSM)	*82*
Collection	**searchBMLByKeywords**(String mm, Collection keywords)	*88*
Collection	**searchByKeywords**(String mm, Collection keywords)	*87*
void	**setService**(Object service)	*81*
Collection	**submitQuery**()	*83*

Field Detail

kbService

```
private org.dbe.kb.service.KBI kbService
```

srService

```
private org.dbe.kb.service.SRI srService
```

---

qmlTool

```
private org.dbe.kb.mdrman.JMItool qmlTool
```

---

sslTool

```
private org.dbe.kb.mdrman.JMItool sslTool
```

---

bmlTool

```
private org.dbe.kb.mdrman.JMItool bmlTool
```

---

sdlTool

```
private org.dbe.kb.mdrman.JMItool sdlTool
```

---

immTool

```
private org.dbe.kb.mdrman.JMItool immTool
```

---

odmTool

```
private org.dbe.kb.mdrman.JMItool odmTool
```

---

sslModelsTool

```
private org.dbe.kb.mdrman.JMItool sslModelsTool
```

---

bmlModelsTool

```
private org.dbe.kb.mdrman.JMItool bmlModelsTool
```

---

sdlModelsTool

```
private org.dbe.kb.mdrman.JMItool sdlModelsTool
```

---

immModelsTool

```
private org.dbe.kb.mdrman.JMItool immModelsTool
```

odmModelsTool

```
private org.dbe.kb.mdrman.JMItool odmModelsTool
```

smTool

```
private org.dbe.kb.sm.SMtool smTool
```

querySessionId

```
private String querySessionId
```

Constructor Detail

Proxy

```
public Proxy(Object service)
```

Method Detail

setService

```
public void setService(Object service)
```

getODM

```
private String getODM()
```

See Also:

org.dbe.kb.service.KBI.getODM()

retrieveODMmodel

```
public String retrieveODMmodel(String modelId)
```

See Also:

org.dbe.kb.service.KBI.retrieveODMmodel(java.lang.String)

getSSL

```
private String getSSL()
```

See Also:

org.dbe.kb.service.KBI.getSSL()

retrieveSSLmodel

```
public String retrieveSSLmodel(String modelId,
                               String nodeId,
                               boolean isSM)
```

See Also:
org.dbe.kb.service.KBI.retrieveSSLmodel(java.lang.String)

---

getSDL

```
private String getSDL()
```

See Also:
org.dbe.kb.service.KBI.getSDL()

---

retrieveSDLmodel

```
public String retrieveSDLmodel(String modelId)
```

See Also:
org.dbe.kb.service.KBI.retrieveSDLmodel(java.lang.String)

---

getBML

```
public String getBML()
```

See Also:
org.dbe.kb.service.KBI.getBML()

---

getIMM

```
private String getIMM()
```

See Also:
org.dbe.kb.service.KBI.getIMM()

---

retrieveIMMmodel

```
public String retrieveIMMmodel(String modelId)
```

See Also:
org.dbe.kb.service.KBI.retrieveIMMmodel(java.lang.String)

---

getQML

```
private String getQML()
```

See Also:
org.dbe.kb.service.KBI.getQML()

---

submitQuery

```
public Collection submitQuery()
                     throws Exception
```

Throws:
Exception
See Also:
org.dbe.kb.service.KBI.submitQuery(java.lang.String, java.lang.String)

---

getResults

```
public Collection getResults()
```

See Also:
org.dbe.kb.service.KBI.getResults(java.lang.String)

---

getResults

```
public Collection getResults(String sessionId)
```

See Also:
org.dbe.kb.service.KBI.getResults(java.lang.String)

---

getSM

```
public String getSM(String smId,
                    String nodeId)
```

See Also:
org.dbe.kb.service.SRI.getSM(java.lang.String)

---

retrieveBMLdata

```
public String retrieveBMLdata(String smId,
                              String nodeId)
```

See Also:
org.dbe.kb.service.SRI.retrieveBMLdata(java.lang.String)

---

retrieveBMLmodel

```
public String retrieveBMLmodel(String modelId,
                               String nodeId,
                               boolean isSM)
```

See Also:
`org.dbe.kb.service.KBI.retrieveIMMmodel(java.lang.String)`

---

retrieveSSLdata

```
public String retrieveSSLdata(String smId,
                              String nodeId)
```

See Also:
`org.dbe.kb.service.SRI.retrieveSSLdata(java.lang.String)`

---

getQMLpackage

```
public org.dbe.kb.metamodel.qml.QmlPackage getQMLpackage()
```

---

getModelPackage

```
public RefPackage getModelPackage(String metamodel)
```

---

getModelPackage

```
public RefPackage getModelPackage(String metamodel,
                                  String nodeId,
                                  String modelId,
                                  boolean isSM)
```

---

getTool

```
private org.dbe.kb.mdrman.JMItool getTool(String metamodel,
                                          String modelId,
                                          String nodeId,
                                          boolean isSM)
```

---

getTool

```
private org.dbe.kb.mdrman.JMItool getTool(String metamodel)
```

---

getBmlTool

```
private org.dbe.kb.mdrman.JMItool getBmlTool()
```

Returns:
Returns the bmlTool.

---

getBmlTool

```
private org.dbe.kb.mdrman.JMItool getBmlTool(String modelId,
                                             String bmlModelId,
                                             String nodeId,
                                             boolean isSM)
```

Returns:

Returns the bmlTool.

---

getImmTool

```
private org.dbe.kb.mdrman.JMItool getImmTool()
```

Returns:

Returns the immTool.

---

getImmTool

```
private org.dbe.kb.mdrman.JMItool getImmTool(String modelId)
```

Returns:

Returns the ImmTool.

---

getOdmTool

```
private org.dbe.kb.mdrman.JMItool getOdmTool()
```

Returns:

Returns the odmTool.

---

getOdmTool

```
private org.dbe.kb.mdrman.JMItool getOdmTool(String modelId)
```

Returns:

Returns the OdmTool.

---

getQmlTool

```
private org.dbe.kb.mdrman.JMItool getQmlTool()
```

Returns:

Returns the qmlTool.

---

getSdlTool

```
private org.dbe.kb.mdrman.JMItool getSdlTool()
```

Returns:
Returns the sdlTool.

---

getSdlTool

```
private org.dbe.kb.mdrman.JMItool getSdlTool(String modelId)
```

Returns:
Returns the SdlTool.

---

getSslTool

```
private org.dbe.kb.mdrman.JMItool getSslTool()
```

Returns:
Returns the sslTool.

---

getSslTool

```
private org.dbe.kb.mdrman.JMItool getSslTool(String modelId,
                                             String bmlModelId,
                                             String nodeId,
                                             boolean isSM)
```

Returns:
Returns the SslTool.

---

isSR

```
public boolean isSR()
```

---

exportModel

```
public void exportModel(ByteArrayOutputStream out)
                 throws IOException
```

Throws:
IOException

---

getModel

```
public void getModel(String metamodel,
                     String id,
                     String nodeId,
                     boolean isSM)
```

getModel

```
public void getModel(String metamodel,
                     String id,
                     boolean isSM,
                     String nodeId)
```

retrieveModel

```
public void retrieveModel(String metamodel,
                          String id,
                          String nodeId,
                          boolean isSM)
```

retrieveModel

```
public void retrieveModel(String metamodel,
                          String id,
                          boolean isSM,
                          String nodeId)
```

importModel

```
private void importModel(String metamodel,
                         String modelId,
                         String id,
                         String nodeId,
                         boolean isSM)
```

addModel

```
private void addModel(String metamodel,
                      String id)
```

findDepSMs

```
public Collection findDepSMs(String sessionId,
                             String metamodel,
                             String modelId)
```

newQuery

```
public void newQuery()
```

searchByKeywords

```
public Collection searchByKeywords(String mm,
                                   Collection keywords)
```

searchBMLByKeywords

```
public Collection searchBMLByKeywords(String mm,
                                      Collection keywords)
```

---

getSMInfo

```
public org.dbe.kb.sm.SMtool getSMInfo(String smid,
                                      String nodeId)
```

---

closeSession

```
public void closeSession()
```

Closes the current session

---

closeSession

```
public void closeSession(String sessionId)
```

---

createQuerySession

```
public void createQuerySession()
```

---

getNewSession

```
public String getNewSession()
```

---

getSSLModel

```
public String getSSLModel(String id,
                          String nodeId)
```

---

getBMLModel

```
public String getBMLModel(String id,
                          String nodeId)
```

---

isAlive

```
public boolean isAlive()
```

---

clearTools

```
public void clearTools()
```

---

getCurrentQuerySessionId

```
public String getCurrentQuerySessionId()
```

Class ProxyFinding

org.dbe.kb.toolkit.proxyutils

```
java.lang.Object
  └org.dbe.kb.toolkit.proxyutils.ProxyFinding
```

---

public class **ProxyFinding**
extends Object

Field Summary		Page
private ProxyHandling	**proxy**	89

Constructor Summary	Page
ProxyFinding()	89

Method Summary		Page
static String	**getContextPathForClass**(String className, String metamodel)	90
static Vector	**getContextVectorItems**(String metamodel)	90
static Vector	**getContextVectorPaths**(String metamodel)	90
ProxyHandling	**getProxyHandling**() Gets the shared instance of the ProxyHandler	89
private ProxyHandling	**getProxyHandling**(ProxyHandling proxy)	90

Field Detail

proxy

```
private ProxyHandling proxy
```

Constructor Detail

ProxyFinding

```
public ProxyFinding()
```

Method Detail

getProxyHandling

```
public ProxyHandling getProxyHandling()
```

Gets the shared instance of the ProxyHandler

Returns:
The shared instance of the ProxyHandler

---

getProxyHandling

```
private ProxyHandling getProxyHandling(ProxyHandling proxy)
```

---

getContextVectorItems

```
public static Vector getContextVectorItems(String metamodel)
```

---

getContextVectorPaths

```
public static Vector getContextVectorPaths(String metamodel)
```

---

getContextPathForClass

```
public static String getContextPathForClass(String className,
                                            String metamodel)
```

Class ProxyHandling

org.dbe.kb.toolkit.proxyutils

```
java.lang.Object
  └org.dbe.kb.toolkit.proxyutils.ProxyHandling
```

---

public class **ProxyHandling**
extends Object

The ProxyHandler class is responsible to connect to a FADA proxy and handle the whole connection and the communication. The Proxy can be set by an independent module, in order not to reconnect from this tool. To connect from an independent module see RegisterDiscoveryTool class.

Author:
George Kotopoulos
Version:
1.0
See Also:
`org.dbe.studio.tools.discovery.RegisterDiscoveryTool`

Field Summary		Page
private static final String	**CONFIG_FILE**	93
protected static Hashtable	**models**	93
protected static Hashtable	**ontologies**	93
private static Properties	**portalClientConfig**	93

private Proxy	**proxy** The Proxy	*93*
protected org.dbe.kb.metamodel.qml.QmlPackage	**qml** The oql Package	*93*

Constructor Summary	***Page***
ProxyHandling() if no query proxy set.	*93*

Method Summary	
protected RefBaseObject	**changeEnvironment**(RefBaseObject obj, String pathItem)
void	**closeSession**()
void	**closeSession**(String sessionId)
protected boolean	**containsOdmProperty**(org.dbe.kb.metamodel.odm.properties.property, org.dbe.kb.metamodel.odm.classes.Odmclass odm
void	**findProxy**() Connects to the FADA server of the properties (dtool.p file and finds a proxy.
Collection	**getAsynchronousResults**() Submits the query and returns a Vector with the results
static String	**getConfigString**(String key)
RefBaseObject	**getContext**(Vector path) For a Vector of String constructs the context reqiered.
String	**getCurrentQuerySessionId**()
protected RefPackage	**getInstantiationModel**() Gets the SSL metamodel
RefPackage	**getMetamodel**(String modelName) Gets a metamodel's RefPackage
String	**getMetamodelName**(int id)
Collection	**getModel**(String metamodel, String id, String nodeId, b isSM) For a string id of a model in a metamodel get the instar RefObject of the root)
protected org.dbe.kb.metamodel.qml.contextdeclarations.QueryContextDecl	**getNewQuery**(Vector path, String name, org.dbe.kb.metamodel.qml.ocl.expressions.OclExpression String result) Gets a new QueryContextDecl for a conetxt path
String	**getNewSessionId**()
Vector	**getObjects**(String metamodel, String modelId, String no Returns all instances of the classes contained in this Pa its contained packages.
Vector	**getObjects**(RefPackage refPack) Returns all instances of the classes contained in this Pa its contained packages.
RefObject	**getOdmClass**(String ontId, String id) For a string id of a model in a metamodel get the instar RefObject of the root)

Vector	**getOdmProperties**(org.dbe.kb.metamodel.odm.classes.Odmc odmclass) For an OdmClass retrieve its properties
Vector	**getOdmRange**(org.dbe.kb.metamodel.odm.properties.ObjectI property) For a string id of a model in a metamodel get the instar RefObject of the root)
private static Properties	**getPortalClientConfig**()
Proxy	**getProxy**() Retrieves the current proxy object which has a number functions
org.dbe.kb.metamodel.qml.QmlPackage	**getQml**() Gets the current Oql Package.
Collection	**getResults**(String sessionId)
RefClass	**getRootClass**(String mm)
RefClass[]	**getRootClasses**()
Vector	**getRoots**(String mm)
protected RefClass	**getSemanticPackage**() Gets the SemanticPackage (the root class of the SSL m
Collection	**getServiceManifest**(String sessionId, String metamodel, modelId)
org.dbe.kb.sm.SMtool	**getSMInfo**(String smId, String nodeId)
protected RefPackage	**getSslModel**() Gets the SSL metamodel
RefPackage	**getSslSpecificPackage**(String packName) Gets an SSL's specific package ("Core", "Types", etc)
boolean	**isAlive**()
boolean	**isInstantatiatable**(String metamodel) Return whether the metmodel specified by id is instant
boolean	**isSM**(String smId)
boolean	**isSR**() Returns true if the proxy coonected to is a ServiceRegi
protected org.dbe.kb.metamodel.odm.OdmPackage	**loadOntology**(String id) For a string id of a model in a metamodel get the instar RefObject of the root)
void	**newQuery**() Instantiates a new Query
void	**printOql**() Prints The QML Expression as an XMI in standard out
Object	**run**()
Collection	**searchBMLByKeywords**(String metamodel, Vector keywords)
Collection	**searchByKeywords**(String metamodel, Vector keywords)
protected void	**setService**(Object service)

Field Detail

proxy

```
private Proxy proxy
```

The Proxy

---

qml

```
protected org.dbe.kb.metamodel.qml.QmlPackage qml
```

The oql Package

---

ontologies

```
protected static Hashtable ontologies
```

---

models

```
protected static Hashtable models
```

---

portalClientConfig

```
private static Properties portalClientConfig
```

CONFIG_FILE

```
private static final String CONFIG_FILE
```

Constructor Detail

ProxyHandling

```
public ProxyHandling()
```

if no query proxy set. Connect to a FADA server and get a proxy.

Method Detail

getContext

```
public RefBaseObject getContext(Vector path)
```

For a Vector of String constructs the context reqiered. i.e A RefClass

---

getPortalClientConfig

```
private static Properties getPortalClientConfig()
```

---

getConfigString

```
public static String getConfigString(String key)
                             throws IOException
```

Throws:
`IOException`

---

findProxy

```
public void findProxy()
```

Connects to the FADA server of the properties (dtool.properties) file and finds a proxy.

---

setService

```
protected void setService(Object service)
```

---

newQuery

```
public void newQuery()
```

Instantiates a new Query

---

getQml

```
public org.dbe.kb.metamodel.qml.QmlPackage getQml()
```

Gets the current Oql Package. If none creates a new one.
Returns:
An Oql Package

---

getNewQuery

```
protected org.dbe.kb.metamodel.qml.contextdeclarations.QueryContextDecl getNewQuery(Vector pat
h,
                                                                                     String nam
e,
                                                                                     org.dbe.kb
.metamodel.qml.ocl.expressions.OclExpression exp,
                                                                                     String res
ult)
```

Gets a new QueryContextDecl for a conetxt path
Parameters:
`path` - The context path (Vector of Strings)

`name` - The Query name
`exp` - The body ocl expression
`result` - The result type

Returns:

A new QueryContextDecl with the specific attributes.

---

getMetamodel

```
public RefPackage getMetamodel(String modelName)
```

Gets a metamodel's RefPackage

Parameters:

`modelName` - the metamodels name (i.e. "SSL", "SDL", etc.)

Returns:

a metamodel's RefPackage

---

getSslModel

```
protected RefPackage getSslModel()
```

Gets the SSL metamodel

Returns:

the SSL metamodel's RefPackage

---

getInstantiationModel

```
protected RefPackage getInstantiationModel()
```

Gets the SSL metamodel

Returns:

the SSL metamodel's RefPackage

---

getSemanticPackage

```
protected RefClass getSemanticPackage()
```

Gets the SemanticPackage (the root class of the SSL metamodel)

Returns:

the SemanticPackage's RefClass

---

getRootClasses

```
public RefClass[] getRootClasses()
```

---

getMetamodelName

```
public String getMetamodelName(int id)
```

---

getRootClass

```
public RefClass getRootClass(String mm)
```

Parameters:
`mm` - the metamodel's name. "SSL" or "SDL".
Returns:
The package's RefClass
Throws:
`NotImplementedException` - if mm is different from "SSL" or "SDL".

---

getSslSpecificPackage

```
public RefPackage getSslSpecificPackage(String packName)
```

Gets an SSL's specific package ("Core", "Types", etc)
Parameters:
`packName` - The SSL's package name ("Core", "Types", etc)
Returns:
A RefPackage object containing the required package

---

getAsynchronousResults

```
public Collection getAsynchronousResults()
```

Submits the query and returns a Vector with the results.
Returns:
A Vector with the results. Each result tuple contains the name of the result, a description and a rank.
Throws:
`RuntimeException` - if an error occurs on server or when submiting the query.

---

printOql

```
public void printOql()
```

Prints The QML Expression as an XMI in standard output.

---

isSR

```
public boolean isSR()
```

Returns true if the proxy coonected to is a ServiceRegistry one. False otherwise
Returns:
True if the proxy coonected to is a ServiceRegistry one. False otherwise

---

getModel

```
public Collection getModel(String metamodel,
                           String id,
                           String nodeId,
                           boolean isSM)
```

For a string id of a model in a metamodel get the instance (the RefObject of the root)

Parameters:

`id` - the id of the model required

Returns:

the RefObject of the root of the model

---

changeEnvironment

```
protected RefBaseObject changeEnvironment(RefBaseObject obj,
                                          String pathItem)
```

---

loadOntology

```
protected org.dbe.kb.metamodel.odm.OdmPackage loadOntology(String id)
```

For a string id of a model in a metamodel get the instance (the RefObject of the root)

Parameters:

`id` - the id of the model required

Returns:

the RefObject of the root of the model

---

getOdmClass

```
public RefObject getOdmClass(String ontId,
                             String id)
```

For a string id of a model in a metamodel get the instance (the RefObject of the root)

Parameters:

`id` - the id of the model required

Returns:

the RefObject of the root of the model

---

getObjects

```
public Vector getObjects(RefPackage refPack)
```

Returns all instances of the classes contained in this Package and its contained packages.

Parameters:

`refPack` - A `RefPackage` denoting the starting package.

Returns:

A `Vector` of `RefObject` Objects

---

getObjects

```
public Vector getObjects(String metamodel,
                         String modelId,
                         String nodeId)
```

Returns all instances of the classes contained in this Package and its contained packages.

Returns:

A `Vector` of `RefObject` Objects

---

getOdmProperties

```
public Vector getOdmProperties(org.dbe.kb.metamodel.odm.classes.Odmclass odmclass)
```

For an `OdmClass` retrieve its properties

Parameters:

`odmclass` - the `OdmClass` whose properties are retrieved

Returns:

A `Vector` with the properties of the odmclass

---

containsOdmProperty

```
protected boolean containsOdmProperty(org.dbe.kb.metamodel.odm.properties.Property property,
                                      org.dbe.kb.metamodel.odm.classes.Odmclass odmclass)
```

Returns:

`true` if the property exists in the class. `false` otherwise.

---

getOdmRange

```
public Vector getOdmRange(org.dbe.kb.metamodel.odm.properties.ObjectProperty property)
```

For a string id of a model in a metamodel get the instance (the RefObject of the root)

Parameters:

`property` - the id of the model required

Returns:

the RefObject of the root of the model

---

isInstantatiatable

```
public boolean isInstantatiatable(String metamodel)
```

Return whether the metmodel specified by id is instantatiatable

Parameters:

`metamodel` - the metamodel name. Either "SSL" or "sdl"

Returns:

`true` if this metamodel is instancetiatable. `false` otherwise.

---

getNewSessionId

```
public String getNewSessionId()
```

---

getServiceManifest

```
public Collection getServiceManifest(String sessionId,
                                     String metamodel,
                                     String modelId)
```

Returns:
a collection of Service Manifests

---

run

```
public Object run()
           throws Exception
```

Throws:
Exception

---

searchByKeywords

```
public Collection searchByKeywords(String metamodel,
                                   Vector keywords)
```

---

searchBMLByKeywords

```
public Collection searchBMLByKeywords(String metamodel,
                                      Vector keywords)
```

---

getSMInfo

```
public org.dbe.kb.sm.SMtool getSMInfo(String smId,
                                      String nodeId)
```

---

closeSession

```
public void closeSession()
```

---

closeSession

```
public void closeSession(String sessionId)
```

---

isSM

```
public boolean isSM(String smId)
```

---

getProxy

```
public Proxy getProxy()
```

Retrieves the current proxy object which has a number of helper functions

Returns:

the current proxy object.

---

getRoots

```
public Vector getRoots(String mm)
```

---

isAlive

```
public boolean isAlive()
```

---

getResults

```
public Collection getResults(String sessionId)
```

---

getCurrentQuerySessionId

```
public String getCurrentQuerySessionId()
```

Class QueryNode

org.dbe.kb.toolkit.proxyutils

```
java.lang.Object
  └org.dbe.kb.toolkit.proxyutils.AbstractNode
      └org.dbe.kb.toolkit.proxyutils.QueryNode
```

All Implemented Interfaces:

IRootNode

---

public class **QueryNode**
extends AbstractNode
implements IRootNode

Author:

gkoto

Field Summary		Page
private int	**childIdx**	102
private ContextRoot[]	**children**	102
private String	**name**	102

Fields inherited from interface org.dbe.kb.toolkit.proxyutils.IRootNode
ALL, HARD, NONE, SOFT

Constructor Summary	***Page***
QueryNode(String name)	*102*
QueryNode(Collection roots, String name, String metamodel)	*102*

Method Summary		***Page***
Object[]	**getChildren**() Returns an array with this nodes children.	*102*
String	**getMetamodel**() Gets a String representation of the query name.	*105*
String	**getName**() Gets a String representation of the name of the node.	*103*
Iterator	**getNodes**() Gets an iterator with the ValueNodes of this query.	*104*
Iterator	**getNodes**(boolean isHard) Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)	*104*
int	**getNodesFlag**() Gets a int flag for the type of the ValueNodes.	*103*
Object	**getObject**() Gets the Object of the current node.	*103*
Vector	**getPath**() Gets a Vector containg the package path to the context Mof class of this query.	*105*
String	**getQueryName**() Gets a String representation of the query name.	*104*
String	**getResultType**() Gets the result type of this context, i.e.	*106*
ContextRoot	**getSelectedChild**()	*105*
boolean	**hasChildren**() Returns true if this node has children.	*102*
boolean	**hasNodes**() Returns true is the query has at least one value node, false otherwise.	*104*
boolean	**isInstance**() Always returns false	*105*
void	**setSelectedMetamodel**(int idx)	*105*
void	**setSelectedMetamodel**(ContextRoot node)	*105*

Methods inherited from class org.dbe.kb.toolkit.proxyutils.AbstractNode
getChildren, getId, getName, getObject, getParent, hasChildren

Methods inherited from interface org.dbe.kb.toolkit.proxyutils.IRootNode
getChildren, getMetamodel, getNodes, getNodes, getNodesFlag, getPath, getQueryName, getResultType, hasNodes, isInstance

Field Detail

name

```
private String name
```

---

children

```
private ContextRoot[] children
```

---

childIdx

```
private int childIdx
```

Constructor Detail

QueryNode

```
public QueryNode(String name)
```

QueryNode

```
public QueryNode(Collection roots,
                 String name,
                 String metamodel)
```

Method Detail

hasChildren

```
public boolean hasChildren()
```

Description copied from class: AbstractNode
Returns true if this node has children. False if it is a leaf node.

Overrides:

hasChildren in class AbstractNode

Returns:

True if this node has children. False if it is a leaf node.

See Also:

`org.dbe.dtool.nodes.AbstractNode.hasChildren()`

---

getChildren

```
public Object[] getChildren()
```

Description copied from class: AbstractNode
Returns an array with this nodes children.

Specified by:

getChildren in interface IRootNode

Overrides:
`getChildren` in class `AbstractNode`
Returns:
An array with this nodes children.
See Also:
`org.dbe.dtool.nodes.AbstractNode.getChildren()`

---

getObject

```
public Object getObject()
```

Description copied from class: `AbstractNode`
Gets the Object of the current node. Each node is holding an object. Usally a MOF object
Overrides:
`getObject` in class `AbstractNode`
Returns:
The Object of the current node
See Also:
`org.dbe.dtool.nodes.AbstractNode.getObject()`

---

getName

```
public String getName()
```

Description copied from class: `AbstractNode`
Gets a String representation of the name of the node.
Overrides:
`getName` in class `AbstractNode`
Returns:
a String representation of the name of the node.
See Also:
`org.dbe.dtool.nodes.AbstractNode.getName()`

---

getNodesFlag

```
public int getNodesFlag()
```

Description copied from interface: `IRootNode`
Gets a int flag for the type of the ValueNodes. Valid values are NONE (0), HARD (1), SOFT (2) and ALL (3)
Specified by:
`getNodesFlag` in interface `IRootNode`
Returns:
NONE (0), HARD (1), SOFT (2) and ALL (3).
See Also:
`org.dbe.dtool.nodes.IRootNode.getNodesFlag()`

---

getNodes

```
public Iterator getNodes(boolean isHard)
```

Description copied from interface: IRootNode
Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)
Specified by:
getNodes in interface IRootNode
Parameters:
isHard - if the ValueNodes to be returned are hard constrains or not.
Returns:
An Iterator with the ValueNodes of a certain type (hard or soft constrains)
See Also:
org.dbe.dtool.nodes.IRootNode.getNodes(boolean)

---

getQueryName

```
public String getQueryName()
```

Description copied from interface: IRootNode
Gets a String representation of the query name.
Specified by:
getQueryName in interface IRootNode
Returns:
a String representation of the query name.
See Also:
org.dbe.dtool.nodes.IRootNode.getQueryName()

---

hasNodes

```
public boolean hasNodes()
```

Description copied from interface: IRootNode
Returns true is the query has at least one value node, false otherwise.
Specified by:
hasNodes in interface IRootNode
Returns:
true is the query has at least one value node, false otherwise.
See Also:
org.dbe.dtool.nodes.IRootNode.hasNodes()

---

getNodes

```
public Iterator getNodes()
```

Description copied from interface: IRootNode
Gets an iterator with the ValueNodes of this query.
Specified by:
getNodes in interface IRootNode

Returns:
The ValueNodes of this query.
See Also:
`org.dbe.dtool.nodes.IRootNode.getNodes()`

---

isInstance

```
public boolean isInstance()
```

Always returns false
Specified by:
`isInstance` in interface `IRootNode`
Returns:
`false`
See Also:
`org.dbe.dtool.nodes.IRootNode.isInstance()`

---

setSelectedMetamodel

```
public void setSelectedMetamodel(int idx)
```

---

setSelectedMetamodel

```
public void setSelectedMetamodel(ContextRoot node)
```

---

getSelectedChild

```
public ContextRoot getSelectedChild()
```

---

getMetamodel

```
public String getMetamodel()
```

Description copied from interface: `IRootNode`
Gets a String representation of the query name.
Specified by:
`getMetamodel` in interface `IRootNode`
Returns:
a String representation of the query name.

---

getPath

```
public Vector getPath()
```

Description copied from interface: `IRootNode`
Gets a Vector containg the package path to the context Mof class of this query.
Specified by:
`getPath` in interface `IRootNode`

Returns:
a Vector containg the package path to the context Mof class of this query.

---

getResultType

```
public String getResultType()
```

Description copied from interface: `IRootNode`
Gets the result type of this context, i.e. the metamodel.
Specified by:
`getResultType` in interface `IRootNode`
Returns:
The result type of this context, i.e. the metamodel.

Class SslInstanceNameAdapter

org.dbe.kb.toolkit.proxyutils

```
java.lang.Object
  └org.dbe.kb.toolkit.proxyutils.SslInstanceNameAdapter
```

All Implemented Interfaces:
NameAdapter

---

public class **SslInstanceNameAdapter**
extends Object
implements NameAdapter

Author:
gkoto

Constructor Summary	*Page*
`SslInstanceNameAdapter()`	*106*

Method Summary		*Page*
`String`	`getId(RefBaseObject object)`	*107*
`String`	`getName(RefBaseObject object)`	*107*

Methods inherited from interface org.dbe.kb.toolkit.proxyutils.NameAdapter
`getId`, `getName`

Constructor Detail

SslInstanceNameAdapter

```
public SslInstanceNameAdapter()
```

Method Detail

getName

```
public String getName(RefBaseObject object)
```

Specified by:
`getName` in interface `NameAdapter`

---

getId

```
public String getId(RefBaseObject object)
```

Specified by:
`getId` in interface `NameAdapter`

Class ValueNode

org.dbe.kb.toolkit.proxyutils

```
java.lang.Object
  └org.dbe.kb.toolkit.proxyutils.AbstractNode
      └org.dbe.kb.toolkit.proxyutils.ValueNode
```

---

public class **ValueNode**
extends AbstractNode

The ValueNode class represents a constraint (either hard or soft) on it's parent AttributeNode. They are nodes of the tree control of FormulateQueryWizardPage class.

Author:
George Kotopoulos
Version:
1.0

Field Summary		Page
static Object[]	**EMPTY_ARRAY**	109
private boolean	**isHard** True if this constraint is a hard one, false otherwise.	109
private String	**operation** a String representation of the operation of this constraint	108
private String	**pathString** A string representation of the path to the tree root.	109
private String	**value** The literal value that constraints the Attribute	108
private int	**weight**	109

Constructor Summary	Page
ValueNode(String op, String value, int weight, AbstractNode parent) Creates a new ValueNode with the specific values	*109*

Method Summary		Page
Object[]	**getChildren**() Returns an array with this nodes children.	*109*
String	**getName**() Gets a String representation of the name of the node.	*110*
Object	**getObject**() Gets the parent object of this node.	*110*
String	**getOperation**() Gets the selected operation	*111*
AbstractNode	**getParent**() Gets the parent of the current node.	*110*
String	**getValue**() The assigned value	*111*
int	**getWeight**()	*111*
boolean	**hasChildren**() Returns true if this node has children.	*109*
boolean	**isHard**() Flag suggesting if it is a hard constraint (true) or not	*111*
void	**setValue**(String op, String val, int weight) Sets the value of this node.	*110*
String	**toString**() Returns a string respresentation.	*111*

Methods inherited from class org.dbe.kb.toolkit.proxyutils.AbstractNode
getChildren, getId, getName, getObject, getParent, hasChildren

Field Detail

operation

```
private String operation
```

a String representation of the operation of this constraint

---

value

```
private String value
```

The literal value that constraints the Attribute

---

pathString

```
private String pathString
```

A string representation of the path to the tree root.

---

isHard

```
private boolean isHard
```

True if this constraint is a hard one, false otherwise.

EMPTY_ARRAY

```
public static Object[] EMPTY_ARRAY
```

---

weight

```
private int weight
```

Constructor Detail

ValueNode

```
public ValueNode(String op,
                 String value,
                 int weight,
                 AbstractNode parent)
```

Creates a new ValueNode with the specific values

Method Detail

hasChildren

```
public boolean hasChildren()
```

Returns true if this node has children. False if it is a leaf node.

Overrides:

hasChildren in class AbstractNode

Returns:

True if this node has children. False if it is a leaf node.

See Also:

`org.dbe.dtool.nodes.AbstractNode.hasChildren()`

---

getChildren

```
public Object[] getChildren()
```

Returns an array with this nodes children.

Overrides:

getChildren in class AbstractNode

Returns:

An array with this nodes children.

See Also:

org.dbe.dtool.nodes.AbstractNode.getChildren()

---

getName

```
public String getName()
```

Gets a String representation of the name of the node.

Overrides:

getName in class AbstractNode

Returns:

a String representation of the name of the node.

See Also:

org.dbe.dtool.nodes.AbstractNode.getName()

---

getObject

```
public Object getObject()
```

Gets the parent object of this node. Parent is usally of type AttributeNode. So, An Mof Attribute is returned.

Overrides:

getObject in class AbstractNode

Returns:

The Object of the current node

See Also:

org.dbe.dtool.nodes.AbstractNode.getObject()

---

getParent

```
public AbstractNode getParent()
```

Gets the parent of the current node.

Overrides:

getParent in class AbstractNode

Returns:

The parent of the current node.

See Also:

AbstractNode.getParent()

---

setValue

```
public void setValue(String op,
                     String val,
                     int weight)
```

Sets the value of this node.
Parameters:
`op` - The operation selected
`val` - the value assigned
`weight` - a flag suggesting whether it is a hard or a soft constraint

---

getOperation

```
public String getOperation()
```

Gets the selected operation
Returns:
the operation

---

getValue

```
public String getValue()
```

The assigned value
Returns:
the assigned value

---

getWeight

```
public int getWeight()
```

---

isHard

```
public boolean isHard()
```

Flag suggesting if it is a hard constraint (true) or not
Returns:
true if it a Hard constraint, false otherwise.

---

toString

```
public String toString()
```

Returns a string respresentation. of this constraint.
Overrides:
`toString` in class `Object`

Package org.dbe.toolkit.portal

Class Summary		Page
DBEPortal		112

DBEPortalToolkitConfig		114

Exception Summary		Page
DBEPortalException		114

Class DBEPortal

org.dbe.toolkit.portal

```
java.lang.Object
  └org.dbe.toolkit.portal.DBEPortal
```

---

public class **DBEPortal**
extends Object

Author:
andy-edmonds

Field Summary		Page
private org.sun.dbe.ClientHelper	**helper**	112

Constructor Summary	Page
DBEPortal(org.sun.dbe.ClientHelper helper)	112

Method Summary		Page
Object[]	**execute**(String smid, String methodName, Object[] parameters)	113
Object	**executeIF**(String smid, HashMap params)	113
String	**getUI**(String smid)	113

Field Detail

helper

```
private org.sun.dbe.ClientHelper helper
```

Constructor Detail

DBEPortal

```
public DBEPortal(org.sun.dbe.ClientHelper helper)
```

Method Detail

execute

```
public Object[] execute(String smid,
                        String methodName,
                        Object[] parameters)
                 throws DBEPortalException
```

Parameters:

`smid` - the SMID of the service to execute

`methodName` - the name of the method to invoke

`parameters` - the list of parameters to supply to the method invocation

Returns:

returns a list containing the results; 1st element is the return value of ServiceProxy.invoke

Throws:

`DBEPortalException`

See Also:

`IDBEPortal`

---

getUI

```
public String getUI(String smid)
             throws DBEPortalException
```

Parameters:

`smid` - the SMID of the service that will supply the service UI

Returns:

a URL represented as a String pointing to the UI that the client can use

Throws:

`DBEPortalException`

See Also:

`IDBEPortal`

---

executeIF

```
public Object executeIF(String smid,
                        HashMap params)
                 throws DBEPortalException
```

Parameters:

`smid` - the SMID of the service that will supply the service UI

Returns:

a URL represented as a String pointing to the UI that the client can use

Throws:

`DBEPortalException`

See Also:

`IDBEPortal`

Class DBEPortalException

org.dbe.toolkit.portal

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └org.dbe.toolkit.portal.DBEPortalException
```

All Implemented Interfaces:

Serializable

---

public class **DBEPortalException**
extends Exception

Field Summary		Page
private static final long	**serialVersionUID**	*114*

Constructor Summary	Page
DBEPortalException(String string)	*114*

Field Detail

serialVersionUID

```
private static final long serialVersionUID
```

Constructor Detail

DBEPortalException

```
public DBEPortalException(String string)
```

Class DBEPortalToolkitConfig

org.dbe.toolkit.portal

```
java.lang.Object
  └org.dbe.toolkit.portal.DBEPortalToolkitConfig
```

---

public class **DBEPortalToolkitConfig**
extends Object

Field Summary		Page
private static final String	**CONFIG_FILE**	*115*
private static Properties	**portalServiceConfig**	*115*

Constructor Summary		Page
private	**DBEPortalToolkitConfig**() Default constructor	*115*

Method Summary		Page
private static Properties	**getPortalServiceConfig**() Gets the portal configuration Properties bundle	*116*
static String	**getString**(String key) Gets a configuration parameter as a String	*115*

Field Detail

portalServiceConfig

```
private static Properties portalServiceConfig
```

CONFIG_FILE

```
private static final String CONFIG_FILE
```

Constructor Detail

DBEPortalToolkitConfig

```
private DBEPortalToolkitConfig()
```

Default constructor

Method Detail

getString

```
public static String getString(String key)
                        throws IOException
```

Gets a configuration parameter as a String

Parameters:

`key` - the requested parameter name

Returns:

the value of the parameter name

Throws:

`IOException`

getPortalServiceConfig

```
private static Properties getPortalServiceConfig()
```

Gets the portal configuration Properties bundle

Returns:

the properties bundle

Package org.dbe.toolkit.portal.client

Class Summary		Page
DBEPortalClient		116
ProxyCache		120
ProxyObject		123

Class DBEPortalClient

org.dbe.toolkit.portal.client

```
java.lang.Object
  └org.dbe.toolkit.portal.client.DBEPortalClient
```

public class **DBEPortalClient**
extends Object

Author:

andy-edmonds

Field Summary		Page
private static final String	CONFIG_FILE	117
private static org.sun.dbe.ClientHelper	helper	117
private static final String	HTTP	117
private static org.apache.log4j.Logger	logger	117
private static final String	PORT_SEP	117
private static Properties	portalClientConfig	118
private static ProxyCache	proxyMan	118

Constructor Summary	Page
DBEPortalClient() Default constructor	118

Method Summary		Page
Object[]	**executeIF**(String smId, String smIdUser, String inObj)	*120*
static String	**getConfigString**(String key)	*119*
static org.apache.log4j.Logger	**getLogger**()	*119*
private static Properties	**getPortalClientConfig**() Gets the portal client configuration bundle	*119*
static ProxyCache	**getProxyManager**() Gets an instance of the proxy manager	*119*
String	**getUI**(String smid) Returns the URL of where the requested UI or IF UI is located on the servent	*118*
void	**init**() Initialises the portal client and enumerates it's configuration parameters	*118*
List	**invoke**(String smid, String methodName, Vector parameters) Invokes a DBE service	*118*
boolean	**isAlive**(String smid) Checks to see if a service is available	*119*
private void	**setLoggingLevel**() Sets the logging level to be used by the Portal client	*118*

Field Detail

PORT_SEP

```
private static final String PORT_SEP
```

HTTP

```
private static final String HTTP
```

CONFIG_FILE

```
private static final String CONFIG_FILE
```

---

helper

```
private static org.sun.dbe.ClientHelper helper
```

---

logger

```
private static org.apache.log4j.Logger logger
```

---

portalClientConfig

```
private static Properties portalClientConfig
```

---

proxyMan

```
private static ProxyCache proxyMan
```

Constructor Detail

DBEPortalClient

```
public DBEPortalClient()
```

Default constructor

Method Detail

init

```
public void init()
```

Initialises the portal client and enumerates it's configuration parameters

---

setLoggingLevel

```
private void setLoggingLevel()
```

Sets the logging level to be used by the Portal client

---

invoke

```
public List invoke(String smid,
                   String methodName,
                   Vector parameters)
```

Invokes a DBE service

Parameters:

`smid` - SMID of the target service to invoke

`methodName` - method of the servuce to invoke

`parameters` - a list of parameter values, ordered as the method signature expects them

Returns:

an ordered list of return results, the first entry in this list is always the return value of Workspace.invoke()

---

getUI

```
public String getUI(String smid)
```

Returns the URL of where the requested UI or IF UI is located on the servent

Parameters:

`smid` - The SMID of the service which has the requested UI or IF UI

Returns:

The URL of the UI or IF UI

---

getLogger

```
public static org.apache.log4j.Logger getLogger()
```

---

getConfigString

```
public static String getConfigString(String key)
                             throws IOException
```

Throws:

`IOException`

---

getProxyManager

```
public static ProxyCache getProxyManager()
                                 throws DBEPortalException
```

Gets an instance of the proxy manager

Returns:

the proxy manager

Throws:

`DBEPortalException`

---

getPortalClientConfig

```
private static Properties getPortalClientConfig()
```

Gets the portal client configuration bundle

Returns:

a properties bundle containing the configuration

---

isAlive

```
public boolean isAlive(String smid)
                throws DBEPortalException
```

Checks to see if a service is available

Returns:

true if the service is reachable, false if it doesn't

Throws:

`DBEPortalException`

---

executeIF

```
public Object[] executeIF(String smId,
                          String smIdUser,
                          String inObj)
```

Parameters:

`smId` - The Service manifest ID

`smIdUser` - THe Service manifest ID of the user service

`inObj` - The data (xml string)that made the service manifest

Returns:

the SM

Author:

mbordin

Class ProxyCache

org.dbe.toolkit.portal.client

```
java.lang.Object
  └org.dbe.toolkit.portal.client.ProxyCache
```

---

public class **ProxyCache**
extends Object

Field Summary		Page
private boolean	**cacheEnabled**	121
private org.sun.dbe.ClientHelper	**helper**	121
private static org.apache.log4j.Logger	**logger**	121
private static long	**OBJ_TTL**	121
private static HashMap	**proxyCache**	121

Constructor Summary	Page
ProxyCache(org.sun.dbe.ClientHelper helper) Constructor	121

Method Summary		Page
org.dbe.toolkit.proxyframework.Workspace	**get**(String smid) Gets a proxy of the corresponding SMID	122
private HashMap	**getCache**() gets the cache of proxies	122
private org.dbe.toolkit.proxyframework.Workspace	**getWorkspace**(String smid, org.dbe.toolkit.proxyframework.Workspace ws) Gets the workspace of the requested service (SMID)	122
private boolean	**isCached**(String smid) Check to see if the proxy is cached locally	123
boolean	**isCacheEnabled**()	121

private ProxyObject	**retreive**(String smid) Retrieves the proxy from the cache corresponding to the supplied SMID	*122*
void	**setCacheEnabled**(boolean cacheEnabled) Check to see if proxy caching is enabled or not	*122*
private void	**store**(String smid, org.dbe.toolkit.proxyframework.Workspace ws) Stores the proxy to the cache	*122*

Field Detail

helper

```
private org.sun.dbe.ClientHelper helper
```

---

cacheEnabled

```
private boolean cacheEnabled
```

---

logger

```
private static org.apache.log4j.Logger logger
```

---

proxyCache

```
private static HashMap proxyCache
```

OBJ_TTL

```
private static long OBJ_TTL
```

Constructor Detail

ProxyCache

```
public ProxyCache(org.sun.dbe.ClientHelper helper)
```

Constructor

Method Detail

isCacheEnabled

```
public boolean isCacheEnabled()
```

---

setCacheEnabled

```
public void setCacheEnabled(boolean cacheEnabled)
```

Check to see if proxy caching is enabled or not

---

getCache

```
private HashMap getCache()
```

gets the cache of proxies

---

get

```
public org.dbe.toolkit.proxyframework.Workspace get(String smid)
                                            throws DBEPortalException
```

Gets a proxy of the corresponding SMID

Parameters:

`smid` - the requested service

Returns:

proxy of the requested service

Throws:

`DBEPortalException`

---

getWorkspace

```
private org.dbe.toolkit.proxyframework.Workspace getWorkspace(String smid,
                                                              org.dbe.toolkit.proxyframework.W
orkspace ws)
                                                       throws DBEPortalException
```

Gets the workspace of the requested service (SMID)

Throws:

`DBEPortalException`

---

store

```
private void store(String smid,
                   org.dbe.toolkit.proxyframework.Workspace ws)
```

Stores the proxy to the cache

Parameters:

`smid` - SMID of the proxy

`ws` - the actual proxy of the service

---

retreive

```
private ProxyObject retreive(String smid)
```

Retrieves the proxy from the cache corresponding to the supplied SMID

Parameters:

`smid` - SMID of the proxy to be retrieved

Returns:

the proxy

---

isCached

```
private boolean isCached(String smid)
```

Check to see if the proxy is cached locally

Class ProxyObject

org.dbe.toolkit.portal.client

```
java.lang.Object
  └org.dbe.toolkit.portal.client.ProxyObject
```

---

public class **ProxyObject**
extends Object

Field Summary		Page
org.dbe.toolkit.proxyframework.Workspace	**proxy**	123
long	**timestamp**	123

Constructor Summary	Page
ProxyObject(org.dbe.toolkit.proxyframework.Workspace ws, long timestamp)	124

Method Summary		Page
org.dbe.toolkit.proxyframework.Workspace	**getProxy**()	124
long	**getTimestamp**()	124

Field Detail

proxy

```
org.dbe.toolkit.proxyframework.Workspace proxy
```

---

timestamp

```
long timestamp
```

Constructor Detail

ProxyObject

```
public ProxyObject(org.dbe.toolkit.proxyframework.Workspace ws,
                   long timestamp)
```

Method Detail

getProxy

```
public org.dbe.toolkit.proxyframework.Workspace getProxy()
```

---

getTimestamp

```
public long getTimestamp()
```

Package org.dbe.toolkit.portal.execution

Class Summary		Page
ExecutionDelegate		124

Class ExecutionDelegate

org.dbe.toolkit.portal.execution

```
java.lang.Object
  └org.dbe.toolkit.portal.execution.ExecutionDelegate
```

---

public class **ExecutionDelegate**
extends Object

Author:
andy-edmonds

Field Summary		Page
private static final String	BOOLEAN	126
private static final String	DATE_TIME	125
private static final String	EXECUTE_IF	126
private org.sun.dbe.ClientHelper	helper	126
private Vector	holderObjects	126
private static final String	INTEGER	126
private static org.apache.log4j.Logger	logger	127
private Vector	outParamHolders	126
private Class[]	outParamHoldersArray	126

private static final String	**REAL**	*126*
private String	**smid**	*127*
private static final String	**STRING**	*126*
private static final String	**TMP_SDL_POSTFIX**	*126*
private static final String	**TMP_SDL_PREFIX**	*126*
private static final String	**URI**	*125*

Constructor Summary	***Page***
ExecutionDelegate(org.sun.dbe.ClientHelper helper, String smid)	*127*

Method Summary		***Page***
private String	**dumpSDL**(String sdlStr) Takes a SDL file and dumps it to string	*128*
Object[]	**execute**(String methodName, Object[] parameters) Executes the method with supplied parameters	*127*
Object	**executeIF**(HashMap parameters)	*127*
private Object	**getHolderValue**(Object holder) Gets the value of a holder object based on type introspection	*129*
private Class	**getInClass**(org.dbe.studio.editor.sdl.Type type) Finds the java	*128*
private Object[]	**getInvocationResults**(Object ret, Object[] parameters) Merges out paramters and result of Workspace.invoke() into one array	*129*
private Class[]	**getMethodParametersTypes**(String methodName) Gets the parameter types for a particular method	*128*
private Object[]	**getMethodParametersValues**(Object[] parameters)	*129*
private Class	**getOutClass**(org.dbe.studio.editor.sdl.Type type, int j) Gets the	*128*
private org.dbe.toolkit.proxyframework.Workspace	**getWorkspace**() Gets a Workspace object	*127*

Field Detail

URI

```
private static final String URI
```

DATE_TIME

```
private static final String DATE_TIME
```

REAL

```
private static final String REAL
```

INTEGER

```
private static final String INTEGER
```

BOOLEAN

```
private static final String BOOLEAN
```

STRING

```
private static final String STRING
```

TMP_SDL_POSTFIX

```
private static final String TMP_SDL_POSTFIX
```

TMP_SDL_PREFIX

```
private static final String TMP_SDL_PREFIX
```

EXECUTE_IF

```
private static final String EXECUTE_IF
```

---

helper

```
private org.sun.dbe.ClientHelper helper
```

---

outParamHolders

```
private Vector outParamHolders
```

---

outParamHoldersArray

```
private Class[] outParamHoldersArray
```

---

holderObjects

```
private Vector holderObjects
```

smid

```
private String smid
```

logger

```
private static org.apache.log4j.Logger logger
```

Constructor Detail

ExecutionDelegate

```
public ExecutionDelegate(org.sun.dbe.ClientHelper helper,
                         String smid)
```

Method Detail

executeIF

```
public Object executeIF(HashMap parameters)
                 throws DBEPortalException
```

Throws:

DBEPortalException

execute

```
public Object[] execute(String methodName,
                        Object[] parameters)
                 throws DBEPortalException
```

Executes the method with supplied parameters

Parameters:

`methodName` - Name of the method to execute

`parameters` - parameters to supply to the the method

Returns:

Results of the exection

Throws:

DBEPortalException

getWorkspace

```
private org.dbe.toolkit.proxyframework.Workspace getWorkspace()
                                                  throws DBEPortalException
```

Gets a Workspace object

Returns:

The workspace object

Throws:

DBEPortalException

---

getMethodParametersTypes

```
private Class[] getMethodParametersTypes(String methodName)
                                 throws DBEPortalException
```

Gets the parameter types for a particular method

Parameters:

`methodName` - The method to get the parameters types

Returns:

A list of method parameter types in the order of the method's signature

Throws:

`DBEPortalException`

---

dumpSDL

```
private String dumpSDL(String sdlStr)
```

Takes a SDL file and dumps it to string

Parameters:

`sdlStr` - the absolute path of a SDL file

Returns:

The SDL as a string

---

getInClass

```
private Class getInClass(org.dbe.studio.editor.sdl.Type type)
              throws DBEPortalException
```

Finds the java

Parameters:

`type` - type to get Class object of

Returns:

Class object supplied

Throws:

`DBEPortalException`
`Exception`

See Also:

object of the type supplied

---

getOutClass

```
private Class getOutClass(org.dbe.studio.editor.sdl.Type type,
                          int j)
               throws DBEPortalException
```

Gets the

Parameters:

`type` - Type to find the
`j` - a reference counter

Returns:
Class object of the supplied type
Throws:
DBEPortalException
Exception
See Also:
object of the supplied type, object of

---

getMethodParametersValues

```
private Object[] getMethodParametersValues(Object[] parameters)
                                  throws DBEPortalException
```

Throws:
DBEPortalException

---

getInvocationResults

```
private Object[] getInvocationResults(Object ret,
                                      Object[] parameters)
```

Merges out paramters and result of Workspace.invoke() into one array
Parameters:
`ret` - Return value of Workspaace.invoke()
`parameters` - out parameters of the rpc call
Returns:
merged array

---

getHolderValue

```
private Object getHolderValue(Object holder)
                       throws Exception
```

Gets the value of a holder object based on type introspection
Parameters:
`holder` - Holder to get the value of
Returns:
the value of the holder
Throws:
Exception

Package org.dbe.toolkit.portal.interactionform

Class Summary		Page
IfManager		130

Class IfManager

org.dbe.toolkit.portal.interactionform

```
java.lang.Object
  └org.dbe.toolkit.portal.interactionform.IfManager
```

public class **IfManager**
extends Object

Field Summary		Page
private static org.apache.log4j.Logger	**log**	130

Constructor Summary	Page
IfManager()	130

Method Summary		Page
String	**getSM**()	131
static org.dbe.kb.service.SRI	**getSRI**(String serventURL)	130
String	**publish**(String sm)	130

Field Detail

log

```
private static org.apache.log4j.Logger log
```

Constructor Detail

IfManager

```
public IfManager()
```

Method Detail

getSRI

```
public static org.dbe.kb.service.SRI getSRI(String serventURL)
                                     throws Exception
```

Throws:
Exception

publish

```
public String publish(String sm)
              throws Exception
```

Throws:
`Exception`

---

getSM

```
public String getSM()
```

Package org.dbe.toolkit.portal.network

Class Summary		Page
SDLRetriever		131

Class SDLRetriever

org.dbe.toolkit.portal.network

```
java.lang.Object
  └org.dbe.toolkit.portal.network.SDLRetriever
```

---

public class **SDLRetriever**
extends Object

Author:
andy-edmonds

Field Summary		Page
private org.sun.dbe.ClientHelper	helper	132
private static org.apache.log4j.Logger	logger	132
private static final String	SR	131

Constructor Summary	Page
SDLRetriever(org.sun.dbe.ClientHelper helper)	132

Method Summary		Page
String	getSDL(String smid) Gets the SDL of the supplied SMID	132

Field Detail

SR

```
private static final String SR
```

---

logger

```
private static org.apache.log4j.Logger logger
```

---

helper

```
private org.sun.dbe.ClientHelper helper
```

Constructor Detail

SDLRetriever

```
public SDLRetriever(org.sun.dbe.ClientHelper helper)
```

Method Detail

getSDL

```
public String getSDL(String smid)
              throws org.dbe.servent.NoSuchServiceException,
                     IOException
```

Gets the SDL of the supplied SMID

Parameters:

`smid` - SMID to get the SDL of

Returns:

SDL as a string value

Throws:

`org.dbe.servent.NoSuchServiceException`
`IOException`

Package org.dbe.toolkit.portal.qfsdt

Class Summary		Page
ServiceSearcher		132

Class ServiceSearcher

org.dbe.toolkit.portal.qfsdt

```
java.lang.Object
  └ org.dbe.toolkit.portal.qfsdt.ServiceSearcher
```

---

public class **ServiceSearcher**
extends Object

Field Summary		Page
private static Hashtable	modelElements	134

private static Hashtable	**proxies**	*134*
private ProxyHandling	**proxy**	*134*
private String	**querySessionId**	*134*
private static Hashtable	**users**	*134*

Constructor Summary	***Page***
ServiceSearcher()	*134*

Method Summary		***Page***
private void	**addModelElement**(Vector features)	*136*
void	**closeSession**(String sessionId)	*136*
Vector	**convertRank**(Vector results)	*137*
private org.dbe.kb.qi.adv.QueryExpr[]	**createQueryExpr**(Collection criteria)	*134*
org.dbe.kb.qi.adv.Template	**createTemplate**(String description, Collection templateElementPaths)	*135*
private Collection	**getAsynchronousResults**(String clientQuerySessionId)	*135*
Vector	**getBMLModelFromId**(String id)	*135*
private Vector	**getChildren**(AbstractNode node, Vector model)	*136*
Vector	**getModelElement**(String modelElementParentPath)	*136*
private Vector	**getModelFromId**(String modelId, String metamodel, String nodeId, boolean isSM)	*135*
Vector	**getModelIds**(String keywordsAndId)	*135*
Vector	**getMoreResults**(String clientQuerySessionId)	*135*
private String	**getPath**(AbstractNode node)	*136*
ProxyHandling	**getProxy**()	*137*
String	**getServiceAvailabilityOnWeb**(String id)	*137*
Vector	**getServices**(String keywordsAndId)	*135*
Vector	**getSMInfo**(String id)	*135*
Vector	**getSSLModelFromId**(String id)	*135*
private Vector	**getTokens**(String keywords)	*136*
String	**getURLToExecuteService**(String id)	*136*
private void	**manageSessions**(String sessionId, boolean isNewQuery)	*135*
Vector	**performAdvancedSearch**(String arguments)	*136*
Object	**performSearch**(String description, Collection templateElementPaths, Collection criteria)	*134*
Object	**performSearch**(org.dbe.kb.qi.adv.Template template, Collection criteria)	*134*
private void	**removeProxyEntry**(String sessionId)	*136*
private Vector	**reverseVector**(Vector vector)	*136*
private Vector	**setRoot**(ContextRoot root, String modelId, String metamodel)	*136*

Field Detail

querySessionId

```
private String querySessionId
```

---

proxy

```
private ProxyHandling proxy
```

---

users

```
private static Hashtable users
```

---

proxies

```
private static Hashtable proxies
```

---

modelElements

```
private static Hashtable modelElements
```

Constructor Detail

ServiceSearcher

```
public ServiceSearcher()
```

Method Detail

performSearch

```
public Object performSearch(String description,
                            Collection templateElementPaths,
                            Collection criteria)
```

---

performSearch

```
public Object performSearch(org.dbe.kb.qi.adv.Template template,
                            Collection criteria)
```

---

createQueryExpr

```
private org.dbe.kb.qi.adv.QueryExpr[] createQueryExpr(Collection criteria)
```

---

createTemplate

```
public org.dbe.kb.qi.adv.Template createTemplate(String description,
                                                 Collection templateElementPaths)
```

---

manageSessions

```
private void manageSessions(String sessionId,
                            boolean isNewQuery)
```

---

getServices

```
public Vector getServices(String keywordsAndId)
```

---

getModelIds

```
public Vector getModelIds(String keywordsAndId)
```

---

getSSLModelFromId

```
public Vector getSSLModelFromId(String id)
```

---

getBMLModelFromId

```
public Vector getBMLModelFromId(String id)
```

---

getSMInfo

```
public Vector getSMInfo(String id)
```

---

getAsynchronousResults

```
private Collection getAsynchronousResults(String clientQuerySessionId)
```

---

getMoreResults

```
public Vector getMoreResults(String clientQuerySessionId)
```

---

getModelFromId

```
private Vector getModelFromId(String modelId,
                              String metamodel,
                              String nodeId,
                              boolean isSM)
```

---

setRoot

```
private Vector setRoot(ContextRoot root,
                       String modelId,
                       String metamodel)
```

---

getChildren

```
private Vector getChildren(AbstractNode node,
                           Vector model)
```

---

addModelElement

```
private void addModelElement(Vector features)
```

---

getModelElement

```
public Vector getModelElement(String modelElementParentPath)
```

---

performAdvancedSearch

```
public Vector performAdvancedSearch(String arguments)
```

---

getURLToExecuteService

```
public String getURLToExecuteService(String id)
```

---

getTokens

```
private Vector getTokens(String keywords)
```

---

reverseVector

```
private Vector reverseVector(Vector vector)
```

---

getPath

```
private String getPath(AbstractNode node)
```

---

closeSession

```
public void closeSession(String sessionId)
```

---

removeProxyEntry

```
private void removeProxyEntry(String sessionId)
```

getProxy

```
public ProxyHandling getProxy()
```

convertRank

```
public Vector convertRank(Vector results)
```

getServiceAvailabilityOnWeb

```
public String getServiceAvailabilityOnWeb(String id)
```

Package org.dbe.toolkit.portal.service

Interface Summary		Page
DBEPortal		137

Class Summary		Page
DBEPortalAdapter		138
DBEPortalRegistrationClient		143
DBEPortalServiceConfig		148
SMIDGenerator		150

Interface DBEPortal

org.dbe.toolkit.portal.service

All Known Implementing Classes:

DBEPortalAdapter

public interface **DBEPortal**

Author:

andy-edmonds

Method Summary		Page
void	**getUIURL**(String smid, StringHolder url)	138
void	**getURL**(StringHolder url)	138

Method Detail

getURL

```
public void getURL(StringHolder url)
            throws RemoteException
```

Parameters:

`url` - the return value - this is the URL where the portal is at

Throws:

`RemoteException`
`java.rmi.RemoteException`

---

getUIURL

```
public void getUIURL(String smid,
                     StringHolder url)
              throws RemoteException
```

Parameters:

`smid` - the SMID of the requested UI
`url` - the return value - this is the URL where the UI is at

Throws:

`RemoteException`
`java.rmi.RemoteException`

Class DBEPortalAdapter

org.dbe.toolkit.portal.service

```
java.lang.Object
  └org.dbe.toolkit.portal.service.DBEPortalAdapter
```

All Implemented Interfaces:

org.dbe.servent.Adapter, DBEPortal

---

public class **DBEPortalAdapter**
extends Object
implements DBEPortal, org.dbe.servent.Adapter

Author:

andy-edmonds

Field Summary		Page
private boolean	**configured**	141
private org.dbe.servent.ServiceContext	**context**	141
private static String	**DEPLOY_ROOT**	140
private static final String	**EXTN**	140
private static String	**GET_SM**	140
private static final String	**HTTP**	140

private static final String	**LOCALHOST**	*140*
private static org.apache.log4j.Logger	**logger**	*141*
private static final String	**NIX_FILEPATH_SEP**	*140*
private static final String	**PORT_PREFIX**	*140*
private static String	**PORTAL_CONTEXT**	*141*
private static String	**PORTAL_PORT**	*141*
private static String	**PORTAL_SM**	*141*
private static boolean	**PROCESS_IF**	*141*
private static String	**REL_UI_DEPLOY_DIR**	*140*
private static String	**REL_UI_DROP_DIR**	*140*
private static String	**UICACHE**	*140*

Constructor Summary	***Page***
DBEPortalAdapter()	*141*

Method Summary		***Page***
void	**destroy**() Called when the service is shutdown	*143*
static org.apache.log4j.Logger	**getLogger**()	*143*
private String	**getPort**() Gets the port on which the portal runs on	*142*
void	**getUIURL**(String smid, StringHolder url) Zips up the requested UI (corresponding to the SMID) and returns back the URL of where it is available	*142*
void	**getURL**(StringHolder url) Gets the URL where the portal runs at	*141*
void	**init**(org.dbe.servent.ServiceContext context) Initialises the PortalService	*142*
private void	**initialiseParameters**() Sets the parameters associated with this service.	*142*
private boolean	**isPortalPublished**() Publishes the service to the SR only if it has never been Called only once when the service runs for the first time	*143*
private void	**processIF**(String smid)	*143*
private void	**setLoggingLevel**() Sets the approprite logging level from the portal service configuration file TODO replace this with a configuration from log4j.properties file	*142*

Methods inherited from interface org.dbe.toolkit.portal.service.DBEPortal
getUIURL, getURL

Methods inherited from interface org.dbe.servent.Adapter
destroy, init

Field Detail

NIX_FILEPATH_SEP

```
private static final String NIX_FILEPATH_SEP
```

LOCALHOST

```
private static final String LOCALHOST
```

PORT_PREFIX

```
private static final String PORT_PREFIX
```

HTTP

```
private static final String HTTP
```

EXTN

```
private static final String EXTN
```

GET_SM

```
private static String GET_SM
```

DEPLOY_ROOT

```
private static String DEPLOY_ROOT
```

UICACHE

```
private static String UICACHE
```

REL_UI_DEPLOY_DIR

```
private static String REL_UI_DEPLOY_DIR
```

REL_UI_DROP_DIR

```
private static String REL_UI_DROP_DIR
```

PORTAL_CONTEXT

```
private static String PORTAL_CONTEXT
```

PORTAL_SM

```
private static String PORTAL_SM
```

PROCESS_IF

```
private static boolean PROCESS_IF
```

---

context

```
private org.dbe.servent.ServiceContext context
```

---

configured

```
private boolean configured
```

PORTAL_PORT

```
private static String PORTAL_PORT
```

---

logger

```
private static org.apache.log4j.Logger logger
```

Constructor Detail

DBEPortalAdapter

```
public DBEPortalAdapter()
```

Method Detail

getURL

```
public void getURL(StringHolder url)
            throws RemoteException
```

Gets the URL where the portal runs at

Specified by:

getURL in interface DBEPortal

Parameters:

`url` - the return value - the portal's URL

Throws:
`RemoteException`

---

getPort

```
private String getPort()
```

Gets the port on which the portal runs on
Returns:
the port number

---

getUIURL

```
public void getUIURL(String smid,
                     StringHolder url)
              throws RemoteException
```

Zips up the requested UI (corresponding to the SMID) and returns back the URL of where it is available
Specified by:
`getUIURL` in interface `DBEPortal`
Parameters:
`smid` - the SMID of the service which implements the requested UI
`url` - a return value - the URL of the zip file containing the UI
Throws:
`RemoteException`

---

init

```
public void init(org.dbe.servent.ServiceContext context)
```

Initialises the PortalService
Specified by:
`init` in interface `org.dbe.servent.Adapter`
Parameters:
`context` - the service context containing runtime information of this service

---

setLoggingLevel

```
private void setLoggingLevel()
```

Sets the approprite logging level from the portal service configuration file TODO replace this with a configuration from log4j.properties file

---

initialiseParameters

```
private void initialiseParameters()
```

Sets the parameters associated with this service. Called by the init method of the service

---

isPortalPublished

```
private boolean isPortalPublished()
                          throws IOException
```

Publishes the service to the SR only if it has never been Called only once when the service runs for the first time

Throws:

`IOException`

---

destroy

```
public void destroy()
```

Called when the service is shutdown

Specified by:

`destroy` in interface `org.dbe.servent.Adapter`

---

processIF

```
private void processIF(String smid)
```

Parameters:

`smid` - the Service manifest ID

Author:

mbordin

---

getLogger

```
public static org.apache.log4j.Logger getLogger()
```

Class DBEPortalRegistrationClient

org.dbe.toolkit.portal.service

```
java.lang.Object
  └org.dbe.toolkit.portal.service.DBEPortalRegistrationClient
```

---

public class **DBEPortalRegistrationClient**
extends Object

Field Summary		Page
private static final String	**BIZ_DOM_ATTR**	*145*
private static final String	**COUNTRY_ATTR**	*145*

private static final String	**DESCRIPTION_ATTR**	*145*
private static final String	**IMM_DATAVALUE**	*145*
private static final String	**LOCALITY_ATTR**	*146*
private org.apache.log4j.Logger	**logger**	*146*
private static final String	**PORTAL_BIZDOM**	*146*
private static final String	**PORTAL_DESCR**	*146*
private static final String	**PORTAL_LOC**	*146*
private static final String	**PORTAL_PAYS**	*146*
private static final String	**PORTAL_REGION**	*146*
private static Properties	**props**	*146*
private static final String	**REGION_ATTR**	*145*
private static String	**SERVENT_HOME**	*145*
private static String	**SM_FILE**	*145*
private static final String	**SR**	*145*
private org.dbe.servent.ServiceContext	**svcCtx**	*146*
private static final String	**VALUE**	*145*
private static final String	**XMI_ID**	*145*

Constructor Summary	***Page***
DBEPortalRegistrationClient(org.dbe.servent.ServiceContext svcCtx) Constructor	*146*

Method Summary		***Page***
org.w3c.dom.Document	**createXMLDoc**(String xmlFile) Reads in a file containing XML and creates a DOM of it's content.	*147*
private void	**doSMEDataEntry**(String serviceManisfestPath) Enters the BML Data into the PortalService's SM	*147*
private void	**doSMEPortalPublish**(String SMID, String serviceManifestPath) Publishes the SM to the SR with the supplied SMID	*148*
static void	**main**(String[] args)	*148*
void	**publish**(String serviceManifestPath) Publishes the PortalService to the SR along with the BML Data	*147*
private org.w3c.dom.Document	**replaceAllXMLAttributeValue**(String xmlFile, String elementName, HashMap replacmentValues) Replaces the value of the specfied element in an XML file	*147*
private void	**setIsPublished**(String serviceManifestPath)	*148*
void	**setProps**(Properties props1)	*148*
void	**setSM_FILE**(String sm_file)	*148*
private String	**smToString**(String serviceManifestPath) Converts the supplied file to a string	*148*
void	**writeXmlFile**(org.w3c.dom.Document doc, String filename) Writes a XML DOM to file.	*147*

Field Detail

SERVENT_HOME

```
private static String SERVENT_HOME
```

SM_FILE

```
private static String SM_FILE
```

SR

```
private static final String SR
```

IMM_DATAVALUE

```
private static final String IMM_DATAVALUE
```

XMI_ID

```
private static final String XMI_ID
```

VALUE

```
private static final String VALUE
```

DESCRIPTION_ATTR

```
private static final String DESCRIPTION_ATTR
```

BIZ_DOM_ATTR

```
private static final String BIZ_DOM_ATTR
```

COUNTRY_ATTR

```
private static final String COUNTRY_ATTR
```

REGION_ATTR

```
private static final String REGION_ATTR
```

LOCALITY_ATTR

```
private static final String LOCALITY_ATTR
```

PORTAL_DESCR

```
private static final String PORTAL_DESCR
```

PORTAL_BIZDOM

```
private static final String PORTAL_BIZDOM
```

PORTAL_PAYS

```
private static final String PORTAL_PAYS
```

PORTAL_REGION

```
private static final String PORTAL_REGION
```

PORTAL_LOC

```
private static final String PORTAL_LOC
```

---

svcCtx

```
private org.dbe.servent.ServiceContext svcCtx
```

---

props

```
private static Properties props
```

---

logger

```
private org.apache.log4j.Logger logger
```

Constructor Detail

DBEPortalRegistrationClient

```
public DBEPortalRegistrationClient(org.dbe.servent.ServiceContext svcCtx)
                            throws Exception
```

Constructor

Throws:
`Exception`

Method Detail

publish

```
public void publish(String serviceManifestPath)
            throws Exception
```

Publishes the PortalService to the SR along with the BML Data

Throws:
`Exception`

---

doSMEDataEntry

```
private void doSMEDataEntry(String serviceManisfestPath)
```

Enters the BML Data into the PortalService's SM

---

replaceAllXMLAttributeValue

```
private org.w3c.dom.Document replaceAllXMLAttributeValue(String xmlFile,
                                                         String elementName,
                                                         HashMap replacmentValues)
```

Replaces the value of the specfied element in an XML file

Returns:
the modified xml document

---

createXMLDoc

```
public org.w3c.dom.Document createXMLDoc(String xmlFile)
                                 throws FactoryConfigurationError
```

Reads in a file containing XML and creates a DOM of it's content.

Returns:
the xml DOM

Throws:
`FactoryConfigurationError`

---

writeXmlFile

```
public void writeXmlFile(org.w3c.dom.Document doc,
                         String filename)
```

Writes a XML DOM to file.

Parameters:
`filename` - the location where the file should be written to

---

doSMEPortalPublish

```
private void doSMEPortalPublish(String SMID,
                                String serviceManifestPath)
```

Publishes the SM to the SR with the supplied SMID

Parameters:

`SMID` - the SMID that is to be published with the SM

---

smToString

```
private String smToString(String serviceManifestPath)
```

Converts the supplied file to a string

---

setIsPublished

```
private void setIsPublished(String serviceManifestPath)
```

---

setProps

```
public void setProps(Properties props1)
```

---

setSM_FILE

```
public void setSM_FILE(String sm_file)
```

---

main

```
public static void main(String[] args)
```

Class DBEPortalServiceConfig

org.dbe.toolkit.portal.service

```
java.lang.Object
  └org.dbe.toolkit.portal.service.DBEPortalServiceConfig
```

---

public class **DBEPortalServiceConfig**
extends Object

Field Summary		Page
private static String	**CONFIG_FILE**	149
private static Properties	**portalServiceConfig**	149

Constructor Summary		Page
private	**DBEPortalServiceConfig**()	*149*

Method Summary		Page
static Properties	**getPortalServiceConfig**()	*149*
static String	**getString**(String key)	*149*
static void	**setCONFIG_FILE**(String config_file)	*150*
static void	**setPortalServiceConfig**(Properties portalServiceConfig)	*150*
static void	**store**(String path)	*149*

Field Detail

portalServiceConfig

```
private static Properties portalServiceConfig
```

CONFIG_FILE

```
private static String CONFIG_FILE
```

Constructor Detail

DBEPortalServiceConfig

```
private DBEPortalServiceConfig()
```

Method Detail

getString

```
public static String getString(String key)
                        throws IOException
```

Throws:

IOException

---

getPortalServiceConfig

```
public static Properties getPortalServiceConfig()
```

---

store

```
public static void store(String path)
```

---

setCONFIG_FILE

```
public static void setCONFIG_FILE(String config_file)
```

---

setPortalServiceConfig

```
public static void setPortalServiceConfig(Properties portalServiceConfig)
```

Class SMIDGenerator

org.dbe.toolkit.portal.service

```
java.lang.Object
  └org.dbe.toolkit.portal.service.SMIDGenerator
```

---

public class **SMIDGenerator**
extends Object

Field Summary		Page
private static org.apache.log4j.Logger	logger	150

Constructor Summary	Page
SMIDGenerator() Default constructor	151

Method Summary		Page
private org.w3c.dom.Document	createXMLDoc(String xmlFile)	151
String	getUID() Generates a unique ID	151
static void	main(String[] args)	152
private void	printUsage() Prints the command line usage	151
private org.w3c.dom.Document	replaceXMLElementValue(String xmlFile, String elementName, String newElementValue)	151
void	writeSMIDtoFile(String smid, String serviceDeployConf) Writes a SMID to the services deployment descriptor	151
void	writeXmlFile(org.w3c.dom.Document doc, String filename)	152

Field Detail

logger

```
private static org.apache.log4j.Logger logger
```

Constructor Detail

SMIDGenerator

```
public SMIDGenerator()
```

Default constructor

Method Detail

printUsage

```
private void printUsage()
```

Prints the command line usage

---

getUID

```
public String getUID()
```

Generates a unique ID

Returns:

the unique ID as a string

---

writeSMIDtoFile

```
public void writeSMIDtoFile(String smid,
                            String serviceDeployConf)
```

Writes a SMID to the services deployment descriptor

Parameters:

`serviceDeployConf` - the location of the service's deployment descriptor

---

replaceXMLElementValue

```
private org.w3c.dom.Document replaceXMLElementValue(String xmlFile,
                                                    String elementName,
                                                    String newElementValue)
                                             throws FactoryConfigurationError
```

Throws:

`FactoryConfigurationError`

---

createXMLDoc

```
private org.w3c.dom.Document createXMLDoc(String xmlFile)
                                  throws FactoryConfigurationError
```

Throws:

`FactoryConfigurationError`

writeXmlFile

```
public void writeXmlFile(org.w3c.dom.Document doc,
                         String filename)
```

main

```
public static void main(String[] args)
```

Package org.dbe.toolkit.portal.ui

Class Summary		Page
UICache		152
UIRetrievalDelegate		154

Class UICache

org.dbe.toolkit.portal.ui

```
java.lang.Object
  └org.dbe.toolkit.portal.ui.UICache
```

public class **UICache**
extends Object

Field Summary		Page
boolean	cacheEnabled	153
HashMap	uiCache	153

Constructor Summary	Page
UICache()	153
UICache(String cacheDirectory)	153

Method Summary		Page
String	get(String smid)	153
private HashMap	getUiCache()	153
boolean	isCached(String smid)	153
boolean	isCacheEnabled()	153
void	setCacheEnabled(boolean cacheEnabled)	153
void	store(String smid, String ui)	153

Field Detail

uiCache

```
HashMap uiCache
```

---

cacheEnabled

```
boolean cacheEnabled
```

Constructor Detail

UICache

```
public UICache()
```

UICache

```
public UICache(String cacheDirectory)
```

Method Detail

isCached

```
public boolean isCached(String smid)
```

---

store

```
public void store(String smid,
                  String ui)
```

---

getUiCache

```
private HashMap getUiCache()
```

---

get

```
public String get(String smid)
```

---

isCacheEnabled

```
public boolean isCacheEnabled()
```

---

setCacheEnabled

```
public void setCacheEnabled(boolean cacheEnabled)
```

Class UIRetrievalDelegate

org.dbe.toolkit.portal.ui

```
java.lang.Object
  └org.dbe.toolkit.portal.ui.UIRetrievalDelegate
```

public class **UIRetrievalDelegate**
extends Object

Author:
andy-edmonds

Field Summary		Page
private static String	**EXTN**	156
private org.sun.dbe.ClientHelper	**helper**	156
private static String	**HTTP**	156
private static String	**HTTP_LOCALHOST_80_PORTAL_UICACHE**	156
private boolean	**initialised**	156
private static org.apache.log4j.Logger	**logger**	156
private static String	**NIX_FILEPATH_SEP**	156
private static String	**PORT_PREFIX**	156
private static String	**PORTAL_DEFAULT_PORTAL_ID**	155
private static String	**PORTAL_DEFAULT_UI_ENTRY_POINT**	155
private static String	**PORTAL_GET_UIURL**	155
private static String	**PORTAL_HOST**	155
private static String	**PORTAL_PORT**	156
private static String	**PORTAL_UI_CACHE_CONTEXT**	156
private static String	**PORTAL_UI_CACHE_DIR**	156
private static String	**SERVENT_DEPLOY_DIR**	155
private org.dbe.studio.tools.smcreator.core.eb.SM	**sm**	157
private org.dbe.studio.tools.smcreator.core.SMCreator	**smc**	157

Constructor Summary	Page
UIRetrievalDelegate(org.sun.dbe.ClientHelper helper)	157

Method Summary		Page
private String	**getCacheLocalUI**(String smid) Downloads the remote UI as a zip file, extracts it and generates a URL for it	158
private String	**getLocalUIURL**(File outputFile, String smid) Unzips the recieved remote UI zip, extracts it to a web context and returns back the URL of that locally cached UI	159
private String	**getPortalEndpoint**(String smid)	157

private String	**getPortalID**(String smid) Gets the SMID of the portal where the requested UI is located	*159*
private String	**getServiceUI**(String smid) This gets the URL where the service UI is located	*160*
String	**getUI**(String smid) Gets the URL of a locally cached copy of the requested UI	*157*
private File	**getUIArchive**(String smid, String uiZipURL) Downloads the zipped UI from the remote servent	*159*
private String	**getUIArchiveURL**(String smid) Gets the URL of the remote UI zip file	*158*
private String	**getUIEntryPoint**(String url) This gets the main entry point for an openlaszlo UI	*160*
private String	**getUIIF**(String smid)	*158*
private org.dbe.toolkit.proxyframework.Workspace	**getWorkspace**(String smid) Gets a Workspace object	*160*
private boolean	**ifisIF**(String smid)	*158*
private void	**initParams**() Initialises parameters from the portalClientConfig.properties file	*157*
private boolean	**isPortal**(String smid)	*158*

Field Detail

SERVENT_DEPLOY_DIR

```
private static String SERVENT_DEPLOY_DIR
```

PORTAL_HOST

```
private static String PORTAL_HOST
```

PORTAL_DEFAULT_UI_ENTRY_POINT

```
private static String PORTAL_DEFAULT_UI_ENTRY_POINT
```

PORTAL_DEFAULT_PORTAL_ID

```
private static String PORTAL_DEFAULT_PORTAL_ID
```

PORTAL_GET_UIURL

```
private static String PORTAL_GET_UIURL
```

PORTAL_UI_CACHE_DIR

```
private static String PORTAL_UI_CACHE_DIR
```

PORTAL_UI_CACHE_CONTEXT

```
private static String PORTAL_UI_CACHE_CONTEXT
```

PORTAL_PORT

```
private static String PORTAL_PORT
```

HTTP_LOCALHOST_80_PORTAL_UICACHE

```
private static String HTTP_LOCALHOST_80_PORTAL_UICACHE
```

HTTP

```
private static String HTTP
```

EXTN

```
private static String EXTN
```

PORT_PREFIX

```
private static String PORT_PREFIX
```

NIX_FILEPATH_SEP

```
private static String NIX_FILEPATH_SEP
```

---

helper

```
private org.sun.dbe.ClientHelper helper
```

---

initialised

```
private boolean initialised
```

---

logger

```
private static org.apache.log4j.Logger logger
```

---

sm

```
private org.dbe.studio.tools.smcreator.core.eb.SM sm
```

Author:
mbordin This variable it is needed to IF serviceManifest

---

smc

```
private org.dbe.studio.tools.smcreator.core.SMCreator smc
```

Constructor Detail

UIRetrievalDelegate

```
public UIRetrievalDelegate(org.sun.dbe.ClientHelper helper)
```

Method Detail

initParams

```
private void initParams()
```

Initialises parameters from the portalClientConfig.properties file

---

getUI

```
public String getUI(String smid)
            throws DBEPortalException
```

Gets the URL of a locally cached copy of the requested UI

Parameters:
`smid` - the SMID of the service which has the requested UI

Returns:
the URL of the UI

Throws:
DBEPortalException

---

getPortalEndpoint

```
private String getPortalEndpoint(String smid)
                         throws DBEPortalException
```

Throws:
DBEPortalException

---

isPortal

```
private boolean isPortal(String smid)
```

---

getUIIF

```
private String getUIIF(String smid)
               throws Exception
```

Parameters:
`smid` - it is the ServiceManifestID
Returns:
String It return the url to lunch
Throws:
`Exception` - If something it is wrong it throw an exception the method create the correct UI for an IF serviceManifest
Author:
mbordin

---

ifisIF

```
private boolean ifisIF(String smid)
```

Parameters:
`smid` - It is the service manifest ID
Returns:
boolean It return TRUE if a serviceManifst it is an iteraction form.
Author:
mbordin

---

getCacheLocalUI

```
private String getCacheLocalUI(String smid)
                       throws DBEPortalException
```

Downloads the remote UI as a zip file, extracts it and generates a URL for it
Parameters:
`smid` - SMID of the service implementing the requested UI
Returns:
URL of the locally cached UI
Throws:
`DBEPortalException`

---

getUIArchiveURL

```
private String getUIArchiveURL(String smid)
                       throws DBEPortalException
```

Gets the URL of the remote UI zip file

Parameters:

`smid` - SMID of the service implementing the service

Returns:

URL of the remote UI zip file

Throws:

`DBEPortalException`

---

getPortalID

```
private String getPortalID(String smid)
                    throws DBEPortalException
```

Gets the SMID of the portal where the requested UI is located

Parameters:

`smid` - SMID of the service implementing the UI requested

Returns:

the ID of the hosting portal

Throws:

`DBEPortalException`

---

getUIArchive

```
private File getUIArchive(String smid,
                          String uiZipURL)
                   throws DBEPortalException
```

Downloads the zipped UI from the remote servent

Parameters:

`smid` - SMID of the service that implements the UI

`uiZipURL` - the URL of the zipped UI

Returns:

a

Throws:

`DBEPortalException`

See Also:

object that corresponds to the zip file containing the UI

---

getLocalUIURL

```
private String getLocalUIURL(File outputFile,
                             String smid)
                      throws DBEPortalException
```

Unzips the recieved remote UI zip, extracts it to a web context and returns back the URL of that locally cached UI

Parameters:

`outputFile` - the file to unzip

`smid` - the SMID of the service implementing the UI

Returns:

the local URL of the UI

Throws:

`DBEPortalException`

getServiceUI

```
private String getServiceUI(String smid)
```

This gets the URL where the service UI is located

Parameters:

`smid` - SMID of the service implementing the UI

Returns:

The URL of the UI

---

getUIEntryPoint

```
private String getUIEntryPoint(String url)
```

This gets the main entry point for an openlaszlo UI

Parameters:

`url` - The URL to extract the main entry point from

Returns:

The main entry point

---

getWorkspace

```
private org.dbe.toolkit.proxyframework.Workspace getWorkspace(String smid)
                                                 throws DBEPortalException
```

Gets a Workspace object

Returns:

The workspace object

Throws:

`DBEPortalException`

Package org.dbe.toolkit.portal.ui.tools

Class Summary		Page
UIResourcesUnzipper		*160*
UIResourcesZipper		*163*

Class UIResourcesUnzipper

org.dbe.toolkit.portal.ui.tools

```
java.lang.Object
  |  org.dbe.toolkit.portal.ui.tools.UIResourcesUnzipper
```

---

public class **UIResourcesUnzipper**
extends Object

Author:
andy-edmonds

Field Summary		Page
private static org.apache.log4j.Logger	**logger**	*161*
private static final String	**NIX_FILEPATH_SEP**	*161*

Constructor Summary	Page
UIResourcesUnzipper()	*161*

Method Summary		Page
private static org.apache.log4j.Logger	**getLogger**()	*162*
static void	**unzip**(String zipName, String extractionPath) Extracts a zip file	*161*

Field Detail

NIX_FILEPATH_SEP

```
private static final String NIX_FILEPATH_SEP
```

---

logger

```
private static org.apache.log4j.Logger logger
```

Constructor Detail

UIResourcesUnzipper

```
public UIResourcesUnzipper()
```

Method Detail

unzip

```
public static void unzip(String zipName,
                         String extractionPath)
                  throws IOException
```

Extracts a zip file

Parameters:

`zipName` - the absolute name of the zip file

`extractionPath` - the absolute path to where the zip file is to be extracted to

Throws:

`IOException`

---

getLogger

```
private static org.apache.log4j.Logger getLogger()
```

Class UIResourcesZipper

org.dbe.toolkit.portal.ui.tools

```
java.lang.Object
  └org.dbe.toolkit.portal.ui.tools.UIResourcesZipper
```

---

public class **UIResourcesZipper**
extends Object

Author:
andy-edmonds

Field Summary		Page
private static final String	**NIX_FILEPATH_SEP**	163
private static boolean	**relative**	163
private static String	**relativeDir**	163

Constructor Summary	Page
UIResourcesZipper()	164

Method Summary		Page
static void	**setRelative**(boolean relative) Sets archive creation to 'relative'	164
private static void	**zip**(org.apache.log4j.Logger logger, File[] files, ZipOutputStream zos)	164
static void	**zip**(org.apache.log4j.Logger logger, String file, String saveTo) create a zip file from a list of files it'll recursively zip if a file's a directory	164

Field Detail

NIX_FILEPATH_SEP

```
private static final String NIX_FILEPATH_SEP
```

---

relative

```
private static boolean relative
```

---

relativeDir

```
private static String relativeDir
```

Constructor Detail

UIResourcesZipper

```
public UIResourcesZipper()
```

Method Detail

zip

```
public static void zip(org.apache.log4j.Logger logger,
                       String file,
                       String saveTo)
                throws IOException
```

create a zip file from a list of files it'll recursively zip if a file's a directory

Parameters:

`file` - Name of the zip file to create

`saveTo` - location to save the created zip file to

Throws:

`IOException`

`Exception`

---

zip

```
private static void zip(org.apache.log4j.Logger logger,
                        File[] files,
                        ZipOutputStream zos)
                 throws IOException
```

Throws:

`IOException`

---

setRelative

```
public static void setRelative(boolean relative)
```

Sets archive creation to 'relative'

Parameters:

`relative` - a boolean to set the flag