Defence Research and Development Canada
Recherche et développement pour la défense Canada

DEFENCE R&D DÉFENSE

Simulation network exploitation

SEDRA and SEDReDE portals: specifications document

Oliver Schoenborn

Defence R&D Canada – Ottawa
CONTRACT REPORT
DRDC Ottawa CR 2008-315
April 2009

Canada

Simulation network exploitation

SEDRA and SEDReDE portals: specifications document

Oliver Schoenborn
CAE Professional Services

Prepared By:
CAE Professional Services
1135 Innovation Dr. Suite 300
Ottawa, ON K2K 3G7
Contract Project Manager: Leo Roberts, 613-293-8993
W8475-06BM04
CSA: Nacer Abdellaoui, Defence Scientist, 613-998-4582

Defence R&D Canada – Ottawa

Contract Report
DRDC Ottawa CR 2008-315
April 2009

Contract Scientific Authority

Original signed by Nacer Abdellaoui

Nacer Abdellaoui

Defence Scientist

Approved by

Original signed by Julie Tremblay

Julie Tremblay

H/CARDS

Approved for release by

Original signed by Pierre Lavoie

Pierre Lavoie

DRP Chair

Abstract

This document specifies the functionality that should be supported by the Synthetic Environment Distributed Resources Access (SEDRA) portal and Synthetic Environment Distributed Resources Data Entry (SEDReDE) portal. These applications are intended to facilitate the description and access to simulation resources, such as documents, video clips, audio captures, and computer programs, available on a distributed network of computers. The resources can be described in one or more databases via SEDReDE and browsed using SEDRA. The latter delegates to a separate application (named SEDReC – Synthetic Environment Distributed Resources Control) the task of executing those simulation resources which are executable.

Résumé

Ce document spécifie les fonctionnalités soutenues par le portail d'accès aux ressources distribuées des environnements synthétiques (SEDRA) ainsi que le portail de saisie de données de ces ressources (SEDReDE). Ces applications sont destinées à faciliter la description et l'accès aux ressources de simulation, tels que des documents, des clips vidéo, capture audio, et logiciels; ces ressources étant évidemment disponibles sur un réseau distribué. Les ressources peuvent être décrites dans une ou plusieurs bases de données via SEDReDE et navigables en utilisant SEDRA. SEDRA délègue la tâche d'exécution des composantes exécutables à une autre application du nom SEDReC (Control des ressources distribuées des environnements synthétiques).

This page intentionally left blank.

Executive summary

Simulation network exploitation: SEDRA and SEDReDE portals: specifications document

Oliver Schoenborn; DRDC Ottawa CR 2008-315; Defence R&D Canada – Ottawa; April 2009.

Introduction or background: This document specifies the functionality that should be supported by the Synthetic Environment Distributed Resources Access (SEDRA) portal and Synthetic Environment Distributed Resources Data Entry (SEDReDE) portal.

Results: These applications are intended to facilitate the description and access to simulation resources, such as documents, video clips, audio captures, and computer programs, available on a distributed network of computers. The resources can be described in one or more databases via SEDReDE and browsed using SEDRA. The latter delegates to a separate application (named SEDReC – Synthetic Environment Distributed Resources Control) the task of executing those simulation resources which are executable.

Significance: This document covers several aspects of the system specification: Functional and Non-functional Requirements, constraints, system components and system actors, and, finally, Use Cases. Only a subset of these use cases will be implemented in the first iteration of development. Subsequent iterations must revise this document, e.g. by removing, changing or adding use cases as appropriate.

This document also specifies, as an Annex, the XML element structure of the information received from the database, and gives an example XML file instance. Finally, it provides a preliminary guideline for the information that should be stored in the database used by the Portal.

Sommaire

Simulation network exploitation: SEDRA and SEDReDE portals: specifications document

Oliver Schoenborn; DRDC Ottawa CR 2008-315; R & D pour la défense Canada – Ottawa; Avril 2009.

Introduction ou contexte: Ce document spécifie les fonctionnalités qui devraient être soutenues par le portail d'accès des ressources distribuées des environnements synthétiques (SEDRA) et le portail de saisie de données des ressources distribuées des environnements synthétiques (SEDReDE).

Résultats: Ces applications sont destinées à faciliter la description et l'accès aux ressources de simulation, tels que des documents, des clips vidéo, capture audio, ainsi que logiciels, disponibles sur un réseau distribué d'ordinateurs. Les ressources peuvent être décrites dans une ou plusieurs bases de données via SEDReDE et navigué en utilisant SEDRA. SEDRA délègue la tâche d'exécution de ces ressources à une autre application distincte du nom de SEDReC – Station de Contrôle des ressources distribuées des environnements synthétiques.

Importance: Ce document couvre plusieurs aspects des spécifications du système: exigences fonctionnels et non fonctionnels, les contraintes, les composantes et les acteurs/rôles du système, et les cas d'utilisation. Durant cette première itération du développement du projet, seul un sous-ensemble de ces cas d'usage sera implémenté. Les Itérations ultérieures doivent impérativement réviser ce document, et ceci par la suppression, la modification ou l'ajout de cas d'utilisation.

L'annexe de ce document spécifie la structure de l'élément XML de l'information reçue de la base de données, et donne un exemple typique de fichier XML. Il offre aussi un guide préliminaire des informations qui doivent être stockés dans la base de données utilisée par le portail.

Table of contents

Abstract …….. i
Executive summary iii
Table of contents v
List of figures vii
List of tables viii
1....Overview 1
2....Functional Requirements 2
2.1 SEDRA Portal 2
2.2 SEDReDE Portal 2
3....Non-Functional requirements 6
3.1 SEDRA Portal 6
3.2 SEDReDE Portal 6
4....System Components 7
5....SEDRA Actors 8
6....SEDRA Portal Use Cases 9
7....SEDRA Portal Test Cases 30
TS-1a. Set which DB(s) accessible from Portal 31
TS-1b. Connect to a Database 32
TS-1c. Edit the DB aliases and associated usernames 33
TS-3. Browse simulation resource tree 34
TS-4. Run simulation component 36
TS-5a. View a simulation document 36
TS-5b. Save a simulation document locally 37
TS-6. Find meta-data by keyword – Basic 39
References 42
Annex A .. Simulation Resources Description Structure 43
Annex B .. State diagram of SEDRA application 49
Annex C .. Simulation Resources Documentation Specification 50
Annex D .. Implementation of SEDRA 52
D.1 Modules and Unit Tests 52
D.2 System Requirements 54
Annex E... Data Model and Naming Conventions for Relational Databases 56
E.1 Naming conventions 56
E.2 Entity Relationships 63
List of symbols/abbreviations/acronyms/initialisms 69

This page is intentionally left blank

List of figures

Figure 1: System Components for SEDReDE portal 3
Figure 2: Relationship between portals and SEDReC station 7
Figure 3: Prototype Tools->Options dialog for SEDRA portal 11
Figure 4: Example SEDRA Portal main window, no databases defined 13
Figure 5: Portal in "connected" state 14
Figure 6: Menu Specification 19
Figure 7: Possible drop-down menus for nodes 22
Figure 8: Example log window 29
Figure 9: SimResources and SimSystem elements 44
Figure 10: SimSystem, ComponentGroup, and LaunchConfig elements 44
Figure 11: Document and Group elements 45
Figure 12: BasicNode, Summary and Field elements 46
Figure 13: Complete data type relationship diagram of all elements 47
Figure 14: State chart of Portal 49
Figure 15: DB schema tables related to sys_group (SystemGroup) 63
Figure 16: DB schema tables related to sim_sys (SimulationSystem) 64
Figure 17: DB schema tables related to comp_group (ComponentGroup) 65
Figure 18: DB schema tables related to sim_comp (SysComponent) 66
Figure 19: DB schema tables related to doc_group (DocumentGroup) 66

List of tables

Table 1: Use Case Template. 9
Table 2: Use Case-0. Give it a reference number and name 9
Table 3: Use Case-1a. Set which DB(s) accessible from Portal 10
Table 4: UC-1b. Connect to a Database 12
Table 5: UC-1c. Edit the DB aliases and associated usernames 14
Table 6: UC-1d. Support new DB types 15
Table 7: UC-1e. Refresh displayed resource tree 16
Table 8: UC-1d. Support new DB types 16
Table 9: UC-1e. Refresh displayed resource tree 17
Table 10: UC-2. Get Help on Portal Application 18
Table 11: UC-3. Browse simulation resource tree 19
Table 12: UC-4. Run simulation component 21
Table 13: UC5a. View a simulation document 22
Table 14: UC-5b. Save a simulation document locally 23
Table 15: UC-6. Find meta-data by keyword: Basic 24
Table 16: UC-7.Find meta-data by keyword: Advanced 25
Table 17: UC-8. Add simulation resource info to DB 26
Table 18: UC-9. Modify simulation resource info in DB 27
Table 19: UC-10. Toggle the visibility of log window 27

1 Overview

This document specifies the functionality that should be supported by the Synthetic Environment Distributed Resources Access (SEDRA) portal and Synthetic Environment Distributed Resources Data Entry (SEDReDE) portal. These applications are intended to facilitate the description and access to simulation resources, such as documents, video clips, audio captures, and computer programs, available on a distributed network of computers. The resources can be described in one or more databases via SEDReDE and browsed using SEDRA. The latter delegates to a separate application (named SEDReC – Synthetic Environment Distributed Resources Control) the task of executing those simulation resources which are executable.

This document covers several aspects of the system specification: Functional and Non-functional Requirements, constraints, system components and system actors, and, finally, Use Cases [1]. Only a subset of these use cases will be implemented in the first iteration of development. Subsequent iterations must revise this document, e.g. removing, changing or adding use cases as appropriate.

Note that this document is intended for software developers and as such assumes familiarity with software development terminology and process, as well as with "technologies" such as UML, databases, XML, GUIs, HTML, etc [2].

2 Functional Requirements

2.1 SEDRA Portal

The following high-level requirements are defined for the SEDRA portal:

- Allow the user to retrieve the availability of simulation resources (hardware, software, documentation). A top-down, hierarchical structure is required for best presentation.
- Allow the user to retrieve those simulation resources that match given keywords in their meta-data.
- Allow the user to retrieve all simulation database (DB) information pertaining to a given keyword(s)
- Allow the user to specify to which DB to connect. All connections to DB's must specify a user logon identifier, with a password. The user is allowed to change the DB connection and the username during a session, to support the notion of role and access control as (and if) supported by the DB.
- Allow the user to define new DB/user pairs. This information is to persist from one session to the next.
- Allow the user to enter new or modify existing database records. A web enabled GUI is required.
- Allow the user to run a distributed simulation exercise/experiment. Partial simulations are to be accommodated to the extent possible with the specific simulation architecture.
- Allow the user to view documentation resources in default "viewers". For example, an ".avi" file type should be viewed in the operating system's default AVI viewer; a ".pdf" type file in the operating system's default PDF viewer, etc. The default viewer is to be launched automatically when the user requests a "view" action.
- The application should:
  - Allow for new databases to be integrated, via the concept of DB proxy plug-in. The DB plug-in should ignore or reject (as appropriate) any requests that cannot be fulfilled by the DB itself.
  - Allow new launchers to be integrated, via the concept of Launcher plug-in. A launcher is an application that can control other applications.

Further details on function requirements for the SEDRA portal are given in section 6 in the form of Use Cases.

2.2 SEDReDE Portal

The SEDReDE web portal is a tool that is intended to enhance the functionality of the SEDRA portal. The SEDRA portal provides a means to easily view information relating to simulation resources. The SEDReDE web portal provides the added capability of editing the information that

was previously only visible in the SEDRA portal. As such, the user interface of the SEDReDE web portal is designed to resemble that of the SEDRA portal. The goal is to provide the user with a familiar and intuitive interface for creating and modifying simulation resource information. In addition to these qualitative requirements, the following functional requirements are also given:

1. The SEDReDE web portal must be a web-based application.

2. The SEDReDE web portal must be capable of accessing and modifying the MySQL implementation of the SEDR database schema.

3. The SEDReDE web portal must be developed to protect the user from manipulating the database in a way that violates the SEDR database schema.

A mock-up of the SEDReDE web portal is given in the next figure:

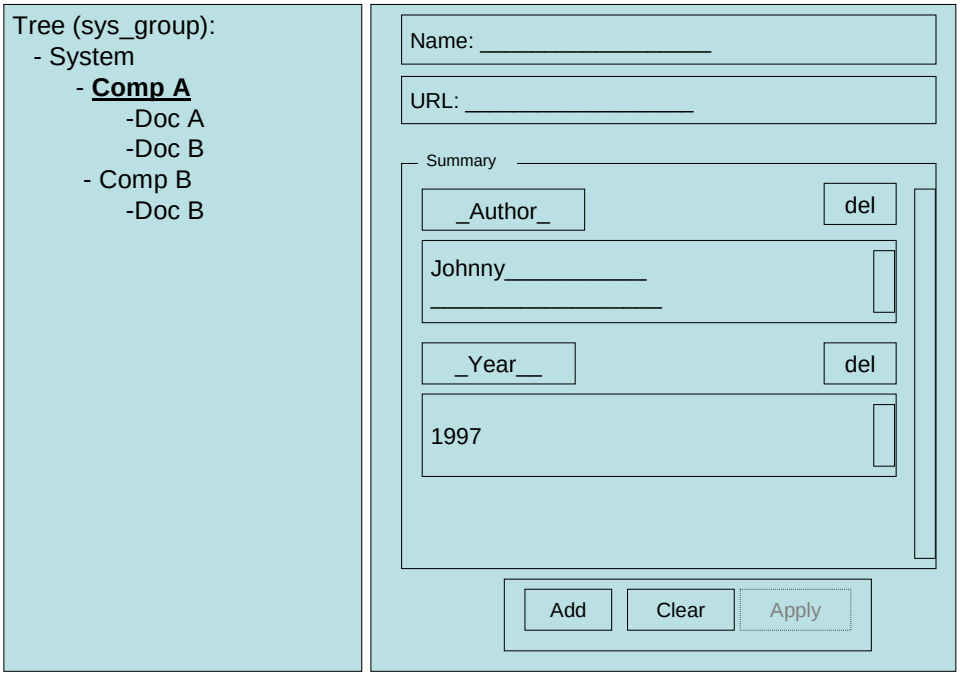

Figure 1: System Components for SEDReDE portal

The SEDReDE web portal is comprised of 3 primary regions with which the user of the system can interact. The first region is the simulation resource tree that appears on the left hand side of the user interface. When the user logs in to the system, the simulation resource tree is populated with any resources that are currently stored in the database.

The second main region of the SEDReDE web portal interface is the node information. Each simulation resource in the simulation tree is represented by a single node and its corresponding

information. A simulation resource must have at least a name and can have an associated URL depending upon its type. Each time a resource is selected from the tree its node information is displayed in the Name and URL fields of the node information region of the user interface. If the selected resource is a document type (i.e. simulation document or control recipe), then its associated URL is also displayed (if it exists). Otherwise, the URL field is empty and disabled.

The third main region of the user interface is the summary information region. Each simulation resource can have any number of summary fields associated with it. When the user selects a simulation resource from the resource tree, all summary fields for the selected node are automatically populated in the summary information region. Each summary field has a name and value as well as its own “del” button. The “del” button removes the summary field from its associated simulation resource in the database.

In addition to these three primary regions, there is also a button panel containing three action buttons which are used to manipulate the summary fields and node information for a given resource.

Using the button panel, the following actions are provided:

1. “Add” – this button adds a new (blank) summary field to the selected simulation resource in the database.

2. “Clear” – this button removes all summary fields from the currently selected simulation resource in the database.

3. “Apply” – this button causes all field changes for the currently selected resource to be applied to the database. The “Apply” button is only available if there are changes pending to the currently selected resource. Pressing “Apply” will cause it to be disabled once changes have been committed. The “Apply” button becomes enabled whenever the user modifies the text in any field of the node information or in a summary field. If the user tries to navigate to a different simulation resource with uncommitted changes pending, appropriate warnings will be provided and the user will be given the opportunity to apply the changes.

Direct interaction with the simulation tree is also available through a “right-click” and the following functions are provided:

1. “Add” – this function will display a popup menu when selected. Depending on the type simulation resource selected, a different list of possible additions is given. For example, if the resource is a “Document Group”, then the user is presented with the option to add either another “Document Group” or a “Simulation Document”. All possible additions are presented in keeping with the rules of the database schema.

2. “Delete” – this function simply removes the selected node from the database.

3. “Rename” – this function renders the resource tree in “Edit” mode. At this stage the user can type in a new name for the selected node and press “Enter”. This causes the new name to be written to the database for the selected resource. The user can also rename a resource by directly typing a new name in the “Name” field of the node information region and then

pressing the “Apply” button. The “Edit” mode of the tree can also be invoked by the user by pressing the “F2” key and proceeding in a similar fashion.

3 Non-Functional requirements

3.1 SEDRA Portal

- The Portal core logic must not have any knowledge of specific databases. The concept of DB proxy plug-ins will provide a uniform interface to extend the Portal for any DB.
- The Portal must not have any launcher intelligence. The Portal will delegate to an external application the task of launching, monitoring, and terminating local and remote applications. This external application is to be created as a "Portal component", or be obtained from a third-party and an interface created to make it interact with the Portal.
- Timeliness: A response time within 30 seconds of the issuance of any user request is considered acceptable by most users. The presentation of the information to the user must be timely. (This may affect some software design aspects that relate to the data transfer from DB to Portal over the network).
- The data stored in the DB must be platform independent. This precludes formats such as MS Word document format (can't be viewed on Linux), but allows PDF.
- Portal must run on MS Windows and Linux.
- The GUI should be desktop based and/or web based.

3.2 SEDReDE Portal

In addition to the list of requirements given above the following non-functional requirements are defined for the SEDReDE web portal.

1. Access to the SEDReDE web portal needs to be secured and access is restricted to those possessing appropriate credentials. These restrictions can be configured through any available web server mechanisms.

2. Network bandwidth usage for database interactions should be made as efficient as possible. This is to improve usability of the system by decreasing the time that the user waits to see database changes being reflected in the user interface. To that end, the resource tree is not completely populated at system start up. Only the root node of the tree and any immediate children are displayed. As the user traverses the tree, selected nodes are continually populated with their respective values and children.

3. The installation of the system will always create a minimally populated database at the very least. This process will install a "System Group" at the very root of the tree as the initial container for every other simulation resource.

4 System Components

1. SEDRA Portal: Client application to view information (meta-data) stored in databases (DB), concerning simulation resources available on the network. The application contains any number of DB Proxy plug-in that are loaded as required, based on the desired connection. The plug-in communicates with a DB server of the associated type.

2. Database (DB): stores data about the simulation resources, and may even store some of the simulation resources proper. There can be different database types, and for a given type there can be several databases.

3. Database Server (DBS): A database is accessed via a DB server. All DB servers are third-party applications that are specific to the DB they mediate, and are not part of the Portal.

4. Network: the Ethernet network that allows the Portal, DB server and simulation programs to be on different machines.

5. SEDReC Station: an application that supports remote application management so simulation components can be run remotely (assuming proper computer and network configuration). Different launchers are possible.

6. SEDReDE Portal: Client application to edit information (meta-data) stored in databases (DB), concerning simulation resources available on the network. The portal cannot verify whether the resources described by the meta-data indeed exist

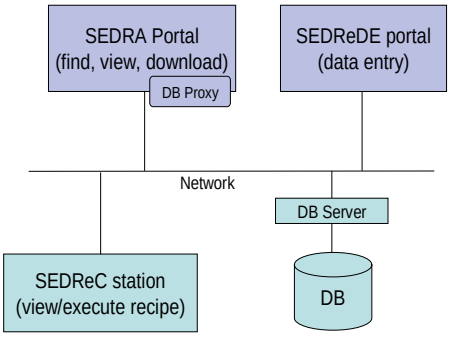

Figure 2: Relationship between portals and SEDReC station

5 SEDRA Actors

The following actors are primary (i.e. act on one or more of the Portal system components in some way):

- User: uses the SEDRA Portal to discover, view and/or execute simulation resources
- Modifier: interacts directly with the DB server or the SEDReDE portal (not the SEDRA Portal) to add, change and remove data from the DB; more than one person can have "modifier" status;
- Portal Admin: in charge of user account management for a given DB, and other chores such as installation and maintenance of Portals and databases, data backup, notifications to users as to URLs of databases etc. This actor interacts with Portals only during installation, update and removal of the application.

The following actors are secondary (i.e. respond to requests from Portal):

- DB server: application that allows for database queries
- Viewer: application that is able to show a document obtained from the database
- Registry Manager: the OS-specific component that is available at runtime for applications to store user settings; for OS that don't have registry, Portal must fake its existence for consistency of interface/logic

6 SEDRA Portal Use Cases

Each use case will be documented in a table that has the following structure. The left column does not change:

Table 1: Use Case Template.

Modification History
Actors
Goal
Pre-conditions
Steps
Variations
Non-functional
Issues
Strategies

Each row is described in the following table, with example if appropriate:

Table 2: Use Case-0. Give it a reference number and name

Modification History	Nov 28th, 2005 Dec 15th, 2005	Author Author	Creation Resolved issues
Actors	Primary: Name primary actors Secondary: Name secondary actors		
Goal	Goal to be achieved by use case		
Pre-conditions	1. First pre-condition or assumption for this use case to succeed 2. Second pre-condition or assumption		
Steps	1. First interaction 2. IF condition THEN Next interaction Next interaction ELSE Next interaction 3. IN PARALLEL (start in (any) order): Next interaction, refer to #2.2 Next interaction 4. WHILE condition DO:		

	Next interaction Next interaction
Variations	#2.2a: List variation for how step 2.2 can occur
Non-functional	Label: description (label is e.g. Performance, Reliability, Fault tolerance, Frequency, Priority)
Issues	Numbered list of any issues that need resolving
Strategies	(Optional) Discuss possible implementation strategies

Each use case is on a separate page.

Table 3: Use Case-1a. Set which DB(s) accessible from Portal

Modification History	Nov 28, 2005 Feb 22, 2006	Schoenborn Schoenborn	Created Sync'd with implementation
Actors	Primary: user		
Goal	Specify which database(s) (DB) the User wishes to have easy access to from a Connections box, and, for each DB, which user name(s) the User wishes to have easy access to.		
Pre-conditions	1. GUI started but not in connected state 2. User knows DB's URL and username(s) defined for him/her by the Portal Admin		
Steps	1. Go to the Tools->Options dialog 2. Disable/Enable DB/User modification buttons based on selection (if any) 3. WHILE user has more DBs to specify, DO: User clicks on the "Add DB" to add a DB: fields are URL and alias name, and database type IF all fields ok THEN accept it ELSE print message, wait for ok, and allow entry again, try 3.2 again IF user cancelled THEN abort the new DB Portal selects the new DB item User creates a user name to use when connecting to that DB IF username ok THEN accept it ELSE print message, wait for ok, and allow entry again, try 3.6 again Repeat #3.5 if more than one username defined Change, if desired, the order in which the DB and users will appear in the connection box, by using the "Move up" and "Move Down" buttons		

	4. Click OK to save changes, or CANCEL to forget them 5. IF at least one DB defined, THEN activate the Connections pane
Variations	#6a: allow the user to drag a node from the resource tree and drop it onto the connection box: this would open the options dialog showing the defined databases, giving the user the ability to change the order, add user names, and defined an associated alias; but due to #2a, only the new addition could be changed. #3.2a: allow user to "get the list of defined usernames" for the selected database. This should require, if possible, a password. #3.1a: allow user to "get the list of available databases" defined in the selected database/user. This presumes that the plug-in mechanism (see below) is able to query the database, that the user has selected a username, and that the user has the password that will connect them to that database.
Non-functional	See Figure 3 for example GUI
Issues	
Strategies	A class interface can be designed to abstract out the DB connection protocol, so different types of DBs can be supported simply by changing the protocol in a class derived from the abstract base.

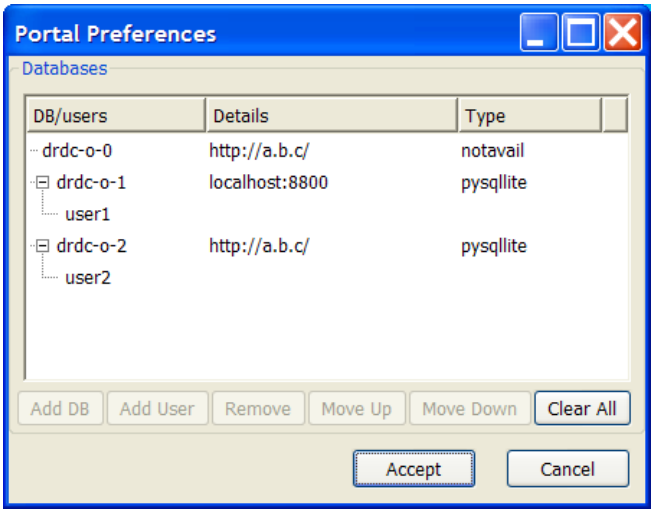

Figure 3: Prototype Tools->Options dialog for SEDRA portal

Table 4: UC-1b. Connect to a Database

Modification History	Nov 28, 2005	Schoenborn	Created
	Feb 22, 2006	Schoenborn	Renamed, rewritten
	Jun 12, 2006	Schoenborn	Renumbered
Actors	Primary: user		
Goal	See what simulation resources are available from a given database		
Pre-conditions	1. GUI software installed (locally) 2. DB plug-in installed (locally) 3. DB created and populated 4. User has the DB URL and (if required) a username valid for that DB, and has entered them in settings as per UC-1a.		
Steps	1. Start GUI 2. User selects which database/user in the Connection Panel 3. User clicks “Connect” button; Portal enters the “connecting” state 4. The GUI changes or shows a dialog from which the connection can be aborted, if it takes too long 5. Portal tells DB proxy to connect, and gives it required parameters 6. Portal shows box if password required. IF user cancels THEN the connection attempt is aborted and the Portal returns to the “unconnected” state 7. IF connection succeeds THEN GUI enters the “connected” state, in which the DB/user cannot be changed GUI changes connect button to “Disconnect” and enables Refresh button Portal gets master sim resources file from DB proxy and validates the XML; if invalid, connection aborts and GUI returns to the “disconnected” state GUI creates tree view of that file in left pane GUI activates the left and right sim resource view panes GUI waits for further input from user ELSE Show error message, with any error info obtained from connection Portal goes back to “disconnected” state Wait for user to OK		
Variations	#2a: IF no DB are defined in options registry THEN o Disable all panels so user can’t do anything o Disable all menu items except Options o Show message saying that main area of GUI can’t be used until some DB/user pairs are defined in the Options menu o Wait for user to OK		

	o Wait till user does use case UC-1a o Resume normal step #2 #7.6a: IF a filter is active from a previous connection, THEN re-apply the filter to the new connection results
Non-functional	• See Figure 4 and Figure 5 for examples of GUI before any databases defined, and after connecting to a database that has just been defined. • Only the parts of the GUI that are valid for the current application state should be enabled; e.g. the “Sim Resources” sub-section of the main GUI should be enabled only after a connection has succeeded.
Issues	
Strategies	

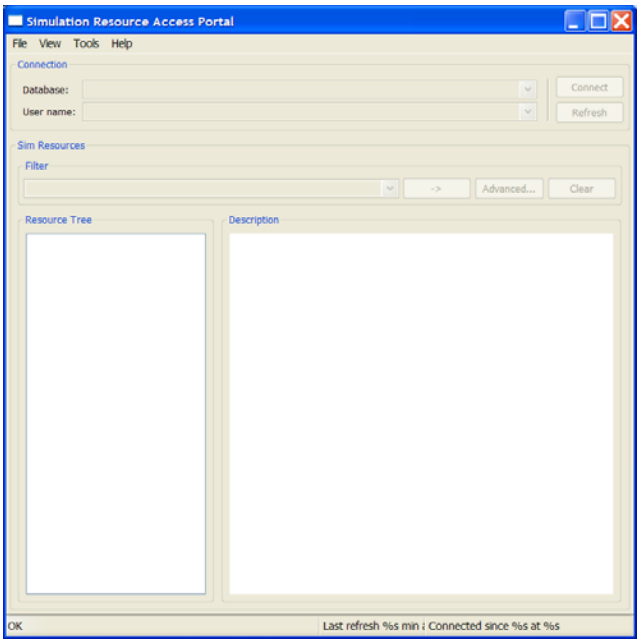

Figure 4: Example SEDRA Portal main window, no databases defined

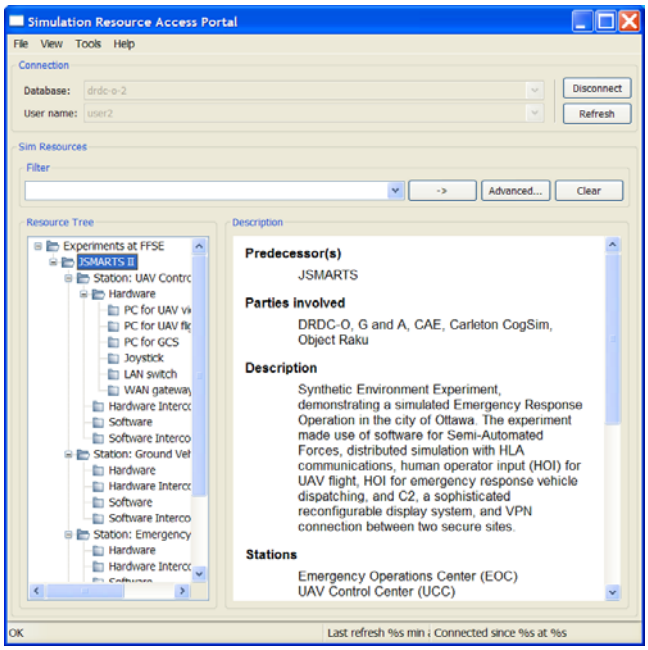

Figure 5: Portal in "connected" state

Table 5: UC-1c. Edit the DB aliases and associated usernames

Modification History	June 14, 2005	Schoenborn	Created
Actors	Primary: user		
Goal	For any database alias defined, allow editing the alias, URL, type or usernames for that database.		
Pre-conditions	1. Same as for Use Case #1b, and 2. Use Case #1b has been completed at least once		
Steps	1. Go to the Tools->Options dialog 2. Disable/Enable DB/User modification buttons based on selection (if any) 3. IF user clicks on a DB or username and then clicks on “Edit” THEN show same text entry as for Use Case #1b, but the fields are populated with the information for the DB alias being edited IF user clicks OK and new DB alias info is invalid THEN show message, allow user to OK it, repeat from #3.1 IF user clicks CANCEL the changes are abandoned 4. Click OK to save changes, or CANCEL to forget them 5. IF at least one DB still defined, THEN activate the Connections		

	pane
Variations	#2a: if in connected state, none of the DB buttons are enabled, so as to prevent user from deleting current (connected) DB/user; in this case #3 will be unavailable and there won't be any changes to save in #4
Non-functional	Invalidity of DB: any field empty or DB alias already in use Invalidity of username: empty or name already defined in DB
Issues	
Strategies	

Table 6: UC-1d. Support new DB types

Modification History	May 12th, 2006	Schoenborn	Created
Actors	Primary: User		
Goal	Support new database types without having to modify the application		
Pre-conditions	A plug-in exists for the database type of interest		
Steps	1. User gets the plug-in (e.g. from Administrator via email, website, etc) 2. User puts the plug-in files in the plug-in folder on user's computer 3. User starts portal 4. User defines a database using the new type (Use Case #1b); this can be done at any of these steps 5. User connects to database		
Variations	Steps 3 to 5 may not be necessary and can be done before or after steps 1 and 2. #4a: Eventually, the plug-in system could: - find all available DB plug-ins (defined on local host) - enable user to select which of those types is a given DB It could even possibly find all available DB plug-ins defined on a server specified by user, download the selected plug-ins, install them, and add them to its list of available DB types in the Options windows.		
Non-functional			
Issues	There is no way for Portal to verify that the type chosen by user is the correct type for the DB pointed to by the URL for the DB		
Strategies			

Table 7: UC-1e. Refresh displayed resource tree

Modification History	May 12th, 2006	Schoenborn	Created
Actors	Primary: User		
Goal	Allow user to refresh the displayed resource tree without having to disconnect and reconnect (esp. useful if passwords would normally be involved)		
Pre-conditions	Connected to a DB		
Steps	1. The user clicks “Refresh” 2. Portal asks for confirmation to proceed. IF user clicks no, THEN operation is cancelled 3. Portal enters the “refreshing” state: portal attempts a new connection, without closing the current connection; the user can abort the refresh; the user can continue browsing while the refresh is taking place 4. IF the refresh fails, THEN operation is cancelled 5. IF the refresh succeeds, Portal regenerates tree view, loosing which node(s) were expanded/collapsed/selected/x-included, and goes back to the “connected” state		
Variations			
Non-functional	The confirmation is necessary since the tree view will be regenerated from scratch and therefore reset (thereby loosing expansions, inclusions, etc): there is no way for the system to know which parts of the simulation resource tree have changed (it may be completely different).		
Issues			
Strategies			

Table 8: UC-1d. Support new DB types

Modification History	May 12th, 2006	Schoenborn	Created
Actors	Primary: User		
Goal	Support new database types without having to modify the application		
Pre-conditions	A plug-in exists for the database type of interest		

Steps	6. User gets the plug-in (e.g. from Administrator via email, website, etc) 7. User puts the plug-in files in the plug-in folder on user's computer 8. User starts portal 9. User defines a database using the new type (Use Case #1b); this can be done at any of these steps 10. User connects to database
Variations	Steps 3 to 5 may not be necessary and can be done before or after steps 1 and 2. #4a: Eventually, the plug-in system could: - find all available DB plug-ins (defined on local host) - enable user to select which of those types is a given DB It could even possibly find all available DB plug-ins defined on a server specified by user, download the selected plug-ins, install them, and add them to its list of available DB types in the Options windows.
Non-functional	
Issues	There is no way for Portal to verify that the type chosen by user is the correct type for the DB pointed to by the URL for the DB
Strategies	

Table 9: UC-1e. Refresh displayed resource tree

Modification History	May 12th, 2006 Schoenborn Created
Actors	Primary: User
Goal	Allow user to refresh the displayed resource tree without having to disconnect and reconnect (esp. useful if passwords would normally be involved)
Pre-conditions	Connected to a DB
Steps	6. The user clicks "Refresh" 7. Portal asks for confirmation to proceed. IF user clicks no, THEN operation is cancelled 8. Portal enters the "refreshing" state: portal attempts a new connection, without closing the current connection; the user can abort the refresh; the user can continue browsing while the refresh is taking place 9. IF the refresh fails, THEN operation is cancelled 10. IF the refresh succeeds, Portal regenerates tree view, loosing which node(s) were expanded/collapsed/selected/x-included, and goes back to the "connected" state

Variations	
Non-functional	The confirmation is necessary since the tree view will be regenerated from scratch and therefore reset (thereby loosing expansions, inclusions, etc): there is no way for the system to know which parts of the simulation resource tree have changed (it may be completely different).
Issues	
Strategies	

Table 10: UC-2. Get Help on Portal Application

Modification History	Nov 28, 2005	Schoenborn	Created
Actors	Primary: user		
Goal	User wants to know more about the application, or wants usage information		
Pre-conditions	GUI started		
Steps	1. IF user goes to menu Help->About to get information about the application THEN 1.1 Show a modal dialog box with information on version number of software and (if available) version number of underlying libraries (wxPython, etc) 1.2 Wait for user to enter OK 2. IF user goes to menu Help->Manual to get information on how to use application THEN The user manual in HTML format is shown via the operating system's default web browser The Portal continues without waiting for user input, independently of what happens in web browser		
Variations			
Non-functional			
Issues			
Strategies	See Figure 6 for possible prototype		

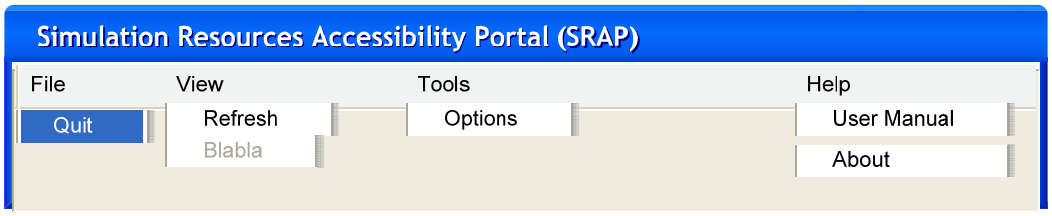

Figure 6: Menu Specification

Table 11: UC-3. Browse simulation resource tree

Modification History	Nov 28, 2005 Schoenborn Created
Actors	Primary: user
Goal	See what simulation resources are available and meta-data about each resource
Pre-conditions	Portal in "connected" state (use case #1a succeeded)
Steps	1. User clicks on the top level node of interest 2. Portal shows the information displayed in the right pane for the selected node 3. IF the node is not a leaf node THEN 3.1 Expand the node to reveal children nodes 3.2 IF the nodes children data has not yet been downloaded, THEN 3.2.1 Show expanded node as containing a "waiting for data" node 3.2.2 Query DB to get data, without loosing responsiveness of GUI 3.2.3 When the data arrives, the nodes are updated (even if they are not in view) 4 User clicks on a lower level node and continues as desired at #2.
Variations	
Non-functional	*User* sees hierarchy of what systems have been created; top level of hierarchy always visible, lower levels only when "opened" by *User*. Note that some of those systems may be in different databases, unbeknownst to the user, if e.g. the database is a meta-database. Though several executables may be available for a given simulation component (e.g. different versions, or several hosts have that component), only one executable "node" is shown; the selection of which executable on which host is allowed/managed by the launcher. Having one component contain several executables does not make

sense; rather, several components can be grouped into a “component group”.

System summary description panel (example info, depends on what has been entered in database):
- Project lead (or Point of Contact)
- When was created (period)
- When last modified
- Description: objective, goals, conclusion
- Classification/Network

Executable component summary description panel:
- Role
- How obtained (created, purchased, open-source), when
- Author or vendor
- Version
- Location of executable (which computer, building, city)
- Classification/Network

Document node summary description panel:
- Number of pages
- Date last updated
- Version
- Intended audience
- Assumed experience/background/expertise
- Summary
- Where stored
- Classification/Network

Group nodes (nodes that have children) can have an “Operational Summary” that describes the essential collaboration between its sub nodes, i.e. what resources (nodes) are required to run the “group”: who (person, software, hardware) does what (role) from where (location, machine), in what sequence. Hyperlinks from parent groups to children groups may be worthwhile.

A document node can be text, image, video, sound, etc. Documents are stored in the database in platform-independent format (e.g. PDF rather than MS Word DOC, etc)

Performance: downloading all meta-data at connection time may take too long; downloading only when user expands a node first time may take too long; perhaps download should take place in the background in parallel based on mouse position (if mouse hovers over a node its children data is acquired)

Issues	
Strategies	

Table 12: UC-4. Run simulation component

Modification History	Nov 28, 2005 Schoenborn Created
Actors	Primary: user Secondary: launcher, and local or remote host
Goal	Allow the user to select a catalogued simulation component for running on a local or remote host
Pre-conditions	1. The Portal is in "connected" mode 2. A launcher is available 3. A launch script (specified in sim tree meta-data) is available to be given to the launcher
Steps	1. User browses to executable node (leaf node within a component node) 2. User right-clicks on node to see operations available on node 3. IF a "Launch via MARCI" is available THEN user selects it 4. IF launch script exists for node THEN Portal creates config file for launcher from that launch script 5. Portal tries to run the launcher, pointing it to created config file (if any) 6. IF no launcher can be run THEN Show error Wait for user OK 7. Resume Portal event handling
Variations	#1a: User browses to a group node; if the group is a component then #4 uses the launch script of the unique executable within component node; otherwise (i.e. group is just a group or a system), the config file is created by aggregating the launch scripts of all nodes beneath. #2a: User can go to File menu item to see same operations available for node (so File menu is updated based on selected node). #2b: Within the Sim Resources pane, above the description pane, is an Operations pane that is updated for every selected node, showing one button per available operation.
Non-functional	Launcher runs in parallel, asynchronously and independently from Portal once launched.

	Failure to run executable is not detected by Portal since better managed by external launcher: - For local run, could fail if minimum system requirements not met, or program fails to start for any other reason (failure to find a dependency such as a shared library); - For remote run, in addition to above cases, could also fail if unable to take control of remote host (insufficient permissions, firewall, user logged in, etc). A launch script must be passive: it is actually a config file that defines parameters so the launcher knows how to launch the executable. This will guarantee that several launch scripts can be combined in any order.
Issues	
Strategies	

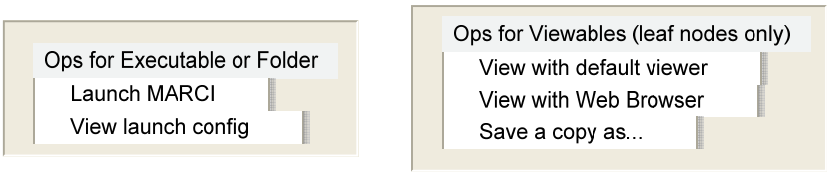

Figure 7: Possible drop-down menus for nodes

Table 13: UC5a. View a simulation document

Modification History	Nov 28, 2005 Schoenborn Created
Actors	Primary: user Secondary: database, external (but local) viewer
Goal	Allow the user to select a catalogued simulation document for viewing on user's machine
Pre-conditions	1. The Portal is in "connected" mode 2. A viewer for the type of document pointed to is available
Steps	1. User browses to the desired document node within any group node 2. User right-clicks on node to see operations available on node 3. IF a "View" is available THEN user selects it 4. Portal sends query to DB proxy to download the document 5. IF document NOT received (link broken) THEN

	5.1 Show error 5.2 Wait for user OK 6. Portal attempts to “start” the document, which will open the default viewer for the document type 7. IF Portal fails to find a default viewer THEN 7.1 Show error 7.2 Wait for user OK 8. Resume Portal event handling
Variations	#2a and #2b: same as for Use Case #4
Non-functional	Failure modes: - no viewer can handle that document type - document not in database (link broken) - user has insufficient privileges to access
Issues	Large downloads will temporarily disable user input.
Strategies	

Table 14: UC-5b. Save a simulation document locally

Modification History	Nov 28, 2005 Schoenborn Created
Actors	Primary: user Secondary: database, external (but local) viewer
Goal	Allow the user to select a catalogued simulation document for saving on user’s machine
Pre-conditions	The Portal is in “connected” mode
Steps	1. User browses to the desired document node within any group node 2. User right-clicks on node to see operations available on node 3. IF a “Save as” is available THEN user selects it 4. Portal sends query to DB proxy to download the document 5. IF username/password required (file not in database but in a secured HTTP area, e.g.), THEN IF username/password not cached THEN ask user ELSE get from cache Repeat query with username/password 6. IF document NOT received (link broken) THEN Show error Wait for user OK 7. Portal opens file browser, asking user where to save file 8. IF file selected already exists THEN ask for confirmation 9. IF user cancels THEN abort the operation and resume normal GUI 10. Copy file to selected location

	11. IF copy fails (inadequate permissions, etc) THEN Show error Wait for user OK Repeat from #7 12. Resume Portal event handling
Variations	#5.1a: When asking user for username/password, user can cancel, in which case operation is aborted
Non-functional	Failure modes: - no viewer can handle that document type - document not in database (link broken) - user has insufficient privileges to access
Issues	Large downloads will temporarily disable user input.
Strategies	

Table 15: UC-6. Find meta-data by keyword: Basic

Modification History	Nov 28, 2005 Schoenborn Created
Actors	Primary: user
Goal	Allow user to quickly find all simulation meta-data pertaining to a given keyword
Pre-conditions	GUI in “connected” state
Steps	1. User enters text in the filter box 2. User clicks the right arrow to indicate “apply” 3. The Portal searches the simulation resource tree node data for the given keywords 4. The Portal grays out the nodes that don’t match, highlights in red those that do, and highlights in bold the matching nodes’ parent nodes 5. Portal saves the search to the “filter history” 6. IF user clicks the “Clear” button THEN Portal clears the filter box (except for the default “empty” filter) Portal shows unfiltered tree 7. Portal resumes event handling
Variations	#1a: User clicks the down arrow and selects one of the previous filters #4a: The bold for matching nodes’ parent nodes is required only for those parents that are collapsed; any collapsed parent that has children with matches could be bolded, and unbolded if it is expanded, and re-bolded if collapsed again.
Non-functional	There is always at least one filter available in the filter history drop

	down combo box: the “empty” filter (no filtering)
Issues	
Strategies	

Table 16: UC-7.Find meta-data by keyword: Advanced

Modification History	Nov 28, 2005 Schoenborn Created
Actors	Primary: user Secondary: database
Goal	Allow user to quickly find all simulation DB information pertaining to a given keyword
Pre-conditions	
Steps	1. User clicks on the “Advanced” button in the filter pane 2. Portal open the Advanced Filtering dialog 3. User changes the filter settings 4. IF user clicks the “Apply” button THEN IN PARALLEL Portal applies current filter settings: The Portal searches the simulation resource tree node data for the given keywords The Portal hides the nodes that don’t match IF the user selects one of the filtered nodes THEN the associated description panel is shown with the keywords highlighted (assuming user checked the appropriate box in the Advanced Filtering dialog, or entered proper filtering command) Portal allows user to change filter settings (go back to #3) ELSE IF user clicks “Cancel” button THEN Portal clears the filter box and cancels filtering of tree Portal closes dialog (go to #8) ELSE (user clicked “OK”) Portal applies current filter settings to the tree Portal puts equivalent “filter rule” in filter text field to match the advanced settings (like Google) Portal closes dialog (go to #8) 5. Portal resumes event handling
Variations	#3a: Portal checks if database supports filtering 1. Ask DB proxy, answers true or false 2. IF supports filtering THEN enable the option to set/unset the checkbox for “Search in database” #4.1.1a: IF user selected “Search in database” THEN 1. In addition to 4.1.1, send query to DB proxy

	2. Proxy translates query to command understood for its database type 3. DB returns a list of matching links 4. DB Proxy translates links to corresponding references to tree nodes
Non-functional	The specific filter abilities of the Advanced Filter are not specified at this time, as it requires further analysis: what kinds of advanced filtering is required by user, how does this play with the open DB architecture for those advanced filtering capabilities that require DB access (some DB may not have certain search capabilities)
Issues	Translating a filter query to a DB search query may not be possible. E.g., a subset of the filter settings may not be supported by DB.
Strategies	

Table 17: UC-8. Add simulation resource info to DB

Modification History	Nov 29, 2005	Schoenborn	Created
Actors	Primary: modifier Secondary: database		
Goal	Have new simulation resource info available to Portal users		
Pre-conditions	Admin has created and configured the database Modifier has appropriate access rights for the parts of the DB that will be affected		
Steps	1. Modifier creates new entries into DB, according to DB protocol and Figure 13 of Annex A. 2. Modifier adds associated documents to DB according to DB protocol 3. Modifier runs Portal and connects to DB and verifies that new system appears 4. Modifier test all links within new System branch in Portal to make sure there are no dead links		
Variations			
Non-functional	**Portal is NOT used for this**. See Appendix for specification of XML file format. The SEDRA portal requires an XML file to describe the database contents (simulation resources), but the DB does not. The only requirement on the database is that it supports a hierarchical breakdown of the set of simulation resources (systems) according to Figure 13 of		

	Annex A. This still allows for a large variety of databases, from relational to object-oriented to file system based. It is up to the DB Proxy to create the XML file from the database structure (if necessary), and to convert the tree-based queries of the Portal into proper database queries (if necessary).
Issues	
Strategies	

Table 18: UC-9. Modify simulation resource info in DB

Modification History	Dec 2, 2005	Schoenborn	Created
Actors	Primary: modifier Secondary: database		
Goal	Modify a system described in DB for Portal users		
Pre-conditions	1. Admin has created and configured the database 2. A modifier (not necessarily same as primary actor) has previously added the system information to DB 3. The (primary actor) modifier has required privileges to make changes to the DB in parts of the DB that will be affected		
Steps	1. Modifier makes necessary changes to DB according to DB protocol 2. Modifier adds/removes any documents according to modifications in system file 3. Modifier runs Portal and connects to DB and verifies that new system appears 4. Modifier test all links within new System branch in Portal to make sure there are no dead links		
Variations			
Non-functional	**Portal is NOT used for this**. See Annex A for specification of XML file format.		
Issues			
Strategies			

Table 19: UC-10. Toggle the visibility of log window

Modification History	Jun 12th, 2006	Schoenborn	Creation

Actors	Primary: User
Goal	The log window can be made visible or hidden
Pre-conditions	The application has started
Steps	1. IF the log window is visible THEN 1.1 The user can close it the usual way (clicking on the X icon in the upper right corner) or via a widget on that window (menu item, button, etc) 1.2 The log window disappears 2. IF the log window is invisible THEN The user can select the appropriate View->Log Window menu item The window appears
Variations	#2.1a: A keyboard shortcut is available to make the window visible if it is not already visible
Non-functional	The log window shows a time-stamped history of status messages that give a trace of user operations (Figure 8). All errors, including trace backs, should be copied to this window. The window should remain open, with a proper error message, even in the case where a fatal problem occurs at program start-up. This may not be possible for certain classes of problems, such as e.g. the Python interpreter not being found.
Issues	
Strategies	Use the native logging tools of the GUI library as much as possible

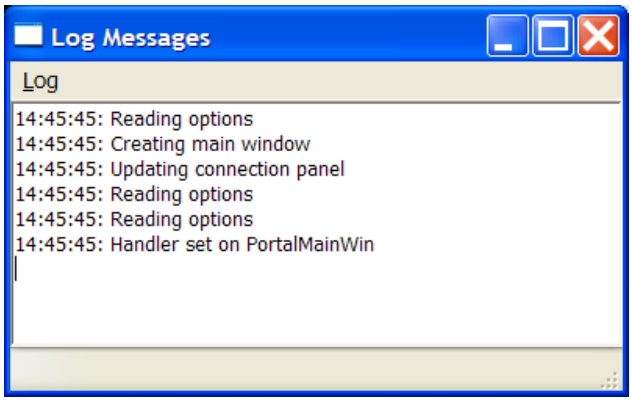

Figure 8: Example log window

7 SEDRA Portal Test Cases

This section documents the test cases for each use case deemed of sufficient complexity to warrant an explicit test case. The test case descriptions aim at specifying as accurately as possible the specific preconditions, inputs, steps, and post-conditions that lead to the successful completion of the use case or to one of its known variations. Only variations that warrant a test case will have one.

In general, there will be several tests for each use case, forming a test suite (**TS**). Test suites are numbered after the use case that they are associated with: TS-*ucNum* for use case *ucNum*. In each test suite, the test cases (**TC**) are numbered according to the step at which they start: TC1 would start applying at step 1 of the use case, TC1.1 at step 1.1, etc. Test cases for variations are therefore naturally covered by this scheme. Note that if more than one test case starts at same step of use case, an extra "-#" counter is added, as in TC4-1 and TC4-2 for test cases # 1 and 2 of step 4 (of the test suite to which this test case belongs).

A test case refers to another test case within the same suite as TCtcNum, within another test suite as TS-ucNum:TCtcNum, and to a whole test suite as TS-tcNum. TCtcNum refers to all tests if there are more than one (i.e. TC1-1, TC1-2, etc). An example table follows:

TS-10. Same name as UC-10				
TC	Modification History	Jun 12th, 2006	Schoenborn	Creation
1-1	Briefly describe test #1 that starts at step 1 of UC-10			
	Pre-conditions	• TS-9 (the whole test suite validated), • TS-8:TC1 (all test case(s) for step 1 of TS-8), • TS-8:TC2-2 (only second test case for step 2 of TS-8)		
	Steps			
	Post-conditions			
1-2	Briefly describe test #2 that starts at step 1 of UC-10			
	Pre-conditions			
	Steps			
	Post-conditions			
2.1a	Briefly describe the only test needed for variation (a) of step 2.1 of UC-10			
	Pre-conditions			
	…			

TS-1a. Set which DB(s) accessible from Portal				
TC	Modification History	Jun 12th, 2006	Schoenborn	Creation
1-1	Test basic add/remove/edit capabilities, fishing with cancel			
	Pre-conditions	• GUI is open but not in connected state • Application never started yet		
	Steps	1. Open Tools->Options 2. See default list of DB/usernames, no selection, all items expanded, only “Add DB” and “Clear All” buttons available (together with “Accept” and “Cancel”, of course) 3. Verify that selecting a DB or username activates all buttons except the two move buttons 4. Verify that removing users and databases removes them from the dialog view 5. Verify that adding users and databases adds them to the dialog view 6. Verify that adding a user adds the user to the currently selected database, or to the database of the currently selected user 7. Verify that selection moves to new database when a database is added, but remains unchanged when adding a user 8. Verify that SPACEBAR activates editing of the selected item 9. Verify that double-clicking an item edits it 10. Verify that “Clear All” removes all entries 11. Cancel the Options dialog		
	Post-conditions	Nothing has changed: no new databases entered, the connections panel is still disabled		
1-2	Accept default database settings			
	Pre-conditions	TC1-1 just completed		
	Steps	1. Open the Tools->Options dialog, see the default database information 2. Remove user1 of drdc-o-2, add ‘user3’ to drdc-o-1 3. Accept to commit the new settings 4. Verify that three databases are now available for connection 5. Exit application and restart Portal 6. Verify that same three database (in same order as before exit) are available		
	Post-conditions	Three default databases are now “accessible” from the connection panel The rest of the main view is still disabled		

TS-1b. Connect to a Database

TC	Modification History	Jun 12th, 2006	Schoenborn	Creation
1-1	Test bad connections to URL			
	Pre-conditions	TS-1a:TC1		
	Steps	1. Do a “connect” operation on drdc-o-0 2. Verify that descriptive error message pops up 3. OK the message 4. Do a “connect” on drdc-o-1 with user=user3 5. Verify that descriptive error message pops up 6. OK the message 7. Do a “connect” on drdc-o-2 with user=”badXML” 8. Verify that descriptive error message pops up 9. OK the message		
	Post-conditions	Nothing has changed: “Connect” button still active, etc.		
1-2	Test connection to an unsecured database			
	Pre-conditions	• TS-1a:TC1 • Added a DB: alias=testHttpDBNoSec, URL=http://hostname/portalTestDBNoSecurity/, type=HTTP, no usernames • Web server configured for unrestricted access to above URL, and proper files contained therein		
	Steps	1. Select DB drdc-o-1 or 2 2. Do a “connect” operation 3. Verify that all panels below connection get activated, and tree view generated, all collapsed 4. Disconnect 5. Verify that all panels below connection are once more inactive 6. Repeat at 2 after selecting “testHttpDBNoSec” 7. Add a username ‘oliver’ under “testHttpDBNoSec” 8. Select new username then repeat at 6		
	Post-conditions	Same as pre, but a greyed-out tree view of the last connection is still visible		
1-3	Test connection to an secured database			
	Pre-conditions	• TS-1a:TC1 • Added a DB: alias=testHttpDB,		

	URL=http://hostname/portalTestDB/, type=HTTP, usernames are “valid” and “invalid” • Web server configured for restricted access to above URL, and proper files contained therein
Steps	1. Select DB “testHttpDB” and username “invalid” 2. Try a “connect” operation 3. Enter password when requested 4. Verify that error says username not valid 5. OK message, verify that username/password dialog appears again 6. Cancel username/password dialog 7. Select username “valid” for same DB 8. Try a “connect” operation 9. Enter *wrong* password when requested 10. Verify that error says username/password not valid 11. OK message, verify that username/password dialog appears again 12. Enter *correct* password when requested 13. Verify that tree view appears 14. Disconnect 15. Connect again 16. Verify that username/password were requested a second time
Post-conditions	Same as pre, but a greyed-out tree view of the last connection is still visible

TS-1c. Edit the DB aliases and associated usernames

TC	Modification History	Jun 12th, 2006	Schoenborn	Creation
1	Test basic add/remove/edit capabilities, fishing with cancel			
	Pre-conditions	• GUI is open but not in connected state • Application never started yet		
	Steps	1. Open Tools->Options 2. See default list of DB/usernames, no selection, all items expanded, only “Add DB” and “Clear All” buttons available (together with “OK” and “CANCEL”, of course) 3. Verify that starting an edit of a database or user, then cancelling the edit box, doesn’t change the information 4. Verify that attempting to change a username to a username that is already defined for the username’s database gets rejected 5. Verify that attempting to give a database an alias that is already		

		in use by another database fails, and the user must correct or cancel 6. Verify that attempting to change a username or any database info field to "" (nothing) fails, and the user must correct or cancel 7. Verify that Clear All removes all entries 8. Cancel the Options dialog
	Post-conditions	Nothing has changed: no new databases entered, the connections panel is still disabled
2a	Prevent user from editing database settings while "connected"	
	Pre-conditions	TC1 and TS-1b:TC1-2
	Steps	1. Select one of the two drdc-o databases from connection pane 2. Connect 3. Open the Tools->Options dialog 4. Verify that none of the database-info editing buttons are enabled 5. Cancel or Accept should return to the main view
	Post-conditions	Connected to one of the two databases Can see simulation resources tree in bottom left of window

TS-3. Browse simulation resource tree

TC	Modification History	Jun 12th, 2006	Schoenborn	Creation
1-1	See the description of a node			
	Pre-conditions	• URL for DB drdc-o-1 changed to "DB/" • Just connected to drdc-o-1 as user user1		
	Steps	1. Verify that icons for "groups" are folders 2. Verify that icon for "Overview of FFSE" is a document 3. Select the "Overview of FFSE" node 4. Verify what appears in description pane: a "Description" field and some text appears beneath it 5. Expand the "JSMARTS" node and all nodes below it 6. Verify that the "RepoEditor" node has an icon that indicates an executable 7. Expand the "Has Inclusions" node 8. Verify that it *momentarily* shows *one* child labelled as "(Need download …" 9. Verify that it (almost immediately after above step) shows three children		

		10. Select its “Tech Memo 2” child 11. Verify that description pane show number of pages 12. Expand the “more components” node 13. Verify that its children are the same as the parent “Has inclusions” node 14. Repeat 11 and 12 several times 15. Expand the “Has Bad Inclusions” node 16. Verify that error message shows can’t be resolved 17. Collapse the “Has Bad Inclusions” node 18. Repeat steps 15-16 19. Select the bad node 20. Verify that description pane explains it is being downloaded (which is false, it has already been downloaded, but this is the current behaviour when error occurs)
	Post-conditions	Portal shows the drdc-o-1 DB with at least one “more components” nodes and a bad node
1-2	Verify x-inclusions work for HTTP type of databases	
	Pre-conditions	• TS-1b:TC1-2 • Just connected to DB testHttpDBNoSec
	Steps	1. Expand “SE ALIX” and select the “SE ALIX/Media” node 2. Verify that nothing appears in description pane 3. Expand the selected node 4. Verify that selection hasn’t changed, and a description (taken from x-inclusion) appears 5. Expand all remaining nodes (note: expanding is almost always instantaneous but has taken up to 2 minutes, reason unknown but server claims response times always under 0.2 secs) 6. Verify that JSMARTS and MarSIE each have one document child 7. Verify that “SE ALIX” has two document children and one subgroup 8. Verify that “SE ALIX/Media” has two document children 9. Disconnect and connect to the secure DB, testHTTPDB, using username = ‘oliver’ and password = ‘test’ 10. Expand all nodes 11. Verify that no username/password was requested from user (since already available from connection to DB)
	Post-conditions	Portal shows the contents of testHttpDBNoSec, all expanded
1-3	Verify x-inclusions show proper info when slow	
	Pre-conditions	• TC1-2 • Uncommented the line in Apache server configuration file, starting with “EnableSendfile off” (or, have some other way of making the acquisition of data from database slow)

		• Just connected to DB testHttpDBNoSec
	Steps	1. Expand all nodes of tree 2. Verify that some nodes have a question mark and a label saying they are being downloaded 3. Click on one of those nodes 4. Verify that the description pane explains that they are being downloaded and that the tree will be updated as soon as the data arrives.
	Post-conditions	Portal shows the contents of testHttpDBNoSec, all expanded

TS-4. Run simulation component				
TC	Modification History	Jun 12th, 2006	Schoenborn	Creation
1-1	n/a			
	Pre-conditions			
	Steps			
	Post-conditions			

TS-5a. View a simulation document				
TC	Modification History	Jun 12th, 2006	Schoenborn	Creation
1-1	View a document mentioned in local database of type folder; some documents are in DB, others are on web			
	Pre-conditions	• TS-3 • Connected to drdc-o-1 as user1 • An application that can view MS Word documents is on local host		
	Steps	1. Select the “Overview…” node of tree 2. Right-click and select “View” 3. Verify that browser opens and shows a directory listing		

		4. Select the “JSMARTS/Tech Memo” node of tree 5. Right-click and select “View” 6. Verify that an “MS Word” viewer (usually MS Word itself on MS Windows) shows the document 7. Select the “Has inclusions/Tech Memo 2” node of tree 8. Right-click and select “View” 9. Verify that an error message appears saying file not found, and ok it
	Post-conditions	Same as pre
1-2	View a document mentioned in unsecured database of type HTTP, some documents are in DB, others are on web in unsecured and secured sites	
	Pre-conditions	• TC1-1 • Connected to testHttpDBNoSec • A viewer is available for each of the document nodes (doc, mp2, pdf, ppt, txt, jpg and msg) • No web browsers are open (so that password caching is not interfering) • The localhost has a web server running, and is configured to give access to the DB and the URL’s specified in the DB (those URL’s may have to be modified to incorporate port number, depending on web server settings); the .htaccess files may need editing to accommodate the web server settings.
	Steps	1. Expand all nodes of the tree view 2. View each of the 7 document nodes available (each node is a different type of document) 3. Verify that each document can be viewed (note: depending on system settings, some will be viewed via web browser) 4. Verify that browser requests username/password for the document in JSMARTS (this is not the same as DB username/password since the document is not in DB)
	Post-conditions	Same as pre

TS-5b. Save a simulation document locally

TC	Modification History	Jun 12th, 2006	Schoenborn	Creation
1-1	Test that documents can be saved, for documents mentioned in DB of type folder			
	Pre-conditions	• TS-5a:TC1-1 • Connected to drdc-o-1 as user1		

	Steps	1. Right-click “Overview…” node and do “Save as” 2. Enter a name for file 3. Select “Desktop” as destination folder 4. Click on Save 5. Verify that file is on your desktop and has proper type (html) 6. Repeat at #1 to 4 for “JSMARTS/Tech Memo”, but don’t change the proposed file name 7. Verify that file is on your desktop, has same name as in DB and has proper type (doc) 8. Verify that file is still in DB 9. Repeat at #1 to 4 for “Has inclusions/Tech Memo 2” 10. Verify that error message says file is not found 11. Change the properties of the JSMARTS_II_Stations.doc file you saved on your Desktop to be read-only 12. Repeat at #1 to 4 for “JSMARTS/Tech Memo” again 13. Verify that message asks for confirmation before overwriting 14. Verify that error message then says that permission is denied to write 15. Verify that file browser pops up again to allow user to pick a different file name 16. Click cancel 17. Verify that normal GUI resumes
	Finalization	Remove the files created on your desktop
	Post-conditions	Same as pre
1-2	Test that documents can be saved, for documents mentioned in DB of type HTTP	
	Pre-conditions	• TS-5a:TC1-2 • TC1-1 • Connected to testHttpDBNoSec • Setup of local host web server is same as for TS-5a:TC1-2
	Steps	1. Do a “Save as” on “Cal Poly Paper” 2. When file browser opens, select Desktop as destination folder and accept default name 3. Verify that file on your desktop, has same name as in DB and has proper type (pdf) 4. Repeat #1 to 2 for “JSMARTS/Tech Memo…” 5. Verify that username/password dialog appears 6. Enter “oliver” and “test2” in appropriate text fields 7. Verify error message about authorization required due to wrong username/password 8. Repeat #1 to 2 for “JSMARTS/Tech Memo…” 9. Verify that username/password dialog appears 10. Enter “oliver” and “test” in appropriate text fields 11. When file browser appears, select Desktop as destination folder and accept default name proposed

		12. Verify that file on your desktop, has same name as in DB and has proper type (doc) 13. Do a “Save as” on “SE ALIX/GD ALTAIR” 14. When file browser opens, select Desktop as destination folder 15. Verify that type “msg” is not in list of available extensions in file browser dialog 16. Don’t change default type, accept proposed filename 17. Verify that file is saved to desktop with the “msg” suffix, not with the default extension that was selected in file browser
	Finalization	Remove the files created on your desktop
	Post-conditions	Same as pre

TS-6. Find meta-data by keyword – Basic

TC	Modification History	Jun 12th, 2006	Schoenborn	Creation
1-1	Find all nodes that have a keyword			
	Pre-conditions	• TS-3:TC1-2 • Connected to DB testHttpDBNoSec • All nodes collapsed (even root)		
	Steps	1. Type in “Unsec” in filter field and press ENTER 2. Verify that the only visible node is bold and red 3. Expand the node to see its direct children 4. Verify that all children are grayed out 5. Type in “Cal” in filter field and press ENTER 6. Verify that only the “Cal Poly” node is bold and red, all other nodes are grayed out 7. Collapse the root node 8. Verify that it changes to bold black 9. Expand root node 10. Verify that it changes to grayed out 11. Type in “snap” in filter field and press ENTER 12. Verify that all visible nodes are grayed out 13. Expand “SE ALIX” 14. Verify that all visible nodes are grayed out 15. Expand “Media” 16. Verify that “SE ALIX/Media/UTCS1…” node is bold red, all others are still grayed out 17. Collapse “SE ALIX/Media”		

		18. Verify that the collapsed node is now bold black 19. Expand “Media” 20. Verify **#16** 21. Collapse “SE ALIX” 22. Verify that the collapsed node is now bold black 23. Expand “SE ALIX” 24. Verify **#16**
	Finalization	Collapse all nodes of tree
	Post-conditions	Same as pre
1-2	Test filter history	
	Pre-conditions	• TC1-1 • Connected to DB testHttpDBNoSec
	Steps	1. Click on the down-arrow of filter to see saved filters 2. Verify that drop-down is completely blank 3. Type in “f1” and press ENTER 4. Click on the down-arrow of filter to see saved filters 5. Verify that first item is “f1”, next is blank line 6. Type in “f2” and press ENTER 7. Click on the down-arrow of filter to see saved filters 8. Verify order of items: “f2” (top), “f1”, blank line (bottom) 9. Click on the down-arrow of filter to see saved filters 10. Select “f1” and press ENTER 11. Click on the down-arrow of filter to see saved filters 12. Verify order of items: “f1” (top), “f2”, blank line (bottom)
	Finalization	Disconnect from DB
	Post-conditions	Tree all greyed out
1-3	Test filter history “clear”	
	Pre-conditions	• TC1-2 • Connected to DB testHttpDBNoSec • No filters saved
	Steps	1. Click “Clear Hist” 2. Verify that filter history still blank 3. Enter several filters: “f1”, “f2”, “f3” 4. Verify that tree nodes all grayed out 5. Click “Clear Hist” 6. Verify that filter history still blank and tree no longer grayed out
	Post-conditions	Same as pre
1-4	Test that active filter is automatically applied when connecting to a DB	
	Pre-conditions	• TC1-1

	• URL for DB drdc-o-1 changed to “DB/” • Connected to DB drdc-o-1 as user1
Steps	1. Type in “p” in filter box and press ENTER 2. Verify that some nodes are gray, some are bold red and some are bold black 3. Refresh the DB 4. Verify that nothing has changed from #2 5. Disconnect and connect to testHttpDBNoSec 6. Verify that all visible nodes are bold red
Post-conditions	Same as pre

References

[1] Nacer Abdellaoui, Paul Hubbard and Oliver Schoenborn. (2004) Evolution of the Synthetic Environment in the FFSE Section Survey of Current Usage of Synthetic Environment within FFSE and Recommendations for the Future. (DRDC Ottawa TM 2005-141).

[2] RFC 1738 (rfc 1738) - Uniform Resource Locators (URL) (online), http://www.faqs.org/rfcs/rfc1738.html (Access date: 12 Nov. 2008)

Annex A Simulation Resources Description Structure

This section specifies what Simulation Resources can be described by the SEDRA portal. It contains a mixture of data type diagrams as well as textual descriptions of the diagram elements. Note that UML notation is used to indicate inheritance and aggregation. As such a line ending in a white triangle indicates that a data type inherits the attributes of the data type "above" it (touching the triangle), whereas a line starting in a dark diamond indicates that a data type element can refer to a specified number of entities of another data type.

The set of all simulation resources describable in SEDRA is represented as a hierarchy from most general to most detailed. There are 4 basic types of simulation resources:

- Simulation systems: Groups of simulation components that work together more or less directly to form a "system";
- System components: typically individual computer programs (executables);
- Documents: text, audio, video and other types of information stored somewhere as a as files somewhere, and related to simulation systems or system components;
- Recipes: an "executable document" that can be processed by SEDRA to document how to start or stop etc a system or component; or that can be given to the SEDReC station for execution of the "recipe".

In addition, systems and components can be grouped into system groups and component groups respectively, to facilitate organization of knowledge.

The root element of the hierarchy is a unique system group named **SimResources** which groups simulation systems and "smaller" system groups. Simulation systems are represented by **SimSystem** elements and system groups by **SystemGroup** elements. SystemGroups can contain 0 or more SystemGroup elements and 0 or more SimSystem elements (Figure 8). Because SimResources "is-a" SystemGroup, the SimResources element can contain the same. Conversely, any SimSystem, and any SystemGroup, must either be in a SystemGroup or the unique SimResources element.

Note that the notion of *what constitutes a system* is to some extent subjective. However, the broad definition would be "a group of simulation components that collaborate towards a common goal". The sequence of SimSystem elements is not ordered, but is likely to be ordered by the GUI (e.g. alphabetically). Note also that SimResources has the same attributes as SystemGroup, but is special in that it is "the root system group" at the base of the knowledge tree (it doesn't have a parent).

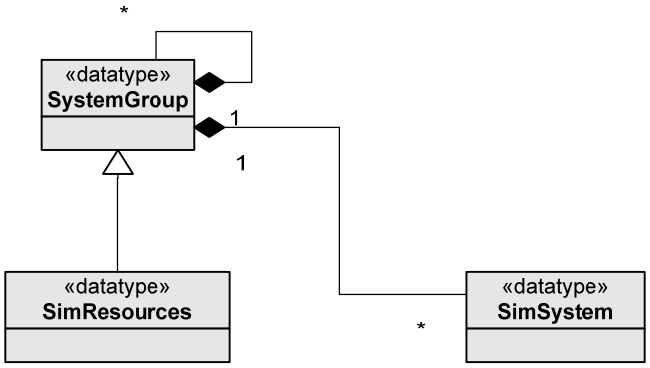

Figure 9: SimResources and SimSystem elements

In turn, simulation systems are broken down into system components and groups of system components (Figure 9). Since a simulation system "is-a" grouping of components, the figure also shows that SimSystem inherits from ComponentGroup. A ComponentGroup can contain other component groups or system components; hence SimSystem can hence contain the same. Conversely, any ComponentGroup is either in another ComponentGroup or in a SimSystem, and the same goes for any SysComponent: it is either in a ComponentGroup or in a SimSystem. Note that each SysComponent element is meant to represent one executable simulation resource. E.g., it could be a federate, or a tool such as a visualization panel, etc.

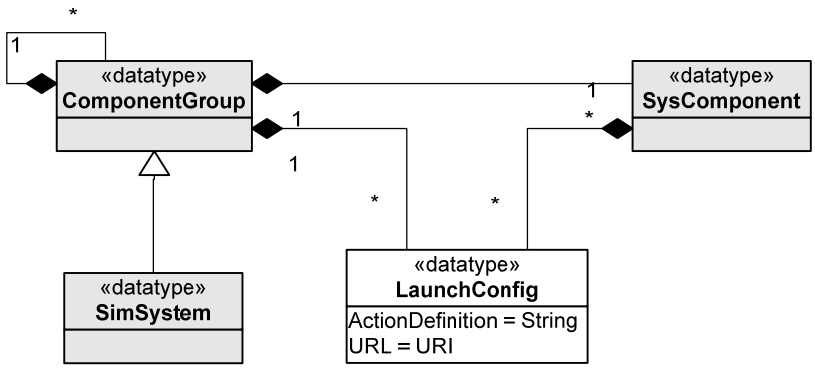

Figure 10: SimSystem, ComponentGroup, and LaunchConfig elements

An executable document that can be used to document how a simulation system, system component, or component group can be launched is represented by the **LaunchConfig** element (Figure 9). Such element can refer to the executable document (the "Control Recipe") via a URL

or may directly contain the recipe as an XML text string. This can be given to the SEDReC station for execution.

All elements that are group types share the ability to "contain" document resources (**Document**) and groups of documents (**DocumentGroup**) (Figure 10). Any element deriving from **Group** has an optional **Dir** attribute whose interpretation is database-dependent (e.g. in an web-server database implementing the schema describe in this appendix could use the dir to indicate a folder in which all child items can be found).

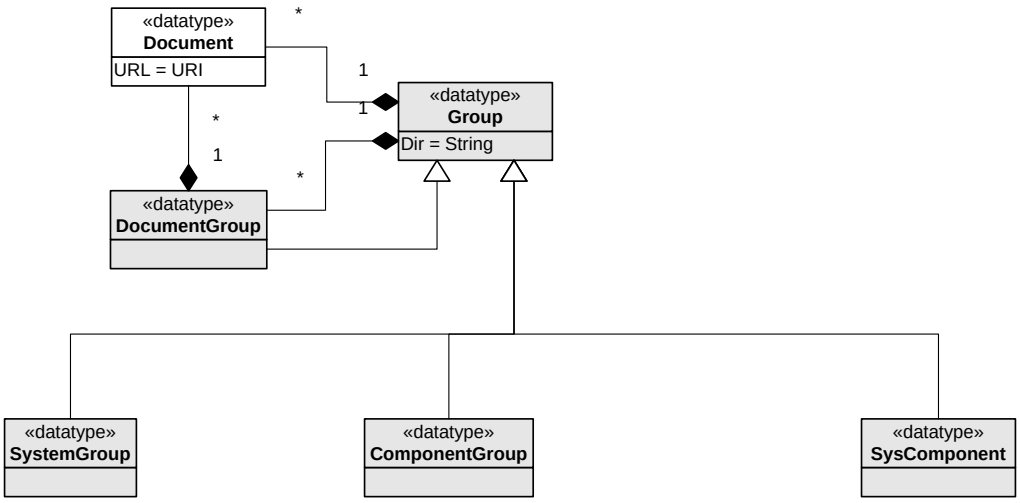

Figure 11: Document and Group elements

The **Document** elements represent any kind of file that is not an executable, such as text files, audio files, movie files, CAD drawings, etc. A document is not normally stored in the simulation resources database, though some databases may support that. Instead it is normally stored in a "decentralized repository" of documents available via HTTP requests to one or more web servers.

All simulation resources are given a name: documents launch configs (control recipes) and any group-type element (Figure 11). The name is an "alias" for the simulation resource, and will be used by SEDRA in its GUI.

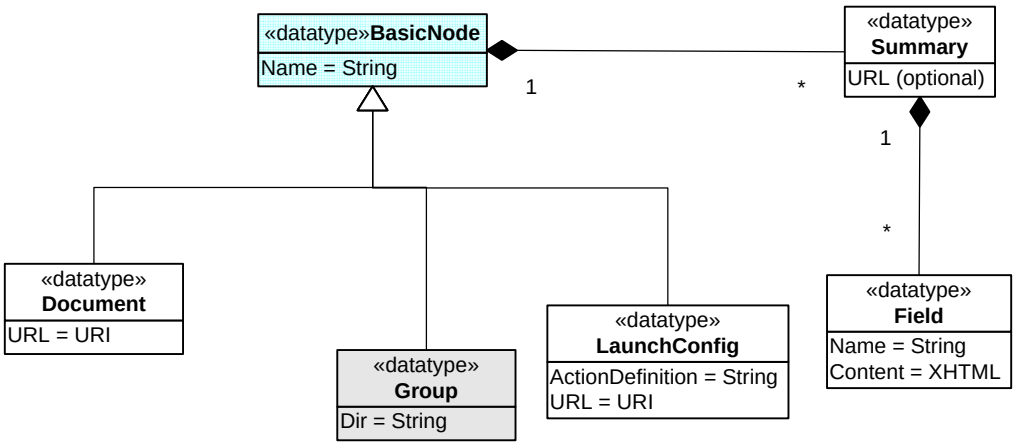

Figure 12: BasicNode, Summary and Field elements

All simulation resources all have a “Summary”, which is meta-data about the document (data) or group (Figure 11). The Summary element can point to an XML document consisting of fields (name-content pairs), or can directly contain the name-content field pairs. This sequence of fields will be formatted and displayed by SEDRA as the “description” of the simulation resource being browsed. Each Field element contains valid XHTML.

The complete diagram, combining the above sub-diagrams and adding any missing relationships described in the text but not shown, appear in Figure 12.

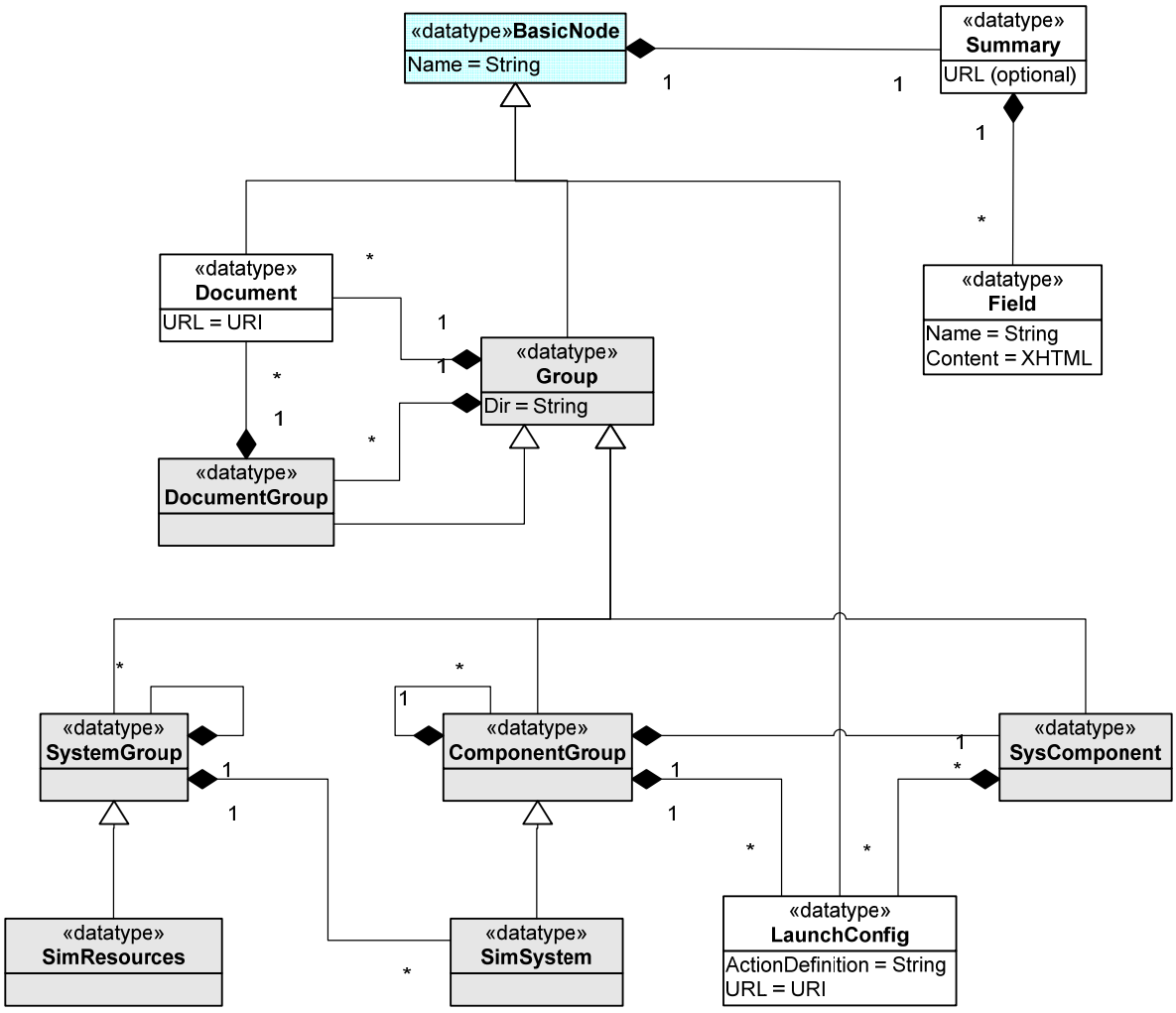

Figure 13: Complete data type relationship diagram of all elements

The following "syntax colored" text shows an example XML representation that adheres to Figure 12. I.e., even though different databases may implement Figure 12 differently (in terms of tables, files and folders, nested objects, etc), all would result, via the DB Proxy plug-in system used by SEDRA, as an XML representation as shown below.

```
<launchProcedure>
    This launches an RTI exec and a JSAF for White forces. It also
    launches a MAK stealth viewer that speaks DIS, and a DIS
    gateway federate so the viewer can view the HLA federation.
    Only the RTI exec requires a license server to be available.

    <actionSeq name="MAK RTI">
        First need to start the exec. Only after it has started
        can federates connect to it.
```

```
        <run name="RTI Exec" path="rtiexec" onHost="131.136.161.98"
            wdir="/home/malo_dev/makRti2.4.2/bin">
            If license manager not available, an error window
            will eventually appear and this action will fail.

            <envVar name="RTI_CONFIG" value="/home/malo_dev/makRti2.4.2">
               Root of RTI exec distrib</envVar>
            <envVar name="MAKLMGRD_LICENSE_FILE" value="@JSAF-FLEXOR">
               Flex LM file location</envVar>
            <envVar name="LD_LIBRARY_PATH"
               value="/usr/local/lib:/home/malo_dev/makRti2.4.2/lib">
               DSO's for RTI exec</envVar>
        </run>

        Now the gateway can be run:

        <run name="DIS Gateway Fed" path="gateway" onHost="131.136.161.98"
            wdir="/home/malo_dev/JSAF-latest/src/Gateway">
            Allows any simulation component that speaks DIS to
            appear as an HLA federate.

            <param name="-terrain"     value="Malo_Cabot">OpenFlight</param>
            <param name="-unicast"     value="224.0.1.1"/>
            <param name="-convert_agl" value=""/>
        </run>

        To operate the simulation, use the force, Luke. Also, pretend
        a file must be uploaded, to host that won't be found:

        <upload name="Setup files" destPath="path2" toHost="131.136.161.42">
            These are the database files from Object Raku:
            <files pattern="c:/test/*.dtf">The terrains</files>
            <files pattern="c:/test/*.xml">The configurations</files>
        </upload>

    </actionSeq>
</launchProcedure>
```

Annex B State diagram of SEDRA application

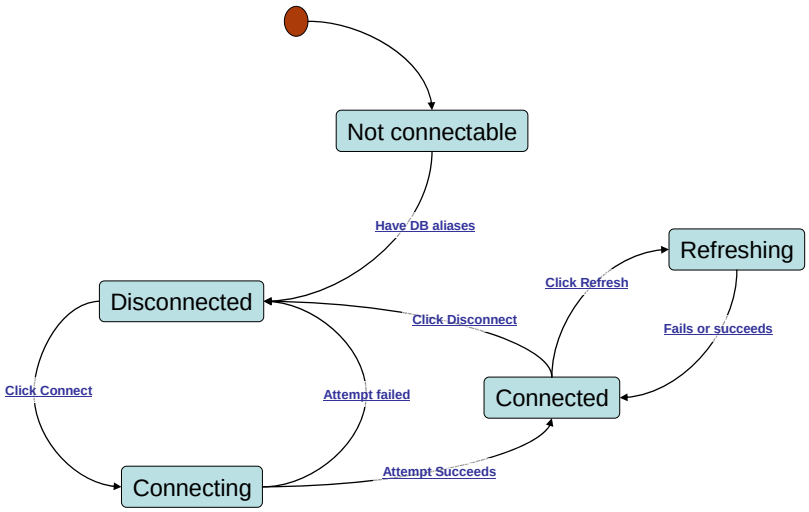

Figure 14: State chart of Portal

Note that the transition out of “Refreshing” is always to “Connected” regardless of failure or success: if failure, the connection to database prior to refresh is maintained and assumed to be temporarily unavailable (no new meta-data); if success, the connection is also maintained but new meta-data is received.

Annex C Simulation Resources Documentation Specification

An assessment conducted in between April and September 2005 established clearly that available documentation for the simulation resources is inadequate, in that it is either absent, insufficient, inaccessible, or obsolete. The Portal should help make it accessible and the database should facilitate maintaining the documentation, however the lack of documentation must be addressed by a separate, and parallel, effort.

The following outline list describes what should be documented, for each simulation system described in the database. All of this would end up in the database, either as data or meta-data. This appendix should be tested on a real project, JSMARTS2, and eventually made into a stand-alone document and template that people can use.

4. Summary description of system:
    a. Project lead or Point of Contact
    b. When was created (period)
    c. When last modified
    d. Summary: objective, goals, conclusion of exercise
    e. Classification/Network
5. List of simulation components; for each one:
    f. Role
    g. How obtained (created, purchased, open-source), when
    h. Author or vendor
    i. Version
    j. Classification/Network
    k. List of documents available (a document can be text, image, video, sound, etc); for each one:
        i. Number of pages
        ii. Date last updated
        iii. Version

iv. Intended audience

v. Assumed experience/background/expertise

vi. Summary

vii. Where stored, custodian

viii. Classification/Network

l. List of executables available (several copies may exist – e.g. a local copy, one on a server, one on another user's machine, one in a database; each may have different minimum system requirements):

i. Location of executable (which computer, building, city)

ii. Hardware system requirements

iii. Software system requirements

6. What resources are required to run system:

m. who (person, software, hardware)

n. does what (role)

o. from where (location, machine),

p. in what sequence;

Note that the most critical item is #3. This is the item that most affects re-usability. A user wanting a better understanding of why a simulation must be run a specific way could refer to information falling under #2. Item 1 is just an "executive summary" of the system for a quick high-level view of the purpose and scope of the system.

Annex D Implementation of SEDRA

D.1 Modules and Unit Tests

This annex discusses the software modules and associated unit tests. Some unit tests use Test Cases directly. For those unit tests, the associated Test Case is simply mentioned. The modules that end with "Glade" are automatically generated by wxGlade: they are tested via the derived class (same name but without "Glade") that adds event handling and data updates. The start-up script is srapGUI.py, which just runs the application: instantiates the logger, main GUI, and AppController and starts the main GUI loop to let the other modules react to user input.

Each of the following Python files is a module which, when run directly by the interpreter (rather than being imported by another file), runs a unit test.

AboutBox.py:

- Shows a dialog box without any buttons
- Contains a short description of application, and version numbers of application (hard-coded), Python (determined from Python), and wxPython (determined from wxPython)
- Test shows the box
- Developer should verify that the information in the box makes sense

AbstractTreeNode.py:

- Abstracts the tree representation of the simulation resources XML "document" to keep track of tree state (which nodes are collapsed/expanded etc)
- Tests the creation of children with append/replace and the matching flags
- Developer should exercising TS-3 (expand/collapse/x-inclusions) and TS-6 (filter) for integration testing

AppController.py:

- Controller module for the application: encapsulates application behavior (logic) by handling user input; any action that requires presenting something to the user is delegated back to the main GUI proxy
- Tests initialization, connection good/bad, changing settings, disconnection, refresh
- Developer should exercise all test cases for integration testing

AppOptions.py:

- Tests that the application options can be saved and restored properly via assertions

BasicLogging.py:

- Provides basic (non-GUI) logging functionality; the LoggingWin.py makes use of it to add a GUI front end to the log, and all non-GUI modules make use of it so that no knowledge of GUI is necessary

DlgAddNewDB.py:

- Shows the dialog box for defining a new database URL in the Portal
- Tests the creation with and without initial values for the text entry fields
- Developer must exercise TS-1a and TS-1c

DlgAppOptions.py:

- Shows the dialog box for the application options
- GUI tested by showing dialog with editDB=True and False
- Developer must exercise TS-1a and TS-1c

DlgGetUserPwd.py:

- Shows the dialog box for entering the username and password for realm/host when doing secure HTTP requests of documents
- GUI tested by showing dialog once: just click OK without having entered anything, this is like having done cancel; showed a second time, type anything then OK, should be no asserts.

Filtering.py:

- Encapsulates the filtering functions, filter sub-window of main GUI, and tree node coloring for matched nodes
- Uses mock objects to test filter function, filter panel manager, colorization of tree items and application of filter
- Developer should also exercise TS-6 for integration testing

LoggingWin.py:

- Formats the wx loggers so that messages, warning and errors generate message boxes in addition to being logged to the window, and verbose message just go to window
- Tests the 4 types of log messages (tester must click ok for each test)
- Developer must verify that icons are correct for each message type and that each message (5 of them) appears in log window with proper type

NetServices.py:

- Provides network-related services such as viewing and downloading documents from the net or DB.
- Tests view and download from DB and fetching from authenticated web-sites
- Developer must exercise TS-1b and TS-3 to TS-6 for integration testing

PortalMainWin.py:

- Main view for the application, adds behavior to the "pure-view" PortalMainWinGlade.py, delegates to Controller (MainApp.py)
- Shows main GUI window (frame, menu, subpanes, status bar), without any event handling
- Tests delegation of event handling via mock handler
- Developer must exercise all TS except TS-1a and TS-1b

SimResourcesDoc.py:

- Represents the meta-data received from the database as a "document" with allowed change/query operations
- Tests basic XML representation and resolving an x-inclusion

The following modules do not have automated testing. They are tested via the Test Cases (Section 8).

ConnectionPanel.py:

- Manages the state of the connection sub-panel of the main GUI
- Tested by exercising TS-1b

SimResPanel.py:

- Manages the part of the GUI representing the tree view of the meta-data and the description panel for the selected node of the tree
- Tested by exercising TS-3 to TS-6

It is possible to run all the unit tests automatically via the testAll.py script. From the command line, "python testAll.py" will take every module and run its tests. Any text output for non-GUI modules can be ignored, except for tracebacks: they point to a bug having been introduced (or a test having become out-of-date). Any text from the GUI modules indicates what to do: click or cancel etc. In some of those modules, the instruction is in the dialog itself, and giving a wrong input (contrary to instruction) will fail that test, giving the impression of a failure. Once the script has run all the tests, it will list those modules for which any traceback appeared.

D.2 System Requirements

- Python 2.4 or later
- wxPython 2.6 or later
- KID 0.8 or later
- Elementtree 1.2.6

- Lxml 1.0.2
- Zip file containing the test databases for ‘HTTP’ plugin type, as well as the diffs to apply to Apache’s httpd.conf to access those databases from Portal

Annex E Data Model and Naming Conventions for Relational Databases

This section describes the data model that was used for the MySQL database. This model is generic and could be used for many different types of databases (MS SQL, Oracle, etc). What changes based on the particular database engine used is the SQL commands that realize the data model.

This section also describes the naming conventions used for the database schema. These naming conventions should be usable for any relational database that can use the described data model, and greatly facilitates the development of the plug-in for the database since many names can be derived algorithmically from a small set of names.

E.1 Naming conventions

The following words have been abbreviated in the naming of tables:

- System – sys
- Simulation – sim
- Component – comp
- Document – doc

All table names are purely lower case, do not contain plurals, and use the above abbreviations. When a table contains several words, the words are separated by one underscore '_' character.

Tables that relate two simpler entities (called many-to-many tables, e.g. the table that identifies which documents are in which document groups) use the table name for the entities, separated by two underscores. E.g. sys_group__sim_doc is the table that identifies which simulation documents are in which system groups.

The following tables were created, as per the data model described below:

- comp_group (for ComponentGroup resources)
- comp_group__comp_group
- comp_group__doc_group
- comp_group__sim_c2
- comp_group__sim_comp
- comp_group__sim_doc
- comp_group__summary_field
- doc_group (for DocumentGroup resources)
- doc_group__doc_group

- doc_group__sim_doc
- doc_group__summary_field
- sim_c2 (for LaunchConfig resources)
- sim_c2__summary_field
- sim_comp (for SysComponent resources)
- sim_comp__doc_group
- sim_comp__sim_c2
- sim_comp__sim_doc
- sim_comp__summary_field
- sim_doc (for Document resources)
- sim_doc__summary_field
- sim_sys (for SimSystem resources)
- sim_sys__comp_group
- sim_sys__doc_group
- sim_sys__sim_c2
- sim_sys__sim_comp
- sim_sys__sim_doc
- sim_sys__summary_field
- sim_url
- summary_field
- sys_group (for SystemGroup resources)
- sys_group__doc_group
- sys_group__sim_doc
- sys_group__sim_sys
- sys_group__summary_field
- sys_group__sys_group

There are two types of tables in the above list that are worth describing further: simulation resources tables (with the element name as described in Annex A), and resource relationship tables. The former simply list the various simulation resources catalogued, whereas the latter defines what resource is defined within which other resource. The sys_group, sim_sys, comp_group, sys_comp, doc_group, sim_doc, and sim_c2 tables are all simulation resources tables.

The table columns were named according to the following rules:

- Primary key: name of table + “_id”. E.g. if the table name is “sim_sys” then the primary key column name for that table would be “sim_sys_id”.
- Foreign Key: name of table + “_fk”. E.g. if a table has a foreign key column referring to the primary key column of table “sim_sys”, then the foreign key column name would be “sim_sys_fk”.
- Constraint name: table name abbreviation + “__” + primary key + “__constraint”. The abbreviation is the first letter of words forming the table name, i.e. “sg” for sys_group, etc.
- Foreign key constraint name: same as for constraint, but ends with “__fk” instead of “__constraint”

The following list describes each table:

1. comp_group table:

```
comp_group_id       INTEGER UNSIGNED       NOT NULL AUTO_INCREMENT
name                VARCHAR(80)                        NOT NULL
dir                 VARCHAR(255)                           NULL
xinclude            BOOLEAN                          DEFAULT TRUE
```

2. comp_group__comp_group table:

```
comp_group_id_fk    INTEGER UNSIGNED                   NOT NULL
comp_group_id_fk2   INTEGER UNSIGNED                   NOT NULL
pred_code           VARCHAR(20)                            NULL
name_override       VARCHAR(80)                            NULL
```

3. comp_group__doc_group table:

```
comp_group_id_fk    INTEGER UNSIGNED                   NOT NULL
doc_group_id_fk     INTEGER UNSIGNED                   NOT NULL
pred_code           VARCHAR(20)                            NULL
name_override       VARCHAR(80)                            NULL
```

4. comp_group__sim_c2 table:

```
comp_group_id_fk    INTEGER UNSIGNED                   NOT NULL
sim_c2_id_fk        INTEGER UNSIGNED                   NOT NULL
pred_code           VARCHAR(20)                            NULL
name_override       VARCHAR(80)                            NULL
```

5. comp_group__sim_comp table:

```
comp_group_id_fk    INTEGER UNSIGNED                   NOT NULL
sim_comp_id_fk      INTEGER UNSIGNED                   NOT NULL
pred_code           VARCHAR(20)                            NULL
name_override       VARCHAR(80)                            NULL
```

6. comp_group__sim_doc table:

```
comp_group_id_fk    INTEGER UNSIGNED                    NOT NULL
sim_doc_id_fk       INTEGER UNSIGNED                    NOT NULL
pred_code           VARCHAR(20)                             NULL
name_override       VARCHAR(80)                             NULL
```

7. comp_group__summary_field table:

```
comp_group_id_fk    INTEGER UNSIGNED                    NOT NULL
summary_field_id_fk INTEGER UNSIGNED                    NOT NULL
name_override       VARCHAR(80)                             NULL
```

8. doc_group table:

```
doc_group_id        INTEGER UNSIGNED     NOT NULL  AUTO_INCREMENT
name                VARCHAR(80)                         NOT NULL
dir                 VARCHAR(255)                            NULL
xinclude            BOOLEAN                           DEFAULT TRUE
```

9. doc_group__doc_group table:

```
doc_group_id_fk     INTEGER UNSIGNED                    NOT NULL
doc_group_id_fk2    INTEGER UNSIGNED                    NOT NULL
pred_code           VARCHAR(20)                             NULL
name_override       VARCHAR(80)                             NULL
```

10. doc_group__sim_doc table:

```
doc_group_id_fk     INTEGER UNSIGNED                    NOT NULL
sim_doc_id_fk       INTEGER UNSIGNED                    NOT NULL
pred_code           VARCHAR(20)                             NULL
name_override       VARCHAR(80)                             NULL
```

11. doc_group__summary_field table:

```
doc_group_id_fk     INTEGER UNSIGNED                    NOT NULL
summary_field_id_fk INTEGER UNSIGNED                    NOT NULL
name_override       VARCHAR(80)                             NULL
```

12. sim_c2 table:

```
sim_c2_id           INTEGER UNSIGNED      NOT NULL AUTO_INCREMENT
name                VARCHAR(80)                         NOT NULL
app_type            VARCHAR(255)                            NULL
actions_defn        VARCHAR(255)                            NULL
sim_url_id_fk       INTEGER UNSIGNED                        NULL
```

13. sim_c2__summary_field table:

```
sim_c2_id_fk        INTEGER UNSIGNED                    NOT NULL
summary_field_id_fk INTEGER UNSIGNED                    NOT NULL
```

```
name_override       VARCHAR(80)                              NULL
```

14. sim_comp table:

```
sim_comp_id         INTEGER UNSIGNED      NOT NULL AUTO_INCREMENT
name                VARCHAR(80)                        NOT NULL
dir                 VARCHAR(255)                           NULL
xinclude            BOOLEAN                         DEFAULT TRUE
```

15. sim_comp__doc_group table:

```
sim_comp_id_fk      INTEGER UNSIGNED                   NOT NULL
doc_group_id_fk     INTEGER UNSIGNED                   NOT NULL
pred_code           VARCHAR(20)                            NULL
name_override       VARCHAR(80)                            NULL
```

16. sim_comp__sim_c2 table:

```
sim_comp_id_fk      INTEGER UNSIGNED                   NOT NULL
sim_c2_id_fk        INTEGER UNSIGNED                   NOT NULL
pred_code           VARCHAR(20)                            NULL
name_override       VARCHAR(80)                            NULL
```

17. sim_comp__sim_doc table:

```
sim_comp_id_fk      INTEGER UNSIGNED                   NOT NULL
sim_doc_id_fk       INTEGER UNSIGNED                   NOT NULL
pred_code           VARCHAR(20)                            NULL
name_override       VARCHAR(80)                            NULL
```

18. sim_comp__summary_field table:

```
sim_comp_id_fk      INTEGER UNSIGNED                   NOT NULL
summary_field_id_fk INTEGER UNSIGNED                   NOT NULL
name_override       VARCHAR(80)                            NULL
```

19. sim_doc table:

```
sim_doc_id          INTEGER UNSIGNED      NOT NULL AUTO_INCREMENT
name                VARCHAR(80)                        NOT NULL
dir                 VARCHAR(255)                           NULL
sim_url_id_fk       INTEGER UNSIGNED                       NULL
```

20. sim_doc__summary_field table:

```
sim_doc_id_fk       INTEGER UNSIGNED                   NOT NULL
summary_field_id_fk INTEGER UNSIGNED                   NOT NULL
name_override       VARCHAR(80)                            NULL
```

21. sim_sys table:

```
sim_sys_id          INTEGER UNSIGNED       NOT NULL AUTO_INCREMENT
name                VARCHAR(80)                        NOT NULL
dir                 VARCHAR(255)                           NULL
xinclude            BOOLEAN                         DEFAULT TRUE
```

22. sim_sys__comp_group table:

```
sim_sys_id_fk       INTEGER UNSIGNED                   NOT NULL
comp_group_id_fk    INTEGER UNSIGNED                   NOT NULL
pred_code           VARCHAR(20)                            NULL
name_override       VARCHAR(80)                            NULL
```

23. sim_sys__doc_group table:

```
sim_sys_id_fk       INTEGER UNSIGNED                   NOT NULL
doc_group_id_fk     INTEGER UNSIGNED                   NOT NULL
pred_code           VARCHAR(20)                            NULL
name_override       VARCHAR(80)                            NULL
```

24. sim_sys__sim_c2 table:

```
sim_sys_id_fk       INTEGER UNSIGNED                   NOT NULL
sim_c2_id_fk        INTEGER UNSIGNED                   NOT NULL
pred_code           VARCHAR(20)                            NULL
name_override       VARCHAR(80)                            NULL
```

25. sim_sys__sim_comp table:

```
sim_sys_id_fk       INTEGER UNSIGNED                   NOT NULL
sim_comp_id_fk      INTEGER UNSIGNED                   NOT NULL
pred_code           VARCHAR(20)                            NULL
name_override       VARCHAR(80)                            NULL
```

26. sim_sys__sim_doc table:

```
sim_sys_id_fk       INTEGER UNSIGNED                   NOT NULL
sim_doc_id_fk       INTEGER UNSIGNED                   NOT NULL
pred_code           VARCHAR(20)                            NULL
name_override       VARCHAR(80)                            NULL
```

27. sim_sys__summary_field table:

```
sim_sys_id_fk       INTEGER UNSIGNED                   NOT NULL
summary_field_id_fk INTEGER UNSIGNED                   NOT NULL
name_override       VARCHAR(80)                            NULL
```

28. sim_url table:

```
sim_url_id          INTEGER UNSIGNED       NOT NULL AUTO_INCREMENT
url_address         VARCHAR(1024)                      NOT NULL
```

29. summary_field table:

```
summary_field_id    INTEGER UNSIGNED       NOT NULL AUTO_INCREMENT
name                VARCHAR(255)                       NOT NULL
content             VARCHAR(255)                           NULL
```

30. sys_group table:

```
sys_group_id        INTEGER UNSIGNED       NOT NULL AUTO_INCREMENT
name                VARCHAR(80)                        NOT NULL
dir                 VARCHAR(255)                           NULL
xinclude            BOOLEAN                          DEFAULT TRUE
```

31. sys_group__doc_group table:

```
sys_group_id_fk     INTEGER UNSIGNED                   NOT NULL
doc_group_id_fk     INTEGER UNSIGNED                   NOT NULL
pred_code           VARCHAR(20)                            NULL
name_override       VARCHAR(80)                            NULL
```

32. sys_group__sim_doc table:

```
sys_group_id_fk     INTEGER UNSIGNED                   NOT NULL
sim_doc_id_fk       INTEGER UNSIGNED                   NOT NULL
pred_code           VARCHAR(20)                            NULL
name_override       VARCHAR(80)                            NULL
```

33. sys_group__sim_sys table:

```
sys_group_id_fk     INTEGER UNSIGNED                   NOT NULL
sim_sys_id_fk       INTEGER UNSIGNED                   NOT NULL
pred_code           VARCHAR(20)                            NULL
name_override       VARCHAR(80)                            NULL
```

34. sys_group__summary_field table:

```
sys_group_id_fk      INTEGER UNSIGNED                  NOT NULL
summary_field_id_fk  INTEGER UNSIGNED                  NOT NULL
name_override        VARCHAR(80)                           NULL
```

35. sys_group__sys_group table:

```
sys_group_id_fk     INTEGER UNSIGNED                   NOT NULL
sys_group_id_fk2    INTEGER UNSIGNED                   NOT NULL
pred_code           VARCHAR(20)                            NULL
name_override       VARCHAR(80)                            NULL
```

E.2 Entity Relationships

The column names from the above tables, following the naming conventions mentioned above, indicate relationships that are more easily seen with diagrams. Those are given next.

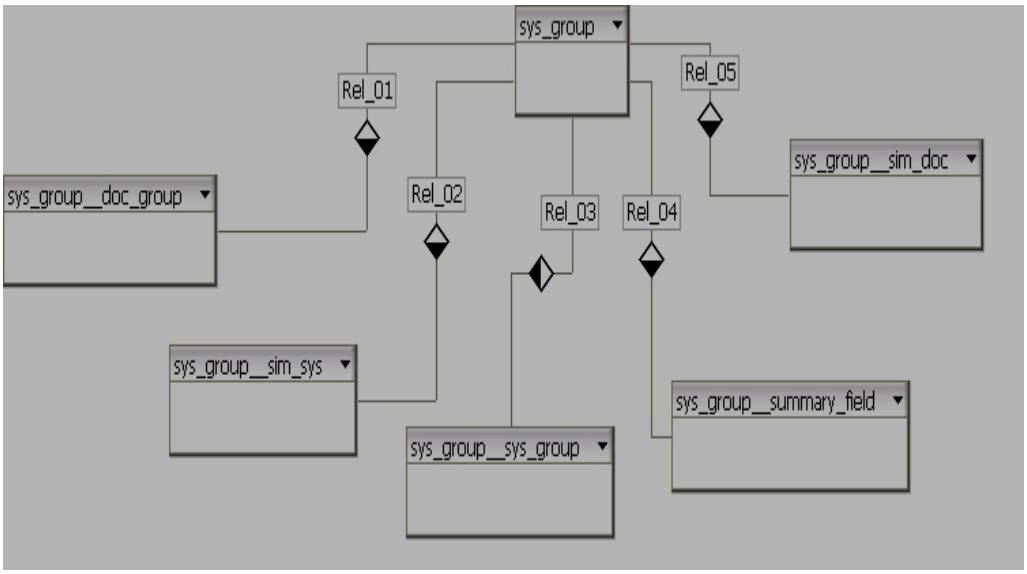

Figure 15: DB schema tables related to sys_group (SystemGroup)

Figure 14 shows:

- The SystemGroup data model shows the relationship between the sys_group table and other simulation resource tables that it is linked to.
- The sys_group table has a one-to-many relationship with the sys_group__doc_group table and the doc_group table also has a one-to-many relationship with the sys_group__doc_group table to form the sys_group__doc_group table.
- The sys_group__sim_sys table comprises of foreign keys from the sys_group table and the sim_sys table.
- The sys_group__sys_group table has two foreign keys both from the sys_group table to form a table of system groups that can contain other system groups.
- The sys_group__summary_field table comprises of foreign keys from the sys_group and summary_field tables.
- The sys_group__sim_doc table comprises of foreign keys from the sys_group and sim_doc tables.

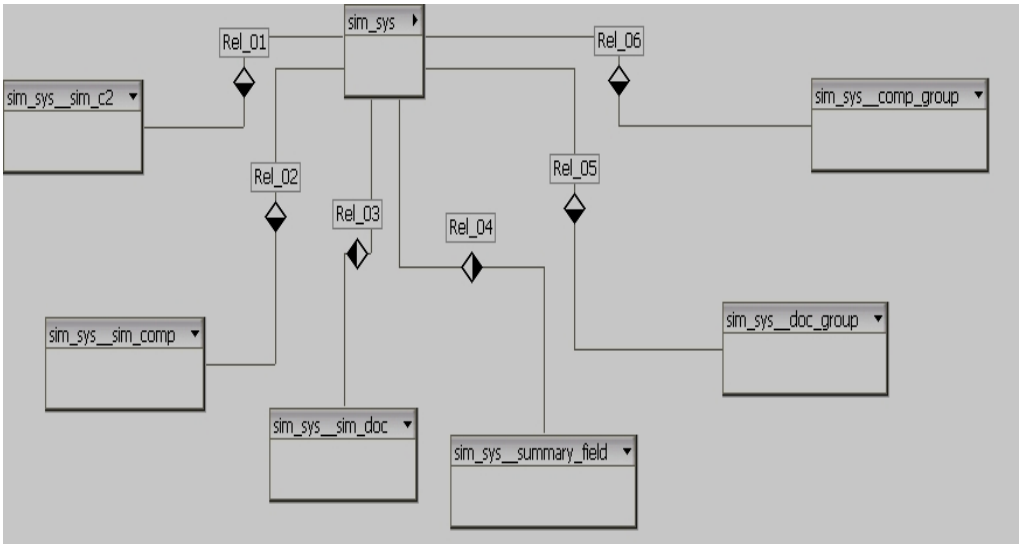

Figure 16: DB schema tables related to sim_sys (SimulationSystem)

Figure 15 shows:

- The SimSystem data model shows the relationship between the sim_sys table and other simulation resource tables that it is linked to.
- The sim_sys table has a one-to-many relationship with the sim_sys__sim_c2 table and the sim_c2 table has a one-to-many relationship with the sim_sys__sim_c2 to form the sim_sys__sim_c2 table.
- The sim_sys__sim_comp table comprises of foreign keys from the sim_sys and sim_comp table.
- The sim_sys__sim_doc table comprises of foreign keys from the sim_sys and sim_doc table.
- The sim_sys__summary_field table comprises of foreign keys from the sim_sys and summary_field table.
- The sim_sys__doc_group table comprises of foreign keys from the sim_sys and doc_group table.
- The sim_sys__comp_group table comprises of foreign keys from the sim_sys and comp_group table.

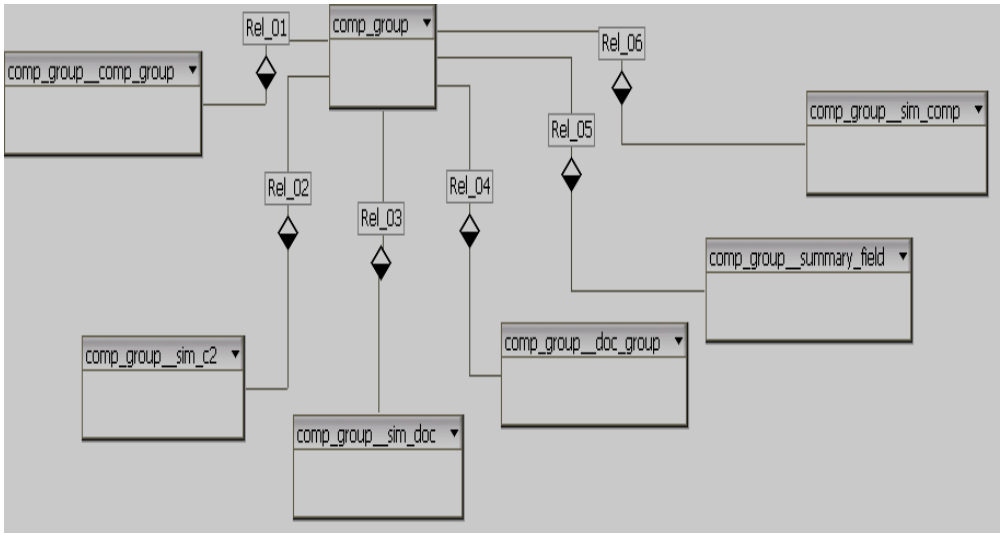

Figure 17: DB schema tables related to comp_group (ComponentGroup)

Figure 16 shows:

- The ComponentGroup simulation model shows the relationship between the comp_group table and other simulation resource tables.
- The comp_group__comp_group table has two foreign keys both from the com_group table to form a table of component groups that can contain other component groups.
- The comp_group__sim_c2 table comprises of foreign keys from the comp_group and sim_c2 tables.
- The comp_group__sim_doc table comprises of foreign keys from the comp_group and sim_doc tables.
- The comp_group__doc_group table comprises of foreign keys from the comp_group and doc_group tables.
- The comp_group__summary_field table comprises of foreign keys from the comp_group and summary_field tables.
- The comp_group__sim_comp table comprises of foreign keys from the comp_group and sim_comp tables.

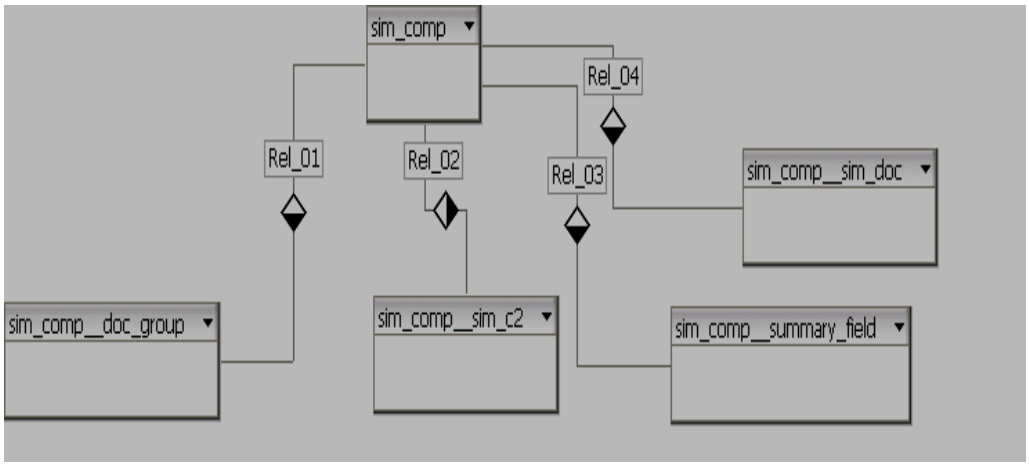

Figure 18: DB schema tables related to sim_comp (SysComponent)

Figure 17 shows:

- The SysComponent data model shows the relationship between the sim_comp table and other simulation resource tables that it is linked to.
- The sim_comp__doc_group table comprises of foreign keys from the sim_comp and the doc_group tables.
- The sim_comp__sim_c2 table comprises of foreign keys from the sim_comp and the sim_c2 tables.
- The sim_comp__summary_field table comprises of foreign keys from the sim_comp and summary_field tables.
- The sim_comp__sim_doc table comprises of foreign keys from the sim_comp and sim_doc tables.

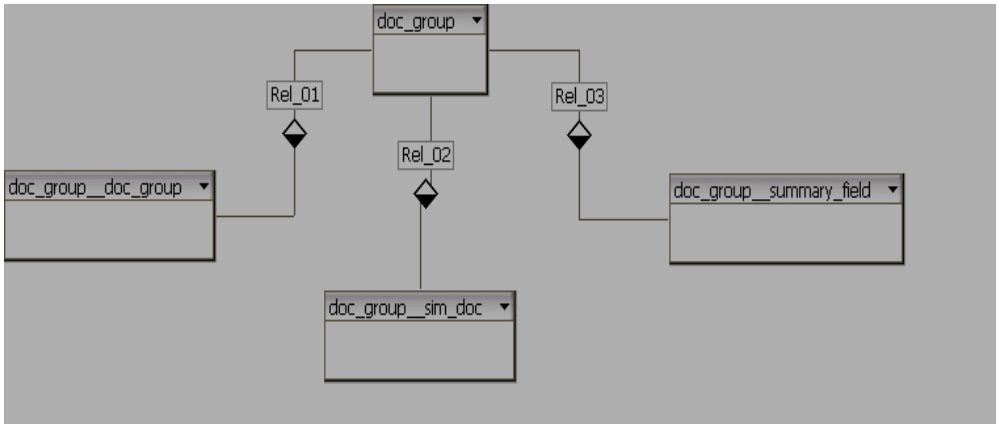

Figure 19: DB schema tables related to doc_group (DocumentGroup)

Figure 18 shows:

- The DocumentGroup data model shows the relationship between the doc_group table and other simulation resource tables that it is linked to.
- The doc_group__doc_group table has two foreign keys both from the doc_group table to form a table of document groups that can contain other document groups.
- The doc_group__sim_doc table comprises of foreign keys from the doc_group table and sim_doc table.

Note: The data model diagram has been reverse engineered from the database as created by the creation SQL scripts. Further updates to this diagram are expected as implementation proceeds.

This page intentionally left blank.

List of symbols/abbreviations/acronyms/initialisms

AVI	Audio Video Interleave
BMH	The company that develops and maintains MARCI
CAD	Computer-Assisted Design
DB	Database
DLL	Dynamically Loaded Library
DND	Department of National Defence
DRDC	Defence Research & Development Canada
FOM	Federation Object Model
GUI	Graphical User Interface
HTML	Hypertext Markup Language
JSAF	Joint Semi-Automated Forces
JSMARTS	Joint Simulation and Modeling for Acquisition, Requirements, Training and Support
MARCI	Multi Host Automated Remote Control and Instrumentation
PDF	Portable Document Format
R&D	Research & Development
Sim	Simulation
SE	Synthetic Environments
SEDR	Synthetic Environment Distributed Resources
SEDRA	SEDR Access
SEDReC	SEDR Control
SEDReDE	SEDR Data Entry
SNE	Simulation Network Exploitation
SQL	Structured English Query Language (originally SEQUEL)
Sync'd	Synchronized
TS	Test suite
TC	Test case
UML	Unified Modeling Language
URL	Uniform Resource Locator
XHTML	Extensible HTML (i.e., HTML formatted according to XML rules)

XML	Extensible Markup Language

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1. ORIGINATOR (The name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.)

CAE Professional Services
1135 Innovation Dr. Suite 300
Ottawa, ON K2K 3G7

2. SECURITY CLASSIFICATION (Overall security classification of the document including special warning terms if applicable.)

UNCLASSIFIED

3. TITLE (The complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S, C or U) in parentheses after the title.)

Simulation network exploitation: SEDRA and SEDReDE portals: specifications document

4. AUTHORS (last name, followed by initials – ranks, titles, etc. not to be used)

Schoenborn, O.

5. DATE OF PUBLICATION (Month and year of publication of document.)

April 2009

6a. NO. OF PAGES (Total containing information, including Annexes, Appendices, etc.)

84

6b. NO. OF REFS (Total cited in document.)

0

7. DESCRIPTIVE NOTES (The category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)

Contract Report

8. SPONSORING ACTIVITY (The name of the department project office or laboratory sponsoring the research and development – include address.)

Defence R&D Canada – Ottawa
3701 Carling Avenue
Ottawa, Ontario K1A 0Z4

9a. PROJECT OR GRANT NO. (If appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant.)

13jb

9b. CONTRACT NO. (If appropriate, the applicable number under which the document was written.)

W8475-06BM04

10a. ORIGINATOR'S DOCUMENT NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document.)

10b. OTHER DOCUMENT NO(s). (Any other numbers which may be assigned this document either by the originator or by the sponsor.)

DRDC Ottawa CR 2008-315

11. DOCUMENT AVAILABILITY (Any limitations on further dissemination of the document, other than those imposed by security classification.)

Unlimited

12. DOCUMENT ANNOUNCEMENT (Any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in (11) is possible, a wider announcement audience may be selected.))

Unlimited

13. ABSTRACT (A brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual.)

This document specifies the functionality that should be supported by the Synthetic Environment Distributed Resources Access (SEDRA) portal and Synthetic Environment Distributed Resources Data Entry (SEDReDE) portal. These applications are intended to facilitate the description and access to simulation resources, such as documents, video clips, audio captures, and computer programs, available on a distributed network of computers. The resources can be described in one or more databases via SEDReDE and browsed using SEDRA. The latter delegates to a separate application (named SEDReC – Synthetic Environment Distributed Resources Control) the task of executing those simulation resources which are executable

Ce document spécifie les fonctionnalités soutenues par le portail d'accès aux ressources distribuées des environnements synthétiques (SEDRA) ainsi que le portail de saisie de données de ces ressources (SEDReDE). Ces applications sont destinées à faciliter la description et l'accès aux ressources de simulation, tels que des documents, des clips vidéo, capture audio, et logiciels; ces ressources étant évidemment disponibles sur un réseau distribué. Les ressources peuvent être décrites dans une ou plusieurs bases de données via SEDReDE et navigables en utilisant SEDRA. SEDRA délègue la tâche d'exécution des composantes exécutables à une autre application du nom SEDReC (Control des ressources distribuées des environnements synthétiques).

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (Technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus, e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

Modeling & Simulation, Synthetic Environments, Distributed Simulation, Data Management, Knowledge Management, Remote Access, Software

Defence R&D Canada

Canada's leader in Defence
and National Security
Science and Technology

R & D pour la défense Canada

Chef de file au Canada en matière
de science et de technologie pour
la défense et la sécurité nationale

www.drdc-rddc.gc.ca