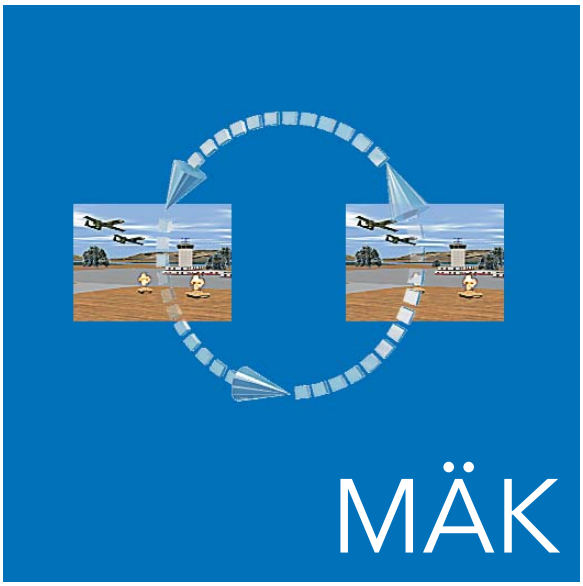
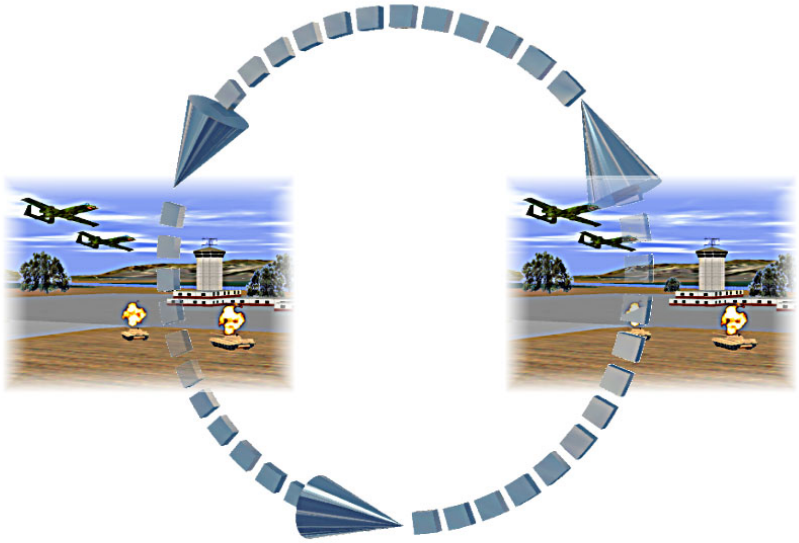


MÄK Gateway

User's Guide

MÄK TECHNOLOGIES



MÄK Gateway

User's Guide

MÄK TECHNOLOGIES

MÄK Technologies
185 Alewife Brook Parkway
Cambridge, MA 02138 USA

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

info@mak.com

www.mak.com

Revision GWY-4.0-1-030508

Contents

Preface

How this Manual is Organized v
Other MAK Products vi
How to Contact Us vii
Document Conventions viii
Hardware and Operating System Naming Conventions..... viii
Mouse Button Naming Conventions viii

Chapter 1 Introduction

1.1. Overview 1-2
1.2. Translating to and from HLA 1-3
1.2.1. FOM Support 1-3
1.2.2. Handle Mapping 1-4
1.2.3. Attributes Required for Sending DIS PDUs 1-4
1.3. Translating Between DIS and the RPR FOM 1-6
1.3.1. Entity Information and Entity Interaction 1-8
1.3.2. Warfare 1-9
1.3.3. Simulation Management 1-9
1.3.4. Radio Communications 1-10
1.3.5. Distributed Emission Regeneration 1-11
1.3.6. Logistics 1-12
1.3.7. Entity Management 1-12
1.3.8. Synthetic Environment 1-12

Chapter 2 Installing the Gateway

2.1. Installing the Gateway 2-2
2.1.1. Installing the Gateway On A PC 2-2
2.1.2. Installing the Gateway On A UNIX System 2-3
2.1.3. Directory Structure 2-4

2.2. Setting Up and Using the FLEXlm License Manager 2-5
2.2.1. The FLEXlm Client-Server Architecture 2-5
2.2.2. Installing and Configuring the License Manager 2-6
2.2.3. Installing the License Manager Software 2-6
2.2.4. Requesting A License File 2-7
2.2.5. Putting the License File in the flexlm Directory 2-8
2.2.6. Setting the MAKLMGRD_LICENSE_FILE Variable 2-8
2.2.7. Adding Additional License Files 2-10
2.2.8. Running the License Server 2-11
2.2.9. Troubleshooting the FLEXlm License Manager 2-11
2.3. Installing An RTI 2-14
2.3.1. Installing the MÄK RTI 2-14
2.3.2. Installing the DMSO RTI 2-15

Chapter 3 Using the Gateway

3.1. Overview 3-2
3.2. Setting Up the RTI Environment 3-2
3.3. Starting the Gateway 3-2
3.3.1. Starting the Gateway in UNIX 3-2
3.3.2. Starting the Gateway on A PC 3-3
3.3.3. Starting Message Translation Immediately After Startup 3-3
3.3.4. Gateway Startup Options 3-4
3.3.5. Closing the Gateway 3-5
3.4. Configuring the Gateway 3-5
3.4.1. Configuring the Gateway in the GUI 3-6
3.4.2. Specifying the FOM Mapper and FED File to Use 3-7
3.4.3. Choosing the Objects and Interactions to Translate 3-8
3.5. Translating DIS and HLA Messages 3-10
3.5.1. Viewing Objects and Interactions 3-10
3.5.2. Stopping Message Translation 3-11

Appendix A Gateway MTL Parameters

A.1. MÄK Technologies Lisp (MTL) Files A-2
A.2. Gateway Configuration Parameters A-3

Index

Preface

The MÄK Gateway allows DIS simulations to interoperate with HLA federations that use the Real-Time Platform Level Reference (RPR) FOM (and other FOMs using VR-Link FOM mappers).

How this Manual is Organized

This manual is written for persons who will install or use the MÄK Gateway. The manual assumes that you are familiar with the basic operation of your operating system and that you know how to work in your computer's graphical window environment.

The chapters are organized as follows:

Chapter 1, *Introduction*, briefly describes the features and uses of the Gateway.

Chapter 2, *Installing the Gateway*, explains how to install the Gateway and other necessary software, including license management software and an RTI.

Chapter 3, *Using the Gateway*, explains how to start the Gateway, configure it, and translate state update messages.

Appendix A, *Gateway MTL Parameters* describes MTL file syntax and the parameters used in the *gateway.mtl* file.

Other MAK Products

The Gateway is a member of the MÄK Technologies® line of software products designed to streamline the process of developing and using networked simulated environments.

In addition to the Gateway, the MÄK Technologies product line includes:

- The 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 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/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.

  VR-Link is the basis of the other MÄK products.

- The 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, RTIspy, that allows you to extend the RTI using plug-in modules. It also has a graphical user interface for the MÄK RTI rtiexec.
- MÄK VR-Forces®. VR-Forces is a computer generated forces application and toolkit. It provides an application with a graphical user interface, that gives you a two dimensional view of a terrain database.

  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.

- The MÄK Stealth. The Stealth displays a realistic, three-dimensional 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.
- The 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, or in reverse, and locate areas of interest quickly. The Logger has a graphical user interface and a command line interface. The Logger API allows you to extend the Logger using plug-in modules or embed the Logger into your own application.

The Logger comes with a set of utility programs that let you inspect, combine and edit Logger recordings.

- The MÄK Plan View Display. The Plan View Display (PVD) provides a two-dimensional, "big picture" of a virtual environment. It shows simulated entities, such as vehicles, soldiers, and tanks moving over a 2D-map display.

How to Contact Us

For Gateway 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

Internet

MÄK web site home page:	www.mak.com
License key requests:	www.mak.com/licenses.htm
Product version and platform information:	www.mak.com/product_version.htm
For the free MÄK RTI evaluation download:	ww.mak.com/rti.htm
Support FAQ:	www.mak.com/faq.htm

Post

Send postal correspondence to:	MÄK Technologies 185 Alewife Brook Parkway 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:

`Monospaced`	Indicates commands or values you enter.
`Monospaced Italic`	Indicates command variables that you replace with appropriate values.
`{ }`	Indicates required arguments.
`[ ]`	Indicates optional arguments.
|	Separates options in a command where only one option may be chosen at a time.
/	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.
Italic	Indicates a file name, pathname, or a class name.
sansSerif	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.
	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 Gateway home directory. For example, the directory *gateway/doc* appears in the text as *./doc*.

Hardware and Operating System Naming Conventions

Unless otherwise specified, references to UNIX include Linux, regardless of the type of computer it is running on. References to PCs refer to Intel processor-compatible computers running a version of Microsoft Windows.

Mouse Button Naming Conventions

An instruction to click the mouse button, refers to clicking the primary mouse button, typically 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.

1

Introduction

This chapter introduces you to the Gateway and its features.

Overview 1-2
Translating to and from HLA 1-3
FOM Support 1-3
Handle Mapping 1-4
Attributes Required for Sending DIS PDUs 1-4
Translating Between DIS and the RPR FOM 1-6
Entity Information and Entity Interaction 1-8
Warfare 1-9
Simulation Management 1-9
Radio Communications 1-10
Distributed Emission Regeneration 1-11
Logistics 1-12
Entity Management 1-12
Synthetic Environment 1-12

1.1. Overview

The MÄK Gateway provides a means for DIS applications to interoperate with HLA federations that use the Real-Time Platform Level Reference (RPR) FOM. The Gateway is a stand-alone translator application that connects DIS exercises to an HLA RPR FOM federation execution. The Gateway acts simultaneously as a member of the DIS exercise, and together with one or more DIS applications, as a federate in the HLA federation execution (Figure 1-1). Because the Gateway is a stand-alone interface, no modification of the DIS applications is required.

The Gateway is based on the VR-Link Toolkit. Therefore, it can take advantage of VR-Link's FOM agility feature. This means that if you write a FOM Mapper, you can use the Gateway with FOMs other than the RPR FOM. See the VR-Link documentation for details.

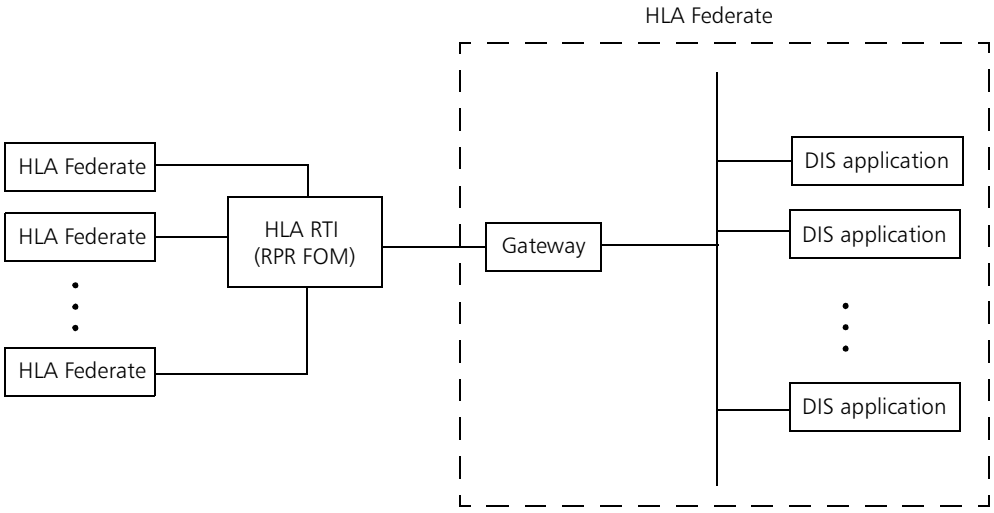


Figure 1-1. Gateway interoperability

DIS to HLA translation works as follows:

1. On the DIS side of the Gateway, data packets called Protocol Data Units (PDUs) are formatted, sent, and received.
2. The Gateway receives the PDUs.
3. It puts them into a VR-Link state repository.
4. The state repository is copied to an HLA state repository.
5. The FOM mapper converts the data in the state repository into and encodes HLA objects and attributes.
6. The HLA data is sent over the network.

In order to communicate with an HLA federation execution, the Gateway must perform some HLA services that have no corresponding function in DIS. These functions include creating, destroying, joining, and resigning federations and publishing and subscribing to RPR FOM classes.

A detailed description of how DIS PDUs are translated into the RPR FOM object and interaction classes is in "Translating Between DIS and the RPR FOM," on page 1-6 and Guidance Rational Interoperability Modalities for the Real-Time Platform Reference (GRIM RPR) FOM, Simulation Interoperability Standards Organization (SISO) RPR FOM SDG, http://www.sisostds.org/stdsdev/rpr-fom/index.htm.

1.2. Translating to and from HLA

This section discusses general issues involved in translating update messages to and from HLA.

1.2.1. FOM Support

As noted previously, the Gateway provides native support for several versions of the RPR FOM. (See the *MÄK Gateway Release Notes* for a list of supported versions.) That is, the Gateway can translate data between DIS and supported versions of the RPR FOM without any additional code. Like other MÄK products, the Gateway comes with a FED file called *VR-Link.fed*, which contains all of the RPR FOM 1.0 objects and interactions, plus a few extra classes described in the file's comments. By default, the Gateway joins the federation execution called VR-Link, and uses this FED file, which is distributed with the Gateway, and with other MÄK tools. For details about choosing the version of the RPR FOM to use, see "Specifying the FOM Mapper and FED File to Use," on page 3-7.

Using FOMs other than the RPR FOM

If you want to use the Gateway with FOMs other than the RPR FOM, you can write a VR-Link FOM mapper to provide the information necessary to properly map between VR-Link's internal object model and your FOM. See *VR-Link Developer's Guide* for complete details about creating FOM mappers.

The Gateway SOM

When a DIS application uses the Gateway to participate in an HLA federation execution, the DIS application and Gateway taken together comprise a federate. Like any other federate, this federate has a SOM that must be submitted when the federate undergoes HLA compliance testing. The contents of the SOM depend on which elements of the FOM are needed by the DIS application. For example, if the DIS application does not care about Collision PDUs, you can instruct the Gateway to not subscribe to (and thus translate) Collision interactions, meaning that Collision interactions will not be part of your federate's SOM.

1.2.2. Handle Mapping

The RTI identifies classes, attributes, and parameters using handles (integer identifiers). A federate discovers these handles by submitting the class, attribute, or parameter string name for which it needs a handle. The RTI returns the handle. For efficiency reasons, the Gateway performs this mapping between string names and handles only once and stores the mapping information in an internal database.

Handle mapping is performed immediately prior to publication or subscription. So, the Gateway performs handle mapping only for those classes that are published or subscribed. Therefore, the Gateway can operate with a FED file that contains only those classes that it publishes and subscribes. The Gateway can also tolerate missing attributes or parameters.

During a handle mapping operation, the Gateway attempts to map the complete set of attributes or parameters for each published or subscribed class. If an attribute or parameter is missing, the Gateway gets the default value specified by the VR-Link state repository, usually 0 or type unknown.

1.2.3. Attributes Required for Sending DIS PDUs

HLA object updates contain only the information that has changed since the previous update. Therefore, when the Gateway discovers an HLA object, it is possible that the update message will not include all of the information necessary to complete a PDU. The Gateway will not send the PDU for this object until it has all of the information it needs. Table 1-1 lists the required attributes, the containing class, and comments on any indirect requirements.

Table 1-1: Mandatory Attributes

Class	Attribute / Parameter	Comment
EmitterSystem	EmitterName, EmitterNumber, HostObjectID, (active beams)	HostObjectID (emitting entity) must refer to an object that has been registered or discovered by the Gateway. The system must have active beams except when being removed.
EmitterBeam	BeamID, EmittingSystemID, BeamParameterIndex, ERP	EmittingSystemID must refer to an object that has been registered or discovered by the Gateway. Heartbeat updates require BeamParameterIndex and ERP to be non-zero.
Designator	HostObjectID	HostObjectID (designating entity) must refer to an object that has been registered or discovered by the Gateway.
RadioTransmitter	HostObjectID	HostObjectID (transmitting entity) must refer to an object that has been registered or discovered by the Gateway.

Table 1-1: Mandatory Attributes

Class	Attribute / Parameter	Comment
RadioReceiver	HostObjectID	HostObjectID (receiving entity) must refer to an object that has been registered or discovered by the Gateway.
IFF	HostObjectID, SystemType	HostObjectID (emitting/receiving entity) must refer to an object that has been registered or discovered by the Gateway.
BaseEntity	EntityID, EntityType, Position	All attributes must be defined.
ResupplyCancel	ReceivingObject, SupplyingObject	Objects IDs must refer to an object that has been registered or discovered by the Gateway.
ResupplyOffer	ReceivingObject, SupplyingObject	Objects IDs must refer to an object that has been registered or discovered by the Gateway.
ResupplyReceived	ReceivingObject, SupplyingObject	Objects IDs must refer to an object that has been registered or discovered by the Gateway.
ServiceRequest	RequestingObject, ServicingObject	Objects IDs must refer to an object that has been registered or discovered by the Gateway.
RepairComplete	ReceivingObject, RepairingObject	Objects IDs must refer to an object that has been registered or discovered by the Gateway.
RepairResponse	ReceivingObject, RepairingObject	Objects IDs must refer to an object that has been registered or discovered by the Gateway.

1.3. Translating Between DIS and the RPR FOM

The Gateway handles the discrepancy between the DIS concept of a heartbeat and the HLA's lack of such as follows:

- The Gateway provides heartbeat updates for the appropriate PDUs when the time between HLA updates exceeds the heartbeat threshold.
- The Gateway deletes HLA objects when the time between DIS PDUs exceeds the timeout threshold (2.5*heartbeat). RPR FOM embedded system objects (radio transmitter/receiver and emitter system/beams) are deleted only if their host entity no longer exists.

The Gateway does not support the full set of PDUs in the 1278.1 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.

The following sections describe the supported PDUs in greater detail.

1.3.1. 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.

1.3.2. 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.

1.3.3. 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

1.3.4. 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 the Gateway. 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
		IffTransponder

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.

Table 1-5: RadioSignal Interaction Class

Class1	Class2
RadioSignal	ApplicationSpecificRadioSignal
	DatabaseIndexRadioSignal
	EncodedAudioRadioSignal
	RawBinaryRadioSignal

1.3.5. 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 the Gateway 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 *EmittingSystemID* that establishes the association between it and its EmitterSystem object instance.

The Gateway creates a Delta State EE PDU for each *EmitterSystem* or *EmitterBeam* reflection. The Gateway does not generate a delta EE PDU if the *EmitterSystem* has no active beams (unless that beam is being removed). The Gateway 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. The Gateway 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.

1.3.6. 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.

1.3.7. 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.

1.3.8. Synthetic Environment

The Gateway 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.

2

Installing the Gateway

This chapter explains how to install Gateway software, RTI software, and license management software.

Installing the Gateway 2-2
- Installing the Gateway On A PC 2-2
- Installing the Gateway On A UNIX System 2-3
- Directory Structure 2-4

Setting Up and Using the FLEXlm License Manager 2-5
- The FLEXlm Client-Server Architecture 2-5
- Installing and Configuring the License Manager 2-6
- Installing the License Manager Software 2-6
- Requesting A License File 2-7
- Putting the License File in the flexlm Directory 2-8
- Setting the MAKLMGRD_LICENSE_FILE Variable 2-8
- Adding Additional License Files 2-10
- Running the License Server 2-11
- Troubleshooting the FLEXlm License Manager 2-11

Installing An RTI 2-14
- Installing the MÄK RTI 2-14
- Installing the DMSO RTI 2-15

2.1. Installing the Gateway

To use the Gateway, you need to install the Gateway software and a supported RTI.

2.1.1. Installing the Gateway On A PC

This section explains how to install the Gateway from CD-ROM or from an ftp download on a PC.

Installing the Gateway from A CD

The Gateway may be supplied to you on an "All Products" CD with a common installation interface or a CD that does not support common installation.

To install the Gateway from a CD:

1. Insert the Gateway CD into your CD-ROM drive.
   - If this is an All Products CD, and the autorun feature is enabled, the MÄK PC Products Installation window opens. The PC Products Installation window is similar to Figure 2-1.

     If the MÄK PC Products Installation window does not open, open the Windows Explorer. In the root directory for your CD drive, double-click *makInst.exe*. The MÄK PC Products Installation window opens.
   - If this is not an All Products CD, and autorun is enabled, an installation program may start immediately. Follow the instructions of the installation program.

     If the installation does not start automatically, open the Windows Explorer and select the CD drive. In the root directory for the Gateway, double-click *setup.exe*. Go to step 3.
2. In the MÄK PC Products Installation window, click Gateway.

The PC Products Installation window may list products by the RTI that they support. If you plan to use the HLA version of the Gateway, install the version that supports the RTI that you plan to use. You can install all versions if you have enough disk space.

3. Follow the instructions of the installation program.

Figure 2-1. **MÄK** PC Products Installation window

Installing the Gateway from A Downloaded File

The Gateway is distributed via ftp as a compressed file. It may be a zip file or an executable file.

To install the Gateway from a zip file:

1. Copy the zip file to a temporary directory.
2. Unzip the zip file.
3. Run *setup.exe*.
4. Follow the instructions of the installation program.

➤ To install the Gateway from an executable file, run the executable.

2.1.2. Installing the Gateway On A UNIX System

This section explains how to install the Gateway on UNIX systems from CD-ROM or from an ftp download.

Installing the Gateway from A CD-ROM

To install the Gateway:

1. Mount the CD-ROM.
2. Run:

   ```
   ./install
   ```

3. Follow the on-screen instructions. You can install the Gateway into any directory for which you have write permissions.

Installing the Gateway from A Downloaded File

If you obtained the Gateway via ftp, you received a compressed tar file.

To install the Gateway:

1. Create the directory in which you want to install the Gateway.
2. Copy the tar file to the install directory.
3. Uncompress the tar file:

   ```
   gunzip gateway.tar.gz
   ```

4. Untar the file:

   ```
   tar xvf gateway.tar
   ```

2.1.3. Directory Structure

The directory structure for the installed files is:

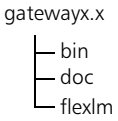


Figure 2-2. Gateway directory structure

Table 2-1 describes the contents of the Gateway directories

Table 2-1: Gateway directory contents

Directory	Contents
bin	Executables, FED files, RID files, and configuration files.
doc	Gateway documentation.
flexlm	License management files.

2.2. Setting Up and Using the FLEXlm License Manager

The Gateway, like other MÄK Technologies products, uses the FLEXlm license manager. Before you can use the Gateway, you must obtain a valid license file and configure the license server and client machines. (For general information about FLEXlm, see www.globetrotter.com.) Contact the MÄK Technologies sales department for information about special licensing agreements.

2.2.1. The FLEXlm Client-Server Architecture

This section explains the FLEXlm architecture to provide a context for the setup instructions that follow. FLEXlm uses a client-server architecture. The FLEXlm executable, *maklmgrd*, runs on one computer, known as the license server. This machine services every machine (including possibly itself) on which you want to run a MÄK Technologies product. The machines on which the MÄK Technologies products are running are called clients.

The license server has a license file (supplied by MÄK Technologies). This file identifies the MÄK Technologies products for which you have licenses. On each client machine, the MAKLMGRD_LICENSE_FILE environment variable (which you must set) points to the server.

Licenses "float", so you can install product software on as many computers as you want, but at any given time, you can run only as many instances of a product as you have licensed.

! The license key in the license file is tied to the host ID of the server machine. Therefore, if you decide to change the server (or replace the network card), you need to get a new license file. Then, you must update the MAKLMGRD_LICENSE_FILE environment variable on every client. Depending on how many clients you have, this may not be a trivial process. Keep this in mind when you choose the machine to use as a license server.

Figure 2-3 illustrates the client-server architecture for FLEXlm and MÄK products.

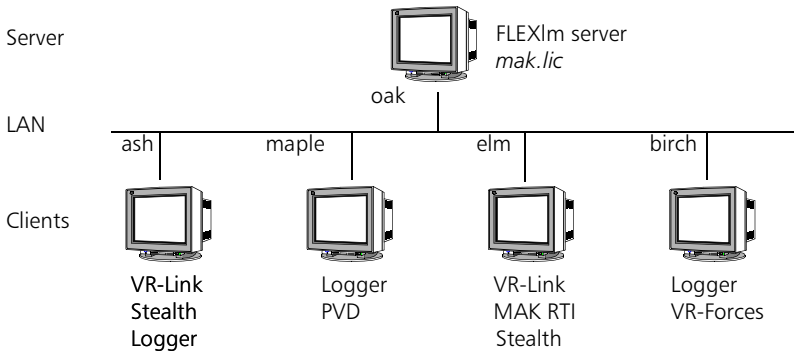


Figure 2-3. FLEXlm example architecture

2.2.2. Installing and Configuring the License Manager

The general procedure for installing and configuring license management is as follows (each step is explained in the rest of this section):

1. Install license manager software.
2. Request a license file.
3. Put the license file in the *flexlm* directory on the license server.
4. Set the MAKLMGRD_LICENSE_FILE environment variable on all client machines.

If you have MÄK Technologies products installed and you purchase additional licenses or products, you must get a license file for the new products or licenses. For this procedure, see "Adding Additional License Files," on page 2-10.

2.2.3. Installing the License Manager Software

As part of the installation of all MÄK products, the FLEXlm software is installed in the *flexlm* directory under the product's root directory.

1. Install your MÄK products on the machines on which you want to run them.
2. Copy the *flexlm* directory to the server machine. For a PC, copy *flexlm* to *C:\flexlm*. For a UNIX machine, a typical location is */usr/local/flexlm*.

2.2.4. Requesting A License File

When you request a license file, you need to know the:

- Host ID of the license server
- Name of the license server
- MÄK invoice number of your purchase
- Basic contact information for you and your company
- Platforms for which you have purchased MÄK products
- Number of licenses for each platform and product.

When you have this information, go to http://www.mak.com/licenses.htm and complete the license key request form. MÄK will e-mail you a license file.

Identifying the Host ID and License Server Name in Windows

To find out the Host ID and license server name in Windows:

1. In the flexlm directory, run *lmtools.exe*. The LMTOOLS application opens.
2. Select the System Settings tab. The Host ID is the string in the Ethernet Address text box. The license server name is the name in the Computer/Hostname text box.

Identifying the Host ID and License Server Name on UNIX Computers

To find out your host ID:

1. On the license server, in a command window, change to the *flexlm* directory.
2. Execute *lmhostid*. This program reports a unique number that MÄK uses to generate your license activation codes. Write down the number.

The host ID is keyed to your ethernet card, so if you change your network card, you must request a new license file.

➤ To find out the name of the server machine, at a command prompt, type `hostname`.

2.2.5. Putting the License File in the flexlm Directory

The license file you receive may be an attachment or embedded in an email.

1. If the attachment is not named *mak.lic*, rename it *mak.lic*.

   If the license is embedded in an email, copy it into an empty text file. Name the file *mak.lic*.

2. Put the license file in the *flexlm* directory on the license server.

! If you already have a *mak.lic* file in the *flexlm* directory, do not overwrite the file. See the instructions in "Adding Additional License Files," on page 2-10.

2.2.6. Setting the MAKLMGRD_LICENSE_FILE Variable

The MAKLMGRD_LICENSE_FILE environment variable identifies the server machine so that the FLEXlm software on a client can check out a license when you run a MÄK Technologies product. You must set MAKLMGRD_LICENSE_FILE on every client machine.

---

i If you are running MÄK Technologies products on the license server machine, it is also a client, so you must set MAKLMGRD_LICENSE_FILE on that machine.

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 Technologies products run on.

Windows NT

On Windows NT, you set environment variables in the Environment tab of the Properties dialog box for the machine:

1. Right-click the My Computer icon on the Desktop.
2. On the pop-up menu, select Properties. The System Properties dialog box opens (Figure 2-4).
3. Select the Environment tab.
4. In the Variable text field, type MAKLMGRD_LICENSE_FILE.
5. In the Value field, type the value, for example:

   ```
   @oak
   ```

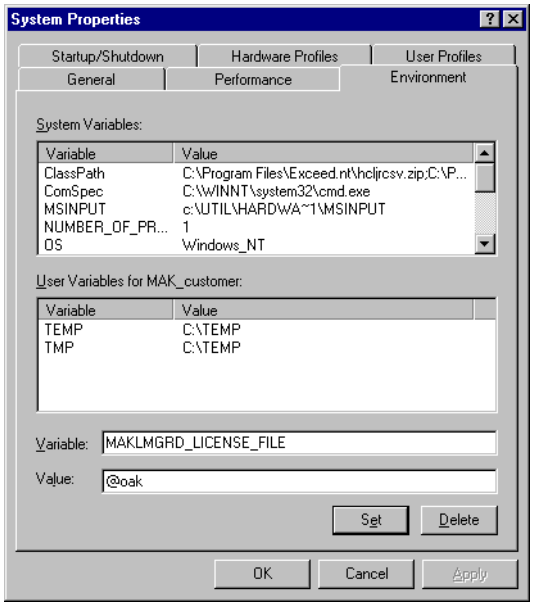


Figure 2-4. Setting an environment variable in Windows NT

Windows 2000

To add the MAKLMGRD_LICENSE_FILE in Windows 2000:

1. Right-click the My Computer icon on the Desktop.
2. On the pop-up menu, select Properties. The System Properties dialog box opens.
3. Click the Advanced tab.
4. Click the Environment Variables button. The Environment Variables dialog box opens.
5. Click the New button. The New System Variable dialog box opens.
6. In the Variable Name field, enter MAKLMGRD_LICENSE_FILE.
7. In the Variable Value field, enter `@server_name`, where `server_name` is the name of the license server.
8. Click OK to back out of each dialog box and set the variable.

Windows XP

To add the MAKLMGRD_LICENSE_FILE in Windows XP:

1. On the Start menu, choose My Computer.
2. In the My Computer window, under System Tasks, click View System Information. The System Properties dialog box opens.
3. Follow steps 3 through 8 in the Windows 2000 procedure.

UNIX

On UNIX, 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. See "Running the License Server," on page 2-11.

2.2.7. Adding Additional License Files

If you buy additional MÄK products, or additional licenses for products you already have, you must get an additional license file.

➤ To request an additional license file, follow the same procedure as for your first license file. See "Requesting A License File," on page 2-7.

To add a new license file to your license server:

1. When you get the new license file, name it *mak2.lic*, or some other name ending in *.lic*, so that it has a different name from the license files currently in the *flexlm* directory on the license server.
2. Put the renamed file in the *flexlm* directory on the license server, with the other license files.
3. Restart the license server.

You can now use the new products or additional product licenses.

2.2.8. Running the License Server

The FLEXlm license server must be running on the server machine before you can run your MÄK applications.

➤ To start the license server, on the server machine only, execute `runLm` from the *./flexlm* directory.

➤ To obtain the current status of the server, run `lmstat`.

! Do not execute more than one instance of `runLm` at a time. If you aren't sure whether or not the license server is running, execute `lmstat`.

Shutting Down the License Server

➤ To force all applications to check in their licenses and shut down the license server, run `lmdown`.

2.2.9. Troubleshooting the FLEXlm License Manager

For general troubleshooting information, refer to the FLEXlm FAQ at www.globetrotter.com. It describes common problems, including the majority that have been reported to MÄK support engineers. Please consult the FAQ first when you have a problem.

Unable to Get A License

If you are unable to get a license for a MÄK application and the FLEXlm log file (*LOG*) indicates that the port is in use, it is possible that you have more than one lmgrd or maklmgrd processes running. It is also possible that an application that was using a license is thought to have terminated, but is still alive. To verify and resolve these possibilities:

1. See if more than one lmgrd or maklmgrd process is running.

   On Windows 95/98/NT, check the Task Manager. To open the Task Manager, press Ctrl-Alt-Delete.

   On UNIX, enter the following commands:

   ```
   ps -aux | grep lmgrd
   ps -aux | grep maklmgrd
   ```

   If more than one instance of either process is running, write down the process ids (PID).

Some UNIX systems use `ps -aux`, others `ps -ef` to check processes. Use the command appropriate for your operating system.

2. If there are old processes present that may be using a license, kill them.

   On UNIX, kill a process with the kill command, for example:

   ```
   kill -9 1385 878
   ```

   where 1385 and 878 are process IDs.

   On Windows NT, select a process in the Task Manager Processes window and click End Process.

   It is important that you kill all the processes to ensure a clean restart for the license server.

3. If in step 2 you killed the license server, start it with the `runLm` command.

Preventing Multiple License Manager Processes

We recommend that you keep the license server running and that you not start and stop it as you run applications. If you prefer to stop the license server when you are not using it, always use `lmdown` to stop it.

Whenever you start the license manager, first run `lmstat` to be sure that there is not already a license server process running. Following this practice will ensure that you do not start multiple license servers.

License Manager Cannot Find MÄK License Management Executable

If your license manager gives you an error message indicating that it cannot find the license management executable, try putting the pathname in the license file as follows:

1. Open the license file (*mak.lic*) in a text editor.
2. Edit the license file to specify the location of the *maklmgrd* executable on the server machine. The file supplied by MÄK represents this location with a dot as shown below:

   ```
   DAEMON maklmgrd .
   ```

   Replace the dot with your absolute path to *maklmgrd*, for example:

   ```
   DAEMON maklmgrd "C:\Program Files\MAK\flexlm\maklmgrd"
   ```

On PCs, you must enclose the path in quotes.
The drive letter must be uppercase.

The License Manager Reports an Unsupported Product

If you receive an error message indicating an unsupported product, you are trying to use a MÄK product for which you do not have a license.

If you are trying to run your application with the DMSO RTI and you get the unsupported product message, it is possible that your application is finding the DLLs for an unlicensed version of the MÄK RTI before it finds the DMSO RTI. To fix this problem do the following:

- Make sure that the DMSO RTI path precedes the MÄK RTI path in your path environment variable.
- Check the directory in which the application is located to be sure that there are no DLLs from the MÄK RTI in that directory.

2.3. Installing An RTI

An RTI (Run-Time Infrastructure) is a software library (and perhaps supporting executables) that implements the 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.

All MÄK products have been tested against the MÄK RTI (which comes with all MÄK products), and the DMSO RTI. For the most recent information about the RTI versions supported by the Gateway, see the Product Versions page on the MÄK Technologies web site at: http://www.mak.com/product_version.htm.

2.3.1. Installing the MÄK RTI

The MÄK RTI is distributed with all MÄK products and you can install it when you install the Gateway.

UNIX

The installation process creates a link called *RTI* in each MÄK product's root directory, which points to the *makRti* directory, in which the MÄK RTI software is located.

PCs

Install the MÄK RTI from the PC Products Installation screen just as you installed the Gateway.

the

Configuring Your System to Use the MÄK RTI

The RTI dynamic libraries need to be located somewhere on your dynamic library search path. This path is indicated by an environment variable as follows:

- For IRIX6n32 use LD_LIBRARYN32_PATH
- On other UNIX platforms use LD_LIBRARY_PATH
- On PCs, use PATH.

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

- The FED file (*.fed*) 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 most 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 Flexlm license server is running, then start the applications, and they should be able to communicate. (For information about the license server, see "Setting Up and Using the FLEXlm License Manager".) See *MÄK RTI Reference Manual* for information about the rtiexec and changing the networking parameters such as multicast addresses and UDP ports.

2.3.2. Installing the DMSO RTI

DMSO does not distribute the DMSO RTI any more, but you may be able to obtain it from other sources.

Consult the DMSO RTI Release Notes to determine version information and whether or not any patches are required. Install the RTI according to the instructions provided.

Configuring Your System to Use the DMSO RTI

Follow the configuration instructions provided by your version of the DMSO RTI.

Running Applications Using the DMSO RTI

To use the DMSO RTI, you need to run the RTI executive (rtiexec). This executable is in the *bin* directory or a subdirectory. Run the rtiexec only on one machine, regardless of how many HLA applications you are running.

Once the rtiexec is running, you can run HLA applications.

3

Using the Gateway

This chapter explains how to configure the Gateway and use it.

Overview 3-2
- Setting Up the RTI Environment 3-2
- Starting the Gateway 3-2
- Closing the Gateway 3-5

Configuring the Gateway 3-5
- Configuring the Gateway in the GUI 3-6
- Specifying the FOM Mapper and FED File to Use 3-7
- Choosing the Objects and Interactions to Translate 3-8

Translating DIS and HLA Messages 3-10
- Viewing Objects and Interactions 3-10
- Stopping Message Translation 3-11

3.1. Overview

The general steps for using the Gateway are as follows:

1. Set up your RTI environment.
2. Start the Gateway application.
3. Configure the DIS and HLA connections. (You can also configure the Gateway with a configuration file or with command line options.)
4. Optionally, edit the default set of objects and interactions that get translated.
5. Start translating messages.

3.2. Setting Up the RTI Environment

To use the Gateway you must set up an RTI environment. See "Installing An RTI," on page 2-14 for information about setting up your environment with either the MÄK RTI or DMSO RTI.

➤ If you are using the DMSO RTI-NG or are using MÄK RTI features that require it, start the rtiexec.

3.3. Starting the Gateway

This section explains how to start the Gateway on the different platforms it supports.

3.3.1. Starting the Gateway in UNIX

To start the Gateway:

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

```
Gateway [startup_options]
```

The Gateway window opens (Figure 3-1). The Gateway startup options are described in "Gateway Startup Options," on page 3-4.

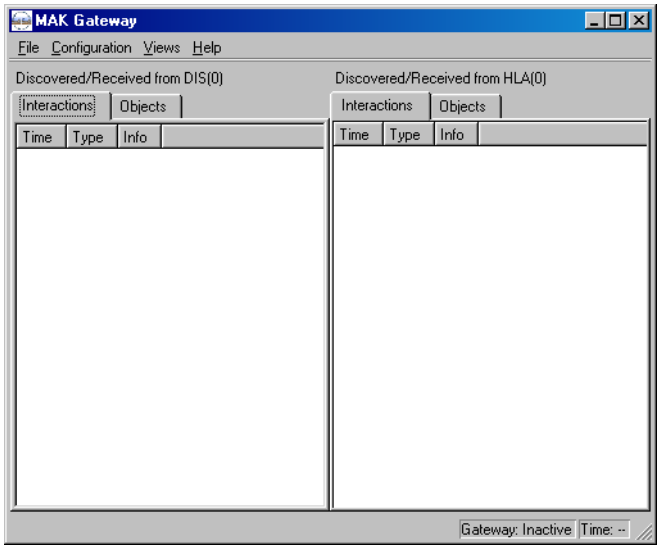


Figure 3-1. Gateway window

3.3.2. Starting the Gateway on A PC

➤ To start the Gateway, on the Start menu, choose Programs → MÄK Technologies → Gateway.

The Gateway window opens (Figure 3-1).

If you want to start the Gateway with startup options, start it from a command prompt or create a shortcut icon for it on your desktop.

3.3.3. Starting Message Translation Immediately After Startup

At the beginning of this chapter, we list several steps that you may need to take between the time you start the Gateway and the time you start translating state update messages. However, if you have specified all the configuration options that you want in the *gateway.mtl* file or with startup options, you can specify that the Gateway start translating messages as soon as it opens.

➤ To start translating messages at startup, start the Gateway with the `-s` startup option.

3.3.4. Gateway Startup Options

The Gateway startup options are as follows:

```
Gateway [-A IP_address
   -a application_number
   -b send_port
   -C
   -e exercise_ID
   -F federate_name
   -f FED_file
   -H heartbeat
   -h
   -n notify_level
   -P port
   -R RPRFOM_version
   -S
   -s site_ID
   -T polling_interval]
   -x federation_execution_name
```

where:

-A IP_address specifies the DIS destination address.

-a application_number specifies the DIS application number.

-b *send_port* specifies the Gateway send port.

-C opens a console window (Windows only.)

-e *exercise_ID* specifies the DIS exercise ID.

-F *federate_name* specifies the federate name that the Gateway is using.

-f *FED_file* specifies the FED file for the exercise.

-H *heartbeat* specifies the heartbeat interval, in seconds.

-h displays the list of startup options.

-n *notify_level* specifies the notification level. The range for *notify_level* is 0-3, with 0 providing the least amount of information.

-P *port* specifies the DIS port.

-R *RPRFOM_version* specifies the RPR FOM version being used.

-S specifies that the Gateway starts translating messages as soon as it starts.

-s *site_ID* specifies the DIS site ID.

-T *polling_interval* specifies the frequency, in seconds, with which the Gateway checks the exercise connection.

-x *federation_execution_name* specifies the HLA federation execution.

3.3.5. Closing the Gateway

➤ To close the Gateway, choose File → Quit.

3.4. Configuring the Gateway

Before you start translating update messages, you must configure the Gateway to work with the DIS exercise and HLA federation execution that you are participating in.

You can configure the Gateway using startup options, the *gateway.mtl* file, or through the Gateway graphical user interface. Startup options override the settings in the configuration file. Changes made in the GUI override startup options. Appendix A, *Gateway MTL Parameters* describes the parameters in *gateway.mtl*. Table 3-1 maps the equivalent MTL parameters, startup options, and GUI configuration options.

Table 3-1: Configuration option mappings

MTL Parameter	Startup Option	GUI Option
disPort	`-P`	Port
disExerciseId	`-e`	Exercise ID
disSiteId	`-s`	Site ID
disApplicationNum	`-a`	Application Number
disDestAddr	`-A`	Dest IP Address
hlaFedName	`-F`	Federate Name
hlaExecName	`-x`	Exercise Name
hlaFedFileName	`-f`	FED File Name
hlaRPRFOMVersion	`-R`	RPR FOM Version
gatewaySendPort	`-b`	
heartbeatInterval	`-H`	
notifyLevel	`-n`	
pollInterval	`-T`	
autoStart	`-S`	

3.4.1. Configuring the Gateway in the GUI

To configure the Gateway:

1. Choose Configuration → Preferences. The Preferences dialog box (Figure 3-2) opens.

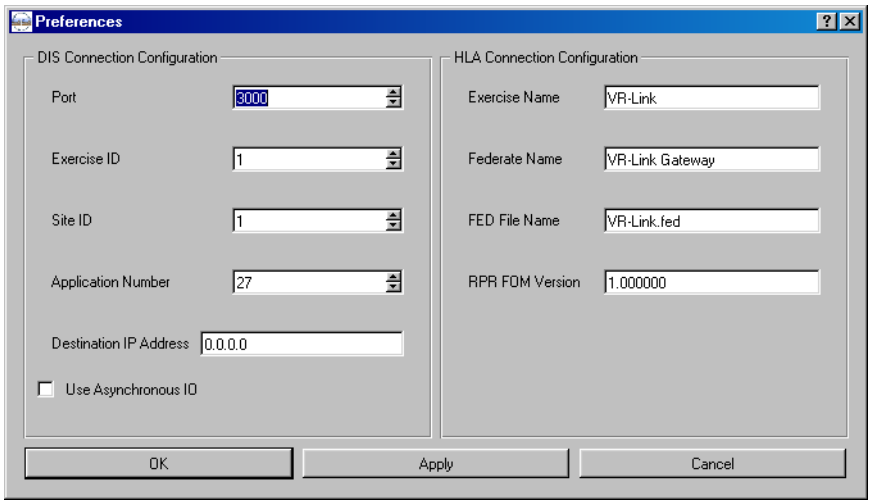


Figure 3-2. Preferences dialog box

2. Specify the DIS and HLA settings for your exercise.

You cannot apply changes made in the Preferences dialog box while the Gateway is translating update messages.

Table 3-2 describes the configuration options in the Preferences dialog box.

Table 3-2: Preferences dialog box parameters (Sheet 1 of 2)

Configuration Option	Description
Port	Specifies the port to send/receive network PDUs on. Default: 3000.
Exercise ID	The DIS exercise ID.
Site ID	The DIS site ID.
Application Number	The DIS application number. It is used for creating entity IDs for PDUs originating from the Gateway. It must be unique.
Dest IP Address	Specifies the address of outgoing DIS PDUs. If set to 0.0.0.0, the Default broadcast address is used.
Federate Name	The name of this federate. Default: VR-Link.fed.

Table 3-2: Preferences dialog box parameters (Sheet 2 of 2)

Configuration Option	Description
Exercise Name	The name of the federation execution. Default: VR-Link.
FED File Name	The name of the FED file being used for the HLA federation execution. It must be the complete name including the three char extension (.fed).
RPR FOM Version	The version of the RPR FOM that you are using. Default: 1.0.

3.4.2. Specifying the FOM Mapper and FED File to Use

This section explains how to specify the FOM Mapper to use with the Gateway. You must specify a FED file that matches the FOM Mapper.

As noted elsewhere in this manual, the Gateway has built-in support for the RPR FOM and VR-Link's ability to support alternative FOMs through FOM Mappers. The Gateway supports different versions of the RPR FOM using FOM Mappers that come with the Gateway and associated FED files. The provided FOM Mappers and FED files are as follows:

- RPR FOM 0.5 – *VR-Link05.fed*
- RPR FOM 0.7 – *VR-Link07.fed*
- RPR FOM 0.8 – *VR-Link08.fed*
- RPR FOM 1.0 – *VR-Link10.fed* and *VR-Link.fed*
- RPR FOM 2.0 draft 6 – *VR-Link20006.fed*
- RPR FOM 2.0 draft 14 – *VR-Link20014.fed*

The default FOM used is RPR FOM 1.0. To use one of the other versions of the RPR FOM, specify the FOM version and FED file in the Preferences dialog box, with the `-f` and `-R` startup options, or with the **hlaFedFileName** and **hlaRPRFOMVersion** parameters in *gateway.mtl*.

If you want to use a FOM Mapper that you have written, you must specify the shared library (DSO or DLL) that contains your FOM Mapper and, optionally, any initialization data the library requires. Specify the FOM Mapper library with the **hlaFomMapperLibName** parameter in *gateway.mtl*. Specify the initialization string with the **hlaFomMapperLibInitData** parameter.

For example, if you write a FOM Mapper contained in a shared library called *myFomMapLib.dll*, which requires the string "ABC" as an initialization parameter, specify the following entries in the *gateway.mtl* file.

```
hlaFomMapperLibName "myFomMapLib.dll"
hlaFomMapperLibInitData "ABC"
```

3.4.3. Choosing the Objects and Interactions to Translate

By default, the Gateway translates all objects and interactions. If you do not want to process all incoming messages, you can specify which ones to translate.

To specify the objects and interactions that the Gateway translates:

1. Choose Configuration → Translations. The Translation Manager opens (Figure 3-3). The interactions and objects that you can translate are listed in a tree-style hierarchy.

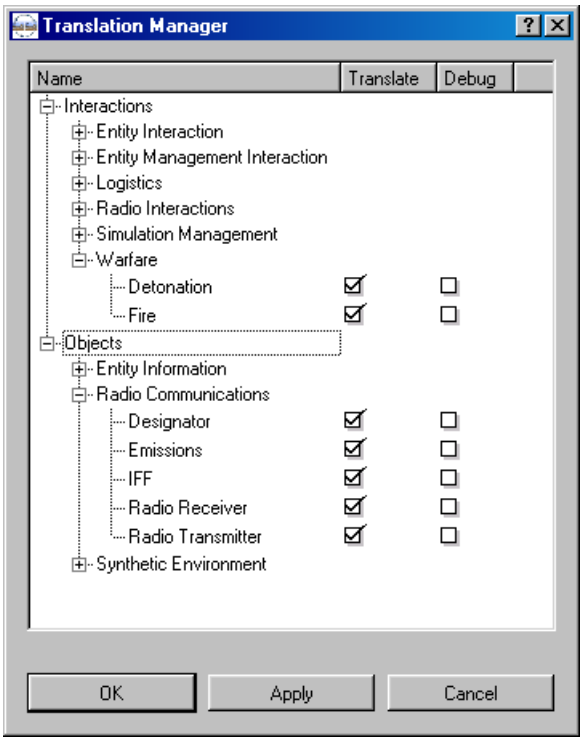


Figure 3-3. Translation Manager

2. Expand the hierarchy to view the interactions and objects available for translation.
3. Check the box in the Translate column for each object or interaction that you want to translate.
4. Check the box in the Debug column if you want to view the contents of the messages that the Gateway is translating.
5. Click OK.

You can change the list of objects and interactions that the Gateway translates while the Gateway is translating messages. Changes take place immediately.

Viewing Debug Data

If you check the Debug box for an interaction or object in the Translation Manager, the Gateway prints information to the *gateway.log* file and the console (if enabled.) The information printed is determined by the **notifyLevel** setting in *gateway.mtl*. To see message information, set **notifyLevel** to at least 2. Messages such as the following are printed:

```
Gateway Packet: ------------ ORIG-(Detonation) DIS at 16.999478
From:               1:15:1
Target:             0:0:0
Munition:           1:15:10
Event:              1:15:2
WorldLocation:      { -2661388.272,  -4359496.289,   3807301.109}
Velocity:           {        0.000,         0.000,         0.000}
Result:             DetResGroundImpact   (3)
FuseType:           FuzeOther   (0)
MunitionType:       2:2:225:1:3:0:0
Quantity:           1
Rate:               0
WarheadType:        WarheadOther   (0)
EntityLocation:     {        0.000,         0.000,         0.000}
NumArtParams:       0
Gateway Packet: ============ NEW-(Detonation) HLA  ============
From:               1:15:1gw
Target:
Munition:           1:15:10gw
Event:              2:Gateway
WorldLocation:      { -2661388.272,  -4359496.289,   3807301.109}
Velocity:           {        0.000,         0.000,         0.000}
Result:             DetResGroundImpact   (3)
FuseType:           FuzeOther   (0)
MunitionType:       2:2:225:1:3:0:0
Quantity:           1
Rate:               0
WarheadType:        WarheadOther   (0)
EntityLocation:     {        0.000,         0.000,         0.000}
```

3.5. Translating DIS and HLA Messages

After you configure the Gateway for your exercise, you need to tell it to start translating DIS and HLA messages.

➤ To start translating messages, choose File → Start Translations.

The Gateway status indicator changes from Inactive to Active (Figure 3-4). The timer displays the Gateway simulation time, in seconds. This time is included in all messages that the Gateway generates.

3.5.1. Viewing Objects and Interactions

The Gateway displays lists of the objects (Figure 3-4) and interactions that it discovers. Interactions and objects are listed on separate tabs and are also separated by simulation standard (DIS and HLA). As objects join and leave the exercise, the Gateway updates the display so that it only shows currently simulated objects. The Gateway does not clear the Interactions list. As more interactions are discovered, they are added to the list.

As noted in "Attributes Required for Sending DIS PDUs," on page 1-4, the Gateway does not send PDUs for an object until it has all of the information it needs to complete the PDU. If the Gateway is waiting for additional information about an object, it lists the object in the Discovered Objects window and displays an asterisk (*) in front of the object's name. When the PDU for an object is sent, the asterisk is removed.

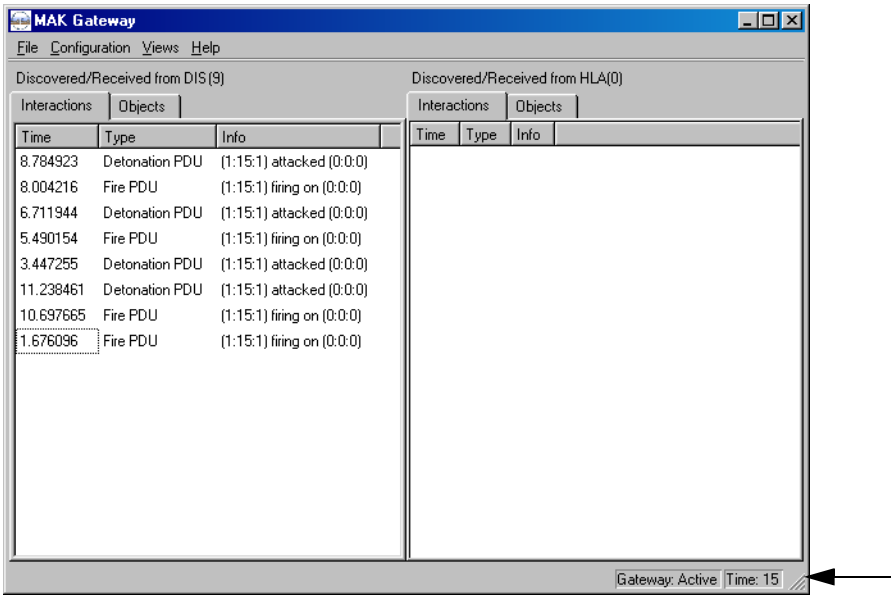


Figure 3-4. Gateway window

➤ To view objects and interactions, select the Interactions or Objects tab for the protocol whose messages you want to view.

Clearing the Interactions List

If you want to, you can clear the Interactions list.

➤ To clear the Interactions list, choose Views → Clear Interactions.

3.5.2. Stopping Message Translation

You can stop the Gateway from processing state update messages (PDUs).

➤ To stop the processing of update messages, choose File → Stop Translations.

A

Gateway MTL Parameters

This chapter describes the parameters in the *gateway.mtl* file.

MÄK Technologies Lisp (MTL) Files A-2

Gateway Configuration Parameters A-3

A.1. MÄK Technologies Lisp (MTL) Files

Configuration files for MÄK Technologies products are ASCII text files that use MÄK Technologies LISP (MTL) to encode configuration information. MTL files have the extension *.mtl*. You can edit them in any text editor. Be sure that your text editor uses the proper end-of-line characters for your platform.

For the Gateway, most parameter entries take the format:

```
(setq parameter_name value)
```

For example:

```
(setq allowRemoteFileDelete 0)
```

To comment out a line, precede the text with two semi-colons (;;). You can also add a comment at the end of a line by preceding it with two semi-colons.

In addition to the `setq` command, a commonly used command is the `load` command. This command instructs the Gateway to load the file specified, for example:

```
(load ./mydata "mysublist.mtl")
```

You can put all MTL arguments in one file, or segregate them into files that have a special purpose, such as a file to list the subscribed classes and interactions. When you start the Gateway, it automatically loads the *Logger.mtl* file in the directory in which the executable is located.

A.2. Gateway Configuration Parameters

Table A-1 describes the configuration parameters in the *gateway.mtl* file.

Table A-1: Gateway configuration parameters (Sheet 1 of 4)

Parameter	Description
disPort	Specifies the port to send/receive network PDUs on. Default: 3000.
disExerciseId	The DIS exercise ID.
disSiteId	The DIS site ID.
disApplicationNum	The DIS application number. It is used for creating entity IDs for PDUs originating from the Gateway. It must be unique.
disDestAddr	Specifies the address of outgoing DIS PDUs. If set to 0.0.0.0, the Default broadcast address is used.
hlaFedName	The name of this federate. Default: VR-Link.fed.
hlaExecName	The name of the federation execution. Default: VR-Link.
hlaFedFileName	The name of the FED file being used for the HLA federation execution. It must be the complete name including the three char extension (.fed).
hlaRPRFOMVersion	The version of the RPR FOM that you are using. Default: 1.0.
pollInterval	Specifies the interval, in seconds, between each call to drainInput() on the exercise connection.
heartbeatInterval	Specifies the number of seconds between heartbeats of published entities. Default: 5.0.
timeoutInterval	Specifies the number of seconds before a reflected entity times out and the Gateway removes it. Default: 10.0.
logFileName	Specifies the name of the file to dump Gateway output to. This flag is only valid on Windows. Default: *gateway.log*
useConsoleOutput	Specifies creation of a Windows console for all Gateway output. This flag is only valid on Windows. Default: 0. ◆ 0 = Do not create a console ◆ 1 = Create a console.

Table A-1: Gateway configuration parameters (Sheet 2 of 4)

Parameter	Description
autoStart	Tells the Gateway to automatically start translating on startup. ◆ 0 = Do not start translating automatically ◆ 1 = Start translating automatically. Default: 0.
Debug Options	
notifyLevel	Specifies the level of notification messages sent by VR-Link. Range: 0-3. Default: 1.
debugGlobalIdMapping	Specifies whether or not to map HLA global object designators to DIS entity IDs in the debug output. ◆ 0 = Do not map HLA global object designators to DIS entity IDs ◆ 1 = Map HLA global object designators to DIS entity IDs. Default: 0.
debugEntityIdMapping	Specifies whether or not to map HLA entity IDs to DIS entity IDs in the debug output ◆ 0 = Do not map HLA entity IDs to DIS entity IDs ◆ 1 = Map HLA entity IDs to DIS entity IDs. Default: 0.
Advanced Configuration	
hlaFomMapperLibName	The name of a FOM mapper library, if you are using one. Default: "". See "Specifying the FOM Mapper and FED File to Use," on page 3-7.
hlaFomMapperLibInitData	A string that is passed to the FOM Mapper library specified in fomMapperLibName. This data is optional. Default: "". See "Specifying the FOM Mapper and FED File to Use," on page 3-7.
gatewaySendPort	Specifies the port the Gateway sends PDU's on. This port is only to be used by the Gateway. It knows that if it receives a message with this port, that it is its own message, so it doesn't translate it. Make sure this port doesn't conflict with others on your network. Default: 2999.

Table A-1: Gateway configuration parameters (Sheet 3 of 4)

Parameter	Description
disDrainTimeout	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 disNumPdusToReadPer-TimeCheck PDUs at a time, until all PDUs are read or the timeout value is reached, whichever comes first. Default: -1.
disNumPdusToReadPerTimeCheck	Specifies the number of PDUs to read before checking to see if the amount of time specified by disDrainTimeout has been reached. Default: 100.
hlaMaxTickTIme	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 hlaMinTickTime, but no more than this value. Default: -1.
hlaMinTickTime	Specifies the minimum amount of time, in seconds, to tick the RTI, before processing incoming messages. Default: 0.0.
disProtocolVersionToSend disProtocolVersionToRecvMin disProtocolVersionToRecvMax	The DIS PDU versions to send and receive, where: ◆ 4 = DIS 2.0.4 ◆ 5 = IEEE 1278.1 ◆ 6 = IEEE 1278.1a. Default version to receive: 4, 5, and 6. Default version to send: 5.
GUI Options	
showObjectUpdates	Specifies display of the last time that the Gateway received an update for this object. ◆ 0 = Do not display the update time ◆ 1 = Display the update time. Default: 1.

Table A-1: Gateway configuration parameters (Sheet 4 of 4)

Parameter	Description
showObjects	Specifies display of discovered and published objects. It does not affect the translation of objects. ◆ 0 = Do not display discovered objects ◆ 1 = Display discovered objects. Default: 1.
showInteractions	Specifies the display of interactions. It does not affect the translation of interactions. ◆ 0 = Do not display interactions ◆ 1 = Display interactions. Default: 1.

Index

A

-A startup option 3-4
-a startup option 3-4
Acknowledge class 1-9
Acknowledge PDU 1-9
Action Request PDU 1-9
Action Response PDU 1-9
ActionRequest class 1-9
ActionResponse class 1-9
Active 3-10
adding
  new license file 2-10
address
  destination 3-6, A-3
application number 3-6, A-3
ApplicationSpecificRadioSignal interaction 1-10
attributes
  missing 1-4
auto start 3-3
automatically starting translations A-4
autoStart parameter 3-5, A-4

B

-b startup option 3-4
BaseEntity class 1-5, 1-8
beam ID 1-11
BeamID parameter 1-4
BeamParameterIndex parameter 1-4
broadcast address 3-6, A-3

C

-C startup option 3-4
class
  Acknowledge 1-9
  ActionRequest 1-9
  ActionResponse 1-9
  BaseEntity 1-5, 1-8
  Collision 1-8
  Comment 1-9
  CreateEntity 1-9
  Data 1-9
  DataQuery 1-9
  Designator 1-4, 1-10
  EmbeddedSystem 1-10, 1-11
  EmitterBeam 1-4
  EmitterSystem 1-4, 1-10, 1-11
  EmittingSystemID 1-11
  EventReport 1-9
  GroundVehicle 1-8
  IFF 1-5, 1-10
  IffInterrogator 1-10, 1-11
  IffTransponder 1-10, 1-11
  JammerBeam 1-11
  MunitionEntity 1-8
  PhysicalEntity 1-8
  RadarBeam 1-11
  RadioReceiver 1-5, 1-10
  RadioTransmitter 1-4, 1-10
  RemoveEntity 1-9
  RepairComplete 1-5, 1-12
  RepairResponse 1-5, 1-12
  ResupplyCancel 1-5, 1-12
  ResupplyOffer 1-5, 1-12
  ResupplyReceived 1-5, 1-12
  ServiceRequest 1-5, 1-12
  SetData 1-9
  StartResume 1-9

StopFreeze 1-9
clearing interaction list 3-11
closing
Gateway 3-5
Collision class 1-8
Collision PDU 1-8
comment
MTL parameter A-2
Comment class 1-9
Comment PDU 1-9
configuration file 3-9
configuring
Gateway 3-5
Gateway through GUI 3-6
GUI options A-5
system for MÄK RTI 2-14
console A-3
conventions
manual viii
Create Entity PDU 1-9
CreateEntity class 1-9

D

data
debug 3-9
Data class 1-9
Data PDU 1-9
Data Query PDU 1-9
DatabaseIndexRadioSignal interaction 1-10
DataQuery class 1-9
Debug
box 3-9
column 3-8
debug data 3-9
debugEntityIdMapping parameter A-4
debugging translations 3-8
debugGlobalIdMapping parameter A-4
Delta State EE PDU 1-11
Designator class 1-4, 1-10
Designator PDU 1-11
destination address 3-6, A-3
Detonation PDU 1-9
DIS
version A-5
DIS protocol 1-2
disApplicationNum parameter 3-5, A-3
discovered object and interaction
viewing 3-10
disDrainTimeout parameter A-5
disExerciseId parameter 3-5, A-3
disNumPdusToReadPerTimeCheck parameter A-5
disPort parameter 3-5, A-3
disProtocolVersionToRecvMax parameter A-5
disProtocolVersionToRecvMin parameter A-5
disProtocolVersionToSend parameter A-5
disSiteId parameter 3-5, A-3
Distributed Emission Regeneration protocol family 1-11
DMSO RTI 2-15

E

-e startup option 3-4
EE PDU 1-11
Electromagnetic Emission PDU 1-11
embedded system object
deletion of 1-6
EmbeddedSystem class 1-10, 1-11
emission 1-11
emitter beam 1-11
emitter system 1-11
EmitterBeam class 1-4
EmitterName parameter 1-4
EmitterNumber parameter 1-4
EmitterSystem class 1-4, 1-10, 1-11
emitting system ID 1-11
EmittingSystemID class 1-11
EmittingSystemID parameter 1-4
EncodedAudioRadioSignal interaction 1-10
entity ID 1-8, 1-10
Entity Information 1-8
Entity Management PDU family 1-12
Entity State PDU 1-8
entity type 1-8
EntityID parameter 1-5
EntityType parameter 1-5
environment variable
MAKLMGRD_LICENSE_FILE 2-8
RTI_CONFIG 2-15
Environmental Process PDU 1-12
EnvironmentProcess class 1-12
ERP parameter 1-4
Event Report PDU 1-9
EventReport class 1-9
executing gateway
UNIX 3-2
Windows 3-3
exercise connection A-3
exercise ID 3-6, A-3

F

-F startup option 3-4
-f startup option 3-4
FED file 1-3, 2-15, 3-7, A-3
federate name 3-6, A-3
federation execution 1-3, 3-7, A-3
file
  FED 2-15
  gateway.log 3-9
  gateway.mtl 3-5, 3-7, 3-9
  license 2-8
  log A-3
  MTL A-2
  RID 2-15
  RTI.RID 2-15
filtering
  objects and interactions 3-8
Fire PDU 1-9
flexlm
  directory location 2-6
  installing 2-6
  license file 2-8
  license server name 2-7
  multiple licenses 2-10
  troubleshooting 2-11
FLEXlm license manager 2-5
FOM
  supported 1-3
FOM agility 1-2
FOM Mapper 1-2, 3-7
FOM mapper library A-4

G

gateway.log file 3-9, A-3
gateway.mtl file 3-5, 3-7, 3-9
gatewaySendPort parameter 3-5, A-4
Gridded Data PDU 1-12
GriddedData class 1-12
GRIM RPR 1-3
GroundVehicle class 1-8
GUI
  configuring A-5

H

-H startup option 3-4
-h startup option 3-4
handle
  mapping 1-4
heartbeat 1-6, 1-11, A-3
heartbeatInterval parameter 3-5, A-3
HLA
  object 1-4, 3-10
  services 1-3
hlaExecName parameter 3-5, A-3
hlaFedFileName parameter 3-5, 3-7, A-3
hlaFedName parameter 3-5, A-3
hlaFomMapperLibInitData parameter 3-7, A-4
hlaFomMapperLibName parameter 3-7, A-4
hlaMaxTickTIme parameter A-5
hlaMinTickTime parameter A-5
hlaRPRFOMVersion parameter 3-5, 3-7, A-3
host entity ID 1-10
host ID
  getting 2-7
host name 2-7
host object ID 1-10
HostObjectID parameter 1-4

I

IFF class 1-5, 1-10
IFF PDU 1-11
IffInterrogator class 1-10, 1-11
IffTransponder class 1-10, 1-11
Inactive 3-10
incomplete
  object information 1-4, 3-10
installing
  DMSO RTI 2-15
  flexlm software 2-6
  MÄK RTI 2-14
interaction 1-8
  ApplicationSpecificRadioSignal 1-10
  clearing list 3-11
  collision 1-8
  DatabaseIndexRadioSignal 1-10
  EncodedAudioRadioSignal 1-10
  MunitionDetonation 1-9
  RadioSignal 1-10
  RawBinaryRadioSignal 1-10
  subscribing to 3-8
  translating 3-8
  viewing list of 3-10
  WeaponFire 1-9

J

JammerBeam class 1-11

L

license file 2-8
  adding new 2-10
license server
  name 2-7
  running 2-11
  status 2-11
  stopping 2-11
  troubleshooting 2-11
lmdown program 2-11
lmhostid 2-7
lmstat program 2-11
log file 3-9, A-3
logFileName parameter A-3
Logistics protocol family 1-12

M

MÄK RTI
  configuring 2-14
  installing 2-14
MÄK Technologies LISP A-2
MAKLMGRD_LICENSE_FILE environment variable 2-8
manual
  conventions viii
mapping
  entity type to object class 1-8
message 3-9
missing
  attributes or parameters 1-4
MTL file 3-9, A-2
MunitionDetonation interaction 1-9
MunitionEntity class 1-8

N

-n startup option 3-4
notification level 3-9
notify level A-4
notifyLevel parameter 3-9, A-4

O

object
  subscribing to 3-8
  time out A-3
  translating 3-8
  updates 1-4, 3-10
  viewing list of 3-10

P

-P startup option 3-4
parameter
  autoStart 3-5, A-4
  BeamID 1-4
  BeamParameterIndex 1-4
  commenting out A-2
  debugEntityIdMapping A-4
  debugGlobalIdMapping A-4
  disApplicationNum 3-5, A-3
  disDrainTimeout A-5
  disExerciseId 3-5, A-3
  disNumPdusToReadPerTimeCheck A-5
  disPort 3-5, A-3
  disProtocolVersionToRecvMax A-5
  disProtocolVersionToRecvMin A-5
  disProtocolVersionToSend A-5
  disSiteId 3-5, A-3
  EmitterName 1-4
  EmitterNumber 1-4
  EmittingSystemID 1-4
  EntityID 1-5
  EntityType 1-5
  ERP 1-4
  gatewaySendPort 3-5, A-4
  heartbeatInterval 3-5, A-3
  hlaExecName 3-5, A-3
  hlaFedFileName 3-5, 3-7, A-3
  hlaFedName 3-5, A-3
  hlaFomMapperLibInitData 3-7, A-4
  hlaFomMapperLibName 3-7, A-4
  hlaMaxTickTIme A-5
  hlaMinTickTime A-5
  hlaRPRFOMVersion 3-5, 3-7, A-3
  HostObjectID 1-4
  logFileName A-3
  missing 1-4
  notifyLevel 3-9, A-4
  pollInterval 3-5, A-3
  Position 1-5
  ReceivingObject 1-5
  RepairingObject 1-5
  RequestingObject 1-5

ServicingObject 1-5
showInteractions A-6
showObjects A-6
showObjectUpdates A-5
SupplyingObject 1-5
SystemType 1-5
timeoutInterval A-3
PDU 1-2
Acknowledge 1-9
Action Request 1-9
Action Response 1-9
Collision 1-8
Comment 1-9
Create Entity 1-9
Data 1-9
Data Query 1-9
delta state EE 1-11
designator 1-11
detonation 1-9
electromagnetic emission 1-11
Entity Management 1-12
Entity State 1-8
Event Report 1-9
fire 1-9
IFF 1-11
incomplete 1-4, 3-10
protocol families 1-6
radio transmitter 1-10
Remove Entity 1-9
Repair Complete 1-12
Repair Response 1-12
Resupply Cancel 1-12
Resupply Offer 1-12
Resupply Received 1-12
sending and receiving 3-6, A-3
service request 1-12
Set Data 1-9
simulation management (SIMAN) 1-9
Start/Resume 1-9
state update EE 1-11
Stop/Freeze 1-9
supported 1-6
Synthetic Environment 1-12
timeout for reading A-5
PhysicalEntity class 1-8
pollInterval parameter 3-5, A-3
port 3-6, A-3, A-4
position attribute 1-8
Position parameter 1-5
Preferences dialog box 3-6
processing
stopping 3-11
products
technical support vii
upgrades vii
protocol
family 1-6
Protocol Data Unit 1-2
publication 1-4

Q

quitting Gateway 3-5

R

-R startup option 3-4
RadarBeam class 1-11
Radio Communications protocol family 1-10
radio object 1-10
Radio Transmitter PDU 1-10
RadioReceiver class 1-5, 1-10
RadioSignal interaction 1-10
RadioTransmitter class 1-4, 1-10
RawBinaryRadioSignal interaction 1-10
reading
PDUs A-5
Real-Time Platform Level Reference 1-2
receiver protocol, signal protocol 1-10
receiving
PDUs 3-6, A-3
ReceivingObject parameter 1-5
Remove Entity PDU 1-9
RemoveEntity class 1-9
Repair Complete PDU 1-12
Repair Response PDU 1-12
RepairComplete class 1-5, 1-12
RepairingObject parameter 1-5
RepairResponse class 1-5, 1-12
RequestingObject parameter 1-5
Resupply Cancel PDU 1-12
Resupply Offer PDU 1-12
Resupply Received PDU 1-12
ResupplyCancel class 1-5, 1-12
ResupplyOffer class 1-5, 1-12
ResupplyReceived class 1-5, 1-12
RID file 2-15
RPR FOM 1-2, 3-7, A-3
RTI
definition of 2-14

DMSO, installing 2-15
environment 3-2
MÄK, installing 2-14
ticking A-5
RTI_CONFIG environment variable 2-15
rtiexec 2-15, 3-2
RTI.rid file 2-15
runLm program 2-11
running
Gateway 3-10
license server 2-11

S

-S startup option 3-4
-S startup option. 3-3
-s startup option 3-4
sending
PDUs 3-6, A-3
Service Request PDU 1-12
ServiceRequest class 1-5, 1-12
ServicingObject parameter 1-5
Set Data PDU 1-9
SetData class 1-9
showInteractions parameter A-6
showObjects parameter A-6
showObjectUpdates parameter A-5
shutting down
license server 2-11
SIMAN PDU 1-9
Simulation Management PDU 1-9
SISO 1-3
site ID 3-6, A-3
SOM 1-3
starting
translations 3-10
translations automatically 3-3
starting gateway
UNIX 3-2
Windows 3-3
StartResume class 1-9
Start/Resume PDU 1-9
startup
automatic A-4
startup option 3-5
startup options 3-4
State Update EE PDU 1-11
status
license server 2-11
status indicator 3-10
Stop 1-9
StopFreeze class 1-9
Stop/Freeze PDU 1-9
stopping
license server 2-11
PDU processing 3-11
subscribing
objects and interactions 3-8
subscription 1-4
SupplyingObject parameter 1-5
support
technical vii
supported PDUs 1-6
Synthetic Environment PDU family 1-12
SystemType parameter 1-5

T

-T startup option 3-4
technical support vii
tick A-5
time out A-3
timeout 1-6
reading PDUs A-5
timeoutInterval parameter A-3
Translation Manager 3-9
Transfer Control Request PDU 1-12
TransferControl class 1-12
translating
objects and interactions 3-8
PDUs 3-10
translation
debugging 3-8
stopping 3-11
Translation Manager 3-8
translations
starting automatically 3-3
transmitter protocol 1-10
troubleshooting
flexlm 2-11

U

update
HLA object 1-4, 3-10
upgrades vii

V

variable
  environment 2-8
version
  DIS A-5
viewing
  objects and interactions 3-10
VR-Link 1-2
VR-Link.fed 1-3

W

warfare protocol family 1-9
WeaponFire interaction 1-9

X-Y-Z

-x startup option 3-4

Link - Simulate - Visualize

GWY-4.0-1-030508

185 ALEWIFE BROOK PARKWAY CAMBRIDGE, MA 02138 617.876.8085 www.mak.com