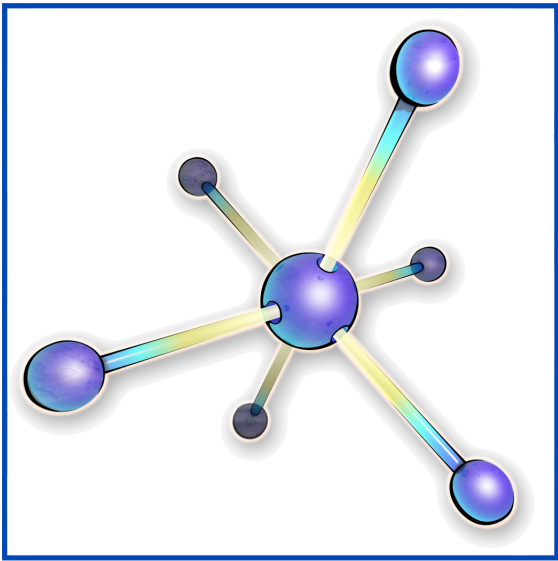
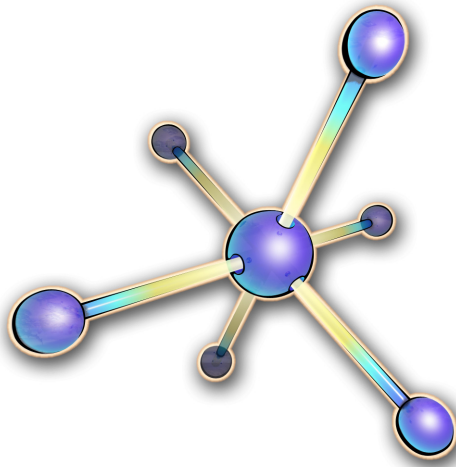




VR-Exchange

Users Guide



VR-Exchange



Users Guide

Copyright © 2012 VT MÄK
All rights Reserved. Printed in the United States.

Under copyright laws, no part of this document may be copied or reproduced in any form without prior written consent of VT MÄK.

VR-Exchange™ and VR-Vantage™ are trademarks of VT MÄK.
MÄK Technologies®, VR-Forces®, RTIspy®, B-HAVE®, and VR-Link® are registered trademarks of VT MÄK.

All other trademarks are owned by their respective companies.

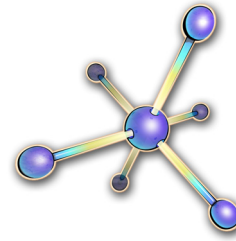
For third-party license information, please see [“Third Party Licenses,”](#) on page xii.

VT MÄK
68 Moulton St.
Cambridge, MA 02138 USA

Voice: 617-876-8085
Fax: 617-876-9208

info@mak.com
www.mak.com

Revision VRE-2.1-1-120726



Contents

Preface

How the Manual is Organized	vii
MÄK Products	viii
How to Contact Us	x
Document Conventions	xi
Mouse Button Naming Conventions.....	xii
Third Party Licenses	xii
Boost License.....	xii
libXML and libICONV	xiii
pThreads Library.....	xiii
EHS, PCRE, and PME	xiii
Freefont OpenType Font Set.....	xiii

Chapter 1 Introduction

1.1. The VR-Exchange Architecture	1-2
1.1.1. Translators	1-3
1.1.2. Using VR-Exchange to Manage a Federation of Federations ...	1-4
1.1.3. The VR-Exchange Portal Application	1-5
1.1.4. FOM-Agility and LROM-Agility	1-5
1.2. The VR-Exchange API	1-6
1.3. Brokers Supplied	1-6
1.4. Translating Between DIS and the RPR FOM	1-8
1.4.1. Entity Information and Entity Interaction	1-10
1.4.2. Warfare	1-11
1.4.3. Simulation Management	1-11
1.4.4. Radio Communications	1-12
1.4.5. Distributed Emission Regeneration	1-13
1.4.6. Logistics	1-14
1.4.7. Entity Management	1-14

1.4.8. Synthetic Environment	1-14
1.4.9. MÄK Logger Control PDUs	1-14
1.5. Hardware Requirements and Product Limitations	1-15
1.6. The Limitations of Translation Software	1-16

Chapter 2 Installing VR-Exchange

2.1. Installing VR-Exchange	2-2
2.1.1. Installing VR-Exchange on Windows	2-2
2.1.2. Installing VR-Exchange on a Linux System	2-2
2.2. Installing and Setting Up the MÄK License Manager	2-3
2.2.1. Specifying the License Server	2-4
2.3. Installing an RTI	2-6
2.3.1. Installing the MÄK RTI	2-6
2.4. Installing TENA Software	2-7

Chapter 3 Using VR-Exchange

3.1. Starting VR-Exchange	3-2
3.1.1. Starting VR-Exchange on Linux	3-2
3.1.2. Starting VR-Exchange on Windows	3-3
3.1.3. VR-Exchange Command-Line Options	3-4
3.1.4. The Portal Application	3-5
3.1.5. Running VR-Exchange without the Portal Application	3-5
3.2. Configuring VR-Exchange	3-6
3.3. Enabling and Disabling Connections	3-8
3.3.1. Enabling Connections when the Portal Starts	3-8
3.3.2. Enabling Connections During Runtime	3-9
3.3.3. Disabling Connections	3-9
3.4. Viewing Objects and Interactions	3-10
3.4.1. Customizing the Display of Object and Interaction Data	3-11
3.4.2. Enabling and Disabling the Display of Object Data	3-11
3.4.3. Sorting the Object List by Column	3-11
3.4.4. Enabling and Disabling the Display of Interactions	3-11
3.4.5. Clearing the Interactions List	3-12
3.5. Filtering Objects	3-13
3.6. Displaying the Queue Status	3-14
3.7. The Data View	3-14
3.8. Closing VR-Exchange	3-15
3.9. Using vrxMessageDump	3-15

Chapter 4 Working with Broker Connections

4.1. Adding and Editing Connections	4-2
4.1.1. Adding Connections	4-2
4.1.2. Specifying Environment Variables for a Connection	4-4
4.1.3. Editing Connections	4-4

4.1.4. Deleting Connections	4-5
4.2. Configuring Connections	4-5
4.2.1. Specifying Collection Values	4-7
4.2.2. Common Broker Settings	4-9
4.2.3. Configuring Publication Conditions	4-10
4.2.4. Protocol Specific Attributes	4-12
4.3. Configuring Network Connections	4-18
4.3.1. HLA Network Connection Parameters	4-19
4.3.2. DIS Network Connection Parameters	4-20
4.3.3. TENA Network Connection Parameters	4-22
4.4. Filtering Objects	4-23
4.4.1. Editing a Filter	4-24
4.4.2. Removing a Filter	4-25
4.5. Specifying Which Translators to Use	4-26
4.6. Viewing Publishing and Reflecting Lists	4-27

Appendix A Broker Details

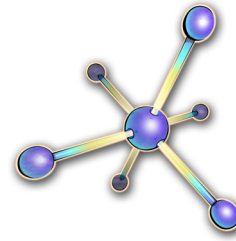
A.1. The HLA Brokers	A-2
A.1.1. HLA Translators	A-3
A.1.2. Generic Translations	A-4
A.1.3. HLA Broker Command-Line Options	A-6
A.2. The DIS Broker	A-8
A.2.1. DIS Broker Command-Line Options	A-9
A.3. The TENA Broker	A-11
A.3.1. TENA Translators	A-12
A.3.2. TENA Broker Command-Line Options	A-13

Appendix B A VR-Exchange/TENA Primer

B.1. Installing TENA Middleware	B-2
B.1.1. Downloading the TENA Middleware	B-2
B.1.2. Installing the TENA Middleware	B-2
B.2. Building the MÄK Object Model	B-3
B.2.1. Downloading the MÄK Object Model	B-3
B.2.2. Installing the MÄK Object Model	B-4
B.2.3. Building the MÄK Object Model Basic Implementation	B-5
B.3. Running the TENA Broker	B-6
B.3.1. TENA Services	B-7

MÄK Products Glossary

Index



Preface

This manual is written for persons who will install or use VR-Exchange. The manual assumes that you are familiar with your operating system and windowing environment.

Please see *VR-Exchange Release Notes* for updates to this manual and the latest information about your version of VR-Exchange.

How the Manual is Organized

This manual is organized as follows:

Chapter 1, *Introduction*, describes VR-Exchange and its architecture. It also contains an important note about the limitations of translation software.

Chapter 2, *Installing VR-Exchange*, describes installation, including setting up the license server and installing an RTI.

Chapter 3, *Using VR-Exchange*, explains how to run VR-Exchange and use it to manage connections and view the data being processed.

Chapter 4, *Working with Broker Connections*, explains how to add, edit, and configure connections.

Appendix A, *Broker Details* describes the configuration parameters for each broker, lists the supported translators, and lists the command-line options that you can use when you start brokers.

Appendix B, *A VR-Exchange/TENA Primer*, provides a brief explanation of how to download and install TENA software.

The *MÄK Products Glossary* defines terms common to real-time simulations and MÄK products.

MÄK Products

VR-Exchange is a member of the VT MÄK line of software products designed to streamline the process of developing and using networked simulated environments. The VT MÄK product line includes the following:

- ♦ VR-Link® Network Toolkit. VR-Link is an object-oriented library of C++ functions and definitions that implement the High Level Architecture (HLA) and the Distributed Interactive Simulation (DIS) protocol. VR-Link has built-in support for the RPR FOM and allows you to map to other FOMs. This library minimizes the time and effort required to build and maintain new HLA or DIS-compliant applications, and to integrate such compliance into existing applications.

VR-Link includes a set of sample debugging applications and their source code. The source code serves as an example of how to use the VR-Link Toolkit to write applications. The executables provide valuable debugging services such as generating a predictable stream of HLA or DIS messages, and displaying the contents of messages transmitted on the network.

- ♦ MÄK RTI. An RTI (Run-Time Infrastructure) is required to run applications using the High Level Architecture (HLA). The MÄK RTI is optimized for high performance. It has an API, RTIsby®, that allows you to extend the RTI using plug-in modules. It also has a graphical user interface (the RTI Assistant) that helps users with configuration tasks and managing federates and federations.
- ♦ VR-Forces®. VR-Forces is a computer generated forces application and toolkit. It provides an application with a GUI, that gives you a 2D and 3D views of a simulated environment.

You can create and view local entities, aggregate them into hierarchical units, assign tasks, set state parameters, and create plans that have tasks, set statements, and conditional statements. VR-Forces also functions as a plan view display for viewing remote entities taking part in an exercise. Using the toolkit, you can extend the VR-Forces application or create your own application for use with another user interface.

- ♦ VR-Vantage™. VR-Vantage is a line of products designed to meet your simulation visualization needs. It includes four end-user applications (VR-Vantage Stealth, VR-Vantage XR, VR-Vantage PVD, and VR-Vantage IG), the VR-Vantage Toolkit, and VR-Vantage FreeView.
 - VR-Vantage Stealth displays a realistic, 3D view of your virtual world. You can view this world from the inside of a simulated moving vehicle, or place the eyepoint at another moving or stationary location. The Stealth lets you switch rapidly among several predefined viewpoints while the simulation is underway.
 - VR-Vantage IG is a configurable desktop image generator (IG) for out the window (OTW) scenes and remote camera views. It has most of the features of the Stealth, but is optimized for its IG function.

- VR-Vantage XR provides a common operational picture using the same 3D views as VR-Vantage Stealth, a 2D plan view, and an exaggerated reality (XR) view. Together these views provide both situational awareness and the big picture of the simulated world.
- VR-Vantage PVD provides a 2D plan view display. It gives you the big picture of the simulated world.
- The VR-Vantage Toolkit is a 3D visual application development toolkit. Use it to customize or extend MÄK's VR-Vantage applications, or to integrate VR-Vantage capabilities into your custom applications. VR-Vantage is built on top of OpenSceneGraph (OSG). The toolkit includes the OSG version used to build VR-Vantage.
- VR-Vantage Free View is a terrain and 3D model viewer that introduces some of the features of VR-Vantage applications in a useful, free utility.
- ♦ MÄK Data Logger. The Data Logger, also called the Logger, can record HLA and DIS exercises and play them back for after-action review. You can play a recorded file at speeds above or below normal and can quickly jump to areas of interest. The Logger has a GUI and a text interface. The Logger API allows you to extend the Logger using plug-in modules or embed the Logger into your own application. The Logger editing features let you merge, trim, and offset Logger recordings.
- ♦ VR-Exchange™. VR-Exchange allows simulations that use incompatible communications protocols to interoperate. For example, within the HLA world, using VR-Exchange, federations using the HLA RPR FOM 1.0 can interoperate with simulations using RPR FOM 2.0, or federations using different RTIs can interoperate. VR-Exchange supports HLA, TENA, and DIS translation.
- ♦ VR-TheWorld™ Server. VR-TheWorld Server is a simple, yet powerful, web-based streaming terrain server, developed in conjunction with Pelican Mapping. Delivered with a global base map, you can also easily populate it with your own custom source data through a web-based interface. The server can be deployed on private, classified networks to provide streaming terrain data to a variety of simulation and visualization applications behind your firewall.
- ♦ VR-inTerra. VR-inTerra is a C++ API for adding terrain agility to applications. It can load, page, or stream terrain from a wide variety of formats or sources into a single, consistent run-time representation, consisting of a collision graph and vector network.

How to Contact Us

For VR-Exchange technical support, information about upgrades, and information about other MÄK products, you can contact us in the following ways:

Telephone

Call or fax us at:	Voice:	617-876-8085 (extension 3 for support)
	Fax:	617-876-9208

E-mail

Sales and upgrade information:	info@mak.com
Technical support:	support@mak.com
VR-Vantage support:	vrv-support@mak.com

Internet

MÄK web site home page:	www.mak.com
License key requests:	www.mak.com/support/get-licenses.html
Product version and platform information:	www.mak.com/support/ product-versions.html
For the free, unlicensed MÄK RTI:	www.mak.com/resources/bonus- material/cat_view/16-bonus-materials/ 24-mak-high-performance-rti.html
MÄK Community Forum:	www.mak.com/community-forum/ 1-forum.html

Post

Send postal correspondence to:	VT MÄK 68 Moulton St. Cambridge, MA, USA 02138
--------------------------------	--

When requesting support, please tell us the product you are using, the version, and the platform on which you are running.

Document Conventions

This manual uses the following typographic conventions:

<code>Monospaced</code>	Indicates commands or values you enter.
Monospaced Bold	Indicates a key on the keyboard.
<i>Monospaced Italic</i>	Indicates command variables that you replace with appropriate values.
Blue text	A hypertext link to another location in this manual or another manual in the documentation set.
{ }	Indicates required arguments.
[]	Indicates optional arguments.
	Separates options in a command where only one option may be chosen at a time.
()	In command syntax, indicates equivalent alternatives for a command-line option, for example, (-h --help).
/	Indicates a directory. Since MÄK products run on both UNIX and Windows PC platforms, we use the / (slash) for generic discussions of pathnames. If you are running on a PC, substitute a \ (backslash) when you type pathnames.
<i>Italic</i>	Indicates a file name, pathname, or a class name.
sans Serif	Indicates a parameter or argument.
➤	Indicates a one-step procedure.
Menu → Option	Indicates a menu choice. For example, an instruction to select the Save option from the File menu appears as: Choose File → Save .
i	Indicates supplemental or clarifying information.
!	Indicates additional information that you must observe to ensure the success of a procedure or other task.

Directory names are preceded with dot and slash characters that show their position with respect to the VR-Exchange home directory. For example, the directory *vrchange/doc* appears in the text as *./doc*.

Mouse Button Naming Conventions

An instruction to click the mouse button, refers to clicking the primary mouse button, usually the left button for right-handed mice and the right button for left-handed mice. The popup menu refers to the menu displayed when you click the secondary mouse button, usually the right button on right-handed mice and the left button on left-handed mice.

Third Party Licenses

MÄK software products may use code from third parties. This section contains the license documentation required by these third parties.

Boost License

VR-Link, and all MÄK software that uses VR-Link uses some code which is distributed under the Boost License. All header files that contain Boost code are properly attributed. The Boost web site is: www.boost.org.

Boost Software License - Version 1.0 - August 17th, 2003

Permission is hereby granted, free of charge, to any person or organization obtaining a copy of the software and accompanying documentation covered by this license (the “Software”) to use, reproduce, display, distribute, execute, and transmit the Software, and to prepare derivative works of the Software, and to permit third-parties to whom the Software is furnished to do so, all subject to the following:

The copyright notices in the Software and this entire statement, including the above license grant, this restriction and the following disclaimer, must be included in all copies of the Software, in whole or in part, and all derivative works of the Software, unless such copies or derivative works are solely in the form of machine-executable object code generated by a source language processor.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

libXML and libICONV

VR-Link and all MÄK software that uses VR-Link, links in libXML and libICONV. On some platforms the compiled libraries and header files are distributed with MÄK Products. MÄK has made no modifications to these libraries. For more information about these libraries please see the following web sites:

- ♦ The LGPL license is available at: <http://www.gnu.org/licenses/lgpl.html>
- ♦ Information about IconV is at: <http://www.gnu.org/software/libiconv/>
- ♦ Information about LibXML is at: <http://xmlsoft.org/>

pThreads Library

VR-Exchange links with the pThreads win32 library. The library is distributed under the GNU Lesser General Public License (LGPL). MÄK has made no modification to this library. For information about the pThreads win32 library please see: <http://sourceware.org/pthreads-win32/>

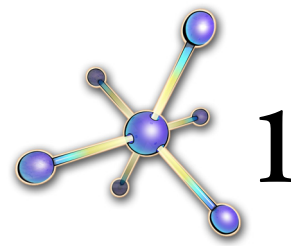
EHS, PCRE, and PME

The MÄK RTI links with the EHS, PCRE, and PME libraries. These libraries are distributed under the GNU Lesser General Public License (LGPL). MÄK has made modification to these libraries only to allow them to compile on the supported platforms. For more information about these libraries please see the following web sites:

- ♦ EHS: <http://xaxxon.slackworks.com/ehs/>
- ♦ PCRE: <http://www.pcre.org/>
- ♦ PME: <http://xaxxon.slackworks.com/pme/index.html>

Freefont OpenType Font Set

VR-Vantage applications and VR-Forces use the Freefont OpenType font set from the Free Software Foundation. It is covered by the General Public License (GPL). For details, please see: <http://www.gnu.org/licenses/gpl.html>



Introduction

This chapter describes VR-Exchange's features and its architecture.

The VR-Exchange Architecture.....	1-2
Translators.....	1-3
Using VR-Exchange to Manage a Federation of Federations.....	1-4
The VR-Exchange Portal Application.....	1-5
FOM-Agility and LROM-Agility	1-5
The VR-Exchange API.....	1-6
Brokers Supplied.....	1-6
Translating Between DIS and the RPR FOM.....	1-8
Entity Information and Entity Interaction.....	1-10
Warfare.....	1-11
Simulation Management	1-11
Radio Communications	1-12
Distributed Emission Regeneration	1-13
Logistics	1-14
Entity Management.....	1-14
Synthetic Environment.....	1-14
MÄK Logger Control PDUs	1-14
Hardware Requirements and Product Limitations.....	1-15
The Limitations of Translation Software	1-16

1.1. The VR-Exchange Architecture

VR-Exchange allows simulations that use incompatible communications protocols to interoperate. For example, within the HLA world, using VR-Exchange, simulations using the HLA RPR FOM 1.0 can interoperate with simulations using RPR FOM 2.0, or simulations using different RTI implementations can interoperate.

VR-Exchange permits simulations to interoperate through the use of a shared memory space (Portal) and brokers (Figure 1-1). Each broker translates between its native protocol and the VR-Exchange common simulation representation. The translated data passes through the Portal and is translated by the other brokers to their protocol.

Given this architecture, it is not necessary to create multiple translation programs that map between specific sets of protocols. You only need to create a broker for a particular protocol, and simulations can interoperate with any other simulations for which a broker exists.

You can run multiple instances of a broker, each configured for a particular simulation. For example, Figure 1-1 shows three different federations. Each uses the broker, but each broker is configured to use different RTIs and different FOMs. Each unique broker configuration is called a connection.

Because the Portal and brokers use shared memory, the Portal and all brokers must run on the same computer.

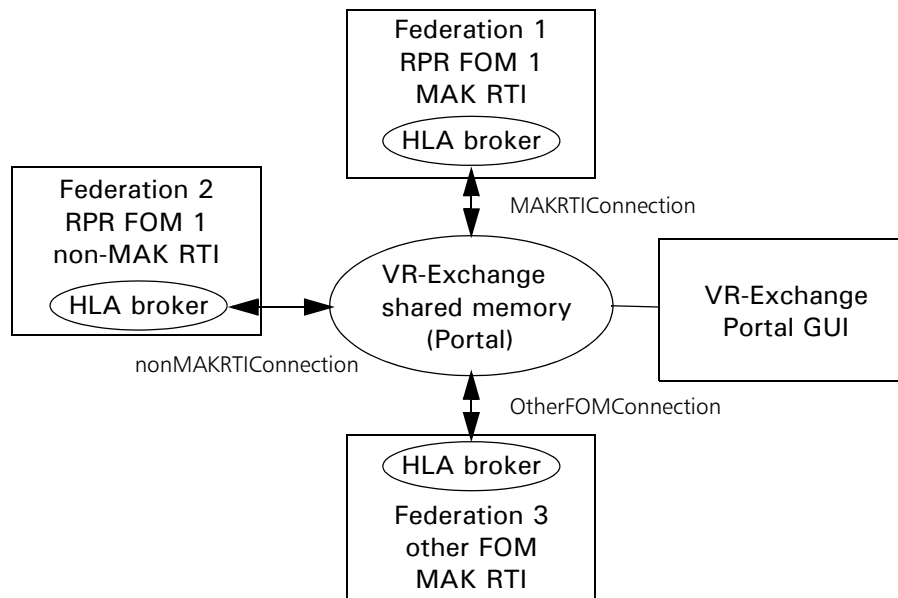


Figure 1-1. VR-Exchange architecture

1.1.1. Translators

Each broker has translators for the object or interaction classes that it supports. [Figure 1-2](#) shows how a translator works. Imagine language-specific HLA FOMs (or TENA LROMs), in this case one for American English and one for British English. Their brokers have translators for the different concepts in the English language. The leftmost federation sends a “lorry” message. The vehicle translator in the broker translates “lorry” to a generic description “wheeled cargo vehicle” and puts the data in the VR-Exchange shared memory area. A second federation, also using the British English FOM, takes the vehicle data from the Portal and translates it back to “lorry”. Finally, a third federation, which uses an American English FOM, reads the vehicle message from the Portal and translates it to its American English equivalent, “truck.”

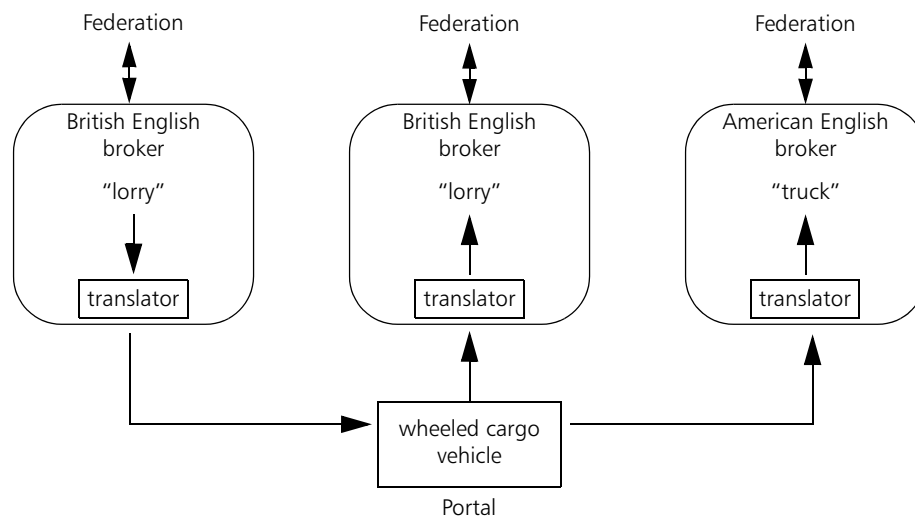


Figure 1-2. How translators work

You can disable translators by editing a connection. Since each connection has its own configuration, you can filter the objects and interactions that the federations are communicating through VR-Exchange.

1.1.2. Using VR-Exchange to Manage a Federation of Federations

In addition to using VR-Exchange to enable heterogeneous federations to interoperate, you can use it to help manage a federation of federations, regardless of whether they use the same FOMs and RTIs. Consider the configuration in [Figure 1-3](#). Four federations are connected via brokers to a VR-Exchange Portal. These Portals are connected to brokers that form their own federation. If any of the individual federations fail, then each instance of VR-Exchange can detect this and drop it from the exercise, while the others can continue.

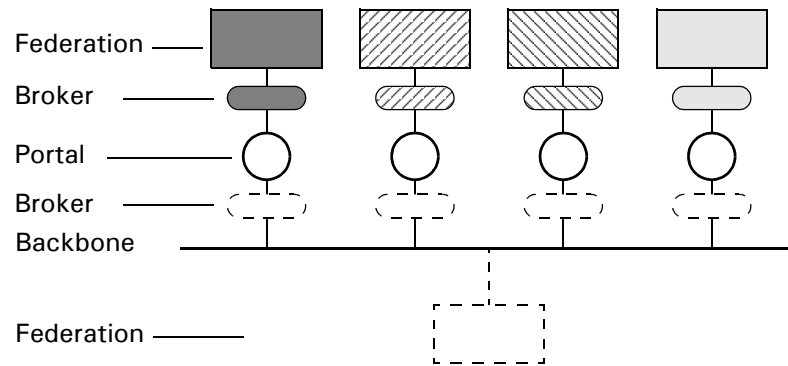


Figure 1-3. Using VR-Exchange over a WAN

1.1.3. The VR-Exchange Portal Application

VR-Exchange has a graphical user interface (Portal application) that provides a view into the shared memory space. It displays the connections in use and the objects and interactions being translated (Figure 1-4). The Portal application does not control the translation process. Translation takes place in the brokers, which are separate programs.

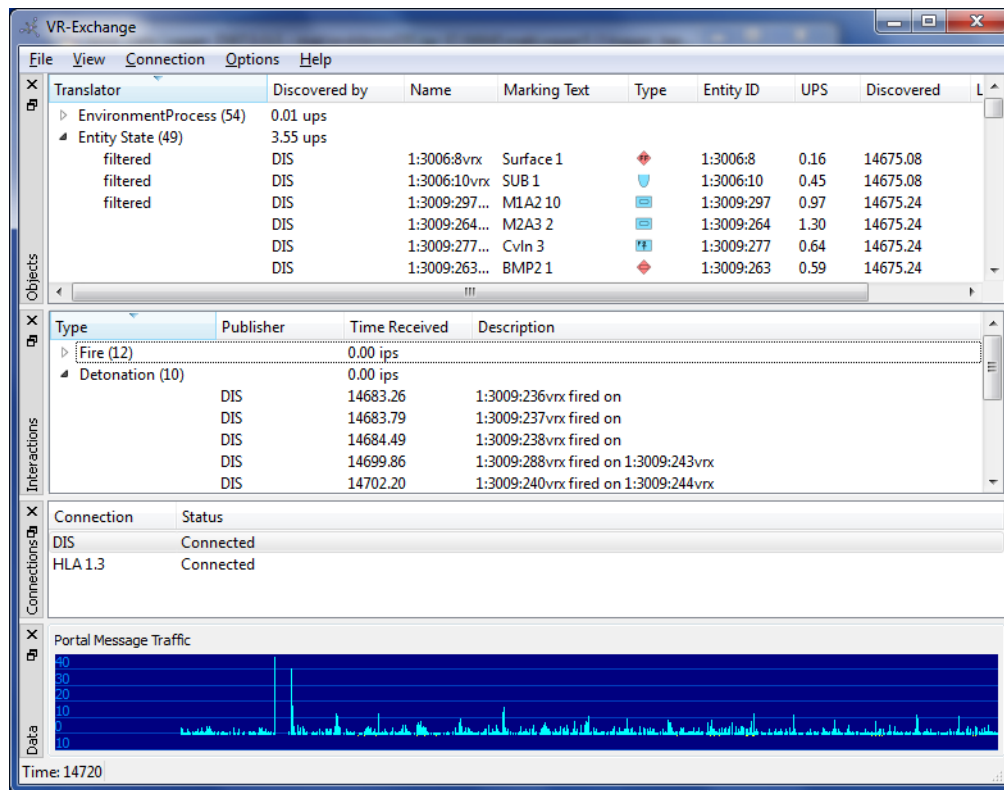


Figure 1-4. VR-Exchange Portal window

1.1.4. FOM-Agility and LROM-Agility

VR-Exchange brokers accommodate different FOMs and LROMs using VR-Link FOM Mappers and LROM Mappers. Mappers are dynamic link libraries that map the objects and concepts of a particular FOM or LROM to the VR-Link internal object model. Since the VR-Exchange brokers are VR-Link applications, they can use these mappers to switch between different FOMs and LROMs. For details about FOM and LROM mapping, please see *VR-Link Developers Guide*.

1.2. The VR-Exchange API

The VR-Exchange API has two primary components, the Broker API and the Portal API. The Broker API lets you create brokers for use with VR-Exchange or extend the brokers provided by MÄK. The Portal API allows brokers to send messages to and receive updates from, the Portal. For details, please see the class documentation.

1.3. Brokers Supplied

VR-Exchange includes brokers for HLA, TENA, and DIS. It has HLA brokers for the HLA 1.3 specification, for the SISO Standard Dynamic Link Compatible C++ API for IEEE 1516 specification, and for the HLA Evolved (IEEE 1516-2010) specification. Subject to licensing, you can run as many instances of these brokers as you want. The HLA brokers support the versions of the RPR FOM supported by VR-Link, or any FOM that has a VR-Link FOM Mapper.

The TENA broker allows VR-Exchange to participate in TENA executions. It uses LROM Mappers to map TENA objects to the VR-Exchange common simulation representation. VR-Exchange provides a common object LROM Mapper and a MÄK LROM Mapper. For details, please see [“The TENA Broker,”](#) on page A-11.

The DIS broker receives DIS 2.0.4, IEEE 1278.1, and IEEE 1278.1a PDUs. It sends IEEE 1278.1 PDUs.

[Table 1-1](#) lists the translation capabilities of each broker. For more information about translators, please see [“HLA Translators,”](#) on page A-3, [“TENA Translators,”](#) on page A-12, and [“The DIS Broker,”](#) on page A-8.

Table 1-1: Broker translators

Translator	HLA	DIS	TENA
Acknowledge	X	X	X
Action Request	X	X	
Action Response	X	X	
Aggregate	X	X	X
Collision	X	X	X
Comment	X	X	X
Create Entity	X	X	
Data	X	X	X
Data Query	X	X	
Designator	X	X	X
Detonation	X	X	X
Emitter System	X	X	X
Entity State	X	X	X
Environment Process	X	X	X
Event Report	X	X	
Fire	X	X	X
Gridded Data	X	X	
IFF	X	X	X
Radio Receiver	X	X	X
Radio Signal	X	X	X
Radio Transmitter	X	X	X
Remove Entity	X	X	
Set Data	X	X	X
Start	X	X	X
Stop	X	X	X

1.4. Translating Between DIS and the RPR FOM

DIS exercises send updates at regular intervals, called heartbeats. HLA does not have this feature. VR-Exchange handles this difference as follows:

- ♦ VR-Exchange provides heartbeat updates for the appropriate PDUs when the time between HLA updates exceeds the heartbeat threshold.
- ♦ VR-Exchange deletes HLA objects when the time between DIS PDUs exceeds the timeout threshold ($2.5 \times \text{heartbeat}$). RPR FOM embedded system objects (radio transmitter/receiver and emitter system/beams) are deleted only if their host entity no longer exists.

VR-Exchange does not support the full set of PDUs in the 1278.1a specification. It supports the following PDUs (listed by protocol families):

- ♦ Entity Information and Entity Interaction PDUs
 - Entity State PDU
 - Collision PDU
- ♦ Warfare PDUs
 - Fire PDU
 - Detonation PDU
- ♦ Simulation Management PDUs
 - Start/Resume PDU
 - Stop/Freeze PDU
 - Acknowledge PDU
 - Action Request PDU
 - Action Response PDU
 - Data Query PDU
 - Set Data PDU
 - Data PDU
 - Event Report PDU
 - Comment PDU
 - Create Entity PDU
 - Remove Entity PDU
- ♦ Radio Communications PDUs
 - Transmitter PDU
 - Signal PDU
 - Receiver PDU

- ♦ Distributed Emission Regeneration PDUs
 - Electromagnetic Emission PDU
 - Designator PDU
 - IFF PDU
- ♦ Logistics PDUs
 - Service Request PDU
 - Resupply Offer PDU
 - Resupply Received PDU
 - Resupply Cancel PDU
 - Repair Complete PDU
 - Repair Response PDU
- ♦ Entity Management PDUs
 - Transfer Control Request PDU
- ♦ Synthetic Environment PDUs
 - Environmental Process PDU
 - Gridded Data PDU.
- ♦ Logger Control PDUs (for MÄK Logger)

The following sections describe the supported PDUs in greater detail.

1.4.1. Entity Information and Entity Interaction

The DIS Entity Information/Interaction PDUs are the Entity State PDU and the Collision PDU.

The Entity State PDU is translated into a RPR FOM object. The RPR FOM has broken down the Entity State PDU into an object class hierarchy rooted at the *BaseEntity* class with several levels of subclasses as shown in Table 1-2. The object class that is registered in the federation execution is dependent on the value of the entity type field. The default mapping between the entity type field and the RPR FOM object class is based on the entity type kind and domain values. For example, a DIS M1 tank would be instantiated as a *GroundVehicle* object in the RPR FOM, whereas a TOW missile would be a *MunitionEntity*.

The data fields of the Entity State PDU are translated to and from the corresponding attributes of the appropriate RPR FOM object class. The entity ID, entity type, and position attributes are required to send an Entity State PDU.

Table 1-2: BaseEntity Object Classes

Class1	Class2	Class3	Class4
BaseEntity	PhysicalEntity	PlatformEntity	Aircraft
			AmphibiousVehicle
			GroundVehicle
			MultiDomainPlatform
			Spacecraft
			SubmersibleVehicle
			SurfaceVessel
		Lifeform	Human
			NonHuman
		CulturalFeature	
		Expendables	
		Munition	
		Radio	
		Sensor	

The Collision PDU is translated into a RPR FOM *Collision* interaction. The *Collision* interaction is defined at a single level, with no inheritance.

1.4.2. Warfare

The DIS Warfare PDUs are the Fire PDU and the Detonation PDU, which are translated into RPR FOM *WeaponFire* and *MunitionDetonation* interactions, respectively. Each interaction class is defined at a single level, with no inheritance.

1.4.3. Simulation Management

The DIS Simulation Management (SIMAN) PDUs are translated into the corresponding RPR FOM interactions as shown in [Table 1-3](#).

Table 1-3: SIMAN Representation in the RPR FOM

DIS PDU	RPR FOM Interaction Class
Create Entity	CreateEntity
Remove Entity	RemoveEntity
Acknowledge	Acknowledge
Set Data	SetData
Data Query	DataQuery
Data	Data
Action Request	ActionRequest
Action Response	ActionResponse
Event Report	EventReport
Comment	Comment
Start/Resume	StartResume
Stop/Freeze	StopFreeze

1.4.4. Radio Communications

The DIS Radio Communications PDUs are the Transmitter, Receiver, and Signal PDUs. The Receiver and Transmitter PDUs are translated into RPR FOM objects as *RadioReceiver* and *RadioTransmitter* object classes, respectively. The RPR FOM has broken down the Radio Transmitter and Receiver protocols into an object class hierarchy as shown in [Table 1-4](#). The *EmbeddedSystem* root object class captures the common attributes that specify the objects as being attached to other objects. The specific object class that is used to register a radio object in the federation execution is dependent on the type of PDU received by VR-Exchange. The data fields of the Transmitter and Receiver protocol are translated to and from the corresponding attributes of the appropriate RPR FOM object class. The host entity ID or host object ID (mapping to an entity ID) is required to send a Radio Transmitter PDU.

Table 1-4: EmbeddedSystem Object Classes

Class1	Class2	Class3
EmbeddedSystem	Designator	
	EmitterSystem	
	RadioReceiver	
	RadioTransmitter	
	IFF	IffInterrogator
		alffTransponder

The Signal protocol is translated into one of several RPR FOM *RadioSignal* interaction classes. A base *RadioSignal* interaction is defined from which the *ApplicationSpecificRadioSignal*, *DatabaseIndexRadioSignal*, *EncodedAudioRadioSignal*, and *RawBinaryRadioSignal* are derived. The HLA signal interaction that VR-Exchange generates is based on the Encoding Class in the DIS Signal PDU.

Table 1-5: RadioSignal Interaction Class

Class1	Class2
RadioSignal	ApplicationSpecificRadioSignal
	DatabaseIndexRadioSignal
	EncodedAudioRadioSignal
	RawBinaryRadioSignal

1.4.5. Distributed Emission Regeneration

The Distributed Emission Regeneration protocol family consists of the Electromagnetic Emission (EE), Designator, and IFF protocols. The EE PDU is transformed into several RPR FOM object classes. The Designator and IFF PDUs are represented in the RPR FOM by a derivation of the *EmbeddedSystem* class as Designator and IFF respectively (see [Table 1-4](#)). The IFF object class is further broken down into the *IffTransponder* and *IffInterrogator* object classes. These classes allow subscription based filtering of IFF data.

The EE PDU translation is different from any of the other PDU transformations in that it is a many-to-one and one-to-many transformation. A separate RPR FOM object is created for each emitter system and emitter beam in an EE PDU. Conversely, the emitter system and emitter beam objects reflected to VR-Exchange must be combined into a single EE PDU (for the State Update). The emitter system data is transformed into an *EmitterSystem* object class. An *EmitterSystem* object is created for each emitter system record in the EE PDU. The *EmitterSystem* is derived from the *EmbeddedSystem* class that associates each *EmitterSystem* object with its host entity object. Based on the beam function, the EE PDU emitter beam data is transformed into either a *RadarBeam* (the default) or a *JammerBeam*. The *EmitterBeam* class contains all of the emitter beam data parameters. Similar to the *EmbeddedSystem*, the *EmitterBeam* defines an *Emitting-SystemID* that establishes the association between it and its *EmitterSystem* object instance.

VR-Exchange creates a Delta State EE PDU for each *EmitterSystem* or *EmitterBeam* reflection. VR-Exchange does not generate a delta EE PDU if the *EmitterSystem* has no active beams (unless that beam is being removed). VR-Exchange must also first receive the *EmitterBeam's* *EmitterSystem* data as well as the emitting entity data. The emitting system ID and beam ID are required for sending an EE PDU. VR-Exchange generates the State Update EE PDU at a heartbeat rate (regardless of intermediate delta updates). The State Update EE PDU contains the emission data for each emitter with active emitter beams.

1.4.6. Logistics

The DIS Logistics PDUs are translated into RPR FOM interaction classes as shown in [Table 1-6](#).

Table 1-6: Logistics Representation in the RPR FOM

DIS PDU	RPR FOM Interaction Class
Service Request	ServiceRequest
Resupply Offer	ResupplyOffer
Resupply Received	ResupplyReceived
Resupply Cancel	ResupplyCancel
Repair Complete	RepairComplete
Repair Response	RepairResponse

The object ID attributes (mappable to DIS IDs) for the participating entities are required to send these PDUs.

1.4.7. Entity Management

The Transfer Control Request PDU is translated to the HLA *TransferControl* interaction class. The *TransferControl* class is available only if you are using RPR FOM version 2, draft 6 or later.

1.4.8. Synthetic Environment

VR-Exchange supports two PDUs in the Synthetic Environment family: Environmental Process PDU and Gridded Data PDU. The Environmental Process PDU is translated to the *EnvironmentProcess* class. The Gridded Data PDU is translated to the *GriddedData* class.

1.4.9. MÄK Logger Control PDUs

In addition to the standard DIS PDUs described in previous sections, VR-Exchange supports Logger Control PDUs for controlling the MÄK Logger.

1.5. Hardware Requirements and Product Limitations

VR-Exchange was designed to work with up to 254 connections. However, the requirement that all brokers run on the same machine limits the actual number of connections that you can run. VR-Exchange has been tested with as many as 4 connections.

It is recommended that VR-Exchange be run with one processor for each connection. Though this is not required, and much testing has been done on single processor machines, a typical instance of VR-Exchange will have three or more processes competing for CPU time.

It is also recommended that each protocol supported exist on a dedicated network. This is not a requirement of VR-Exchange, but is recommended for high traffic exercises.

1.6. The Limitations of Translation Software

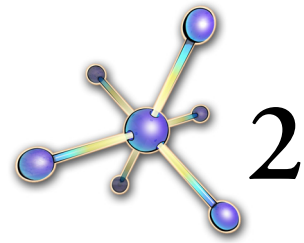
Translating between two protocols, or even two FOMs (in the case of HLA) is similar to translating between two spoken languages, in that you may lose something in translation. The goal of a good translator is to try to minimize that loss. When translating between spoken languages, you cannot simply perform word for word replacement. The source text needs to be examined holistically, and translated so that it conveys the same meaning to the target language. Sometimes idiomatic structures are not directly translatable and can only be approximated in the target language. And some concepts present in one language may not even exist in the other. As a result, some precision may be lost.

This same limitation occurs with all software translators. Messages often cannot be translated on a packet by packet basis; rather a stream of messages as a whole must be examined and translated. Some protocols map easily to each other. Others do not. Some protocols have no valid translation at all.

In the case of DIS to HLA translation, or even FOM-to-FOM translation, there is not always a one-to-one mapping between PDUs received from the DIS side and updates sent to the HLA side (or vice versa). This is due to fundamental differences in the rules and conventions of HLA versus DIS, and to differences in the ways different FOMs choose to represent similar concepts. For example, the DIS broker needs to send DIS heartbeat PDUs every 5 seconds, even though it may not receive HLA updates for an object for a much longer period of time. A single DIS PDU might contain data that must be translated into a series of attribute updates. For example, in DIS, a single Electromagnetic Emission PDU might contain information about a series of emitter systems and beams, but in HLA's RPR FOM, each emitter system and beam exists as a separate object, and must be updated through separate streams of messages.

In general, the strategy that VR-Exchange uses is to decode incoming data into state repositories that always contain the current state of each object. State repository information is passed to the other side of the bridge each frame, independently of when actual updates might have arrived. Then, the publishing rules of the other protocol and/or FOM are used to determine when to send out data describing the object. A consequence of this strategy is that no one piece of code is ever looking at both an incoming message and an outgoing message at the same time. While this strategy is necessary to support FOM agility and generic protocol-translation in VR-Exchange, a limitation is that “something” can get lost in the translation. For example, you should not expect that the timestamp on an outgoing HLA update will match the timestamp of a particular incoming DIS PDU.

Translators are very valuable solutions to real-world interoperability problems, and they often offer a much cheaper and easier solution than re-tooling applications to use a common protocol. But they can never quite provide the same level of close-coupling that you can get if all of your systems communicated natively. MÄK works hard to insure that our translators retain as much of the meaning and semantics of the data as is feasible, given the fundamental differences in the various protocols. But users must be aware that, as with any type of translator, there will always be limitations.



Installing VR-Exchange

This chapter explains how to install VR-Exchange and associated software.

Installing VR-Exchange.....	2-2
Installing VR-Exchange on Windows	2-2
Installing VR-Exchange on a Linux System	2-2
Installing and Setting Up the MÄK License Manager.....	2-3
Specifying the License Server	2-4
Installing an RTI.....	2-6
Installing the MÄK RTI	2-6
Installing TENA Software.....	2-7

2.1. Installing VR-Exchange

A full VR-Exchange installation includes three components: the application, the License Manager software, and for HLA operation, an RTI (or Runtime Infrastructure.)

Before you install VR-Exchange, please read *VR-Exchange Release Notes* to see if there are any special instructions for installation.

2.1.1. Installing VR-Exchange on Windows

Note the following:

- ♦ You must have administrator privileges to install MÄK products on Windows Vista.
- ♦ When you install large applications on Vista, there may be a delay of up to several minutes from the time you try to run the setup program to the time that an installation dialog box is displayed. This is due to how Vista scans setup programs before it executes them. If you experience this problem, turning off User Access Control can reduce or eliminate this delay.

Windows versions of VR-Exchange are provided as executable installer files on CD or DVD, or as downloaded files. The installers are named to indicate the compiler used to build that version of VR-Exchange and whether it is a 32 bit or 64 bit version.

- To install VR-Exchange, run the installer. Follow the instructions in the installation wizard.

2.1.2. Installing VR-Exchange on a Linux System

Before you install VR-Exchange, please read *VR-Exchange Release Notes* to see if there are any special instructions for installation.

Linux versions of VR-Exchange are provided as compressed tar files on CD or DVD, or as downloaded files.

To install VR-Exchange on Linux:

1. Create the directory in which you want to install VR-Exchange.
2. Copy the tar file to the install directory.
3. Uncompress and untar the file:

```
tar -vxzf application.tar.gz
```

where *application* is the product release identification.

2.2. Installing and Setting Up the MÄK License Manager

Before you can use a MÄK product, you must obtain a valid license file, install the MÄK License Manager, and configure the license server and client machines. The License Manager uses a client-server architecture, so you do not need to install the License Manager on every computer on which you install MÄK products. You only need to install it on the computer that you will use as the license server.



If you have already installed the License Manager for another MÄK product, you do not have to install it again. You just need to make sure you have licenses for your newly installed products.

The License Manager installer is included on MÄK installation media. It is separate from the product installers. You can download the installers from our web site at <http://www.mak.com/license-support.html> or you can download directly from:

- ♦ Windows: <ftp://ftp.mak.com/out/MAKLicenseManager-win-setup.exe> (32 bit systems)
<ftp://ftp.mak.com/out/MAKLicenseManager-win64-setup.exe> (64 bit systems)
- ♦ Linux: <ftp://ftp.mak.com/out/MAKLicenseManager-linux-setup.tar.gz>

Complete installation and configuration instructions are included with the License Manager installer. Instructions are also available at the license support page.

Some customers use dongle licenses instead of running a license server. Instructions for using dongles are in the License Manager documentation.

2.2.1. Specifying the License Server

The first time you run a MÄK application on a particular computer, the License Setup dialog box opens (Figure 2-1). It prompts you to enter the hostname of the license server and optionally, a port number.

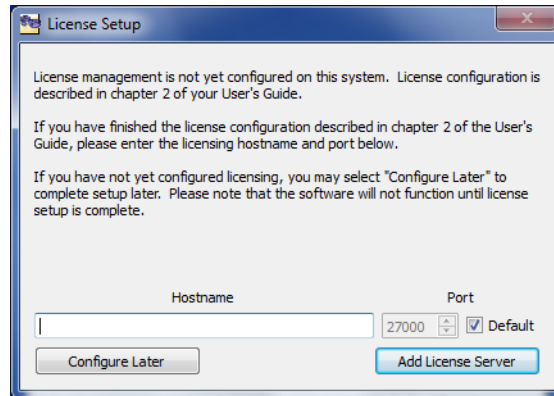


Figure 2-1. License Setup dialog box

If you do not know the hostname of the license server, click **Configure Later**. When you have the hostname, you can start the application again and complete the dialog box. You will not be able to run any MÄK applications until you set up license management.

If you know the hostname, type it in the **Hostname** box. Then click **Add License Server**. The application will start.

i

- ♦ If you are running MÄK products on the license server machine, it is also a client, so you must specify the license server on that machine too.
 - ♦ If you change the license server, the saved configuration will no longer be valid and the License Setup dialog box will open the next time you start a MÄK application.
 - ♦ You can clear the saved license configuration by deleting the cache file. On Windows XP, it is *C:\Documents and Settings\user_name\Application Data\MAK\license<n>.xml*. On Windows 7, it is *C:\Users\user_name\AppData\Roaming\MAK\licenses<n>.xml*. (The *AppData* directory is hidden by default.) On Linux, it is *.mak/license<n>.xml*. (There may be more than one cache file, for example, *licenses1.xml* and *licenses2.xml*.)
-

The MAKLMGRD_LICENSE_FILE Environment Variable

As an alternative to using the License Setup procedure described in the previous section, you can configure the license server in an environment variable. The MAKLMGRD_LICENSE_FILE environment variable identifies the server machine. If you set this environment variable, it overrides the settings stored by the License Setup procedure.

The syntax for the environment variable is: `@Server_name`. For example, if the server machine is oak, set the environment variable to `@oak`.

The following sections explain how to set environment variables on the different platforms that MÄK products run on.

Windows

To add the MAKLMGRD_LICENSE_FILE in Windows:

1. Open the Control Panel.
2. Click System and Security.
3. Click System.
4. In the sidebar menu, click the Advanced System Settings. The System Properties dialog box opens.
5. Click the Environment Variables button. The Environment Variables dialog box opens.
6. Click the New button. The New System Variable dialog box opens.
7. In the Variable Name field, enter MAKLMGRD_LICENSE_FILE.
8. In the Variable Value field, enter `@server_name`, where `server_name` is the name of the license server.
9. Click OK to back out of each dialog box and set the variable.



Different versions of Windows have slightly different wording of the various Control Panel options.

Linux

On Linux, you set environment variables in your `.cshrc` (or equivalent startup file). Set the variable similarly to the following example:

```
setenv MAKLMGRD_LICENSE_FILE @oak
```

If you are using the sh or bash shells, you set environment variables in your `.profile` file (or `.bashrc`). Set the variable similarly to the following example:

```
MAKLMGRD_LICENSE_FILE=@oak
export MAKLMGRD_LICENSE_FILE
```

Do not put spaces around the equal (=) sign.

You are ready to run the license server and use your new licenses or MÄK products.

2.3. Installing an RTI

An RTI is a software library (and perhaps supporting executables) that implements an HLA Interface Specification. In HLA, applications exchange FOM data through RTI calls, which means that all HLA applications need to use an RTI.



Because of differences in the low-level network mechanisms used by different RTI implementations (which include, but are not limited to packet layout), applications that want to interoperate in the same federation execution must use the same RTI implementation.

Because RTIs are usually provided as dynamic libraries that implement a fixed API, a federation can often switch from one RTI implementation to another between runs (without even recompiling the applications), but during each run, all participants must agree on which RTI to use, much as they must also agree on which FOM to use.

For the most recent information about the RTI versions supported by MÄK products, please see the release notes for your MÄK application.

2.3.1. Installing the MÄK RTI

To install the MÄK RTI, follow the instructions in Chapter 2 of *MÄK RTI Users Guide*.

Configuring Your System to Use the MÄK RTI

The RTI dynamic libraries must be located somewhere on your dynamic library search path. The path is specified in the PATH environment variable. The path is indicated by an environment variable as follows:

- ♦ Linux — LD_LIBRARY_PATH
- ♦ Windows — PATH.

The MÄK RTI needs to know where to find the following configuration files:

- ♦ The federation configuration file (*.fed*, *.fdd*, or *.xml*) for your federation execution (required)
- ♦ The RID file (*rid.mtl*) (optional).

Put the configuration files in the directory from which you are running, or set the environment variable RTI_CONFIG to the directory that contains them.

Running Applications with the MÄK RTI

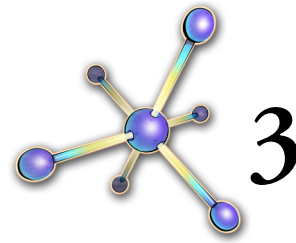
In many cases, you do not need an RTI server, such as the rtiexec, or a central application to use the MÄK RTI, however you can run the rtiexec if you want to. (It is required to use certain features of the MÄK RTI.) Be sure the license server is running, then start the applications, and they should be able to communicate. For more information, please see your RTI documentation.

2.4. Installing TENA Software

If you plan to use the TENA broker, you must install the TENA Middleware and Object Model dynamic libraries. MÄK cannot distribute the majority of these files. They are under the control of the TENA SDA and you must get them from www.tena-sda.org. The software required to participate in a TENA execution is categorized as follows:

- ♦ TENA Middleware is the basic software required to run a TENA execution. It includes the Network Naming Service, Execution Manager, and some of the required dynamic libraries.
- ♦ The MÄK Object Model includes some of the dynamic libraries required to publish and subscribe to TENA messages and objects from the Common Object LROM and the MÄK LROM.
- ♦ The MÄK Object Model Basic Implementation is a set of dynamic libraries that is traditionally built by the end-user using code downloaded from the MÄK Object Model. These files are included in the installation of VR-Exchange.

For details about installing TENA software and the MÄK Object Model, please see Appendix B, [A VR-Exchange/TENA Primer](#).



Using VR-Exchange

This chapter explains how to start the Portal and use it to manage connections.

Starting VR-Exchange.....	3-2
Starting VR-Exchange on Linux	3-2
Starting VR-Exchange on Windows	3-3
VR-Exchange Command-Line Options.....	3-4
The Portal Application	3-5
Running VR-Exchange without the Portal Application	3-5
Configuring VR-Exchange.....	3-6
Enabling and Disabling Connections.....	3-8
Enabling Connections when the Portal Starts	3-8
Enabling Connections During Runtime	3-9
Disabling Connections	3-9
Viewing Objects and Interactions	3-10
Customizing the Display of Object and Interaction Data	3-11
Enabling and Disabling the Display of Object Data	3-11
Sorting the Object List by Column	3-11
Enabling and Disabling the Display of Interactions.....	3-11
Clearing the Interactions List	3-12
Filtering Objects	3-13
Displaying the Queue Status	3-14
The Data View	3-14
Closing VR-Exchange	3-15
Using vrxMessageDump	3-15

3.1. Starting VR-Exchange

This section explains how to start VR-Exchange on the different operating systems it supports.



All VR-Exchange components (the Portal and brokers) must run on the same computer.

3.1.1. Starting VR-Exchange on Linux

To start VR-Exchange on Linux:

1. In a command shell, change directory to *.bin*.
2. Enter:

```
vrx [command-line_options]
```

The VR-Exchange Portal window opens ([Figure 3-1](#)). The VR-Exchange command-line options are described in “[VR-Exchange Command-Line Options](#),” on page 3-4.

3.1.2. Starting VR-Exchange on Windows

- To start VR-Exchange on Windows, on the Start menu, choose **All Programs** → **MÄK Technologies** → **VR-Exchange 2.1** → **VR-Exchange (32-bit)** (or 64-bit).

The VR-Exchange Portal window opens (Figure 3-1).

If you want to start VR-Exchange with command-line options, start it from a command prompt or create a shortcut icon for it on your desktop. The VR-Exchange command-line options are described in “VR-Exchange Command-Line Options,” on page 3-4.

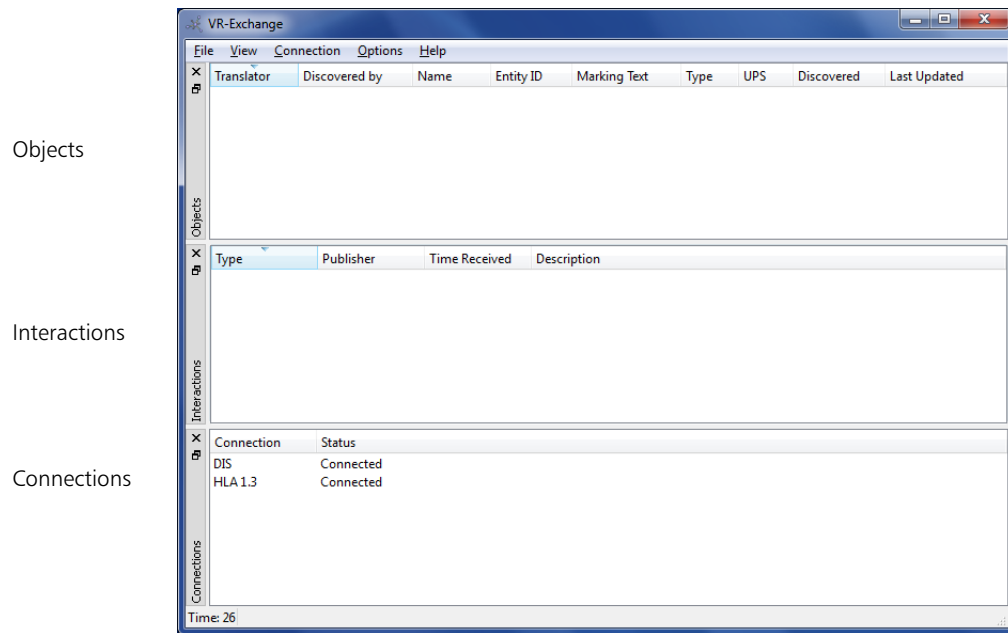


Figure 3-1. VR-Exchange Portal window

3.1.3. VR-Exchange Command-Line Options

The VR-Exchange command-line options override settings in *VR-Exchange.xml*. [Table 3-1](#) describes the command-line options

Table 3-1: VR-Exchange Portal command-line options

Option	Description
(-- --ignore_rest)	Tells VR-Exchange to ignore the arguments following this flag.
--controlQueueName queueName	Specifies the name of the control shared memory queue.
--dontStartBrokers	Specifies that VR-Exchange not enable the connections listed in <i>VR-Exchange.xml</i> . If you use this option, you must start the connections manually.
(-G --noGui)	Specifies console mode (no GUI).
(-h --help)	Displays usage information and exits.
--interactionQueue- Name queueName	Specifies the name of the interaction shared memory queue.
(-l --configFile) filename	Specifies a configuration file other than <i>VR-Exchange.xml</i> .
(-L --logFilename) filename	Specifies a log file.
--maxInteractionsTo- Show count	Specifies the maximum number of interactions to show in the Portal application.
--maxPortalControlMes- sages count	Specifies the maximum number of control messages to process per tick.
--maxPortalInterac- tionMessages count	Specifies the maximum number of interaction messages to process per tick.
--maxPortalObjectMes- sages count	Specifies the maximum number of object messages to process per tick.
(-n --notifyLevel) notify_level	Specifies the notification level for logging messages. Range: 0-4, where: ♦ 0 – Fatal error messages ♦ 1 – Warning messages ♦ 2 – Informational messages ♦ 3 – Verbose information ♦ 4 – Debugging data
--objectQueueName queueName	Specifies the name of the object shared memory queue.
--queueBucketCount count	Specifies the number of buckets inside the shared memory queue.
--queueBucketPayload- Size size	Specifies the size of buckets inside the shared memory queue.

Table 3-1: VR-Exchange Portal command-line options

Option	Description
<code>--tcpPort port_number</code>	Specifies the TCP port for connectivity tests.
<code>(-v --version)</code>	Displays version information and exits.

3.1.4. The Portal Application

The Portal application houses several dockable windows. You can drag any of them off of the Portal and place them anywhere on your desktop. You can close any of the windows and open them to suit your viewing preferences.

The Portal has context-sensitive menus. The windows have a context-sensitive menu that matches the View menu. The individual connections in the Connections window have context-sensitive menus that match the Connection menu.

3.1.5. Running VR-Exchange without the Portal Application

You can run VR-Exchange without the Portal application. It automatically starts any connections that are configured to start when the Portal starts.

To run VR-Exchange without the Portal application:

1. Open `./bin/VR-Exchange.xml` in a text editor.
2. Set the value of the `noGui` attribute to true.

```
<!--Disable Portal GUI-->
<noGui value="true"/>
```
3. Save the file.

3.2. Configuring VR-Exchange

VR-Exchange has several parameters that you can set to optimize performance. Preferences are saved to *VR-Exchange.xml*.

To set VR-Exchange preferences:

1. Choose **Options** → **Preferences**. The VR-Exchange Preferences dialog box opens (Figure 3-2).

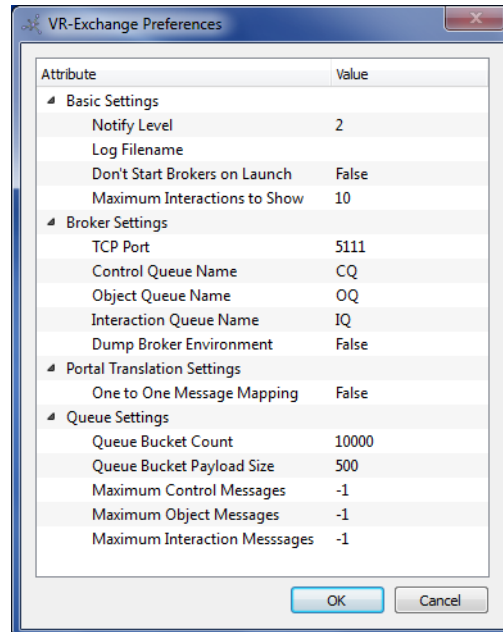


Figure 3-2. VR-Exchange Preferences dialog box

2. Expand the appropriate group of settings.
3. Click the value for the attribute that you want to change. The value becomes editable.
4. Edit the value. Some values can be chosen from a drop-down list. Others are text or numeric values that you edit directly.
5. Click OK. Changes do not take effect until you restart VR-Exchange.

Table 3-2 describes the attributes in the VR-Exchange Preferences dialog box.

Table 3-2: VR-Exchange preferences

Attribute	Description
Basic Settings	
Notify Level	Specifies the output level for the Portal's log file. Default: 2.

Table 3-2: VR-Exchange preferences

Attribute	Description
Log Filename	Specifies the log file for messages.
Maximum Interactions to Show	Specifies the maximum number of interactions to show in the Interactions window for each interaction type. When the Portal runs for a long time, the interaction list can become very long, which can degrade performance. If this number is set to -1 all interactions are shown. If it is set to any positive number, only the most recent maximum number of interactions is displayed. Default: 10.
Don't Start Brokers on Launch	Specifies that connections should not be enabled automatically when VR-Exchange starts.
Broker Communication Settings	
TCP Port	Specifies the TCP port to use to test broker lifetime. Each broker connects to this port. If the Portal crashes, the brokers will know and then shut down. If TCP Port is set to 0, this feature is disabled.
Object Queue Name Control Queue Name Interaction Queue Name	Specifies the names of the object, control, and interaction queues in the shared memory pool. If you run more than one instance of VR-Exchange, the queue names must be different for each Portal.
Dump Broker Environment	Print the brokers environment to a file at startup.
Portal Translation Settings	
One to One Message Mapping	Specifies whether or not to generate an outbound message for each inbound message. Changing this setting requires restarting all brokers.
Queue Settings	
Queue Bucket Count	Specifies the maximum number of messages in the shared memory queue.
Queue Bucket Payload Size	Specifies the maximum message size, in bytes.
Maximum Control Messages Maximum Interaction Messages Maximum Object Messages	Specifies the maximum number of Portal messages to process per call to drainInput(), where -1 means process all messages. Adjust these parameters to avoid process starvation if one of the queues is heavily used.

3.3. Enabling and Disabling Connections

The brokers are executable files that are independent of the Portal. You start them using connection configurations that specify a broker, a connection name, and configuration parameters. You can enable a connection automatically when the Portal is started or you can enable a connection manually. For details about adding and configuring connections, please see Chapter 4, [Working with Broker Connections](#).



The TENA broker will not run if TENA is not installed. If TENA is not installed, you will be able to add a connection that specifies the TENA broker, but you will not be able to configure the connection or enable it. If you want to use the TENA broker, it is your responsibility to ensure that all required TENA files are installed. For more information, please see Appendix B, [A VR-Exchange/TENA Primer](#).

3.3.1. Enabling Connections when the Portal Starts

To specify that a connection be enabled when the Portal starts:

1. Start VR-Exchange.
2. In the Connections window ([Figure 3-1](#)), select the connection that you want to configure.
3. If the connection is enabled, choose **Connection** → **Disable Connection**. The connection is disabled.
4. Choose **Connection** → **Edit Connection**. The Edit Connection dialog box opens ([Figure 3-3](#)).

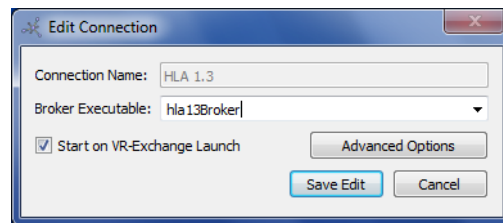


Figure 3-3. Edit Connection dialog box

5. Select the Start on VR-Exchange Launch check box.
6. Click Save Edit.
7. Enable the connection, if applicable.

3.3.2. Enabling Connections During Runtime

To enable a connection after you start the Portal:

1. In the Connections window ([Figure 3-1](#)), select the connection that you want to enable.
2. Choose **Connection** → **Enable Connection**. The status changes to Connected.

3.3.3. Disabling Connections

Connections are disabled and brokers are shut down automatically when the Portal is shut down. Connections can also be disabled independently. If a broker crashes, or otherwise exits, and if the broker is connected using TCP, the Portal removes any connections that use it.

To disable a connection:

1. In the Connections window ([Figure 3-1](#)), select the connection you want to disconnect.
2. Choose **Connection** → **Disable Connection**.

3.4. Viewing Objects and Interactions

VR-Exchange displays lists of the objects and interactions that it discovers, which connection is publishing them, and statistics about them (Figure 3-4). As objects join and leave the exercise, VR-Exchange updates the display so that it only shows currently simulated objects. You can configure the number of interactions kept in the interaction list. (For details, please see “Configuring VR-Exchange,” on page 3-6.)

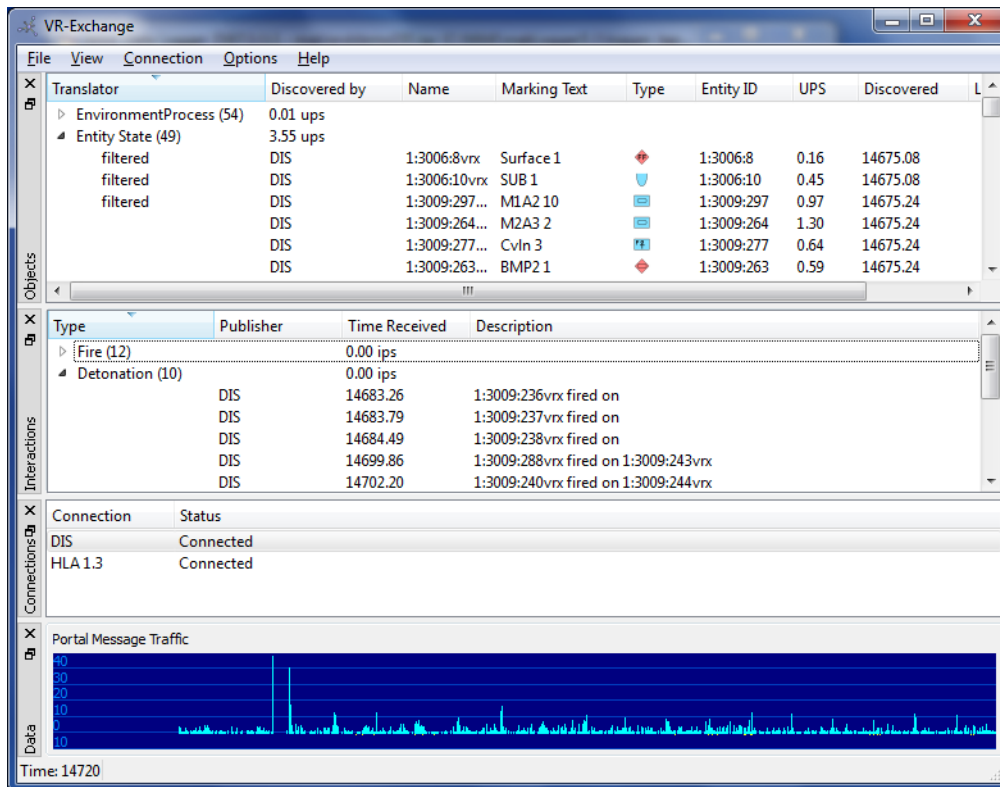


Figure 3-4. Object and interaction display



If you mouse over the icon for an entity in the Type column, VR-Exchange displays a tooltip with the entity type enumeration for that object.

3.4.1. Customizing the Display of Object and Interaction Data

You can choose which columns of data to display and you can change the order in which columns are displayed.

To show or hide a column:

1. In the Objects window or Interactions window (Figure 3-4), right-click any column label. A list of columns is displayed.
 2. To display a hidden column or hide a currently displayed column, click its entry in the list.
- To reorganize the order of data columns, drag the column title to the location where you want to display it.

3.4.2. Enabling and Disabling the Display of Object Data

You can enable and disable display of the following types of object data:

- ♦ Objects – show or hide all objects
 - ♦ Updates – show or hide the Updated column.
- To enable or disable display of object data, choose **View** → **Objects** → *option*, where *option* is one of the following:
- Show Objects
 - Show Updates.

3.4.3. Sorting the Object List by Column

- To sort the object list by column, click a column heading.

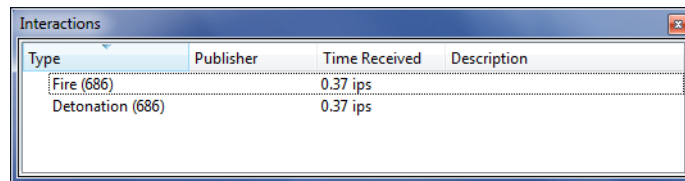
3.4.4. Enabling and Disabling the Display of Interactions

You can hide or display the interaction list.

- To enable or disable the display of interactions, choose **View** → **Interactions** → **Show Interactions**.

3.4.5. Clearing the Interactions List

You can clear the Interactions list. When you clear the interactions list, VR-Exchange continues to maintain a running count of the number of interactions. [Figure 3-5](#) illustrates the interactions list after it is cleared. Notice the cumulative statistics next to Fire and Detonation.



Type	Publisher	Time Received	Description
Fire (686)		0.37 ips	
Detonation (686)		0.37 ips	

Figure 3-5. Cleared Interactions list

- To clear the Interactions list, choose **View** → **Interactions** → **Clear Interactions**.

Resetting Statistics

You can clear the cumulative statistics maintained by the Interactions window.

- To clear interaction statistics, choose **View** → **Interactions** → **Reset Statistics**.

3.5. Filtering Objects

You can filter out entire classes of objects (for details, please see [“Specifying Which Translators to Use,”](#) on page 4-26). You can also filter individual objects. You might do this if you are interested in particular objects and want to reduce network traffic by filtering out the objects you are not interested in. You can filter objects by name, entity type, and marking text. Filtering by entity type and marking text only applies to entities and aggregates.

This section explains how to filter out objects from the Objects window. You can also set up filters in the Connection Information dialog box for a broker. For details, please see [“Filtering Objects,”](#) on page 4-23.

To filter out an object:

1. In the Objects window, select the objects that you want to filter.
2. Right-click the objects. A context-sensitive menu is displayed.
3. Choose **Add Filter**. A submenu is displayed.
4. Choose the type of filter that you want to add. The word “filtered” is displayed in the Translators column for each filtered object ([Figure 3-6](#)).

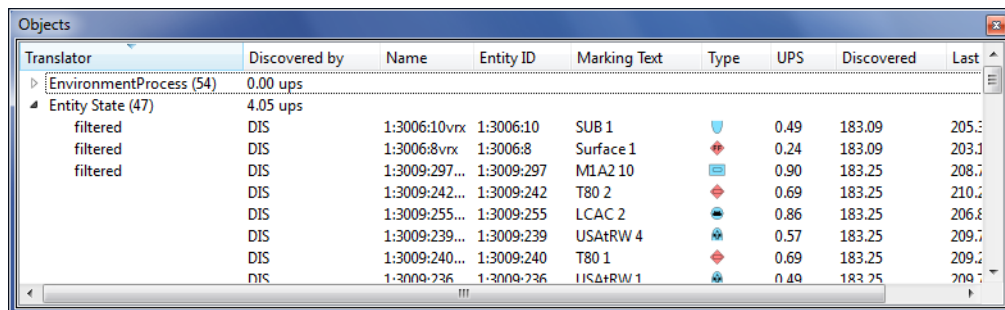


Figure 3-6. Filtered objects

To remove a filter:

1. In the Objects window, select the objects that you want to unfilter.
2. Right-click the objects. A context-sensitive menu is displayed.
3. Choose **Remove Filter**. A submenu is displayed.
4. Choose the type of filter that you want to remove. The filter is removed and the “filtered” text in the Translator column is cleared.

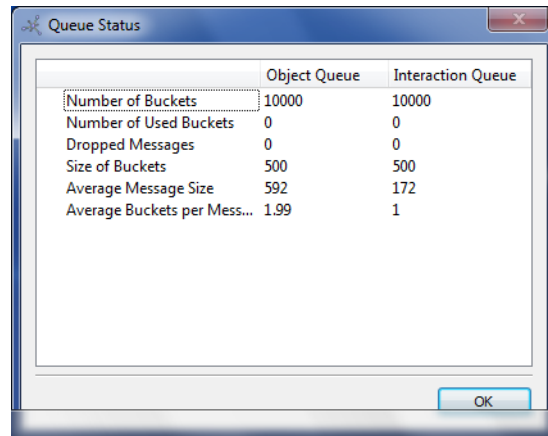


Only filters created from the context-sensitive menu can be removed from the context-sensitive menu. Filters created on the Filters page of the Connection Information dialog box cannot be removed from the context-sensitive menu.

3.6. Displaying the Queue Status

You can display details about the shared memory queue.

- To display the queue status, choose **View** → **Queue Status**. The Queue Status dialog box opens (Figure 3-7).

A screenshot of the 'Queue Status' dialog box. It has a title bar with a close button. The main area contains a table with two columns: 'Object Queue' and 'Interaction Queue'. The rows show various statistics: 'Number of Buckets' (10000), 'Number of Used Buckets' (0), 'Dropped Messages' (0), 'Size of Buckets' (500), 'Average Message Size' (592), and 'Average Buckets per Mess...' (1.99). An 'OK' button is at the bottom right.

	Object Queue	Interaction Queue
Number of Buckets	10000	10000
Number of Used Buckets	0	0
Dropped Messages	0	0
Size of Buckets	500	500
Average Message Size	592	172
Average Buckets per Mess...	1.99	1

Figure 3-7. Queue Status dialog box

3.7. The Data View

You can display a histogram of the number of packets being processed by VR-Exchange (Figure 3-8). Blue lines (above the baseline) represent the number of object packets; yellow lines (below the baseline) represent the number of interaction packets; red lines represent VR-Exchange messages.

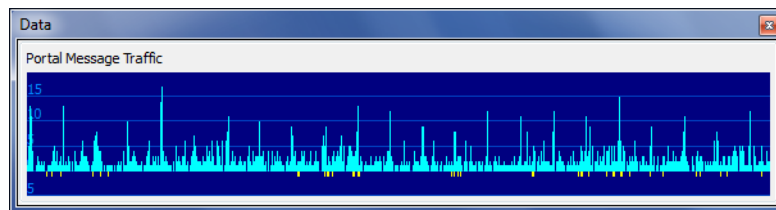


Figure 3-8. Data View

- To hide or display the Data View, choose **View** → **Show Data View**.

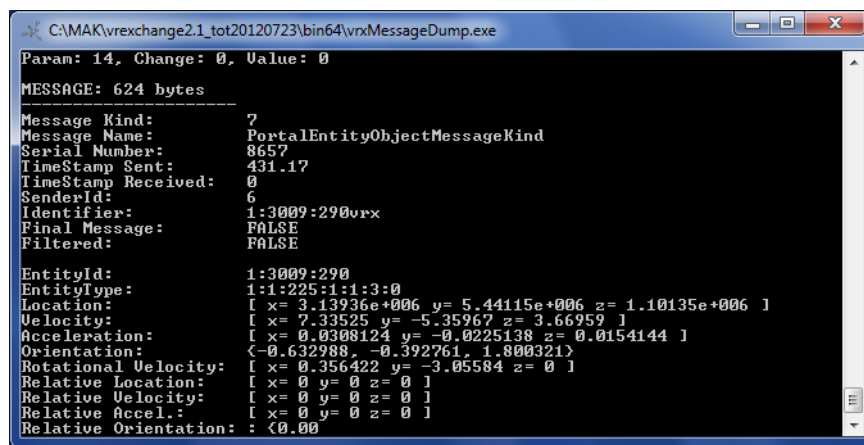
3.8. Closing VR-Exchange

- To close VR-Exchange, choose **File** → **Exit**.

3.9. Using *vrxMessageDump*

VR-Exchange includes a utility called *vrxMessageDump*. It displays all messages passing through shared memory in a console window (Figure 3-9). *vrxMessageDump* exits automatically when the Portal is shut down.

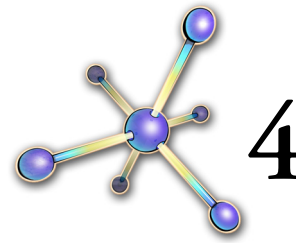
- To run *vrxMessageDump*, start VR-Exchange, then run the *vrxMessageDump* executable in the *./bin* directory.



```

C:\MAK\vrexchange2.1_tot20120723\bin64\vrxMessageDump.exe
Param: 14, Change: 0, Value: 0
MESSAGE: 624 bytes
-----
Message Kind:          7
Message Name:          PortalEntityObjectMessageKind
Serial Number:         8657
TimeStamp Sent:        431.17
TimeStamp Received:    0
SenderId:              6
Identifier:            1:3009:290vrx
Final Message:         FALSE
Filtered:              FALSE
EntityId:              1:3009:290
EntityType:            1:1:225:1:1:3:0
Location:              [ x= 3.13936e+006 y= 5.44115e+006 z= 1.10135e+006 ]
Velocity:              [ x= 7.33525 y= -5.35967 z= 3.66959 ]
Acceleration:          [ x= 0.0308124 y= -0.0225138 z= 0.0154144 ]
Orientation:           [ <-0.632988 -0.392761 1.800321>
Rotational Velocity:   [ x= 0.356422 y= -3.05584 z= 0 ]
Relative Location:     [ x= 0 y= 0 z= 0 ]
Relative Velocity:     [ x= 0 y= 0 z= 0 ]
Relative Accel.:       [ x= 0 y= 0 z= 0 ]
Relative Orientation:  : <0.00
  
```

Figure 3-9. *vrxMessageDump* display



Working with Broker Connections

This chapter explains how to add, edit, and configure connections.

Adding and Editing Connections.....	4-2
Adding Connections.....	4-2
Specifying Environment Variables for a Connection	4-4
Editing Connections	4-4
Deleting Connections.....	4-5
Configuring Connections	4-5
Specifying Collection Values.....	4-7
Common Broker Settings	4-9
Configuring Publication Conditions	4-10
Protocol Specific Attributes	4-12
Configuring Network Connections.....	4-18
HLA Network Connection Parameters.....	4-19
DIS Network Connection Parameters	4-20
TENA Network Connection Parameters	4-22
Filtering Objects	4-23
Editing a Filter	4-24
Removing a Filter	4-25
Specifying Which Translators to Use.....	4-26
Viewing Publishing and Reflecting Lists.....	4-27

4.1. Adding and Editing Connections

A connection runs a broker executable with a unique set of parameters. This allows you to, for example, run an instance of the HLA 1.3 broker with RTI A and another instance of the HLA 1.3 broker with RTI B.

A connection must have a name and it must specify the broker that it uses and a configuration file. It can include command-line options and can specify environment variables. For lists of command-line options, please see Appendix A, [Broker Details](#).

VR-Exchange ships with connections for HLA 1.3, DIS, and TENA. The HLA and DIS connections are started automatically when you start VR-Exchange. The TENA connection is not displayed unless you have TENA installed.



The TENA broker will not run if TENA is not installed. If TENA is not installed, you will be able to add a connection that specifies the TENA broker, but you will not be able to configure the connection or enable it. If you want to use the TENA broker, it is your responsibility to ensure that all required TENA files are installed. For more information, please see Appendix B, [A VR-Exchange/TENA Primer](#).

4.1.1. Adding Connections

To add a connection:

1. Choose **Connection** → **Add New Connection**. The Add New Connection dialog box opens ([Figure 4-1](#)).

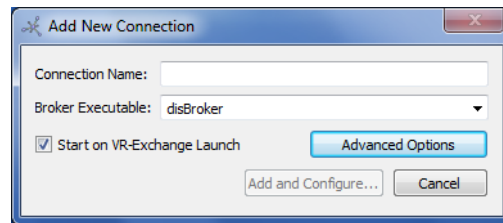


Figure 4-1. Add New Connection dialog box

2. In the Connection Name box, type a name for the connection.
3. In the Broker Executable box, select a broker from the drop-down list or type the name of a broker executable.
4. Optionally, specify advanced options, as follows:
 - a. Click Advanced Options. The dialog box expands ([Figure 4-2](#)).

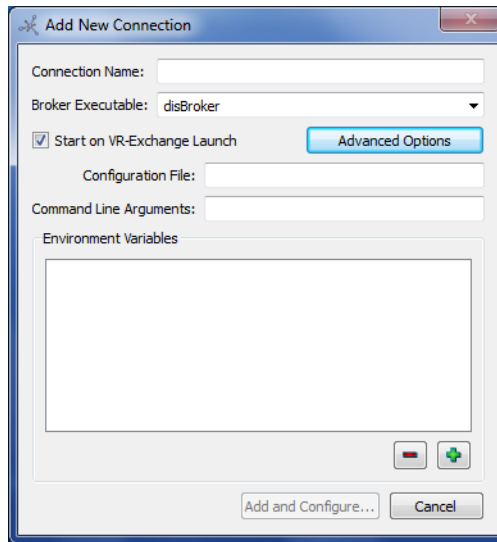


Figure 4-2. Add New Connection dialog box, Advanced Options

- b. In the Configuration File box, type a name for the connection configuration file. (VR-Exchange will append the extension *.bcf*.) If you do not specify a configuration file name, VR-Exchange uses the connection name to name the configuration file.
- c. Optionally, type command-line arguments in the Command Line Arguments box.
- d. Optionally, specify environment variables for this connection, as described in [“Specifying Environment Variables for a Connection,”](#) on page 4-4.
- e. Click Advanced Options to collapse the dialog box.
5. If you want to enable the connection when VR-Exchange starts, select the Start on VR-Exchange Launch check box.
6. Click Add and Configure. The Connection Information dialog box for this connection opens.
7. Optionally, edit configuration parameters for the new connection. For details, please see [“Configuring Connections,”](#) on page 4-5.

4.1.2. Specifying Environment Variables for a Connection

You can specify environment variables that only apply to a specific connection. You might want to do this to have different HLA connections use different RTIs.

To specify an environment variable for a connection:

1. In the Add New Connection dialog box or Edit Connection dialog box, click Advanced Options. The dialog box expands (Figure 4-2).
2. In the Environment Variables group box, click the Add button (+). The Environment Variable dialog box opens.
3. In the Variable Name box, type a name for the environment variable.
4. In the Variable Value box, type the value for the variable.
5. Click OK. The environment variable is added to the list.
6. Click Next.

4.1.3. Editing Connections

To edit a connection:

1. In the Connections window (Figure 3-1), select the connection that you want to edit.
2. If the connection is connected, choose **Connection** → **Disable Connection**.
3. Choose **Connection** → **Edit Connection**. The Edit Connection dialog box opens (Figure 4-3).

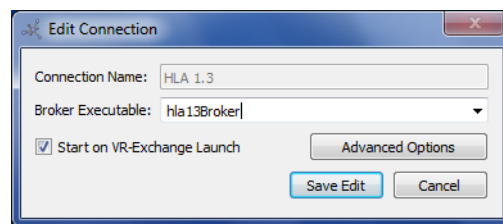


Figure 4-3. Edit Connection dialog box

4. Click Advanced Options to display all connection details.
5. Edit the connection as desired.
6. Click Next.

4.1.4. Deleting Connections

Deleting a connection deletes the association of a connection name with a broker and a configuration file. It does not affect the broker executable referenced by the configuration file and it does not affect the configuration file it references (if one exists).

To delete a connection:

1. In the Connections window ([Figure 3-1](#)), select the connection that you want to delete.
2. If the connection is connected, choose **Connection** → **Disable Connection**.
3. Choose **Connection** → **Remove Connection**.
4. Click OK. A confirmation dialog box opens.
5. Click OK.

4.2. Configuring Connections

This section describes the attributes you can configure on the Configure page of the Connection Information dialog box.

Each connection has a set of attributes that you can set. If you use the default attribute values, VR-Exchange does not create a configuration file for the connection. If you change any of the attributes, VR-Exchange saves them to the configuration file you specified when you created the connection.

To configure a connection:

1. In the Connections window ([Figure 3-1](#)), select the connection that you want to configure.
2. Choose **Connection** → **Edit Connection Information**, or double-click the connection. The Connection Information dialog box opens ([Figure 4-4](#)). (When you add a new connection, the Connection Information dialog box opens automatically after you create the connection.)

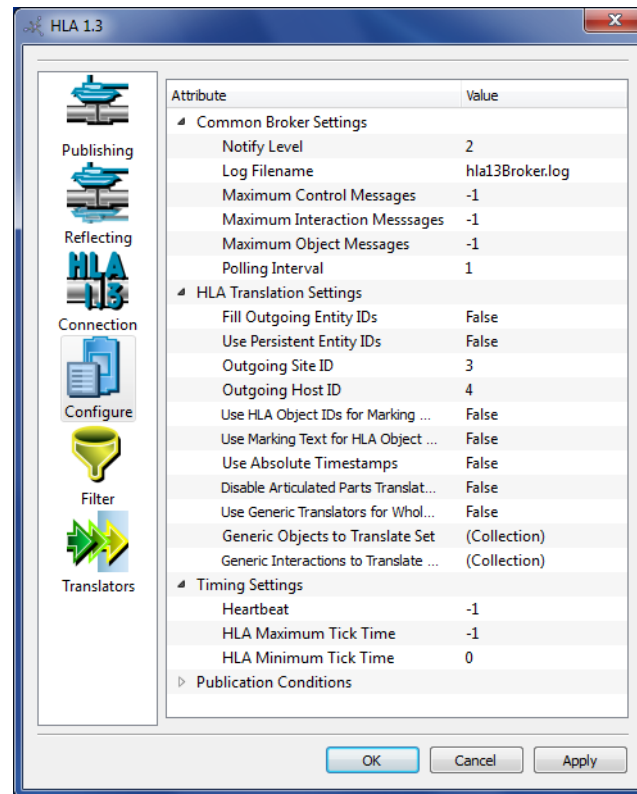


Figure 4-4. Connection Information dialog box

3. Select the Configure page. It lists attributes for the connection, organized in logical groupings, except for the network connection settings. These are configured on the Connections page.
4. Expand the attribute tree to find the attributes you want to edit.
5. To edit an attribute, click its value. The field becomes editable.
 - If the field has a single text or numeric value, change the value as desired. Some values are available on drop-down lists. The attributes are listed and described in later sections in this chapter.
 - If the attribute value is a collection, edit it as described in [“Specifying Collection Values,”](#) on page 4-7.
6. Click OK. If the connection was enabled, the changes do not take effect until you restart the connection. VR-Exchange prompts you to restart the connection.

4.2.1. Specifying Collection Values

Some attributes can take multiple values (collections).

To specify the values for a collection:

1. In the Configure page for a connection, expand the attribute list to display the collection attribute you want to edit.
2. Click the word Collection in the Value column (Figure 4-5). It changes to a button labeled Click to Edit.

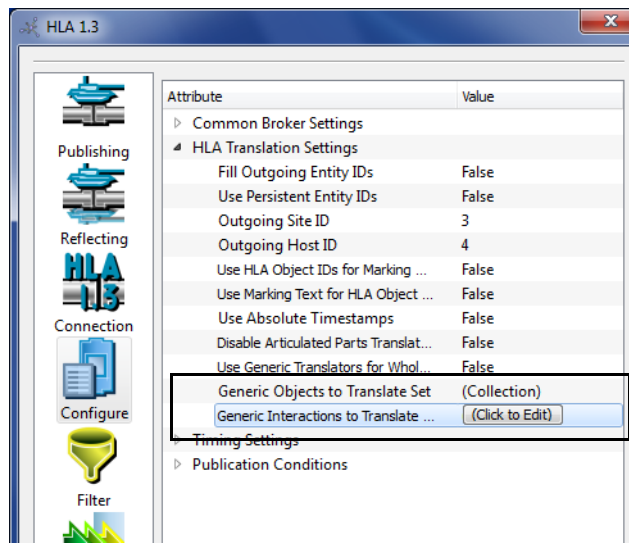


Figure 4-5. Collection value

3. Click the button. The Edit Collection dialog box opens (Figure 4-6).

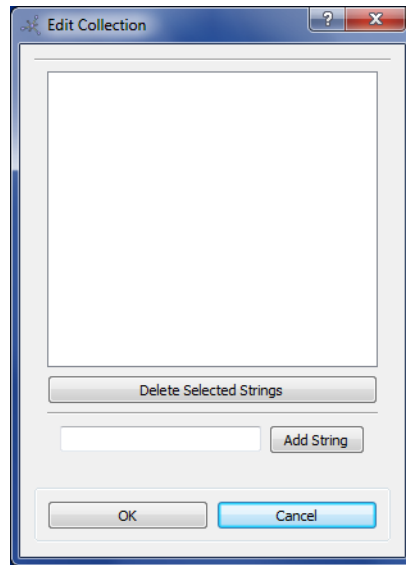


Figure 4-6. Edit Collection dialog box

4. In the text box, type the string you want to add.
5. Click Add String. The string is added to the list window.
6. Click OK.

Removing Values from a Collection

To remove values from a collection:

1. In the Configure page for a connection, click the word Collection in the Value column ([Figure 4-5](#)). It changes to the text Click to Edit.
2. Click the button. The Edit Collection dialog box opens ([Figure 4-6](#)).
3. In the list window, select the strings that you want to delete.
4. Click Delete Selected Strings.
5. Click OK.

4.2.2. Common Broker Settings

Table 4-1 describes the attributes in the Common Broker Settings group. They are the same for all of the brokers.

Table 4-1: Common broker settings attributes

Attribute	Description
Notify Level	Specifies the logging output level for the broker's log file. Range 0-4, where: 0 = Fatal error messages 1 = Warning messages 2 = Informational messages 3 = Verbose information 4 = Debugging data Default: 2.
Log Filename	Specifies the name of the file to log broker output to. Default: "" (no log file).
Maximum Control Messages	Specifies the maximum number of Portal control messages to process per call to <code>drainInput()</code> . If set to -1, then all messages are processed. Default: -1.
Maximum Interaction Messages	Specifies the maximum number of Portal interaction messages to process per call to <code>drainInput()</code> . If set to -1, then all messages are processed. Default: -1.
Maximum Object Messages	Specifies the maximum number of Portal object messages to process per call to <code>drainInput()</code> . If set to -1, then all messages are processed. Default: -1.
Polling Interval	Specifies the interval, in seconds, between each call to <code>drainInput()</code> on the exercise connection. Default: 1.

4.2.3. Configuring Publication Conditions

By default, VR-Exchange does not translate certain HLA objects until certain conditions are met. For example, an entity is not translated until its entity ID is a non-zero value. This is done because various mappings exist between DIS and HLA identifiers. However, there are situations in which IDs and other attributes are not specified. Turning off publication conditions helps allows you to translate these objects.

If a condition is set to True, an HLA object is not published until the condition is true. If a condition is set to false, the object will be published without satisfying this condition.

You can set discovery criteria for the following objects:

- ♦ Aggregate:
 - Aggregate needs valid Entity ID
 - Aggregate needs valid Location
 - Aggregate needs valid Type
- ♦ Designator:
 - Designator needs valid Host ID
 - Designator needs valid Entity ID
- ♦ Emitter System:
 - Emitter System needs valid Entity ID
 - Emitter System needs valid Host ID
- ♦ Entity State:
 - Entity needs valid Entity ID
 - Entity needs valid Location
 - Entity needs valid Type
- ♦ Environment Process:
 - Environment Process needs valid Process ID
- ♦ Gridded Data:
 - Gridded Data needs valid Grid ID
- ♦ IFF:
 - IFF needs valid Entity ID
 - IFF needs valid Host ID
- ♦ Radio Receiver:
 - Radio Receiver needs valid Entity ID
 - Radio Receiver needs valid Host ID

- ♦ Radio Transmitter:
 - Radio Transmitter needs valid Entity ID
 - Radio Transmitter needs valid Host ID

To configure publication conditions:

1. In the Connections window (Figure 3-1), select the connection that you want to configure.
2. Choose **Connection** → **Edit Connection Information**, or double-click the connection. The Connection Information dialog box opens.
3. Select the Configure page (Figure 4-7).

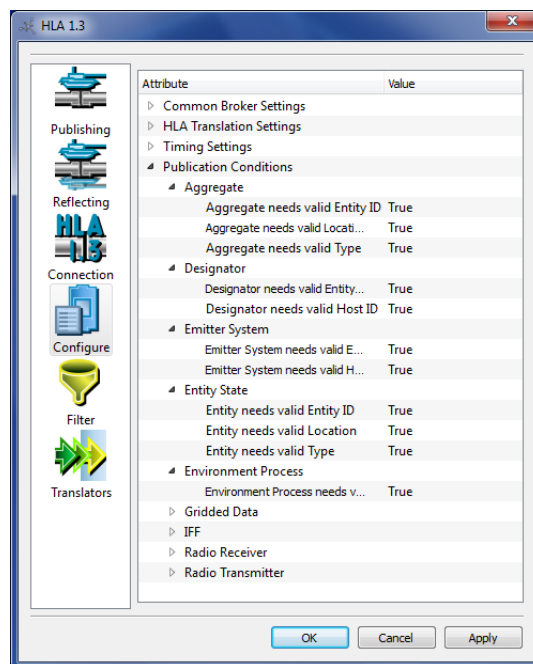


Figure 4-7. Publication conditions

4. Expand the Publication Conditions section.
5. Expand the hierarchy for the message type whose publication conditions you want to change.
6. Click the value to make it editable.
7. Select the value that you want.
8. Click OK.

4.2.4. Protocol Specific Attributes

The Configure page for the different brokers has some attribute groupings used in one or more brokers, but the attributes themselves vary by broker. The following sections describe them.

HLA Translation Settings and Timing Settings

[Table 4-2](#) describes the HLA-specific attributes grouped under translation settings and timing settings.

Table 4-2: HLA connection attributes (Sheet 1 of 2)

Attribute	Description
HLA Translation Settings	
Fill Outgoing Entity IDs	Enables or disables filling out the entity ID (site:host:id) for all outgoing HLA entities. Use this parameter to spoof gateways that require DIS entity IDs for RPR FOM objects. Default: False.
Use Persistent Entity IDs	Specifies that entity IDs persist after disconnects. If this feature is enabled, when you replay an exercise, entities with common names will use the same entity IDs that they did in previous runs. To enable this feature, set Use Persistent Entity IDs to True. Default: False. You must also enable Fill Outgoing Entity IDs. This attribute saves entity IDs to the file <i>hlaPersistentEntityNumMap.csv</i> . Default: False.
Outgoing Site ID	The site ID to use when Fill Outgoing Entity IDs is true. Default: 3.
Outgoing Host ID	The host ID to use when Fill Outgoing Entity IDs is set to 1. Default: 4.
Use HLA Object IDs For Marking Text	When enabled, the broker uses the HLA object ID from reflected HLA objects to set the marking text when it sends information to the Portal. This is useful for mapping non-RPR FOM-based FOMs that do not have marking text to RPR FOM-based FOMs that might rely on it. Default: False.
Use Marking Text for HLA Object IDs	When enabled, the broker uses the marking text for objects it discovers in the Portal to name the HLA objects it is sending to its federation. If this parameter is enabled, HLA objects will not be published until a valid marking text is supplied. Default: False.
Use Absolute Time-stamps	Enables or disables use of absolute time stamping. Default: False.
Disable Articulated Parts Translation	Specifies whether or not to translate articulated parts. This can be used when your exercise has circular dependencies or otherwise non-standard configurations of articulated parts that might confuse the brokers.

Table 4-2: HLA connection attributes (Sheet 2 of 2)

Attribute	Description
Use Generic Translators for Whole FOM	Enables or disables use of generic translation for the entire FOM. Default: False.
Generic Objects to Translate Set	A list of HLA object classes to translate using the generic object translator. The generic object translator does not work with a FOM Mapper and the classes represented in this list must be exactly the same in all receiving FOMs. In cases where there are more than two brokers, only the two for which they are exactly the same should have this parameter set. (For information about generic translation, please see “Generic Translations,” on page A-4.) Default: an empty list.
Generic Interactions to Translate Set	A list of HLA interaction classes to translate using the generic interaction translator. (For information about generic translation, please see “Generic Translations,” on page A-4.) Default: an empty list.
Timing Settings	
Heartbeat	The frequency with which to send an update if there is no change. Default: -1 (means do not heartbeat).
HLA Maximum Tick Time	Specifies the maximum amount of time, in seconds, to tick the RTI before processing incoming messages. -1 = Tick the RTI until all messages are read. >0 = Tick the RTI at least HLA Minimum Tick Time, but no more than this value. Default: -1.
HLA Minimum Tick Time	Specifies the minimum amount of time, in seconds, to tick the RTI before processing incoming messages. Default: 0.0.

DIS Translation Settings, Debug Settings, and Timing Settings

Table 4-3 lists DIS-specific attributes grouped under translation settings, debug settings, and timing settings.

Table 4-3: DIS connection attributes (Sheet 1 of 2)

Parameter	Description
DIS Translation Settings	
Parse Unknown IDs	Tells the DIS Broker to try to parse a DIS ID out of a Portal Object ID if there is not already a known entity available. For example, if the Portal ID (a string) is "1:2:3", and if the DIS Broker does not already know about string "1:2:3", it tries to create a DIS ID from the string, in this case 1:2:3. Default: True.
Remap DIS IDs	Enables or disables mapping of all DIS entity IDs to match the site and host IDs specified in the configuration file. This corrects errors caused by rogue DIS objects in a DIS-to-HLA-to-DIS exercise. Default: False.
Use Persistent Entity IDs	Specifies that entity IDs persist after disconnects. If this feature is enabled, when you replay an exercise, entities with common names use the same entity IDs that they did in previous runs. To enable this feature, set Use Persistent Entity IDs to True. You must also enable Fill Outgoing Entity IDs. This attribute saves entity IDs to the file <i>disPersistentEntityNumMap.csv</i> . Default: False.
Application Number	The DIS application number. It is used for creating entity IDs for PDUs originating from the DIS broker. It must be unique. Default: 27.
Site ID	Specifies the DIS site ID. Default: 1.
Use Absolute Time-stamps	Specifies the use of absolute timestamps on PDUs or interactions. False = Do not use absolute timestamps. True = Use absolute timestamps. Default: False.
DIS Protocol Version to Send	The DIS PDU version to send, where: 4 = DIS 2.0.4 5 = IEEE 1278.1 6 = IEEE 1278.1a Default: 5.
DIS Minimum Protocol Version to Receive	The minimum DIS PDU version to receive. Default: 4.
DIS Maximum Protocol Version to Receive	The maximum DIS PDU version to receive. Default: 6.

Table 4-3: DIS connection attributes (Sheet 2 of 2)

Parameter	Description
Disable Articulated Parts Translation	Specifies whether or not to translate articulated parts. This can be used when your exercise has circular dependencies or otherwise non-standard configurations of articulated parts that might confuse the brokers.
Debug Settings	
Debug Global ID Mappings	Enables or disables mapping of Portal object IDs to DIS entity IDs in the debug output. <code>notifyLevel</code> must be set to at least 2 for this to work. Default: False.
Debug Entity ID Mappings	Enables or disables mapping of Portal entity IDs to DIS entity IDs in the debug output. Default: False.
Debug Event ID Mappings	Enables or disables mapping of Portal event IDs to DIS event IDs in the debug output. Default: False.
Dump GUI to Text	Enables or disables logging of GUI updates to the log file. When enabled, the Portal outputs a line each time an update comes from the translator. This information is already displayed in the GUI, but using Dump GUI to Text makes a better record for debugging. Notify Level must be set to at least 2 for this to work. Default: False.
Timing Settings	
Heartbeat Interval	Specifies the number of seconds between heartbeats of published entities. Default: 5.0 seconds.
Timeout Interval	Specifies the number of seconds before a reflected entity times out and the DIS broker removes it. Default: 10.0 seconds.
Drain Timeout	Specifies the timeout value, in seconds, for reading PDUs from the network, as follows: -1 = No timeout. Poll network until all PDUs are read. >0 = Poll the network, reading the number of PDUs specified by Number of PDUs Read Per Tick, until all PDUs are read or the timeout value is reached, whichever comes first. Default: -1.
Number of PDUs Read Per Tick	Specifies the number of PDUs to read before checking to see if the amount of time specified by Drain Timeout has been reached. Default: 100.
Multicast Time-to-Live	Specifies a multicast time-to-live value. Default: 0 (use the system default).

TENA Translation Settings, AMO Settings, Debug Settings, and Timing Settings

Table 4-4 lists TENA-specific attributes that are grouped under translation settings, TENA AMO settings, debug settings, and timing settings for the TENA connection. Attributes that take a list of translator names use the names listed in “TENA Translators,” on page A-12.

Table 4-4: TENA connection attributes (Sheet 1 of 2)

Attribute	Description
TENA Translation Settings	
Parse Unknown IDs	When set, the broker will try to parse string identifiers into DIS IDs. Default: True.
Generate Fire PDU for Indirect Fire	If the broker receives a TENA detonation for which it does not have a corresponding Fire Interaction, it generates a Fire Interaction.
Fill Outgoing Entity IDs	Enables or disables filling out of the entity ID (site:host:id) for all outgoing TENA entities. Default: False.
Use Persistent Entity IDs	Specifies that entity IDs persist after disconnects. If this feature is enabled, when you replay an exercise, entities with common names will use the same entity IDs that they did in previous runs. To enable this feature, set Use Persistent Entity IDs to True. You must also enable Fill Outgoing Entity IDs. This attribute saves entity IDs to the file <i>hlaPersisten-tEntityNumMap.csv</i> . Default: False.
Site ID	Specifies the TENA site ID. Default: 1.
Application Number	The TENA application number. It is used for creating entity IDs for PDUs originating from the TENA broker. It must be unique. Default: 3338.
Use TENA Object ID for Marking Text	Enables or disables use of the TENA object ID from reflected TENA objects to set the marking text when it sends information to the Portal. Default: False.
Use Marking Text for TENA Object ID	Enables or disables use of the marking text for objects it discovers in the Portal to name the TENA objects it is sending to its execution. If enabled, TENA objects are not published until a valid marking text is supplied. Default: False.
Guise is Entity Type	Translate the TENA Guise attribute as the Entity Type when moving back and forth from DIS or HLA. Default: True.
Force Updates	Forces all objects to update on tick even if they have not changed at all. Default: False.
TENA AMO Settings	
AMO Name	Specifies the TENA broker's AMO name. Default: "".
AMO Hostname	Specifies the TENA broker's AMO hostname. Default: "".

Table 4-4: TENA connection attributes (Sheet 2 of 2)

Attribute	Description
AMO Host IP Address	Specifies the TENA broker's AMO hostIpAddress. Default: "".
AMO Operator Name	Specifies the TENA broker's AMO operatorName. Default: "".
AMO Operator Telephone Number	Specifies the TENA broker's AMO operatorTelephoneNumber. Default: "".
Debug Settings	
Debug Global ID Mappings	Enables or disables debug output for global ID mapping. Global ID mapping maps TENA names to Portal entity IDs. Notify Level must be set to at least 2 for this to work. Default: False.
Debug Entity ID Mappings	Enables or disables mapping of Portal entity IDs to TENA entity IDs in the debug output. Default: False.
Dump GUI to Text	Enables or disables logging of GUI updates to the log file. When enabled, the Portal outputs a line each time an update comes from the translator. This information is already displayed in the GUI, but using Dump GUI to Text makes a better record for debugging. Notify Level must be set to at least 2 for this to work. Default: False.
Timing Settings	
TENA Tick Time	Specifies the TENA exercise connection's tick time, that is, the time to spend within drainInput().
	 drainInput() will not exit early if there is no more input from the exercise.
	For more detail concerning this parameter, refer to VR-Link's TENA DtExerciseConn::drainInput() documentation. Default: 0.1 seconds
Object Throttle Rates	Specifies a list of entity types and update values. Limits the maximum number of updates per second for specific TENA object entity types. Default: 10.

4.3. Configuring Network Connections

Each broker connection dialog box has a Connection page that lets you configure the network connections for the broker. The parameters are protocol-specific.

To configure a broker's network connection:

1. In the Connections window (Figure 3-1), select the connection that you want to configure.
2. Choose **Connection** → **Edit Connection Information** or double-click the connection. The Connection Information dialog box opens.
3. Select the Connections page. Figure 4-8 illustrates the Connections page for a DIS broker.

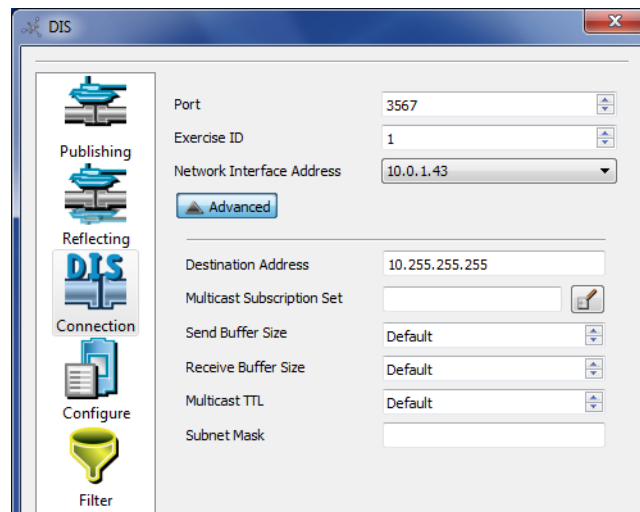


Figure 4-8. DIS Network Connections page

4. Edit the parameters. The parameters are described in the following sections.

4.3.1. HLA Network Connection Parameters

Table 4-2 describes the HLA-specific parameters that you can configure for connections that use the HLA broker.

Table 4-5: HLA network connection parameters (Sheet 1 of 2)

Attribute	Description
HLA Execution Name	Specifies the name of the HLA federation execution that this broker will join. Default: VR-Link.
HLA FED File Name	Specifies the name of the FED file being used for the federation execution. It must be the complete name including the three character extension (<i>.fed</i>). Default: VR-Link.fed.
Federate Type	Specifies the HLA Federate type. Federate Type can be used to distinguish different categories of federates, and its only practical use is in save-and-restore. Default: VR-Exchange.
FOM Mapper Library Name	Specifies the name of a FOM mapper library, if you are using one. If you want to use one of the FOM Mappers supplied with VR-Exchange, leave this attribute blank and specify a FOM Mapper for the Built In FOM Mapper Attribute. Default: "".
FOM Mapper Initialization Data	Specifies a string that is passed to the FOM mapper library specified in FOM Mapper Library Name. This data is optional. Default: "".
Built In FOM Mapper	If the federation is using the RPR FOM, specifies the version, and for RPR FOM 2, draft 17, the revision. RPR FOM revision 2 for version 2.0017 fixes an encoding and decoding size issue with object ID lists to match RPR FOM 2 draft 17. The encoding and decoding issues affected the Aggregates, JammerBeams, and RadarBeams objects and the RemoveObjectRequest, AttributeChangeRequest, and ActionRequestToObject interactions. The built-in FOM Mappers are: ♦ RPR FOM 1.0 ♦ RPR FOM 2, draft 14 ♦ RPR FOM 2, draft 17, Revision 1 ♦ RPR FOM 2, draft 17, Revision 2. Default: RPR FOM 1.0.

Table 4-5: HLA network connection parameters (Sheet 2 of 2)

Attribute	Description
Use HLA Advisories	Specifies whether the broker should use RTI advisory messages when determining what attributes to update. False = Send attribute updates regardless of whether or not someone has subscribed to the attribute with the RTI. True = Send attribute updates only if someone has subscribed to the attribute. Default: False.
Additional Connection Settings for HLA Evolved	
Federate Name	The Federate Name is a unique name that can be assigned to the federate. No two federates within the same federation can have the same name. If no name is specified, the RTI automatically assigns a unique name for the federate.
HLA Evolved FOM Module Names	HLA Evolved allows federates with different FOMs to interoperate without merging their FOMs. This is done by loading FOM modules.
Local Settings Designator	A string that is passed to the RTI during federate connect when using HLA Evolved. It is usually used to provide additional RTI configuration, but every RTI uses it differently. Please consult your RTI documentation for proper usage. This setting is optional for the MÄK RTI. If you specify it, it overrides the RID file settings.

4.3.2. DIS Network Connection Parameters

Table 4-3 lists parameters that you can configure for network connections that use the DIS broker.

Table 4-6: DIS network connection parameters (Sheet 1 of 2)

Parameter	Description
Port	Specifies the port to receive network PDUs from and send them to. Default: 3000.
Exercise ID	Specifies the DIS exercise ID. Default: 1.
Network Interface Address	Specifies the IP address of the network device to use for sending DIS PDUs.
Destination Address	Specifies the address of outgoing DIS PDUs. If set to 0.0.0.0, the default broadcast address is used.
Multicast Subscription Set	Specifies a list of multicast addresses the DIS broker should subscribe to for DIS communication. Default: an empty list.
Send Buffer Size	Sets the socket's send buffer size. Default: 0.

Table 4-6: DIS network connection parameters (Sheet 2 of 2)

Parameter	Description
Receive Buffer Size	Sets the DIS socket's buffer received size. If you think the broker is dropping incoming packets during periods of heavy network traffic, increasing this value might help. A typical default value is 50000. If you set the size to zero, the DIS broker uses the default size for the operating system. Default: 0.
Multicast TTL	Specifies the number of routers that a message can pass through when PDUs are sent to multicast addresses.
Subnet Mask	VR-Exchange can usually figure out the subnet mask from the IP address. However, if it does not do this correctly, you can explicitly specify the subnet mask.

4.3.3. TENA Network Connection Parameters

[Table 4-4](#) lists parameters that you can configure for a TENA network connection. Attributes that take a list of translator names use the names listed in [“TENA Translators,”](#) on page A-12.

Table 4-7: TENA network connection parameters

Attribute	Description
Execution Name	Specifies the name of the TENA execution this broker will join. The Execution Name value must match the TENA <i>executionManager.exe</i> 's <code>-executionname</code> parameter. Default: "VR-Link".
ORB Default Init Ref	Specifies the TENA ORB Initialization Reference. It must match the TENA <i>executionManager.exe</i> 's <code>-ORBdefaultInitRef</code> parameter. Example: <code>corbaloc:iiop:10.0.1.250:50509</code> Default: "".
ORB Default Listen Endpoints	Specifies the TENA ORB Listen Endpoint. It must match the TENA <i>executionManager.exe</i> 's <code>-ORBlistenEndpoints</code> parameter. Example: <code>iiop://10.0.1.250</code> Default: "".
TENA Transport Type	Specifies the transport type – best effort or reliable.
LROM Mapper Name	LROM Mapper DLL name. Default: <i>makLromMapper.dll</i> .
Multicast Time to Live	Specifies a multicast time-to-live value. Default: 1 (use the system default).
Multicast Interface	Specifies the IP address of the network device to use for sending PDUs. Default: 0.0.0.0 (the IP address of the primary network device).

4.4. Filtering Objects

You can filter out objects by their object type, object name, or marking text. Filtering by entity type and marking text only applies to entities and aggregates. The filtering mechanism supports use of wildcards.

This section explains how to add and remove filters in the Connection Information dialog box. You can also do so dynamically in the Objects window. For details, please see “[Filtering Objects](#),” on page 3-13.

To filter an object:

1. Select the broker connection on which you want to configure a filter.
2. Choose **Connection** → **Edit Connection Information** or double-click the connection. The Connection Information dialog box opens.
3. Select the Filter page ([Figure 4-9](#)). It displays any filters that you have set in the Objects window or in this dialog box.

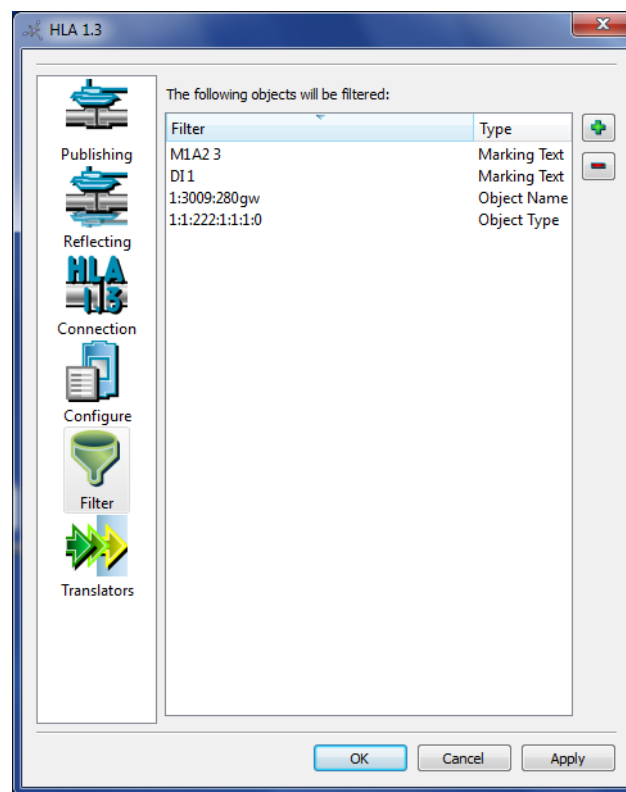


Figure 4-9. Filter page

4. Click the Add icon (+). The New Filter dialog box opens ([Figure 4-10](#)).

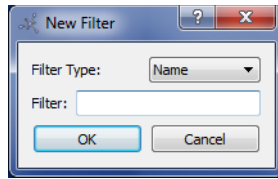


Figure 4-10. New Filter dialog box

5. Select the filter type in the Filter Type drop-down list.
6. Enter the filter value. If you chose Name or Marking Text as the filter type, type the appropriate text for the object you want to filter. You can use wildcards such as *, ?, and [abcd] to define a generalized ID.

If you chose Type, VR-Exchange automatically inserts an enumeration value (all zeros) in the field. Type the enumeration for the object you want to filter.



If you mouse over the entity icon in the Type column of the Objects window, VR-Exchange displays the entity type enumeration for the object. This information may be helpful in deciding the enumeration you want to use when filtering by entity type.

7. Click OK. The new filter is added to the list.

4.4.1. Editing a Filter

To edit a filter:

1. Select the broker connection for which you want to edit a filter.
2. Choose **Connection** → **Edit Connection Information**, or double-click the connection. The Connection Information dialog box opens.
3. Select the Filter page (Figure 4-9).
4. Double-click the filter that you want to edit. The New Filter dialog box opens (Figure 4-10).
5. Edit the values as desired.
6. Click OK.

4.4.2. Removing a Filter

When you delete a filter in the Connection Information dialog box, its status in the Objects window is updated. However, there may be some latency between the two events.

To delete a filter:

1. Select the broker connection from which you want to remove a filter.
2. Choose **Connection** → **Edit Connection Information**, or double-click the connection. The Connection Information dialog box opens.
3. Select the Filter page ([Figure 4-9](#)).
4. Select the filter that you want to delete.
5. Click the Delete icon ().

4.5. Specifying Which Translators to Use

Each broker has a set of translators. You can specify which translators a connection should use during a session.

To specify which translators a connection should use:

1. In the Connections window (Figure 3-1), select the connection that you want to configure.
2. Choose **Connection** → **Edit Connection Information**, or double-click the connection. The Connection Information dialog box opens.
3. Select the Translators page (Figure 4-11).

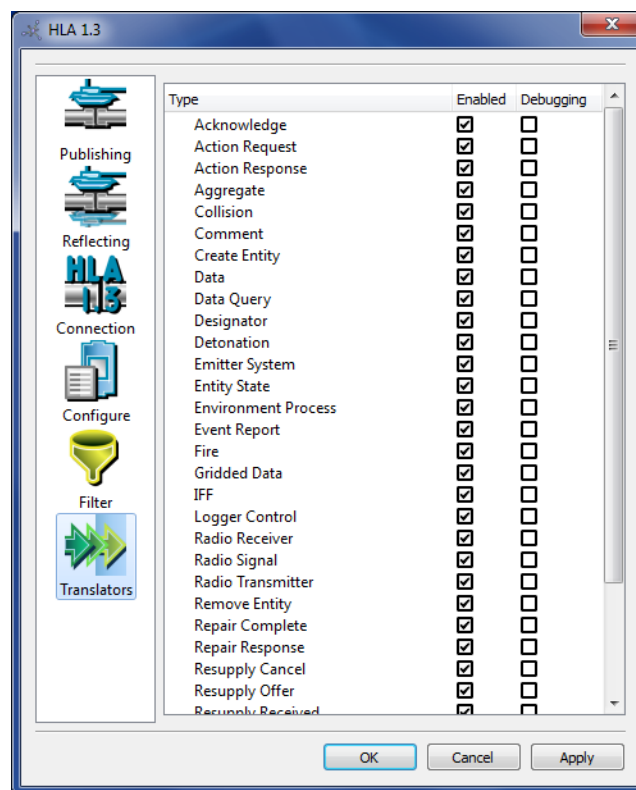


Figure 4-11. Translators page

4. In the Enabled column, select the check box for each translator that you want to use. Clear the check box for each translator that you do not want to use.
5. In the Debugging column, select the check box for a translator if you want VR-Exchange to send debugging messages for this translator. If debugging is enabled, the notify level must be at least 2.
6. Click OK.

4.6. Viewing Publishing and Reflecting Lists

You can view lists of the objects that a connection is publishing and reflecting. You can open information windows for multiple connections at the same time.

- The Publishing page displays the objects that the connection has discovered from the Portal and published to its simulation network.
- The Reflecting page displays the objects that the connection has discovered on its simulation network and published to the Portal.

To view connection information:

1. In the Connections window (Figure 3-1), select the connection whose details you want to display.
2. Choose **Connection** → **Edit Connection Information**, or double-click the connection. The Connection Information dialog box opens.
3. Select the Publishing page or the Reflecting page (Figure 4-12).

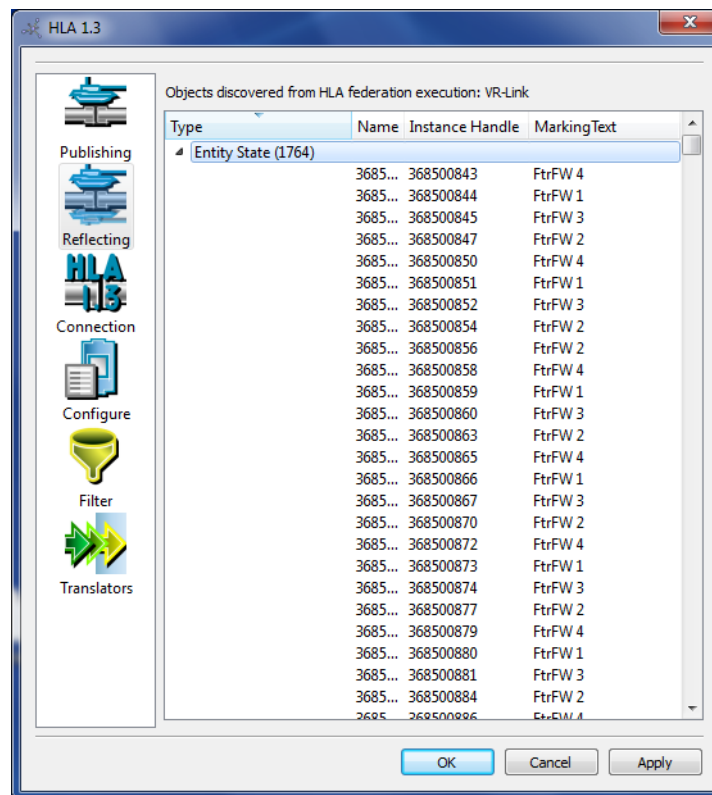
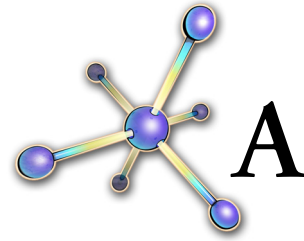


Figure 4-12. HLA connection Reflecting page



Broker Details

This appendix lists the translators provided by each broker and lists their command-line options. You can add command-line options to a connection in the Add Broker dialog box and the Edit Broker dialog box.

The HLA Brokers	A-2
HLA Translators	A-3
Generic Translations	A-4
HLA Broker Command-Line Options	A-6
The DIS Broker	A-8
DIS Broker Command-Line Options	A-9
The TENA Broker	A-11
TENA Translators	A-12
TENA Broker Command-Line Options	A-13

A.1. The HLA Brokers

VR-Exchange includes HLA brokers for the HLA 1.3 specification, for the SISO Standard DLC C++ API for IEEE 1516 specification, and for the HLA Evolved specification. They have built-in support for the RPR FOM using the VR-Link FOM Mappers and FED files provided. They can support other FOMs through the use of user-written FOM Mappers or generic translation.



The brokers work identically and support the same configurations, so for simplicity, in the remainder of this manual we refer to them generically as the HLA broker.

A.1.1. HLA Translators

The HLA broker is composed of translators (described in “[Translators](#),” on page 1-3). Each translator is responsible for a specific FOM object class, or interaction class. [Table A-1](#) lists the translators in the HLA broker. The HLA broker can use a VR-Link FOM Mapper to modify the translations of any of the translators. For information about FOM mapping, please see *VR-Link Developers Guide*.

Table A-1: HLA broker translators

Translator	RPR FOM Class
Acknowledge	InteractionRoot.Acknowledge
Action Request	InteractionRoot.ActionRequest
Action Response	InteractionRoot.ActionResponse
Aggregate	ObjectRoot.BaseEntity.AggregateEntity
Collision	InteractionRoot.Collision
Comment	InteractionRoot.Comment
Create Entity	InteractionRoot.CreateEntity
Data	InteractionRoot.Data
Data Query	InteractionRoot.DataQuery
Designator	ObjectRoot.Designator
Detonation	InteractionRoot.MunitionDetonation
Emitter System	ObjectRoot.EmitterBeam ObjectRoot.EmbeddedSystem.EmitterSystem
Entity State	ObjectRoot.BaseEntity.PhysicalEntity.Platform
Environment Process	ObjectRoot.EnvironmentProcess
Event Report	ObjectRoot.EventReport
Fire	InteractionRoot.WeaponFire
Gridded Data	ObjectRoot.GriddedData
IFF	ObjectRoot.EmbeddedSystem.IFF
Radio Receiver	ObjectRoot.EmbeddedSystem.RadioReceiver
Radio Signal	InteractionRoot.RadioSignal
Radio Transmitter	ObjectRoot.EmbeddedSystem.RadioTransmitter
Remove Entity	InteractionRoot.RemoveEntity
Set Data	InteractionRoot.SetData
Start	InteractionRoot.StartResume
Stop	InteractionRoot.StopFreeze

A.1.2. Generic Translations

The HLA broker has a special translator called a generic translator. Each broker can configure multiple generic translators, each responsible for one specific object or interaction class. You can also specify that all translations be done generically, using none of the translators supplied with VR-Exchange. Generic translation allows you to filter parts of a FOM. It also allows you to create RTI-to-RTI bridges if the federations being bridged use the same FOM.

The generic translator does not actually do any translation. It copies each attribute or parameter byte-for-byte from and to the Portal, and publishes it to the network.

Using the same configuration as [Figure 1-2](#), [Figure A-1](#) illustrates how a generic translator works. This time, suppose that these FOMs do not have a translator for vehicle names. The leftmost federation sends a “lorry” message. The broker simply sends the string “lorry” to the VR-Exchange shared memory area. The second federation reads the string “lorry” from the Portal. This string is meaningful to the FOM, which is identical to that of the first federation, so it sends the “lorry” message. The third federation does not have a generic translator, so it ignores the string.

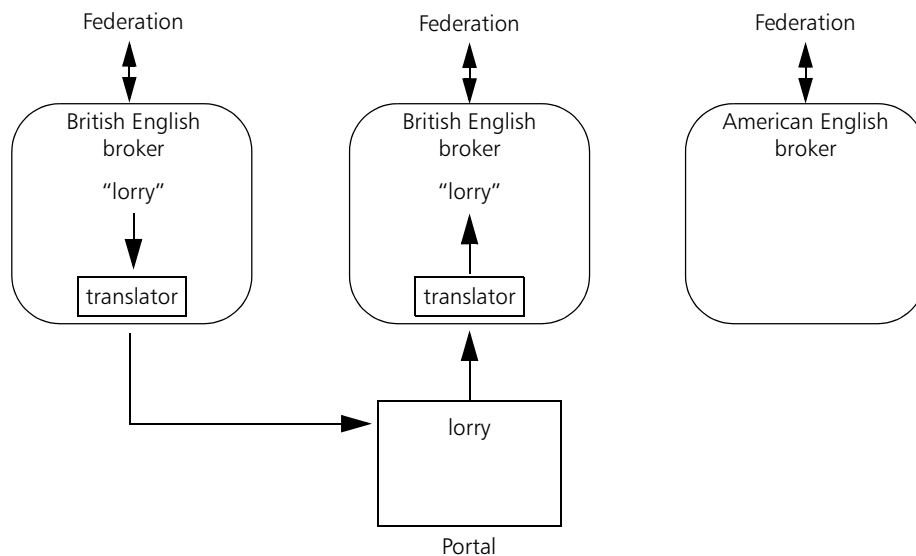


Figure A-1. How generic translators work



To use generic translation, the sending and receiving federations must configure the same generic translators.

Generic translation only works with the HLA brokers, and only with one API at a time. This means:

- ♦ Generic messages do not work between the different HLA specifications
- ♦ The object or interaction class must be exactly the same in all connected FOMs
- ♦ You cannot use a FOM Mapper with generic translators.

Translating Individual Classes Generically

Translating individual classes generically is useful when you only want to translate parts of a FOM or you want to supplement the built-in translators. For example, if your exercise includes an interaction for which VR-Exchange does not have a translator, you can add a generic translator for that interaction. Bear in mind the restrictions on generic translations listed in the previous section.

To translate individual classes generically, you add the class names of the objects you want to translate to the appropriate collection in a connection's configuration. For example, you might add `ObjectRoot.EnvironmentProcess` to the Generic Objects to Translate Set attribute.

Translating a FOM Generically

Translating all FOM classes generically is useful if you have two or more federations that use the exact same FOM, but perhaps different RTI's. When you enable generic translation for all FOM classes, the HLA broker creates a generic translator for all objects and interactions in the FOM (except MOM data, and anything that starts with `RTIPrivate`). This mode prevents any built-in translators from starting.

To enable generic translation for the entire FOM, set the Use Generic Translators for Whole FOM attribute to `True`.

A.1.3. HLA Broker Command-Line Options

The HLA broker command-line options override the settings in a connections configuration file. The HLA broker has the following command-line syntax:

```
{hla13Broker | hla1516Broker | hla1516eBroker} [  
  --  
  --brokerName name  
  --controlQueueName name  
  -E  
  -f library  
  --fomMapperInitData data  
  --fomModules module (HLA Evolved only)  
  -F file  
  -G  
  -h  
  --interactionQueueName name  
  -l filename  
  -L logfile_name  
  -n level  
  -N federate_name (HLA Evolved only)  
  --objectQueueName name  
  -O  
  -p type  
  --rprFomRevision revision  
  --rprFomVersion version  
  --showInfoDialog  
  -S designator (HLA Evolved only)  
  -t port  
  -v  
  -x execution]
```

where:

(-- | --ignore_rest) tells the broker to ignore the arguments following this flag.

--brokerName *name* specifies the name of the broker. The broker name must be unique among all brokers used. Default: Broker.

--controlQueueName *name* specifies the name of the shared memory control queue.

(-E | --dumpEnvironment) print all of the environment variable settings to the log file.

(-f | --fomMapperLibName) *library* specifies the FOM mapper library, if used.

--fomModules *module* specifies a FOM module. This can be used multiple times. (HLA Evolved only.)

(-F | --fedFileName) *file* specifies the FED file name. Default: *VR-Link.fed*.

--fomMapperInitData *data* specifies FOM Mapper initialization data, if it is required.

(-G | --noGui) specifies that the VR-Exchange Portal GUI should not be run.

(-h | --help) displays usage information and exits.

--interactionQueueName *name* specifies the name of the shared memory interaction queue.

(-l | --configFile) *filename* specifies the name of the configuration file for the specified broker. Default: *broker.xml*.

(-L | --brokerLogFileName) *filename* specifies the name of the log file for the broker.

(-n | --notifyLevel) *level* specifies the output level for the broker's log file. Value values are 0 – 4. As the level increases, so does the output generated. Default: 2. The meanings of the various levels are described in “[VR-Exchange Command-Line Options](#),” on page 3-4.

(-N | --federateName) *federate_name* specifies the name of the HLA broker federate. (HLA Evolved only.)

--objectQueueName *name* specifies the name of the shared memory object queue.

(-O | --oneToOneMapping) forces the broker to send an update every time it receives an update from the portal, even if normal rules would dictate that an update is not necessary.

(-p | --federateType) *type* specifies the federate type. Federate Type can be used to distinguish different categories of federates, and its only practical use is in save-and-restore.

--rprFomRevision *revision* specifies the revision of the RPR FOM version, for example, RPR FOM 2, revision 17.

--rprFomVersion *version* specifies the RPR FOM version. Default: 1.0.

--showInfoDialog specifies that the connection information dialog box be displayed at startup.

(-S | --localSettingsDesignator) *designator* specifies a string that is passed to the RTI during federate connect when using HLA Evolved. It is usually used to provide additional RTI configuration, but every RTI uses it differently. Please consult your RTI documentation for proper usage. This attribute is optional in the MÄK RTI. You can use it to override the RID file settings. (HLA Evolved only.)

(-t | --tcpPort) *port* specifies the TCP port to use to maintain connectivity with the Portal. If the Portal dies, then the broker will know by testing this port. If this argument set to 0, the broker does not try to connect. Default: 5111.

(-v | --version) displays version information and exits.

(-x | --execName) *execution* specifies the HLA federation execution.
Default: “VR-Link”.

A.2. The DIS Broker

The DIS broker provides comprehensive translation of DIS PDUs. It receives DIS 4, 5, and 6. It sends DIS 5.

[Table A-2](#) lists the translators supported by the DIS broker.

Table A-2: DIS broker translators

Translator	PDU
Acknowledge	Acknowledge PDU
Action Request	Action Request PDU
Action Response	Action Response PDU
Aggregate	Aggregate PDU
Collision	Collision PDU
Comment	Comment PDU
Create Entity	Create Entity PDU
Data	Data PDU
Data Query	Data Query PDU
Designator	Designator PDU
Detonation	Detonation PDU
Emitter Systems	Electromagnetic Emission PDU
Entity State	Entity State PDU
Environment Process	Environmental Process PDU
Event Report	Event Report PDU
Fire	Weapon Fire PDU
Gridded Data	Gridded Data PDU
IFF	IFF/ATC/NAVAIDS PDU
Radio Receiver	Radio Receiver PDU
Radio Signal	Signal PDU
Radio Transmitter	Radio Transmitter PDU
Remove Entity	Remove Entity PDU
Set Data	Set Data PDU
Start	Start Resume PDU
Stop	Stop Freeze PDU

A.2.1. DIS Broker Command-Line Options

The DIS broker command-line options override the settings in connection's configuration file. The DIS broker has the following command-line syntax:

```
disBroker [
  --
  -a application
  -A address
  --brokerName name
  --controlQueueName name
  --disMcastDevice device
  --disSendPort sendPort
  -E
  -G
  -h
  --interactionQueueName name
  -l filename
  -L logfile_name
  -n level
  --objectQueueName name
  -O
  -P port
  -s site
  --subnetMask mask
  --showInfoDialog
  -t port
  -v
  -x exercise]
```

where:

(-- | --ignore_rest) tells the broker to ignore the arguments following this flag.

(-a | --applicationNumber) *application* specifies the unique DIS application number. It is used for creating entity IDs for PDUs originating from the DIS broker. Default: 27.

(-A | --destinationAddress) *address* is the address of outgoing DIS PDUs. If set to 0.0.0.0 the default broadcast address is used. Default: 0.0.0.0.

--brokerName *name* specifies the name of the broker. The broker name must be unique among all brokers used. Default: Broker.

--controlQueueName *name* specifies the name of the shared memory control queue.

--disMcastDevice *device* is the address of the device to use for multicast traffic. (This option is equivalent to the Network Interface Address parameter on the Connection Information dialog box, Connections page.) If set to 0.0.0.0, the default network address is used. Default: 0.0.0.0.

`--disSendPort sendPort` specifies the port the broker sends PDUs on. This port is only to be used by the broker. It knows that if it receives a message with this port, that it is its own message, so it does not translate it. Federates listen to the port specified by `--disPort`. Make sure this port does not conflict with others on your network.

Default: 2999.

`(-E | --dumpEnvironment)` print all of the environment variable settings to the log file.

`(-G | --noGui)` specifies that the VR-Exchange Portal GUI should not be run.

`(-h | --help)` displays usage information and exits.

`--interactionQueueName name` specifies the name of the shared memory interaction queue.

`(-l | --configFile) filename` specifies the name of the configuration file for the specified broker. Default: *broker.xml*.

`(-L | --brokerLogFileName) filename` specifies the name of the log file for the broker.

`(-n | --notifyLevel) level` specifies the output level for the broker's log file. Value values are 0 – 4. As the level increases, so does the output generated. Default: 2. The meanings of the various levels are described in “[VR-Exchange Command-Line Options](#),” on page 3-4.

`--objectQueueName name` specifies the name of the shared memory object queue.

`(-O | --oneToOneMapping)` forces the broker to send an update every time it receives an update from the portal, even if normal rules would dictate that an update is not necessary.

`(-P | --disPort) port` specifies the port to send and receive network PDUs on. Default: 3000.

`(-s | --siteId) site` specifies the site ID for the current exercise. This needs to be the same for all participants in the DIS exercise. Default: 1.

`--subnetmask mask` specifies the subnet mask. VR-Exchange can usually figure out the subnet mask from the IP address. However, if it does not do this correctly, you can explicitly specify it.

`--showInfoDialog` specifies that the connection information dialog box be displayed at startup.

`(-t | --tcpPort) port` specifies the TCP port to use to maintain connectivity with the Portal. If the Portal dies, then the broker will know by testing this port. If this argument is set to 0, the broker does not try to connect. Default: 5111.

`(-v | --version)` displays version information and exits.

`(-x | --exerciseId) exercise` specifies the exercise ID for the current exercise. This needs to be the same for all participants in the DIS exercise. Default: 1.

A.3. The TENA Broker

TENA (Test and Training Enabling Architecture) is an interoperability architecture used to pass data in the form of object updates and messages (like HLA objects and interactions) from one application to another. It was developed by the Foundation Initiative 2010 (FI2010) program, which is managed by the Director of the Central Test and Evaluation Investment Program (CTEIP). CTEIP is part of the Office of the Undersecretary of Defense for Test and Evaluation (DOTE-OSD). TENA was developed to serve as the interoperability architecture for the Live Range community. The direction of TENA is managed by the TENA Architecture Management Team (AMT) – a group that meets about once a quarter.

TENA's data definition format is called a Logical Range Object Model (LROM) (comparable to an HLA FOM). An LROM is a set of object/message definitions that may be used in an exercise. These object models are available via the TENA Software Development Activity (SDA) web site (<https://tena-sda.org>), and there are some standard TENA object models that were developed by the TENA community. TENA also uses a process called an Execution Manager that each application must connect to in order to be a part of the exercise.

The TENA broker supports the common object LROM and the MÄK LROM using an LROM Mapper. It can support other LROMs through the use of user-written LROM Mappers.



An LROM Mapper is conceptually the same thing as a FOM Mapper. For details, please see *VR-Link Developers Guide*.

The TENA broker:

- ♦ Throttles the rate at which an object's data is published to the TENA execution by entity type
- ♦ Lets you specify the library (LROM Mapper) used to map the objects in an LROM to the VR-Exchange internal format
- ♦ Displays the throttle rate and actual published rate in the TENA broker's GUI.

A.3.1. TENA Translators

The *coLromMapper* LROM Mapper supports the translators listed in [Table A-3](#). Each translator is responsible for the listed TENA LROM class. For a general explanation of translators, please see “[Translators](#),” on page 1-3.

The *makLromMapper* LROM Mapper supports the translators listed in [Table A-3](#) and those listed in [Table A-4](#). The MÄK LROM is based on the HLA RPR FOM and DIS standards.

Table A-3: Common Object LROM Mapper translators

Translator	TENA LROM Class
Entity State	TENA::Platform TENA::PlatformDetails
Fire	TENA::Engagement::WeaponFire
Detonation	TENA::Engagement::Detonation

Table A-4: MÄK LROM Mapper translators

Translator	TENA LROM Class
Acknowledge	MAK::Acknowledge
Aggregate	MAK::Aggregate
Collision	MAK::Collision
Comment	MAK::Comment
Data	MAK::Data
Designator	MAK::Designator
Emitter System	MAK::ElectromagneticEmission
Environment Process	MAK::EnvironmentProcess
IFF	JNTC::Airtransponder
Radio Receiver	MAK::RadioReceiver
Radio Signal	MAK::Signal
Radio Transmitter	MAK::RadioTransmitter
Set Data	MAK::SetData
Start	MAK::StartResume
Stop	MAK::StopFreeze

A.3.2. TENA Broker Command-Line Options

The TENA broker has the following command-line syntax:

```
tenaBroker [
  --
  --brokerName name
  --controlQueueName name
  -e endpoints
  -G
  -h
  -i
  --interactionQueueName name
  -l file
  -L logfile_name
  --lromMapper mapper
  -n level
  --objectQueueName name
  -O
  --showInfoDialog
  -t port
  -v
  -x execution]
```

where:

(-- | --ignore_rest) tells the broker to ignore the arguments following this flag.

--brokerName *name* specifies the name of the broker. The broker name must be unique among all brokers used. Default: Broker.



The broker's name is used as its session name.

--controlQueueName *name* specifies the name of the shared memory control queue.

(-e | --OrbListenEndpoints) *endpoints* specifies the TENA ORB Listen Endpoint. It must match the TENA *executionManager.exe*'s -ORBlistenEndpoints parameter.

(-G | --noGui) specifies that the VR-Exchange Portal GUI should not be run.

(-h | --help) displays usage information and exits.

(-i | --OrbInitRef) *reference* specifies the TENA ORB Initialization Reference. It must match the TENA *executionManager.exe*'s -ORBdefaultInitRef parameter.

--interactionQueueName *name* specifies the name of the shared memory interaction queue.

(-l | --configFile) *filename* specifies the name of the configuration file for the specified broker. Default: *broker.xml*.

--lromMapper *mapper* specifies the LROM Mapper file name.

(-L | --brokerLogFileName) *filename* specifies the name of the log file for the broker.

(-n | --notifyLevel) *level* specifies the output level for the broker's log file. Range: 0 – 4. Default: 2. The meanings of the various levels are described in “[VR-Exchange Command-Line Options](#),” on page 3-4.

--objectQueueName *name* specifies the name of the shared memory object queue.

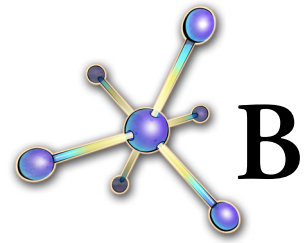
(-O | --oneToOneMapping) translate using one-to-one mapping.

--showInfoDialog specifies that the connection information dialog box be displayed at startup.

(-t | --tcpPort) *port* specifies the TCP port to use to maintain connectivity with the Portal. If the Portal dies, then the broker will know by testing this port. If this argument set to 0, the broker does not try to connect. Default: 5111.

(-v | --version) displays version information and exits.

(-x | --execName) *execution* specifies the TENA execution name. The value for *execution* must match the TENA *executionManager.exe*'s -*executionname* parameter. Default: “VR-Link”.



A VR-Exchange/TENA Primer

TENA can be overwhelming for new users; however detailed documentation about TENA is beyond the scope of this manual. This appendix explains the most basic steps for getting up and running with TENA and the VR-Exchange TENA broker. For complete documentation about TENA, please refer to its home web site: www.tena-sda.org.

i

tena-sda.org is a restricted web site. Access must first be granted by the TENA SDA project office. On the home page, click Account Request.

Installing TENA Middleware.....	B-2
Downloading the TENA Middleware.....	B-2
Installing the TENA Middleware	B-2
Building the MÄK Object Model	B-3
Downloading the MÄK Object Model	B-3
Installing the MÄK Object Model.....	B-4
Building the MÄK Object Model Basic Implementation	B-5
Running the TENA Broker.....	B-6
TENA Services.....	B-7

B.1. Installing TENA Middleware

To use the TENA broker, you must download and install the TENA Middleware.

This section explains how to download, install, and verify the installation of the TENA Middleware.



- ♦ Some TENA files are compiler-specific. In the instructions in this appendix, we use the filenames for VC 7.1 versions. If you are downloading VC 8.0 versions, be sure to choose the correct version.
- ♦ Approximately 2.98 GB of hard drive space is required for building all the required TENA files.

B.1.1. Downloading the TENA Middleware

To download the TENA Middleware:

1. Log into the TENA web site, www.tena-sda.org.
2. On the Welcome page, click the TENA Middleware link.
3. On the TENA Middleware page, click the Middleware Downloads link. The TENA Repository page is displayed.
4. In the Middleware drop-down list, select TENA-v5.2.2.
5. In the table of downloads, click the download button for the version you want (*TENA-xp-vc71-v5.2.2.exe* or *TENA-xp-vc80-v5.2.2.exe*).

B.1.2. Installing the TENA Middleware

To install the TENA Middleware:

1. Close all open windows and running applications.
2. Run the TENA installer (for example, *TENA-xp-vc71-v5.2.2.exe*). This is a self-extracting zip file. In the WinZip Self-Extractor dialog box, make sure to select:

When done unzipping open: `TENA\5.2.2\scripts\post.install.bat`

If you have problems install the TENA Middleware, please contact the TENA-SDA Help Desk.

B.2. Building the MÄK Object Model

You do not have to install or build the MÄK Object Model to run VR-Exchange. It is required if you want to build a customized TENA broker.

B.2.1. Downloading the MÄK Object Model

To download the MÄK Object Model:

1. Log into the TENA web site, www.tena-sda.org.
2. On the Welcome page, click the TENA Repository link.
3. At the top of the page, click Browse Repository. A list of object models is displayed.
4. Expand the MAK entry.
5. In the table of MAK object models, click MAK-LROM-v2.
6. In the Distributions list, select TENA-v5.2.2 (Figure B-1).

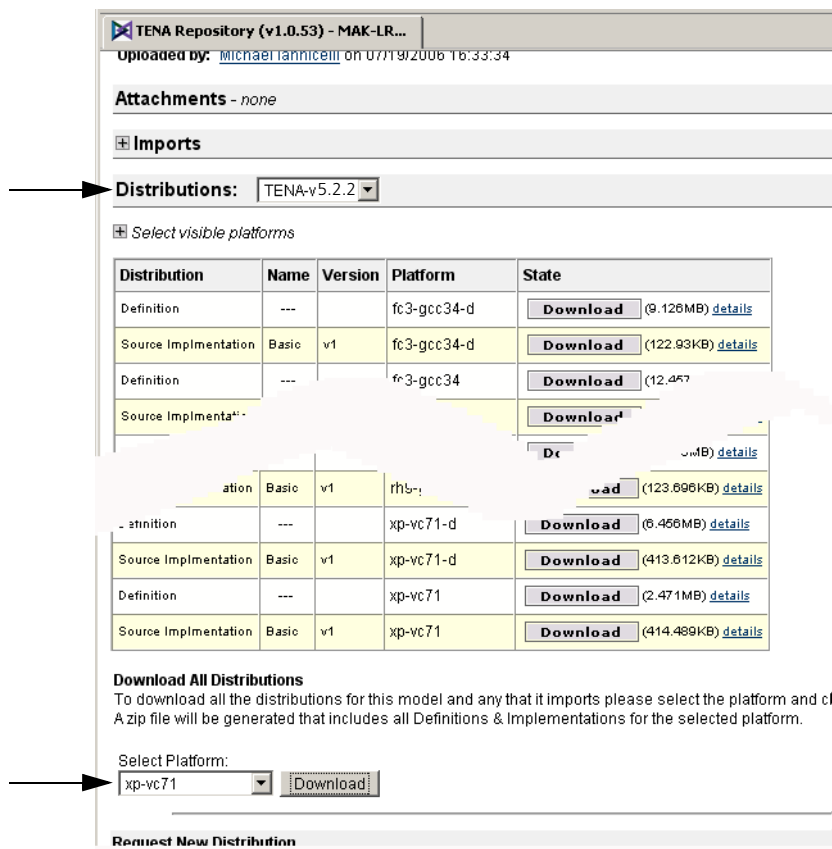


Figure B-1. MÄK Object Model download page

7. In the Select Platform list located below the Distributions table, select xp-vc71.

8. Click the Download button next to Select Platform. The distribution is assembled and displayed under the heading Object Model Imports.
9. Click Download Zip File. The file (*MAK-LROM-v2-distributions-xp-vc71-v5.2.2.zip* or *MAK-LROM-v2-distributions-xp-vc80-v5.2.2.zip*) is downloaded.

B.2.2. Installing the MÄK Object Model

To install the MÄK Object Model:

1. Download the MÄK Object Model files as described in “[Downloading the MÄK Object Model](#),” on page B-3.
2. Extract all files from the object model file (for example, *MAK-LROM-v2-distributions-xp-vc71-v5.2.2.zip*). The extracted files are either *.exe* files or zip files.
3. Run all of the *.exe* files. They are self-extracting zip files. Make sure the destination folder is the same as the TENA installation folder.



When you extract the files, do not select:

When done unzipping open: `TENA\5.2.2\scripts\post.install.bat`

B.2.3. Building the MÄK Object Model Basic Implementation

The dynamic libraries are included in the VR-Exchange installation. You do not need to build them unless you build a customized TENA broker.

To build the MÄK Object Model basic implementation:

1. Download the MÄK Object Model files as described in “[Downloading the MÄK Object Model](#),” on page B-3.
2. Extract all files from the object model zip file (for example, *MAK-LROM-v2-distributions-xp-vc71-v5.2.2.zip*). The extracted files are either *.exe* files or zip files.
3. Unzip each of the zip files that you extracted. Make sure the destination folder is the same as the TENA installation folder.
4. Open MS VC++ 7.1.
5. Open and build *TENA\5.2.2\src\MAK-LROM-v2\MAK-LROM-v2-2003.sln*.

i

MS VC++ 7.1 has experienced problems with large amounts of text in the Additional Include Directories project setting. If header files cannot be found, reduce the amount of text in this project setting. For example, change all instances of `$(TENA_HOME)\$(TENA_VERSION)\include` to `C:\TENA\5.2.2\include`.

6. Create the directory: *TENA\5.2.2\include\impls\MAK-LROM-v2-basic-v1*.
7. Copy *TENA\5.2.2\src\MAK-LROM-v2*.h* (2 files) to *TENA\5.2.2\include\impls\MAK-LROM-v2-basic-v1*.
8. Copy the files in *TENA\5.2.2\src\MAK-LROM-v2\MAK* to *TENA\5.2.2\include\impls\MAK-LROM-v2-basic-v1\MAK*.
9. Copy *TENA\5.2.2\src\MAK-LROM-v2*.lib* and **.exp* (2 files) to *TENA\lib*.
10. Copy *TENA\5.2.2\src\MAK-LROM-v2*.dll* (1 file) to *TENA\5.2.2\bin\xp-vc71*.

B.3. Running the TENA Broker

For the TENA broker to communicate with other TENA executions, two TENA services must be running: the Network Naming Service and the Execution Manager. These services do not have to reside on the same machine as VR-Exchange. However, they must be able to access that machine via TCP/IP.

1. Verify that the following environment variables are set:
 - TENA_BOOST_LIB_VERSION=1_31
 - TENA_BOOST_VERSION=1.31.0a
 - TENA_HOME=C:\TENA (should point to your TENA installation path)
 - TENA_OMC_VERSION=0
 - TENA_OS=xp
 - TENA_PLATFORM=xp-vc71
 - TENA_VERSION=5.2.2
2. Make sure that the PATH includes:

```
%TENA_HOME%\%TENA_VERSION%\bin\%TENA_PLATFORM%;  
%TENA_HOME%\%TENA_VERSION%\scripts;
```
3. Change directory to the TENA installation directory (for example, *TENA\5.2.2\bin\xp-vc71*) .
4. Start the Network Naming Service by running:

```
networkNamingService.exe -ORBlistenEndpoints  
iiop://10.0.1.250:50509
```
5. Start the Execution Manager by running:

```
executionManager.exe -ORBdefaultInitRef  
corbaloc:iiop:10.0.1.250:50509  
-executionname VR-Link
```
6. Start VR-Exchange.
7. Configure the TENA connection. Make sure that the connection attributes match the values you used to start the Network Naming Service and the Execution Manager, for example:
 - ORB Default Init Ref: corbaloc:iiop:10.0.1.250:50509
 - ORB Default Listen Endpoints: iiop://10.0.1.250
 - Execution Name: VR-Link.

For details about configuring a TENA connection, please see [“Configuring Connections,”](#) on page 4-5.

B.3.1. TENA Services

The information in this section is a very basic introduction to the TENA services.

Network Naming Service

The Network Naming Service (NNS) is a registry for TENA Execution Managers. The NNS must specify its ORBlistenEndpoints, which is where it should listen for requests, for example:

```
networkNamingService -ORBlistenEndpoints iiop://10.1.1.29:50509
```

The Exercise Manager

An Execution Manager registers its IP address and execution name with the NNS. Each TENA execution has one Execution Manager. In [Figure B-2](#), two TENA executions are connected to the NNS. Execution VR-Link is running VR-Exchange.

```
executionManager -ORBdefaultInitRef  
corbaloc:iiop:10.1.1.29:50509 -executionname VR-Link
```

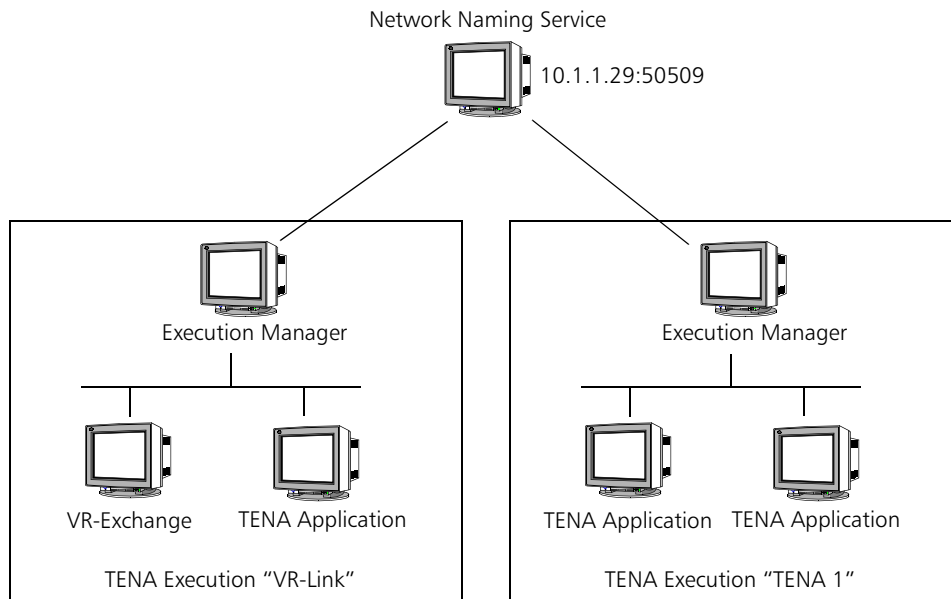
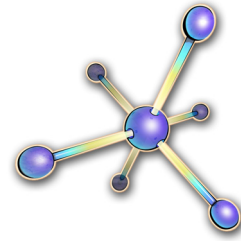


Figure B-2. TENA executions and Execution Manager



MÄK Products Glossary

This glossary defines terms used in MÄK product documentation.

absolute dead reckoning

A method of [dead-reckoning](#) in which an application uses the time stamps contained within state updates if the sender is using absolute time stamping. Contrast with [relative dead reckoning](#).

aggregate

The combination of individual entities, aggregates, or both into a single object. For example, an organizational group such as a platoon.

AMO

See [Application Management Object](#).

Application Management Object

An Application Management Object (AMO) is a standard TENA object that is supposed to be used by every TENA application. It gives other applications information about the local application. The VR-Link exercise connection, by default, publishes an AMO object, updates it at a specified rate (the rate is part of the AMO's state repository; the default is 30 seconds), and updates the number of objects published and proxies (TENA reflected objects) held.

API

Application Programming Interface.

articulated part

A part of an entity that is capable of movement relative to the entity, such as a turret or gun.

attached part

An object that is attached to another object, such as a rocket attached to a jet.

body coordinate frame

A coordinate framework centered on an entity. To represent the orientation of an entity, an application uses Euler angles or a rotation matrix that rotates from a reference frame to the body coordinate frame.

channel

A channel renders the view of an observer in a 3D visualization application.

clipping

A feature that prevents the display of terrain and objects or parts of objects, that are closer than or farther than specific distances from the observer.

clipping planes

The range of distances from the observer (near clipping and far clipping) in which an application clips objects.

culling

The process of discarding database objects which are not within view, and therefore need not be rendered and displayed.

Cartesian coordinates

A system for indicating location by means of three planes intersecting at right angles to one another at the origin.

Data Distribution Management

The set of HLA services used to specify the routing of data between publishers and subscribers. DDM adds greater control over the Declaration Management (DM) services that only specify filtering based on the class of the data (e.g., ground vehicle versus aircraft). DDM uses the data content to express regions of interest and regions of affect whose overlap establish a communication channel. An example is geographic filtering where a publisher indicates a region around the entity or effect and a subscriber indicates a region expressing its sensor range.

dead-reckoning

A process by which an application calculates the expected location of an entity during periods between state updates, based on velocity, acceleration, and rotational velocity.

DDM

[Data Distribution Management.](#)

Defense Modeling and Simulation Office

The former name of the executive secretariat for the Executive Council on Modeling and Simulation (EXCIMS), which provided a full-time focal point for information concerning Department of Defense modeling and simulation (M&S) activities. Renamed Modeling and Simulation Coordination Office. Currently the MSCO promulgates M&S policy, initiatives, and guidance to promote cooperation among DoD components to maximize efficiency and effectiveness.

DIS

[Distributed Interactive Simulation](#).

DIS data packets

See [Protocol Data Unit](#).

display engine

An object in an application that can render 3D data.

Distributed Interactive Simulation

The DIS protocol is a set of standards that govern how participating applications share information about the virtual world. The protocol specifies a set of packets, called protocol data units (PDUs), that communicate this information. Each PDU identifies the sender and contains other information depending on the PDU type. The DIS protocol also specifies when and how frequently PDUs are sent.

The DIS protocol has been superseded by the [High-Level Architecture](#) for use by the Department of Defense.

DMSO

See [Defense Modeling and Simulation Office](#).

emitter

A simulated device on an entity that emits electromagnetic radiation, for example, radar.

entity

An element in a simulation, such as a vehicle or a person, that is represented in the simulation through the issuance of state messages.

entity maintenance

A process by which the MÄK Data Logger compensates for discontinuities following a time jump. The Logger sends out interim state messages for entities that were present before the time jump so that they do not time out before the next update message arrives.

entity-type

A list of seven components as specified by the DIS protocol and the HLA RPR FOM. If none of the seven components contains a wildcard (-1) value, then the entity type refers to a specific, narrowly-defined entity. If some components contain a wildcard, then the entity type refers to a class of entities. A larger number of wildcards indicates a broader, more general class.

The components of an entity-type are:

- ♦ Entity kind
- ♦ Domain
- ♦ Country
- ♦ Category
- ♦ Subcategory
- ♦ Specific
- ♦ Extra.

environmental process

According to the IEEE 1278.1a specification (DIS), environmental process PDUs communicate simple environment variables, small scale environmental updates, and embedded processes.

Euler angles

A set of three angles used to describe the orientation of an entity as a set of three successive rotations about three different orthogonal axes (x, y, and z). These angles specify successive rotations needed to transform from a reference coordinate system to the entity's body coordinate system.

event

An interaction between objects or between an object and the terrain, such as firing of a munition, or a collision of entities.

execution

The TENA term for one or more interacting simulation applications.

Execution Manager

An Execution Manager governs a TENA execution for applications joining, resigning, or changing subscriptions to that execution.

exercise

The DIS term for a one or more interacting simulation applications. Compare to [federation execution](#) in HLA.

exercise connection

The object (*DtExerciseConn*) through which a VR-Link application connects to the network. State messages are sent through the exercise connection and information about remote entities is received through the exercise connection. (All MÄK products are VR-Link applications.)

exercise ID

A numeric identification for a DIS simulation exercise.

FED file

federate

A connection to the [RTI](#). Typically a single simulation application can be thought of as a federate.

federation

A group of HLA federates capable of playing in the same [federation execution](#).

Federation Object Model

Defines the data content of a federation execution.

federation execution

The federation execution represents the actual operation, over time, of a subset of the federates and the RTI initialization data taken from a particular federation. A federation execution is the logical equivalent of a DIS simulation exercise.

field of view

Controls the perspective of the observer.

A wide field of view creates an effect like that of a wide-angle camera lens. Objects appear smaller and farther away from the observer, since the observer coverage spans a wider area. Depth become exaggerated.

A narrow field of view creates an effect like that of a telephoto lens. Objects appear larger and closer to the observer, and the overall scene depth appears flattened. The distances between objects appears compressed.

filter range

A setting that prevents distant entities from being processed while allowing distant terrain to appear normally.

FOM

See [Federation Object Model](#).

FOM Module

In HLA Evolved, defines additional modular data content for a federation execution, usually extensions to an existing FOM. (Also supported by the HLA 1.3 and HLA 1516 versions of the MÄK RTI. Per the HLA Evolved interface specification:

“A partial FOM (containing some or all OMT tables) that specifies a modular component of a FOM. A FOM module may contain classes not inherent to it but upon which the FOM module depends, i.e., superclasses to the modular components. These superclasses will be included in the FOM module either as complete or scaffolding definitions.”

frame rate

The rate at which the application displays updated images.

ground clamping

A process by which the a 3D visualization application keeps an entity anchored to the surface of its terrain database, regardless of the altitude data contained in its entity state message.

geocentric coordinates

A coordinate system calculated with respect to the earth's center. The origin of the geocentric coordinate system is the center of the earth. The positive X-axis passes through the prime meridian at the equator; the positive Y-axis passes through 90 degrees east longitude at the equator; and the positive Z-axis passes through the north pole.

geodetic coordinates

A coordinate system in which position is determined relative to a reference ellipsoid, such as the surface of the earth at sea level. In MÄK applications, geodetic coordinates consist of latitude and longitude in radians, and altitude in meters above the reference ellipsoid.

gridded data

Data that has been processed in a rectangular array of points, in X, Y or latitude/longitude, at which single data values define a two dimensional function. According to the IEEE 1278.1a specification, gridded data transmits information about large-scale or high-fidelity spatially and temporally varying ambient fields and about environmental processes and features.

guise

An alternative entity type used to display an object depending on the force ID. For example, a tank could look like an M1A1 to friendly forces and a T72 to hostile forces.

head-up display

A set of indicators and readouts superimposed onto a graphics display. Also called an overlay.

heartbeat

In DIS, the frequency with which current PDUs are sent to the network regardless of whether or not the entity's state has changed.

High-Level Architecture

The High Level Architecture (HLA) for simulations is a U. S. Department of Defense (DOD)-wide initiative to provide an architecture to support interoperability and reuse of simulations. The HLA supersedes DIS for the DOD.

HLA

See [High-Level Architecture](#).

interaction

A [message](#) describing a simulation event. An interaction describes an event, it does not update an object's state.

Logger control PDUs

logical range

A suite of TENA Resources, sharing a common object model, that work together for a given range event. Similar in concept to the HLA Federation.

Local RTI Component

The libraries on a computer that implement an RTI. A federate communicates with the HLA network through the LRC.

Logical Range Object Model

The data definition file for TENA exercises. It contains the set of object and message definitions that may be used in the exercise.

LRC

See [Local RTI Component](#).

LROM

See [Logical Range Object Model](#).

MÄK Technologies Lisp

An adaptation of the Lisp language used in configuration files for MÄK products.

message

A general term used to refer to [DIS PDUs](#) and [HLA interactions](#) and state updates. In TENA, a message is similar to an HLA interaction.

MIM

The MOM & Initialization Module (MIM) allows for extensions to be made to the HLA standard MOM. It is a subset of the FOM that contains OMT tables that describe the HLA MOM. The MIM also contains additional predefined HLA constructs such as object and interaction roots, data types, transportation types and dimensions. HLA specifies a standard MIM that is incorporated into all FDDs automatically by the RTI. The standard MIM can be replaced with a user-supplied MIM containing the standard MIM plus extensions.

Modeling and Simulation Coordination Office

The executive secretariat for the Executive Council on Modeling and Simulation (EXCIMS), which provided a full-time focal point for information concerning Department of Defense modeling and simulation (M&S) activities. The MSCO promulgates M&S policy, initiatives, and guidance to promote cooperation among DoD components to maximize efficiency and effectiveness. (Formerly DMSO)

MSCO

Modeling and Simulation Coordination Office. (Formerly DMSO.)

MTL

See [MÄK Technologies Lisp](#).

Network Naming Service

Acts as a registry for TENA Execution Managers.

NNS

See [Network Naming Service](#).

orientation clamping

Adjusts the pitch and roll of an entity so that it appears properly seated when the terrain is inclined. For example, if a tank is moving horizontally across the face of a hill, orientation clamping prevents the tank from appearing level, and therefore, partially embedded into the hillside. Used with [ground clamping](#).

object

An element in a simulation that has persistence, as opposed to an interaction, which is a transient element.

object handle

An integer that an application uses to identify a particular object in RTI service calls. An object handle is meaningful only to a particular federate. The same object can be known to different federates by different object handles.

object name

A character string that can be used to identify an object. In HLA, the object name is known to the RTI, and the RTI provides functions to find out an object's name, given its handle, and vice versa. Object names can be chosen by applications that register the objects with the RTI, however if you do not want to choose names for objects, the RTI will assign names for you.

observer

In MÄK 3D applications, the point of view into the scene. (Called the eyepoint in MÄK Stealth 6.x and earlier.)

PDU

See [Protocol Data Unit](#).

Protocol Data Unit

A unit of data message (packet) that is passed on a network between DIS simulation applications.

proxy

In TENA, a reflected object ([Stateful Distributed Object](#)).

radio transmitter

A simulated device on an entity, capable of transmitting radio communications.

radio receiver

A simulated device on an entity, capable of receiving radio communications.

range resource

A participant in a TENA execution (similar in concept to the HLA federate).

Realtime Platform Reference FOM

An [HLA](#) reference [FOM](#) based on the [DIS](#) protocol.

recording

A Data Logger file that stores a history of the interactions in a simulation for playback.

reflected entity list

A list of entities simulated by remote applications (federates in HLA) and about which the local simulation has received information over the network.

reference FOM

A [FOM](#) designed to be used as a whole, or with modification, by a wide variety of similar [federation executions](#).

relative dead reckoning

A method of dead reckoning in which an application approximates the location of an object based on the local time of receipt of the last state update message.

remote display engine

A display engine running in an application that is separate from the master application, and which is controlled by the master application.

RID file

See [RTI Initialization Data](#).

RTI Initialization Data

The initialization data required by the RTI for operation.

Data required by an RTI during initialization, independent of the FOM being used. RID data is usually dependent on a specific implementation of the RTI.

RPR FOM

See [Realtime Platform Reference FOM](#).

RTI

See [Run-Time Infrastructure](#).

Run-Time Infrastructure

A library and other supporting software that implements the HLA interface specification. All federates communicate with one another in an HLA environment through RTI functions.

SDO

See [Stateful Distributed Object](#).

servant

A published object (SDO) in TENA.

Simulation Interoperability Standards Organization

A group that seeks to promote modeling and simulation interoperability and reuse for the benefit of diverse M&S communities, including developers, procurers, and users, world-wide.

simulation time

In VR-Forces, simulation time is used in dead-reckoning of remote entities and thresholding of local entities. Typically, simulation time is set once during each iteration of the application's main simulation loop so that all entities are dead-reckoned based on the same value of current time.

SISO

See [Simulation Interoperability Standards Organization](#).

smooth period

The period of time over which [trajectory smoothing](#) takes place.

smoothing

A method of ensuring that transitions from an entity's dead-reckoned position to its actual position are not so abrupt as to be visually disconcerting.

state

The current status of an object, including location, direction of movement, extent of damage, and so on.

Stateful Distributed Object

An object in a TENA exercise.

tape

An alternative term for a Logger recording. Not a physical magnetic tape.

TENA

See [Test and Training Enabling Architecture](#).

TENA Middleware

“The software that links range resources together into a logical range. The “TENA Middleware” is the software component that is in between (that is, in the middle of) range resources during a logical range execution.” – from www.tena-sda.org

terrain following

In a 3D visualization application, causes the observer (eyepoint) to maintain a constant distance above the terrain surface. The observer's height changes in tandem with the peaks and valleys as it passes over the geography.

Test and Training Enabling Architecture

Test and Training Enabling Architecture (TENA) is an interoperability architecture used to pass data in the form of object updates and messages (like HLA objects and interactions) from one application to another.

timeout

The period of time in which an application continues to display an entity after that entity's update messages have stopped appearing on the network. Time-outs are usually not used in HLA because there is no set frequency (no heartbeat) for transmitting messages.

timescale

A factor by which time is accelerated or slowed during playback of a Data Logger recording.

topographic coordinates

A right-handed Cartesian coordinate system whose X-Y plane is tangent to the earth's surface at the origin, with the positive X axis pointing north, the positive Y axis pointing east, and the positive Z axis pointing down. There are an infinite number of topographic coordinate systems – one for each point on the earth's surface.

topographic coordinate frame

A coordinate frame in the context of the terrain.

trajectory smoothing

A method used by to smooth positional discontinuities that could occur when new state updates arrive.

UDP port

A network channel through which an application sends and receives data for DIS exercises.

Universal Transverse Mercator

In general, a non-cartesian coordinate system in which the X, Y, and Z correspond to easting (nearly east), northing (nearly north), and height above an ellipsoid that approximates the surface of the earth.

UTM

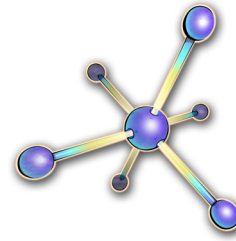
See [Universal Transverse Mercator](#).

view control messages

A set of programmatic messages that let you control the view in VR-Vantage applications remotely.

view floor

In MÄK Stealth 6.x and earlier, a minimum height above the terrain in Absolute or Track mode, below which the observer is not allowed to go. The view floor is measured relative to the terrain surface.



Index

- command-line option 3-4, A-6, A-9, A-13
 - applicationNumber command-line option A-9
 - brokerLogFileName command-line option A-7, A-10, A-14
 - brokerName command-line option A-6, A-9, A-13
 - configFile command-line option 3-4, A-7, A-10, A-14
 - controlQueueName command-line option 3-4, A-6, A-9, A-13
 - destinationAddress command-line option A-9
 - disMcastDevice command-line option A-9
 - disPort command-line option A-10
 - disSendPort command-line option A-10
 - dontStartBrokers command-line option 3-4
 - dumpEnvironment command-line option A-6, A-10
 - execName command-line option A-8, A-14
 - exercisId command-line option A-10
 - federateName command-line option A-7
 - federateType command-line option A-7
 - fedFileName command-line option A-7
 - fomMapperInitData command-line option A-7
 - fomMapperLibName command-line option A-6
 - help command-line option 3-4, A-7, A-10, A-13
 - ignore_rest command-line option 3-4, A-6, A-9, A-13
 - interactionQueueName command-line option 3-4, A-7, A-10, A-13
 - localSettingsDesignator command-line option A-7
 - logFileName command-line option 3-4
 - lromMapper command-line option A-14
 - maxInteractionsToShow command-line option 3-4
 - maxPortalControlMessages command-line option 3-4
 - maxPortalInteractionMessages command-line option 3-4
 - maxPortalObjectMessages command-line option 3-4
 - noGui command-line option 3-4, A-7, A-10, A-13
 - notifyLevel command-line option 3-4, A-7, A-10, A-14
 - objectQueueName command-line option 3-4, A-7, A-10, A-14
 - oneToOneMapping command-line option A-7, A-10, A-14
 - OrbInitRef command-line option A-13
 - OrbListenEndpoints command-line option A-13
 - queueBucketCount command-line option 3-4
 - queueBucketPayloadSize command-line option 3-4
 - rprFomRevision command-line option A-7
 - rprFomVersion command-line option A-7
 - showInfoDialog command-line option A-7, A-10, A-14
 - siteId command-line option A-10
 - subnetmask command-line option A-10
 - tcpPort command-line option 3-5, A-7, A-10, A-14
 - version command-line option 3-5, A-7, A-10, A-14
-
- ## A
- A command-line option A-9
 - a command-line option A-9
 - absolute timestamp 4-14

- Acknowledge* class 1-11
- Acknowledge PDU 1-11
- Action Response PDU 1-11
- ActionRequest* class 1-11
- ActionResponse* class 1-11
- Add New Connection dialog box 4-2, 4-4
- adding, connections 4-2
- address
 - destination, DIS 4-20, A-9
 - multicast 4-20, 4-22
- AMO Host IP Address attribute 4-17
- AMO Hostname attribute 4-16
- AMO Name attribute 4-16
- AMO Operator Name attribute 4-17
- AMO Operator Telephone Number attribute 4-17
- API, VR-Exchange 1-6
- application number
 - configuring DIS 4-14, 4-16
 - DIS A-9
- Application Number attribute 4-14, 4-16
- ApplicationSpecificRadioSignal* interaction 1-12
- architecture 1-2
- attribute
 - AMO Host IP Address 4-17
 - AMO Hostname 4-16
 - AMO Name 4-16
 - AMO Operator Name 4-17
 - AMO Operator Telephone Number 4-17
 - Application Number 4-14, 4-16
 - Built In FOM Mapper 4-19
 - collection 4-7
 - connection 4-5
 - Control Queue Name 3-7
 - Debug Entity ID Mappings 4-15, 4-17
 - Debug Event ID Mappings 4-15
 - Debug Global ID Mappings 4-15, 4-17
 - Destination Address 4-20
 - DIS Maximum Protocol Version to Receive 4-14
 - DIS Minimum Protocol Version to Receive 4-14
 - DIS Multicast Device 4-20
 - DIS Multicast Subscription Set 4-20
 - DIS Protocol Version to Send 4-14
 - Disable Articulated Parts Translation 4-12, 4-15
 - Drain Timeout 4-15
 - Dump GUI to Text 4-15, 4-17
 - Execution Name 4-22
 - Exercise ID 4-20
 - attribute (continued)
 - Federate Name 4-20
 - Federate Type 4-19
 - Fill Outgoing Entity IDs 4-12, 4-16
 - FOM Mapper Initialization Data 4-19
 - FOM Mapper Library Name 4-19
 - Force Updates 4-16
 - Generic Interactions to Translate Set 4-13
 - Generic Objects to Translate Set 4-13
 - Guise is Entity Type 4-16
 - Heartbeat 4-13
 - Heartbeat Interval 4-15
 - HLA 4-12
 - HLA Execution Name 4-19
 - HLA FED File Name 4-19
 - HLA Maximum Tick Time 4-13
 - HLA Minimum Tick Time 4-13
 - Interaction Queue Name 3-7
 - Local Settings Designator 4-20
 - Log Filename 3-7, 4-9
 - LROM Mapper Name 4-22
 - Maximum Control Messages 3-7, 4-9
 - Maximum Interaction Messages 3-7, 4-9
 - Maximum Interactions to Show 3-7
 - Maximum Object Messages 3-7, 4-9
 - Multicast Interface 4-22
 - Multicast Time-to-Live 4-15, 4-22
 - Notify Level 3-6, 4-9
 - Number of PDUs Read Per Tick 4-15
 - Object Queue Name 3-7
 - Object Throttle Rates 4-17
 - One to One Message Mapping 3-7
 - ORB Default Init Ref 4-22
 - ORB Default Listen Endpoints 4-22
 - Outgoing Host ID 4-12
 - Outgoing Site ID 4-12
 - Parse Unknown IDs 4-14, 4-16
 - Polling Interval 4-9
 - Port 4-20
 - Queue Bucket Count 3-7
 - Queue Bucket Payload Size 3-7
 - Remap DIS IDs 4-14
 - Site ID 4-14, 4-16
 - Socket Receive Buffer Size 4-21
 - Socket Send Buffer Size 4-20
 - TCP Port 3-7
 - TENA 4-16
 - TENA Tick Time 4-17
 - TENA Transport Type 4-22
 - Timeout Interval 4-15

attribute (continued)

- Use Absolute Timestamps 4-12, 4-14
- Use Generic Translators for Whole FOM 4-13
- Use HLA Advisories 4-20
- Use HLA Object IDs For Marking Text 4-12
- Use Marking Text for HLA Object IDs 4-12
- Use Marking Text for TENA Object ID 4-16
- Use Persistent Entity IDs 4-12, 4-14, 4-16
- Use TENA Object ID for Marking Text 4-16

B

- BaseEntity* class 1-10
- beam ID 1-13
- broadcast address 4-20
- broker 1-2
 - API 1-6
 - configuring in connection 4-2
 - DIS 1-6, A-8
 - HLA A-2
 - command-line options A-6
 - ignoring command-line options A-6, A-13
 - name, DIS A-9
 - number supported 1-15
 - specifying configuration file for A-14
 - specifying name
 - HLA A-6
 - TENA A-13
 - starting 3-8
 - not 3-7
 - supplied 1-6
- bucket count, specifying 3-4
- bucket payload size 3-4
- buffer, DIS receive 4-21
- building, MÅK Object Model B-5
- Built In FOM Mapper attribute 4-19

C

class

- Acknowledge* 1-11
- ActionRequest* 1-11
- ActionResponse* 1-11
- BaseEntity* 1-10
- Collision* 1-10
- Comment* 1-11
- CreateEntity* 1-11
- Data* 1-11
- DataQuery* 1-11
- Designator* 1-12

class (continued)

- EmbeddedSystem* 1-12, 1-13
- EmitterSystem* 1-12, 1-13
- EmittingSystemID* 1-13
- EventReport* 1-11
- generic translation A-4, A-5
- GroundVehicle* 1-10
- IFF* 1-12
- IffInterrogator* 1-12, 1-13
- IffTransponder* 1-13
- interactions to translate 4-13
- JammerBeam* 1-13
- MunitionEntity* 1-10
- objects to translate 4-13
- PhysicalEntity* 1-10
- RadarBeam* 1-13
- RadioReceiver* 1-12
- RadioTransmitter* 1-12
- RemoveEntity* 1-11
- RepairComplete* 1-14
- RepairResponse* 1-14
- ResupplyCancel* 1-14
- ResupplyOffer* 1-14
- ResupplyReceived* 1-14
- ServiceRequest* 1-14
- SetData* 1-11
- StartResume* 1-11
- StopFreeze* 1-11
- clearing, interaction statistics 3-12
- clearing interaction list 3-12
- closing, VR-Exchange 3-15
- collection 4-7
 - removing values from 4-8
- Collision* class 1-10
- Collision PDU 1-10
- coLromMapper library A-12
- column
 - changing order of in object and interaction
 - windows 3-11
 - displaying and hiding in object and interaction
 - windows 3-11
 - sorting by 3-11
- command-line option 3-4
 - 3-4, A-6, A-9, A-13
 - A A-9
 - a A-9
 - applicationNumber A-9
 - brokerLogFileName A-7, A-10, A-14
 - brokerName A-6, A-9, A-13
 - configFile 3-4, A-7, A-10, A-14

command-line option (continued)

- controlQueueName 3-4, A-6, A-9, A-13
- destinationAddress A-9
- DIS A-9
- disMcastDevice A-9
- disPort A-10
- disSendPort A-10
- dontStartBrokers 3-4
- dumpEnvironment A-6, A-10
- E A-6, A-10
- e A-13
- execName A-8, A-14
- exercisId A-10
- F A-7
- f A-6
- federateName A-7
- federateType A-7
- fedFileName A-7
- fomMapperInitData A-7
- fomMapperLibName A-6
- G A-7, A-10, A-13
- h A-7, A-10, A-13
- help 3-4, A-7, A-10, A-13
- HLA broker A-6
- i A-13
- ignore_rest A-6, A-9, A-13
- interactionQueueName 3-4, A-7, A-10, A-13
- L 3-4, A-7, A-10, A-14
- l 3-4, A-7, A-10, A-14
- localSettingsDesignator A-7
- logFilename 3-4
- lromMapper A-14
- maxInteractionsToShow 3-4
- maxPortalControlMessages 3-4
- maxPortalInteractionMessages 3-4
- maxPortalObjectMessages 3-4
- N A-7
- n 3-4, A-7, A-10, A-14
- noGui 3-4, A-7, A-10, A-13
- notifyLevel 3-4, A-7, A-10, A-14
- O A-7, A-10, A-14
- objectQueueName 3-4, A-7, A-10, A-14
- oneToOneMapping A-7, A-10, A-14
- OrbInitRef A-13
- OrbListenEndpoints A-13
- P A-10
- p A-7
- queueBucketCount 3-4
- queueBucketPayloadSize 3-4
- rprFomRevision A-7

command-line option (continued)

- rprFomVersion A-7
- S A-7
- s A-10
- showInfoDialog A-7, A-10, A-14
- siteId A-10
- subnetmask A-10
- t A-7, A-10, A-14
- tcpPort 3-5, A-7, A-10, A-14
- v 3-5, A-7, A-10, A-14
- version 3-5, A-7, A-10, A-14
- x A-8, A-10, A-14
- Comment* class 1-11
- Comment PDU 1-11
- condition, publication 4-10
- configuration file
 - RTI 2-6
 - specifying alternative 3-4
 - specifying at startup A-7, A-14
 - specifying for broker A-10
- Configure page 4-6, 4-8, 4-11
- configuring
 - connection 4-5
 - collection 4-7
 - DIS application number 4-14, 4-16
 - discovery criteria 4-10
 - HLA connection 4-12
 - network connection 4-18
 - system for MÄK RTI 2-6
 - TENA connection 4-16
 - VR-Exchange 3-6
- connection
 - adding 4-2
 - adding and editing 4-2
 - attribute, collection 4-7
 - configuring 4-5
 - deleting 4-5
 - disabling 3-9
 - discovered and reflected objects 4-27
 - editing 4-4
 - enabling at startup 3-8
 - environment variable 4-4
 - HLA, attributes 4-12
 - network
 - configuring 4-18
 - DIS 4-20
 - HLA 4-19
 - TENA 4-22
 - provided 4-2

- connection (continued)
 - starting 3-8
 - manually 3-9
 - startup, disabling 3-4
 - TENA, attributes 4-16
 - translator, enabling 4-26
- Connection Information dialog box 4-5, 4-11, 4-18, 4-23, 4-24, 4-25, 4-26, 4-27
- Connection menu 3-5
- console mode 3-4, 3-5
- constraint 1-16
- context-sensitive menu 3-5
- control message, number to show 3-4
- Control Queue Name, attribute 3-7
- CPU 1-15
- Create Entity PDU 1-11
- CreateEntity* class 1-11
- criteria, discovery 4-10

D

- Data* class 1-11
- Data PDU 1-11
- Data Query PDU 1-11
- Data View 3-14
- DatabaseIndexRadioSignal* interaction 1-12
- DataQuery* class 1-11
- Debug Entity ID Mappings attribute 4-15, 4-17
- Debug Event ID Mappings attribute 4-15
- Debug Global ID Mappings attribute 4-15, 4-17
- debugging, output level 4-9
- deleting, connection 4-5
- Delta State EE PDU 1-13
- Designator* class 1-12
- Designator PDU 1-13
- destination address, DIS A-9
- Destination Address attribute 4-20
- Detonation PDU 1-11
- dialog box
 - Add New Connection 4-2, 4-4
 - Connection Information 4-5, 4-11, 4-18, 4-23, 4-24, 4-25, 4-26, 4-27
 - Edit Collection 4-7
 - Edit Connection 3-8, 4-4
 - Environment Variable 4-4
 - License Setup 2-4
 - New Filter 4-23
 - Queue Status 3-14
 - VR-Exchange Preferences 3-6

DIS

- application number A-9
- broker 1-6, A-8
- command-line options A-9
- drain timeout 4-15
- entity ID 4-12, 4-14
- event ID 4-15
- exercise ID 4-20
- heartbeat 4-15
- ID 4-14
- multicast address 4-20
- multicast subscriptions 4-20
- multicast time-to-live 4-15
- multicast TTL 4-21
- network connection parameters 4-20
- port 4-20, A-10
- protocol version 4-14
- site ID 4-14
- socket receive buffer size 4-21
- timeout 4-15
- timestamp 4-14
- DIS Maximum Protocol Version to Receive
 - attribute 4-14
- DIS Minimum Protocol Version to Receive
 - attribute 4-14
- DIS Multicast Device attribute 4-20
- DIS Multicast Subscription Set attribute 4-20
- DIS Protocol Version to Send attribute 4-14
- Disable Articulated Parts Translation, attribute 4-12, 4-15
- disabling
 - automatic broker start 3-7
 - automatic connection startup 3-4
 - connections 3-8, 3-9
 - Portal 3-4, 3-5
 - translators 4-26, A-3
- discovery criteria 4-10
 - configuring 4-10
- display, filtering object data 3-11
- displaying
 - columns in object and interaction windows 3-11
 - connection information 4-27
 - Data View 3-14
 - interaction list 3-11
 - shared memory queue status 3-14
 - version A-7, A-10, A-14
 - version information 3-4, 3-5
- Distributed Emission Regeneration protocol
 - family 1-13

- downloading
 - MÄK Object Model B-3
 - TENA Middleware B-2
- drain timeout 4-15
- Drain Timeout attribute 4-15
- drainInput() function 4-9
 - time allowed 4-17
- Dump GUI to Text attribute 4-15, 4-17

E

- E command-line option A-6, A-10
- e command-line option A-13
- Edit Collection dialog box 4-7
- Edit Connection, dialog box 3-8
- Edit Connection dialog box 4-4
- editing
 - collection 4-8
 - connection 4-4
 - connections 4-2
 - filter 4-24
- EE PDU 1-13
- Electromagnetic Emission PDU 1-13
- embedded system object, deletion of 1-8
- EmbeddedSystem* class 1-12, 1-13
- emission 1-13
- emitter beam 1-13
- emitter system 1-13
- EmitterSystem* class 1-12, 1-13
- emitting system ID 1-13
- EmittingSystemID* class 1-13
- enabling
 - connection, manually 3-9
 - connection at startup 3-8
 - connections 3-8
 - translators 4-26
- EncodedAudioRadioSignal* interaction 1-12
- entity ID 1-10, 1-12, 4-12
 - DIS 4-14, 4-15
 - TENA 4-16
- Entity Information 1-10
- Entity Management PDU family 1-14
- Entity State PDU 1-10
- entity type 1-10
 - filtering by 4-23
- environment variable 4-4
 - MAKLMGRD_LICENSE_FILE 2-5
 - RTI_CONFIG 2-6
- Environment Variable dialog box 4-4
- Environmental Process PDU 1-14

- EnvironmentProcess class 1-14
- event ID, DIS 4-15
- Event Report PDU 1-11
- EventReport* class 1-11
- execution
 - specifying execution A-14
 - specifying TENA 4-22
- Execution Manager A-11, B-7
- Execution Name attribute 4-22
- exercise
 - ID A-10
 - DIS 4-20
 - site ID A-10
- exercise connection
 - polling interval 4-9
 - TENA tick time 4-17
- Exercise ID attribute 4-20
- exiting
 - connection 3-9
 - VR-Exchange 3-15

F

- F command-line option A-7
- f command-line option A-6
- FDD file 2-6
- FED file 2-6
 - HLA broker 4-19
 - specifying A-7
- federate
 - name, specifying A-7
- Federate Name, attribute 4-20
- Federate Type attribute 4-19
- federation
 - name 4-19
 - specifying name A-8
- federation of federations 1-4
- file
 - FED 2-6
 - log 3-4
 - RID 2-6
- filename, log file 3-7, 4-9
- Fill Outgoing Entity IDs attribute 4-12, 4-16
- filter
 - editing 4-24
 - removing 4-25
- Filter page 4-23
- filtering
 - object display 3-11
 - objects 3-13, 4-23

- filtering (continued)
 - objects and interactions A-3
 - translators 4-26
- Fire PDU 1-11
- FLEXlm 2-3
- FOM 2-6
 - agility 1-5
 - generic translation A-4, A-5
 - module 4-20
- FOM Mapper 1-5, 1-6
 - library 4-19, A-6
 - initialization data 4-19
 - specifying initialization data A-7
 - specifying library A-6
- FOM Mapper Initialization Data attribute 4-19
- FOM Mapper Library Name attribute 4-19
- Force Updates attribute 4-16
- function, drainInput() 4-9

G

- G command-line option A-7, A-10, A-13
- generic interaction translator 4-13
- Generic Interactions to Translate Set attribute 4-13
- generic object translator 4-13
- Generic Objects to Translate Set attribute 4-13
- generic translation A-4
 - FOM A-5
 - FOM class A-5
 - HLA 4-13
- global ID 4-15
 - TENA 4-17
- graphical user interface, running without 3-5
- Gridded Data PDU 1-14
- GriddedData class 1-14
- GroundVehicle* class 1-10
- GUI 1-5
 - logging to file 4-15, 4-17
 - maximum number of interactions 3-7
 - running without 3-4, 3-5
- Guise is Entity Type attribute 4-16

H

- h command-line option 3-4, A-7, A-10, A-13
- hardware requirement 1-15
- heartbeat 1-8, 1-13
 - DIS 4-15
- Heartbeat attribute 4-13

- Heartbeat Interval attribute 4-15
- help
 - command line A-10
 - displaying command-line option 3-4
- hiding
 - columns in object and interaction windows 3-11
 - Data View 3-14
 - interaction list 3-11
- histogram 3-14
- HLA
 - advisories 4-20
 - broker 1-6, A-2
 - configuring, discovery criteria 4-10
 - connection attributes 4-12
 - generic translation 4-13
 - network connection parameters 4-19
 - object ID 4-12
- HLA broker
 - command-line options A-6
 - FED file name 4-19
 - specifying federate name A-7
- HLA Execution Name attribute 4-19
- HLA FED File Name attribute 4-19
- HLA ID, filtering display of 3-11
- HLA Maximum Tick Time attribute 4-13
- HLA Minimum Tick Time attribute 4-13
- host entity ID 1-12
- host ID 4-12
- host object ID 1-12
- hostname, license server 2-4

I

- i command-line option A-13
- ID
 - DIS 4-14
 - entity 4-15
 - site 4-14
 - entity 4-12
 - exercise 4-20, A-10
 - HLA object 4-12
 - host 4-12
 - mapping Portal to DIS 4-15
 - site 4-12, A-10
 - TENA
 - entity 4-16
 - global 4-17
 - site 4-16
 - using marking text for 4-16

- ID (continued)
 - using object as marking text 4-16
- IFF* class 1-12
- IFF PDU 1-13
- IffInterrogator* class 1-12, 1-13
- IffTransponder* class 1-13
- ignoring command-line options 3-4, A-6
- information
 - connection, displaying 4-27
- installing
 - MÄK Object Model B-3, B-4
 - RTI 2-6
 - TENA Middleware 2-7, B-2
- interaction 1-10
 - ApplicationSpecificRadioSignal* 1-12
 - classes to translate 4-13
 - clearing list 3-12
 - collision 1-10
 - DatabaseIndexRadioSignal* 1-12
 - EncodedAudioRadioSignal* 1-12
 - filtering A-3
 - generic translation A-4, A-5
 - hiding or displaying list of 3-11
 - maximum shown in GUI 3-7
 - MunitionDetonation* 1-11
 - RadioSignal* 1-12
 - RawBinaryRadioSignal* 1-12
 - viewing discovered 3-10
 - WeaponFire* 1-11
 - window, customizing 3-11
- interaction message, number to show 3-4
- Interaction Queue Name, attribute 3-7
- introduction 1-2

J

- JammerBeam* class 1-13

L

- L command-line option 3-4, A-7, A-10, A-14
- l command-line option 3-4, A-7, A-10, A-14
- License Manager 2-3
- license server, hostname 2-4
- License Setup dialog box 2-4
- limitation, translation software 1-16
- Linux, installing on 2-2
- Local Settings Designator, attribute 4-20
- log file 3-4
 - name 3-7, 4-9

- log file (continued)
 - notification level 3-6, A-7, A-10, A-14
 - output level 4-9
- Log Filename attribute 3-7, 4-9
- Logical Range Object Model A-11
- Logistics protocol family 1-14
- LROM A-11
 - agility 1-5
- LROM Mapper 1-5, 1-6, A-12
 - specifying 4-22, A-14
- LROM Mapper Name attribute 4-22

M

- MÄK Object Model 2-7
 - building B-5
 - downloading B-3
 - installing B-3, B-4
- MÄK Object Model Basic Implementation 2-7
- MÄK RTI, configuring 2-6
- MAKLMGRD_LICENSE_FILE, environment
 - variable 2-5
- makLromMapper A-12
- manual, connection startup 3-9
- mapper, FOM and LROM 1-5
- mapping, entity type to object class 1-10
- marking text
 - filtering by 4-23
 - filtering display of 3-11
 - HLA objects 4-12
 - using for object ID 4-16
 - using object ID for 4-16
- Maximum Control Messages attribute 3-7, 4-9
- Maximum Interaction Messages attribute 3-7, 4-9
- Maximum Interactions to Show, attribute 3-7
- Maximum Object Messages attribute 3-7, 4-9
- mcast address
 - configuring for DIS 4-20
 - configuring for TENA 4-22
- menu
 - Connection 3-5
 - context-sensitive 3-5
- message
 - control, number to show 3-4
 - displaying in console 3-15
 - interaction, number to show 3-4
 - maximum number of 3-7

message (continued)
 maximum Portal 3-7
 control 4-9
 interaction 4-9
 object 4-9
 maximum size of 3-7
 notification 3-4, A-10
 object, number to show 3-4
 module, FOM 4-20
 multicast
 device A-9
 time-to-live 4-15, 4-22
 multicast address
 configuring for DIS 4-20
 configuring for TENA 4-22
 subscribed to 4-20
 Multicast Interface attribute 4-22
 Multicast Time-to-Live attribute 4-15, 4-22
 multicast TTL, DIS 4-21
MunitionDetonation interaction 1-11
MunitionEntity class 1-10

N

-N command-line option A-7
 -n command-line option 3-4, A-7, A-10, A-14
 name
 DIS broker A-9
 federation execution 4-19
 filtering by 4-23
 log file 3-7
 specifying federate A-7
 network, frequency of reading 4-9
 network connection
 configuring 4-18
 HLA 4-19
 parameter
 DIS 4-20
 TENA 4-22
 Network Naming Service B-7
 New Filter dialog box 4-23
noGui parameter 3-5
 notification level 3-6, 4-9, A-10
 setting 3-4
 specifying A-7, A-14
 Notify Level, attribute 3-6
 Notify Level attribute 4-9
 Number of PDUs Read Per Tick attribute 4-15

O

-O command-line option A-7, A-10, A-14
 object
 classes to translate 4-13
 discovered by connection 4-27
 entity type, filtering by 4-23
 filtering 3-13, 4-23, A-3
 data display 3-11
 generic translation A-4, A-5
 HLA, configure discovery criteria 4-10
 marking text, filtering by 4-23
 name, filtering by 4-23
 reflected by connection 4-27
 sorting display of 3-11
 viewing discovered 3-10
 window, customizing 3-11
 object ID
 HLA 4-12
 mapping to DIS 4-15
 using as marking text 4-16
 using marking text as 4-16
 object message, number to show 3-4
 Object Queue Name, attribute 3-7
 Object Throttle Rates attribute 4-17
 One to One Message Mapping, attribute 3-7
 option, command-line 3-4
 ORB Default Init Ref attribute 4-22
 ORB Default Listen Endpoints attribute 4-22
 ORB Initialization Reference, specifying 4-22, A-13
 ORB Listen Endpoint B-7
 specifying 4-22, A-13
 ORBlistenEndpoints parameter A-13
 Outgoing Host ID attribute 4-12
 Outgoing Site ID attribute 4-12
 overview 1-2

P

-P command-line option A-10
 -p command-line option A-7
 packet, count 3-14
 parameter
 network connection
 HLA 4-19
 TENA 4-22
 noGui 3-5
 ORBlistenEndpoints A-13
 Subnet Mask 4-21

- paramter
 - network connection, DIS 4-20
- Parse Unknown IDs attribute 4-14, 4-16
- parsing, DIS ID 4-14
- path, specifying for RTI 2-6
- PDU
 - Acknowledge 1-11
 - Action Response 1-11
 - Collision 1-10
 - Comment 1-11
 - Create Entity 1-11
 - Data 1-11
 - Data Query 1-11
 - delta state EE 1-13
 - designator 1-13
 - destination address 4-20
 - detonation 1-11
 - electromagnetic emission 1-13
 - Entity Management 1-14
 - Entity State 1-10
 - Event Report 1-11
 - fire 1-11
 - IFF 1-13
 - protocol families 1-8
 - radio transmitter 1-12
 - Remove Entity 1-11
 - Repair Complete 1-14
 - Repair Response 1-14
 - Resupply Cancel 1-14
 - Resupply Offer 1-14
 - Resupply Received 1-14
 - service request 1-14
 - Set Data 1-11
 - simulation management (SIMAN) 1-11
 - Start/Resume 1-11
 - state update EE 1-13
 - Stop/Freeze 1-11
 - supported 1-8
 - Synthetic Environment 1-14
- performance 1-15, 1-16
 - limiting number of interactions displayed 3-7
 - object throttle rate 4-17
 - tuning message traffic 3-7
- PhysicalEntity* class 1-10
- Polling Interval attribute 4-9
- popup menu 3-5
- port
 - DIS 4-20, A-10
 - send A-10
 - specifying TCP 3-7

- port (continued)
 - TCP A-10
 - TENA A-14
- Port attribute 4-20
- Portal 1-2, 1-5
 - closing 3-15
 - configuring 3-6
 - control message, maximum 4-9
 - disabling 3-4, 3-5
 - interaction message, maximum 4-9
 - interface 1-6
 - maximum number of interactions 3-7
 - object message, maximum 4-9
 - starting
 - Linux 3-2
 - Windows 3-3
 - startup, enabling connection 3-8
 - tuning message traffic 3-7
- Portal window 3-5
- position attribute 1-10
- protocol
 - DIS supported 1-6, A-8
 - DIS version 4-14
 - family 1-8
- publication condition 4-10
- Publishing page 4-27

Q

- queue
 - control shared memory, name 3-4
 - interaction shared memory, name 3-4
 - maximum number of messages 3-7
 - number of messages 3-7
 - object shared memory, name 3-4
 - status 3-14
- Queue Bucket Count, attribute 3-7
- Queue Bucket Payload Size, attribute 3-7
- Queue Status, dialog box 3-14
- quitting, VR-Exchange 3-15

R

- RadarBeam* class 1-13
- Radio Communications protocol family 1-12
- radio object 1-12
- Radio Transmitter PDU 1-12
- RadioReceiver* class 1-12
- RadioSignal* interaction 1-12
- RadioTransmitter* class 1-12

RawBinaryRadioSignal interaction 1-12
 receiver protocol, signal protocol 1-12
 Reflecting page 4-27
 relative timestamp 4-14
 Remap DIS IDs attribute 4-14
 Remove Entity PDU 1-11
RemoveEntity class 1-11
 removing
 filter 4-25
 values from collection 4-8
 Repair Complete PDU 1-14
 Repair Response PDU 1-14
RepairComplete class 1-14
RepairResponse class 1-14
 resetting interaction statistics 3-12
 Resupply Cancel PDU 1-14
 Resupply Offer PDU 1-14
 Resupply Received PDU 1-14
ResupplyCancel class 1-14
ResupplyOffer class 1-14
ResupplyReceived class 1-14
 revision, RPR FOM 4-19
 RID file 2-6
rid.mtl file 2-6
 right-click menu 3-5
 RPR FOM
 revision 4-19
 specifying version A-7
 version 4-19
 RTI 2-6
 advisory 4-20
 definition of 2-6
 ticking 4-13
 RTI to RTI bridge A-4
 RTI_CONFIG environment variable 2-6
 running, TENA broker B-6

S

-S command-line option A-7
 -s command-line option A-10
 SDA 2-7, A-11
 send port, DIS A-10
 Service Request PDU 1-14
ServiceRequest class 1-14
 Set Data PDU 1-11
SetData class 1-11
 setting, notification level 3-4
 shared memory 1-2
 status 3-14

shared memory queue
 bucket count 3-4
 bucket payload size 3-4
 control, name 3-4
 interaction, name 3-4
 object, name 3-4
 SIMAN PDU 1-11
 Simulation Management PDU 1-11
 site ID 4-12, A-10
 DIS 4-14
 TENA 4-16
 Site ID attribute 4-14, 4-16
 socket receive buffer size 4-21
 Socket Receive Buffer Size attribute 4-21
 Socket Send Buffer Size attribute 4-20
 sorting, object display 3-11
 specifying
 broker name for HLA A-6
 broker name for TENA A-13
 execution name for TENA A-14
 FED file A-7
 federation name A-8
 FOM Mapper initialization data A-7
 FOM Mapper library A-6
 log notification level A-7, A-14
 LROM Mapper 4-22, A-14
 notification level 3-6
 ORB Initialization Reference 4-22, A-13
 ORB Listen Endpoint 4-22, A-13
 RPR FOM version A-7
 TCP port 3-7
 starting
 connections 3-8
 Portal
 Linux 3-2
 Windows 3-3
StartResume class 1-11
 Start/Resume PDU 1-11
 startup
 enabling connection 3-8
 ignoring command-line options 3-4
 State Update EE PDU 1-13
 statistics, resetting interaction 3-12
 Stop 1-11
StopFreeze class 1-11
 Stop/Freeze PDU 1-11
 stopping, connection 3-9
 subnet mask A-10
Subnet Mask parameter 4-21
 supported PDUs 1-8

Synthetic Environment PDU family 1-14

T

- t command-line option A-7, A-10, A-14
- TCP port A-10
 - specifying 3-7
 - at startup 3-5
 - TENA A-14
- TENA A-11
 - broker 1-6
 - command-line options A-13
 - running B-6
 - connection attributes 4-16
 - entity ID 4-16
 - execution 4-22
 - execution name A-14
 - exercise connection tick time 4-17
 - global ID 4-17
 - installing required software 2-7
 - LROM Mapper 1-6
 - multicast address 4-22
 - multicast time-to-live 4-22
 - network connection parameters 4-22
 - ORB Initialization Reference 4-22, A-13
 - ORB Listen Endpoint 4-22, A-13
 - site ID 4-16
 - TCP port A-14
 - translators A-12
 - update rate 4-17
 - using object ID as marking text 4-16
- TENA Middleware 2-7
 - downloading B-2
 - installing B-2
- TENA Software Development Activity A-11
- TENA Tick Time attribute 4-17
- TENA Transport Type, attribute 4-22
- Test and Training Enabling Architecture A-11
 - See TENA
- tick 4-13
- timeout 1-8
 - DIS 4-15
 - network drain 4-15
- Timeout Interval attribute 4-15
- timestamp 4-14
- Transfer Control Request PDU 1-14
- TransferControl class 1-14
- translating
 - interactions 4-13
 - objects 4-13

- translation
 - filtering individual objects 3-13
 - generic A-4, A-5
 - limitations of 1-16
- translator A-3
 - enabling and disabling 4-26
 - generic 4-13
 - interaction 4-13
 - object 4-13
 - TENA A-12
- Translators page 4-26
- transmitter protocol 1-12

U

- update
 - filtering display of 3-11
 - maximum per second 4-17
- Use Absolute Timestamps, attribute 4-12
- Use Absolute Timestamps attribute 4-14
- Use Generic Translators for Whole FOM attribute 4-13
- Use HLA Advisories attribute 4-20
- Use HLA Object IDs For Marking Text attribute 4-12
- Use Marking Text for HLA Object IDs attribute 4-12
- Use Marking Text for TENA Object ID attribute 4-16
- Use Persistent Entity IDs attribute 4-12, 4-14, 4-16
- Use TENA Object ID for Marking Text attribute 4-16
- using, RTI advisories 4-20

V

- v command-line option 3-5, A-7, A-10, A-14
- version
 - DIS 4-14
 - displaying 3-4, 3-5, A-7, A-10, A-14
 - RPR FOM 4-19
- viewing, discovered objects and interactions 3-10
- VR-Exchange
 - configuring 3-6
 - running without GUI 3-5
 - starting
 - Linux 3-2
 - Windows 3-3
- VR-Exchange Preferences, dialog box 3-6

VR-Exchange.xml 3-5, 3-6

VR-Link 1-5

vrxMessageDump 3-15

W

warfare protocol family 1-11

WeaponFire interaction 1-11

window, Portal 3-5

Windows, installing on 2-2

X-Y-Z

-x command-line option A-8, A-10, A-14

XML, RTI configuration file 2-6



Link - Simulate - Visualize

VRE-2.1-1-120726