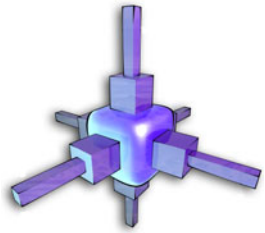

MÄK RTI 3.2 Release Notes

This document provides the following release-specific information for MÄK RTI 3.2:

Document ID: RTI-3.2-2-071106

MÄK RTI 3.0 has been verified by DMSO as fully compliant with the IEEE 1516 version of the HLA Interface Specification (SISO DLC HLA API 1516 (SISO-STD-004.1-2004).)

Supported Systems

Table 1 lists the platforms supported by MÄK RTI 3.2. Please contact MÄK if you need it for another platform. Application code must be built with the indicated compilers in order to link to the MÄK RTI libraries.

Table 1: Platforms supported

Operating System	Compiler
SGI IRIX6.5	MipsPro7.4.2m C++ compiler
Sun Sparc Solaris 10	Sun C++ 5.8
Red Hat Enterprise Linux Workstation 4	Default compiler.
Red Hat Enterprise Linux Workstation 5	Default compiler.
Red Hat Fedora 7	Default compiler.
Windows XP	Microsoft Visual C++ 7.1, 8.0

Compiler Compatibility on Windows

MÄK provides versions of product releases that have been compiled with Microsoft Visual C++ 7.1 and 8.0. When you run MÄK products together on the same computer, for example, the MÄK RTI and the Stealth, we strongly recommend that you run versions compiled with the same compiler. Mixing products compiled with different versions of the compiler can result in program instability.

Libraries and Binaries Built with Visual Studio 2005

All MÄK products built with Microstoft Visual Studio require the C Runtime Library to function. The C runtime libraries have always been available from Microsoft for download, they are also installed on a user's machine when a Microsoft compiler is installed. The C runtime libraries are not part of the normal Windows installation. For customers who plan to use MÄK products on machines that do not have a compiler installed, MÄK has historicaly distributed a copy of the C Runtime Libraries with MÄK products. These libraries were put in the *bin* directory used by the MÄK products. MÄK products would then use the libraries in the *bin* directory and customers would not have a problem if copies of the libraries were not already installed.

Unfortunatly, with the release of the new C Runtime Libraries required by Microsoft Visual Studio 2005 (MSVC++8.0), the libraries can no longer just be copied into the *bin* directory of an application. The libraries need to be installed correctly into Windows system folders. (The process is actually a little more complicated, a manifest file needs to be created to tell Windows where to find the libraries.)

To accommodate this change, MÄK is distributing the Windows installer for the C runtime libraries with all MÄK products released for MSVC++8.0. The installer is named *vcredist_x86.exe* and is in the base directory of any installed MÄK product.

Running the installer requires Administrator privileges for the machine the installer is run on. MÄK has chosen to not integrate the MÄK installer and the Microsoft installer so as not to require users to have Administrator privileges to install MÄK products. Therefore, if you who do not have a compiler installed, or get error messages like "Software has not been installed correctly, please re-install", you must apply the patch.

For more information see this Microsoft URL:

http://msdn2.microsoft.com/en-us/library/ms235299.aspx

FLEXlm Support

MÄK RTI 3.2 uses FLEXlm 11.4 for Windows, Linux, and SGI IRIX. Sun Solaris uses FLEXlm 9.2.

Third Party Library Dependencies

In addition to the Qt dependencies mentioned previously, the MÄK RTI 3.2 has the following dependencies:

- QT - 3.3.5 - http://www.trolltech.com/
- QWT - 0.4.1 - http://qwt.sourceforge.net/
- EHS - 1.3.1 - http://xaxxon.slackworks.com/ehs/
- PME - 1.0.4 - http://xaxxon.slackworks.com/pme/
- PCRE - 6.4.1- http://www.pcre.org/
- ZLIB - 1.2.3 - http://www.zlib.net/
- ICONV - 1.8 - http://www.gnu.org/software/libiconv/
- LIBXML2 - 2.6.22 - http://www.xmlsoft.org/
- prototype.js - 1.5.0 - http://prototype.conio.net/
- script.aculo.us - 1.6.1 - http://script.aculo.us
- VR-Link 3.11.1.

Patch Required for AMD Dual-processor Windows PCs

VR-Link-based products use a high resolution counter for time calculations on Windows PCs. Customers who are running Windows on PCs with multiple AMD Athlon 64-bit processors may notice clock jitter, which may cause time in MÄK products to run backwards. This occurs when the Windows scheduler changes the CPU the MÄK process is using. If the high resolution counters on each processor are not synchronized, the application may witness a decrease in the high resolution counter value stored in the processor causing an incorrect time calculation. To fix this problem customers, apply the AMD Dual-Core Optimizer patch provided by AMD. You can get the patch at:

http://www.amd.com/us-en/Processors/TechnicalResources/0,,30_182_871_9706,00.html

If you get an error when you try to access this URL, reload the page.

RTIspy Web GUI Configuration

The RTIspy Web GUI has the following system considerations:

- The RTIspy Web GUI is optimized for screen resolutions of 1024x768 and larger. For smaller screen sizes, additional window scrolling will be required.
- The RTIspy Web GUI has been tested and will display correctly with Mozilla Firefox version 1.5 and higher and Internet Explorer versions 6 and higher. Other browsers may be compatible, but they have not been tested and are not specifically supported.
- The RTIspy Web GUI requires an RTI Plus license for each LRC. If licenses are available for only a subset of the federation, then those licenses will be consumed in the order that federates are started. Once all licenses are used, additional federates will not be available for monitoring (although they will still be viewable at the federation level).

MÄK RTI Licensing for the RTI Forwarder

You can run the RTI Forwarder without an RTI Plus license if it is used stand-alone (that is, centralized forwarding, in which it is not part of a distribution network and the routes file is empty.) It uses a license if it is run as part of a distributed network with other forwarders. An RTI Plus license allows you to run:

- One federate with GUI and plug-ins
- One rtiexec with GUI
- One distributed forwarder
- Plug-ins for either the rtiexec or the Distributed Forwarder (but not both with a single license).

Backwards Compatibility

The MÄK RTI 3.2 libraries are not run-time compatible with previous versions. (In other words, you cannot have a federation execution in which some federates use MÄK RTI 3.2 and some use a previous version of the MÄK RTI.) However, the MÄK RTI 3.2 is link compatible with previous versions, so a federate does not need to be recompiled to use this version. We recommend that all federates use the same version of the RTI when operating within the same federation (or using the same *rtiexec*).

Network Compatibility

The MÄK RTI is not network compatible with other RTIs. (In general, no two RTI implementations are network compatible, due to differences in the low-level network mechanisms that they use (which include, but are not limited to packet layout.)) Applications that want to interoperate in the same federation execution must all use the same RTI implementation. A federation can easily switch from one RTI implementation to another between runs, but during each run, all participants must agree on which RTI to use, much as they must also agree on which FOM to use.

Compatibility with Other RTIs

The MÄK RTI 1.3 implements the API dictated by the 1.3 version of the HLA Interface Specification, as amended by DMSO's official HLA 1.3 Interpretations document. All RTIs that are verified as compliant with the HLA 1.3 specification are written to this exact API standard. This insures that the MÄK RTI is link-compatible with other RTIs that have been verified as HLA compliant.

The MÄK RTI 1516 implements the SISO DLC HLA API[1] 1516 (SISO-STD-004.1-2004). This API supports dynamic link compatibility, which was problematic with the original IEEE 1516 API. The IEEE 1516 API only supports compile time compatibility (uses C++ templates and implementation-specific header files). The MÄK RTI is compatible with any RTI written to the SISO DLC HLA API 1516.

Because an RTI is typically accessed by an application through a dynamic library (*.dll* on Windows, *.so* on UNIX), applications do not need to be recompiled or relinked to switch among dynamic-link-compatible RTI implementations that have been built using the same compilers. The only requirement is that the appropriate version of the library be located within the shared library search path. For more information, please see Section 4.2, "Compiling and Linking with the MÄK RTI 1.3" in *MÄK RTI Reference Manual*.

1. Simulation Interoperability Standards Organization Dynamic Link Compatible High Level Architecture Application Program Interface

Because the DMSO RTI NG v6 was last built with the MSVC++ 6.0 compiler, it is not link compatible with the MÄK RTI 3.2. However, updated versions of the DMSO RTI provided by other vendors using compilers supported by MÄK RTI 3.2 would be link compatible.

New Features and Updates

MÄK RTI 3.2 is primarily a maintenance release. However it also has the following improvements:

- Support for FOM modules and update rate reduction, which are features in the HLA Evolved draft specification.
- Support for implicit DDM.
- The Qt based LRC GUI has been removed.
- Support for a network map in the web-based RTIspy GUI.
- The rtiexec GUI no longer requires an RTI Plus license. It is the default environment for the rtiexec.
- Support for RTI Forwarder groups.
- Support for asynchronous message processing
- Version control for RTI messages.
- Diagnostic messages include a timestamp and tick count.

Modular FOMs

The MÄK RTI allows a federation to be formed and modified using FOM modules. The use of FOM modules allows you to integrate federates that use the same reference FOM, but which have added different, non-conflicting extensions, without having to merge the FOM files. It supports more agile federations in which only FOM modules pertinent to a particular federate or subset of federates need to be introduced by that federate into the federation. Federates can specify one or more FOM modules when creating a federation, and more significantly, a federate can specify one or more FOM modules when it joins. For details, please see Chapter 12, *Using FOM Modules*, in *MÄK RTI Reference Manual*.

Modular FOMs is one of the SISO HLA Evolved PDG modifications submitted for the update of the HLA standard.

Update Rate Reduction

Update rate reduction is a mechanism that allows federates to specify the rate at which attribute updates can be delivered to the federate. This feature augments the declaration management (DM) and data distribution management (DDM) services that provide mechanisms for reducing or filtering the data that a federate receives based on class and data content. Update rate reduction applies to best effort attribute updates. It does not affect interactions or reliable attribute updates. For details, please see Chapter 13, *Update Rate Reduction*, in *MÄK RTI Reference Manual*.

Update rate reduction is one of the SISO HLA Evolved PDG modifications submitted for the update of the HLA standard.

RTIspy GUI Network Map

The network map displays the RTI components graphically, showing the links between them (Figure 1-1). The links represent the reliable connections (TCP) between the components. Components may communicate through other connectionless methods that are not represented in the network map (UDP). For details, please see Section 9.2.1, "Viewing the Network Map", in *MÄK RTI Reference Manual*.

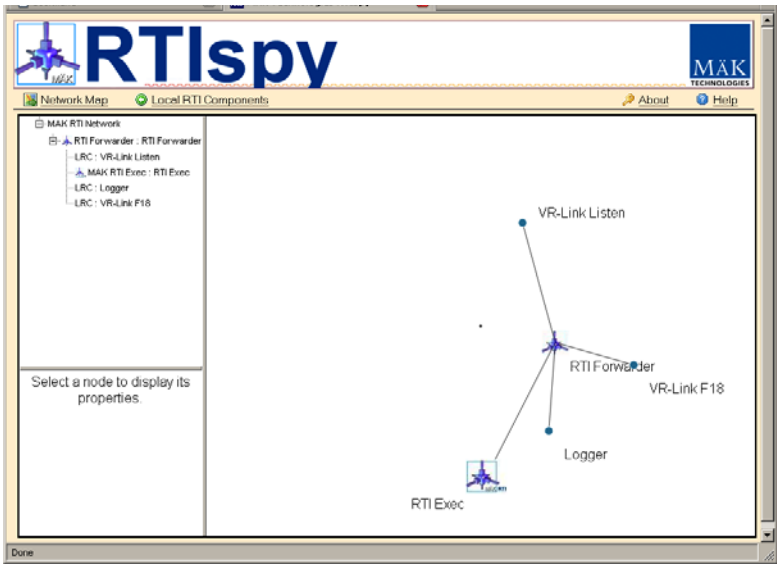

Figure 1-1. Network map, no components selected

Implicit DDM

Implicit data distribution management (DDM) provides the advantages of DDM filtering without a federate using the DDM services. Implicit DDM uses a MÄK RTI plug-in that substitutes non-DDM method calls from a federate with DDM method calls. The DDM calls create regions and modify region extents using a configurable set of criteria based on geographic data as specified by the RPR FOM. For details, please see Section 10.3, "Implicit DDM", in *MÄK RTI Reference Manual*.

Asynchronous Message Processing

Asynchronous message processing adds a third mode to the asynchronous process model, which is half way in between asynchronous I/O (which simply supports asynchronous message buffering) and asynchronous callbacks (which support fully asynchronous processing and callbacks.) In the asynchronous message processing mode, the asynchronous thread handles the network I/O and processes received messages. However, any federate ambassador callbacks that are generated enter a queue to be delivered to the federate synchronously during its call to the tick or evokeCallback RTI Ambassador service.

This processing mode relieves the federate from having to call tick in order to provide the RTI with processing cycles. It may improve the response and efficiency of some distributed algorithms (for example, time management). However, it requires the caching of callback parameters, which results in additional memory allocation and data copy operations.

Diagnostic Message Improvements

Elapsed time timestamps and tick count have been added to the diagnostic message output (notify level > 3).

DDM Improvements

Fixed grid now support multiple extents for subscriptions. While multiple extents is generally allowed with any region, it is still the case that only the first extent will be used in update regions.

Fixed grid configuration now supports a simple flat partitioning of dimensions with no subspace divisions.

Version Control for Messages

Messages exchanged by the RTI are now version controlled. As of this version (3.2), the RTI will discard messages coming from incompatible versions.

RTI Forwarder Groups

RTI Forwarder groups is an option for configuring RTI Forwarders to share network load. Please see Chapter 7, *Configuring Distributed Forwarding*, in *MÄK RTI Reference Manual* for an extensive discussion of options for configuring distributed forwarding.

RTIspy API Change

Access to the RTI's diagnostic output streams has been added via the following four macros: DtRtiFatal, DtRtiWarn, DtRtiInfo, DtRtiVerbose, and DtRtiDebug. A plug-in should use these macros for diagnostic output in order for the output to be captured in the RTI's diagnostic logs.

Miscellaneous Changes

- Instead of exiting when a routes file cannot be found, even if specified in the RID, a forwarder will now just default to centralized forwarding mode.
- Plug-in diagnostic output is now captured by the log files (if enabled) in addition to the console output.
- The time management services are automatically disabled in lighweight mode as they require the rtiexec.
- The RID parameter **RTI_enableBigMessages** has been deprecated. Big messages (> 64K) are now inherently supported by the MÄK RTI. However, the message format still defaults to limit the size of individual attribute and parameter values to < 64K. If value sizes will be greater than 64K, enable the new RID parameter **RTI_use32BitsForValueSize**.

Documentation Updates

MÄK RTI Reference Manual has been updated and reprinted for this release.

MÄK RTI Reference Manual incorrectly states that a MOM module can extend an existing MOM class with additional attributes. It cannot. A MOM module can add subclasses of existing MOM classes. The subclasses can have attributes.

Bug Fixes

The following bugs have been fixed in this release:

- Bundling (**RTI_enablePacketBundling**) is allowed with large messages (> 64k), including messages with individual attribute or parameter values greater than 64K.
- The asynchronous callback mode is now supported by the 1.3 API.
- The rtidump utility will now correctly dump messages when bundling and compression are enabled.
- The RTI no longer throws InvalidRegionContext exceptions for a 1516 federate using DDM and a 1.3 FED.
- Support of the **RTI_addDestAddr** configuration feature has been fixed. It had problems with anything other than a simple network configuration.
- A federate issuing repeated ownership acquisition requests will no longer cause the owning federate to terminate.
- The RTI now correctly handles deleting orphan objects even when the "privilege to delete" attribute has been transferred to another federate.

Known Problems

This release has the following known problems:

- Distributed forwarder multicast subscription does not work for late-joining forwarders. With the introduction of the unified Distributed Forwarder a need arose to communicate between federates on a LAN and forwarders on remote LANs when a DM or DDM multicast group is joined or dropped, such that the forwarders can monitor or drop those groups on its LAN and forward messages back to the federates joined to such multicast groups. However the scheme to accomplish this does not work if the forwarder is not connected to the forwarder network when the federate joins the multicast group. The workaround is to ensure that the forwarder network is completely established prior to launching federates, or at the least, prior to allowing federates to attempt to join DM or DDM multicast groups.
- The RID consistency feature relies on the ability to establish a connection between a federate and the RID tester (presumably the rtiexec), however if the federate's **RTI_enableTcpCompression** parameter and the RID tester's parameter are different, the two cannot communicate, and consequently the RID consistency test request and response messages cannot be exchanged. The workaround is to ensure that the **RTI_enableTcpCompression** parameter is consistent among all participants prior to launch.
- The XML processing of the FDD DTD attribute differs between Linux and Windows. In Linux, the DTD is found relative to the federate's execution directory. In Windows, the file is found relative to the FDD location.

- In 1516, the HLA DTD is not distributed with the FDD file. In order for the LRC to process the FDD file, the HLA DTD must be accessible as specified in the FDD DTD attribute.
- The MÄK Technologies Lisp (MTL) files used by MÄK products to specify configuration settings are parsed at startup by the RTI. When a configuration file has certain errors, such as unmatched parentheses, the parser will not necessarily return an error code. This can cause the RTI to reach a partially initialized state or cause the RTI to initialize itself with default RID values, causing unexpected behavior. This applies to lisp expressions in the RID file as well as the expressions passed to the 1516 createRTIambassador function.
- A workaround has been added to deal with situations where a host running the RTI Forwarder application has multiple network interface adapters (NICs) or has multiple IP addresses assigned to an interface, which may be the case if the RTI Forwarder host is connected to the WAN via a VPN tunnel. Under these circumstances the RTI Forwarder may fail to initiate connections to other forwarders if the IP address corresponding to its host in the Forwarder List section of the *routes.mtl* file does not match the IP address the RTI Forwarder application receives when it queries the host itself for its IP address. To correct this the IP addresses of the interfaces on which it will connect to other RTI Forwarders must be entered into the *routes.mtl* file Interface Address List section. At a minimum one such entry should correspond to the entry identifying the host in the Forwarder List section of the *routes.mtl* file. Please refer to Section 7.4, "Configuring RTI Forwarders for Distributed Forwarding" in *MÄK RTI Reference Manual* for details about the Interface Address List entry format.
- The HLA 1.3 version of the RTI can sometimes fail to throw an "ObjectAlreadyRegistered" exception if a federate tries to register an object with a name that is already being used by another object. The federate can register an object using a name that belongs to another object that is unknown to the local RTI component (LRC). The LRC only maintains the names of objects that are registered or discovered by the federate. While we plan to fix this problem, a good preventive measure is to always subscribe and tick the RTI before registering objects.
- The rtiexec does not connect the output streams to a console until after reading the RID file in Windows. As a result, errors opening or parsing the RID file are not displayed. If **RTI_enablePopUpErrorMsgs** is enabled then these errors are displayed in a popup box, however this solution works only under Windows.
- To use shared memory on operating systems other than Windows, you must start the Shared Memory Queue Manager manually. Automatic launching of the shared memory queue manager has been disabled because the current implementation resulted in a conflict for console I/O.
- If you are using the Linux version of the RTIspy with versions of MÄK products earlier than the most recent releases, the MÄK products may crash. If this happens, contact MÄK technical support and we will provide you with a fix for your application.

- The RTI Dump utility does not work directly with shared memory only mode. It can be used with shared memory if the shared memory mode relays messages to the network and the RTI Dump utility is connected via the network communications.
- Smart Forwarding does not work when using fixed grid DDM. The smart forwarding feature relies on the exchange of DM and DDM information in order to establish a correct destination list. Because fixed grid DDM does not exchange DM or DDM information, enabling smart forwarding causes no messages to be sent because there are no destinations.
- When merging FOM modules, the dimensions are taken from the create FOM module and no others are merged.
- The RTIspy browser-based GUI does not work well with Internet Explorer 6 or 7. While you will be able to see and use the RTIspy GUI with IE7, the page layout may be problematic, and some information may not be displayed correctly. For optimal use, MÄK recommends using Firefox (2.0 or later).
- The RTIspy Network map displays federates on machines on which there is no network web server running. However, you will not be able to click on the node to view the federate's data.
- The RTIspy browser-based GUI does not work on Irix. We plan to address this issue in a future maintence or feature release. If this affects you, please contact MÄK Support.
- Using the MÄK RTI on Fedora Core 7 machines that have Xen Virtualization software installed causes a problem with multicast discovery. This is because INADDR_ANY maps to the virtual interface rather than the default, eth0. To resolve this issue, set **RTI_mcastDevAddrString** to the IP address of the real ethernet card.
- The RTIspy web-server may hang on machines running Solaris. If you are using RTIspy with Solaris, do not run the web-server. Point the web-service address (**RTI_webserviceAddr** in the RID file) to a web-server running on a different machine.
- The federate reconnect fault tolerance feature does not work when the rtiexec has more than one federation. The current implementation uses the federate handle as an identifier and it is possible for the same federate handle to be used in multiple federations.

RTI 1516 Implementation Issues

The MÄK RTI 1516 has the following known issues:

- While the 1516 API uses the std:wstring to exchange string information, internally, the MÄK RTI 1516 operates on narrow strings only (that is, representable by std::string). Therefore, the contents of any wide string passed through the RTI API must be convertible to a narrow string.

- The publish and subscribe services in the 1516 interface specification are additive when successive calls are made, versus replacement for the 1.3 interface specification. The MÄK RTI 1516 implementation follows the additive policy; however, the MÄK RTI 1.3 follows the replacement policy. As a result, the advisories and discoveries may conflict when a federation is composed of both 1.3 and 1516 federates. The **RTI_processUnknownUpdatesForDiscovery** RID parameter should be enabled as a work around for discoveries.