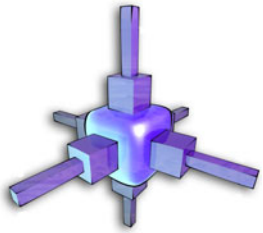

MÄK RTI 3.0.1 Release Notes

This document provides the following release-specific information for MÄK RTI 3.0.1:

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).)

Document ID: RTI-3.0.1-2-060728

Supported Systems

Table 1 lists the platforms supported by the MÄK RTI 3.0.1. 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

Platform	Compiler
SGI IRIX6.5	MipsPro7.4.2m C++ compiler
Sun Sparcstation with Solaris 8	Sun Workshop 6 with C++ 5.
Red Hat Enterprise Linux Workstation 3	GNU C++ (GCC) 3.2.3; glibc 2.3.2
Red Hat Enterprise Linux Workstation 4	GNU C++ (GCC) 3.4.4; glibc 2.3.4
Red Hat Fedora Core 3	GNU C++ (GCC) 3.4.4; glibc 2.3.6
PC with Windows 2000/XP	Microsoft Visual C++ 6.0, 7.1, and 8.0

MÄK RTI 3.0.1 was built with VR-Link 3.9.6.

Qt Release Compatibility

The MÄK RTIspy GUI uses Trolltech's Qt cross-platform GUI toolkit. (The Qt run-time libraries are distributed with the MÄK RTI). If a federate also uses Qt, it must be sure to use the same version of Qt as the MÄK RTIspy GUI, in order for the RTIspy GUI to function properly for that federate. Federates built with other versions of Qt should disable the RTIspy GUI to avoid these incompatibility problems.

The default RTIspy LRC Console GUI Plug-in is built with Qt 3.3.5. For federates such as recent versions of the MÄK Logger, MÄK Stealth, and VR-Forces 3.8, which require Qt 3.3.1, an alternate build of the RTIspy GUI Plug-in is included with MÄK RTI 3.0.1. The Qt 3.3.1 versions of the plug-in for Windows are:

- *LRCconsole-qt3.3.1.dll*
- *LRCconsoled-qt3.3.1.dll*
- *LRCconsole1516-qt3.3.1.dll*
- *LRCconsole1516d-qt3.3.1.dll*.

The Qt 3.3.1 versions of the plug-in for UNIX are:

- *LRCconsole-qt3.3.1.so*
- *LRCconsole1516-qt3.3.1.so*.

To select the Qt 3.3.1 version of the RTIspy LRC Console GUI instead of the Qt 3.3.5 version, disable the **RTI_enableLrcGui** RID file parameter and use the **RTI-addPluginName** parameter. For example, to select the debug build of the Qt 3.3.1 plug-in for HLA 1.3 set the following RID parameters:

```
(setqb RTI_enableLrcGUI 0)
(RTI-addPluginName "LRCconsole-qt3.3.1.dll")
```

The Qt 3.3.1 libraries are not included in the Microsoft Visual C++ 8.0 version of the RTI.

Patching the Microsoft Visual C++ 6.0 Compiler

If you use Microsoft Visual C++ version 6.0, you must install a patch to a standard header file, in order to successfully build applications for HLA 1.3 or 1516. The patch comes from Dinkumware, Ltd (the company that developed the Standard C++ Library header files for Microsoft), but for convenience, we are distributing it with MÄK RTI releases.

The patch fixes a bug that would cause applications to crash if they pass std::sets or std::maps from an application to a DLL or vice-versa. For details about the patch go to http://www.dinkumware.com/vc_fixes.html.

To install the patch:

1. In the top level MÄK RTI directory, is a file called *xtree-msvc++patch*. Rename this file *XTREE*.
2. Make a back-up version of the original *XTREE*.
3. In the Visual Studio *include* directory (for example, *C:\Program Files\Microsoft Visual Studio\VC98\Include*), replace the original version of *XTREE* with the patched version.

Compiler Compatibility on Windows

MÄK provides versions of product releases that have been compiled with Microsoft Visual C++ 6.0, 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.

FLEXlm Support

The Windows VC++ 8.0 version of the MÄK RTI requires FLEXlm 10.8. All other versions use FLEXlm 9.2. If you use the VC 8.0 version, replace your current license server files with those in the *flexlm10.8* directory. If this is a new installation, simply follow the directions for installing a license server in *MÄK RTI Reference Manual*. License files are forward compatible. You do not need to update your license file to use FLEXlm 10.8.

Backwards Compatibility

The MÄK RTI 3.0.1 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 3.0.1 and some use a previous version of the MÄK RTI.) However, the MÄK RTI 3.0.1 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 the DMSO RTI

With the exception of several additional functions (such as, DtReadFileDescriptor(), described in MÄK RTI documentation), the MÄK RTI 3.0.1 for HLA 1.3 has the exact same API as the DMSO RTI 1.3 NG. Unless you are using the RtReadFileDescriptor function, an application built against the DMSO RTI NG does not need to be recompiled in order to be used with the MÄK RTI 3.0.1, or vice versa (except that the Linux version is not compatible with the DMSO RTI).

MÄK RTI 3.0.1 uses the same FED file format as the DMSO RTI, but a different RID file format. (RID file formats depend on the RTI-implementation.)

New Features and Updates

MAK RTI 3.0.1 has no new features. It is a maintenance release.

Documentation Updates

Note the following corrections to *MÄK RTI Reference Manual*:

- In section C.2.2, under the Windows heading, it should read MSVC++ 7.1, not MSVC++ 7.
- In section C.3.2, under the Windows heading, the names of the project files should be *simpleDDMGUI.sln* and *simpleDDMGUI.dsw*.

Bug Fixes

The following bugs have been fixed in this release:

- Significant performance improvements have been made to the distributed forwarder.
- In MÄK RTI 3.0 and 2.4.2 an incompatability existed between versions of static libraries linked simultaneously by the RTI and VR-Link. This lead to a crash on application exit if the MÄK RTI was loaded dynamically at runtime (using the LoadLibrary and FreeLibrary commands) under Windows. This problem has been resolved.
- A problem with the Java Bindings Handle Set Iterators has been resolved.
- A problem with the Java Bindings Announce Synchronization Point callback has been resolved.
- On some UNIX platforms, multicast discovery failed and consumed excessive CPU time during multicast filtering.
- The RTIspy LRC GUI now works with Qt applications that run their QApplication in a secondary thread.
- When **RTI_bigMessage** is enabled, an interaction sent with time will be appropriately processed by receiving federates. Previously, these messages would be erroneously dropped by the RTI.
- Asynchronous callbacks are not implemented for 1.3, and cannot be enabled via the RID file or otherwise. Previously a 1.3 federate could enable asynchronous callbacks resulting in a federate that would throw an exception in createFederationExecution.
- Asynchronous IO had a problem on some platforms, which, depending on a race condition, could allow the asynchronous processing thread to exit, resulting in a federate that would not successfully be able to joinFederationExecution.

Known Problems

This release has the following known problems:

- 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.
- If you are using the Linux version of the MÄK 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 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.
- The asynchronous callback feature is not yet available for HLA 1.3 federates. It is only available for 1516 federates. Enabling this option will have no effect on an HLA 1.3 federate.

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.

Interoperability Between HLA 1.3 and IEEE 1516 Federates

In general, the MÄK RTI, and various other MÄK tools support run-time interoperability between HLA 1.3 federates and IEEE 1516 federates. Federates that are using Federation Management, Declaration Management, Object Management, Time Management, and Data Distribution Management services should be able to interoperate across the 1.3-1516 boundary. For example, you can typically run the HLA 1.3 version of the MÄK Stealth and the IEEE 1516 version of VR-Forces together in the same federation execution, provided you are using the MÄK RTI, or another RTI that supports 1.3-1516 interoperability.

However, there note the following restrictions and requirements:

- MOM is not currently supported across the 1.3-1516 boundary, and must be disabled. (The 1.3 and 1516 Specifications dictate completely different data representations for MOM attributes and parameters.)
- Ownership management is also problematic, and should be avoided. For object names, the 1516 RTI converts all strings from the wide char representation to a narrow character representation. So, the names supplied by the 1516 federates must allow this conversion (that is, only support the ASCII character set). For user supplied tags, the 1516 federates are restricted to the 1.3 string format. The 1.3 RTI cannot represent the complex data types allowed by 1516.
- Although federates might differ in the version of the HLA they use (1.3 versus 1516), they must agree on a common FOM representation to use. All federates must use either an HLA-1.3-style FED file, or an IEEE-1516-style XML file. (VR-Link and the MÄK RTI allow you to use HLA-1.3-style FED files with IEEE-1516-based federates, and vice versa).

The main reason that consistency in FOM format is necessary is that HLA 1.3 and IEEE 1516 use different names for the "Root" classes of the Object and Interaction class hierarchies. A 1.3-style FED file requires a Root class called *ObjectRoot*, whereas a 1516-style XML files requires a Root class called *HLAObjectRoot*. This is a problem because if a federate uses an HLA-1.3-based FED file, it might subscribe to a class called *ObjectRoot.Vehicle*. On the other hand, a federate that is using an IEEE-1516-style XML file might publish a class that is meant to be the same class, but that is actually named *HLAObjectRoot.Vehicle*. Neither the RTI nor the federates will realize that these classes were intended to be the same, and the subscribing federate will fail to discover any objects that the publishing federate registers.

In future releases, we intend to rectify this situation, and allow run-time interoperability between federates that have loaded a 1.3-style FED file and federates that have loaded a 1516-style XML file that otherwise includes the same set of classes, attributes and parameters.

Recording HLA 1.3 Logger Files and Playing Back in HLA 1516 Federations

The issues described in the previous section have a direct effect on the MÄK Data Logger. The Logger uses the same file format for HLA 1.3 and HLA 1516 Logger files. You can record a Logger file from an HLA 1.3 federation, and play it back into an HLA 1516 federation (and vice versa). However, you must be consistent about the FOM representation used during record and playback. For example, if the Logger was using an HLA-1.3-style FED file during recording, you must use an HLA-1.3-style FED file during playback, even if you are using the IEEE 1516 version of the Logger executable, and an IEEE 1516 RTI.