MÄK RTI Release 1.3 Release Notes

The MÄK RTI (Run-Time Infrastructure) 1.3 is an implementation of a subset of the HLA Interface Specification, version 1.3. It implements the same API (exact same header files) as the DMSO RTI, version 1.3v4.

This document covers the following:

Document ID: RTI-1.3-2-981013

System Requirements

Table 1 lists the platforms supported by the MÄK RTI 1.3. Application code must be built with the indicated compilers in order to interface to the RTI libraries.

Table 1: Platforms supported

Platform	Compiler
SGI IRIX6.2 or later (both n32 and o32 ABIs)	MipsPro7.1 C++ compiler (available from SGI)
Sun Sparcstation with SunOS4.1.3 or later	GNU C++ (g++) version 2.7.2.3 and G++ libraries version 2.7.2. (Available via anonymous FTP from prep.ai.mit.edu/pub/gnu.)
Sun Sparcstation with Solaris2.5 or later	SPARCompiler C++ version 4.1 (available from Sun)
Dec Alpha with OSF/1 V4.0	DEC C++ for OSF/1 version 5.5 (available from DEC)
PC with Red Hat Linux 5.1	GNU C++ (g++) distributed with Red Hat distribution. The command: `g++ -v` returns: `gcc version egcs-2.90.27 980315 (ecgs-1.02 release)`
PC with Windows NT or Windows 95	Microsoft Visual C++ 5.0 or later. Note: MÄK RTI 1.0.3 used MSVC++ 4.2.

Compatibility with the DMSO RTI

With the exception of the function RtReadFileDescriptor() (described in MÄK RTI documentation), the MÄK RTI has the exact same API as the DMSO RTI version 1.3v4. Unless you are using RtReadFileDescriptor(), an application built against the MÄK RTI does not need to be recompiled to be used with the DMSO RTI, or vice versa.

The MÄK RTI 1.3 uses the same FED file format as DMSO RTI 1.3v4, but a different RID file format. (RIDs are RTI implementation dependent.)

Network Compatibility

The MÄK RTI is not network compatible with the DMSO RTI. 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 to another between runs, but during each run, all participants must agree on which RTI to use, much as they must agree on which FOM to use.

Backward Compatibility

The MÄK RTI 1.3 is not network compatible with the MÄK RTI 1.0.3. All federates in a federation execution must use the same RTI version.

Documentation

Because the MÄK RTI implements the same interface specification as the DMSO RTI, the documentation provided with the DMSO RTI applies to the MÄK RTI as well, apart from the few differences we describe in this release note and the MÄK RTI Reference Manual. The DMSO RTI documentation is included with the MÄK RTI in the *doc* directory. It is also available on DMSO's HLA web site: http://hla.dmso.mil. Choose HLA Software Distribution Center, then click on the Products button.

Services Implemented

MÄK RTI 1.3 implements a subset of the RTI services, specifically, the functionality that is most likely to be needed by the real-time platform-level simulation community.

The following federate-initiated services are implemented. (Stubs exist for the rest):

- CreateFederateExecution
- DestroyFederateExecution
- JoinFederateExecution
- ResignFederateExecution
- PublishInteractionClass
- PublishObjectClass
- SubscribeInteractionClass
- SubscribeObjectClassAttributes
- UnpublishInteractionClass
- UnpublishObjectClass
- UnsubscribeInteractionClass
- UnsubscribeObjectClass
- DeleteObjectInstance
- LocalDeleteObjectInstance
- RegisterObjectInstance
- RequestClassAttributeValueUpdate
- RequestObjectAttributeValueUpdate
- SendInteraction
- UpdateAttributeValues
- DisableAttributeRelevanceAdvisorySwitch
- DisableClassRelevanceAdvisorySwitch

- DisableInteractionRelevanceAdvisorySwitch
- EnableAttributeRelevanceAdvisorySwitch
- EnableClassRelevanceAdvisorySwitch
- EnableInteractionRelevanceAdvisorySwitch
- GetAttributeHandle
- GetAttributeName
- GetInteractionClassHandle
- GetInteractionClassName
- GetObjectClass
- GetObjectClassHandle
- GetObjectClassName
- GetObjectInstanceHandle
- GetObjectInstanceName
- GetParameterHandle
- GetParameterName
- Tick

The following RTI-initiated services are called by the MÄK RTI:

- StartRegistrationForObjectClass
- StopRegistrationForObjectClass
- TurnInteractionsOn
- TurnInteractionsOff
- DiscoverObjectInstance
- ProvideAttributeValueUpdate
- ReceiveInteraction
- ReflectAttributeValues
- RemoveObjectInstance
- turnUpdatesOffForObjectInstance
- turnUpdatesOnForObjectInstance

In addition, MÄK RTI 1.3 only implements the best-effort communications mechanism, and does not publish any MOM data.

The RID File

The MÄK RTI 1.3 has a new RID configuration parameter. **RTI_notifyLevel** indicates the categories of notification messages to print. Values are: 0 - Fatal 1 - Warn 2 - Notify 3 - Verbose

Reading The FED File

In the RTI 1.3 API, the FED file name is passed to the createFederationExecution() service along with the federation execution name. When joinFederationExecution() is called with a federation execution name, the FED file that was associated with the execution name in an earlier create call is read. In the DMSO RTI, the *rtiexec* manages the list of associations between federation execution names and FED file names.

Since the MÄK RTI does not require a central *rtiexec* server, all federates that use the MÄK RTI need an alternate way to ensure federation execution name/FED file associations, regardless of whether or not you use *rtiexec*.

All federates should call createFederationExecution() before calling joinFederationExecution() (this tells us what FED file to use), even if you know that the federation execution has already been created by another federate. This method is recommended for use with the DMSO RTI as well, but not technically required.

Federation Time

The RTI 1.3 API allows users to choose a method of implementing federation time for time management services by subclassing the abstract base class *RTI::FedTime*, and telling the RTI to use the subclass by providing implementations for the functions RTI::FedTimeFactory::decode() and RTI::FedTimeFactory::makeZero(). (These functions are declared, but not defined by the RTI.)

Although the MÄK RTI does not implement time-management services, and so has no need for a class derived from *RTI::FedTime*, we require that definitions for these functions be provided outside of the *libRTI.so* library for consistency with the DMSO RTI. (If we simply included definitions of these functions in our *libRTI.so*, then applications that were built against the DMSO RTI, which therefore contain definitions for these functions, would find that these functions are multiply defined when they ran with the MÄK RTI.)

If your application was built using VR-Link, it automatically contains definitions for the required *FedTime* functions. However, if it was not, you can link against the *libfedtime.so* shared library that is provided with the MÄK RTI. It defines, and instructs the RTI to use a class called *DtVrlFedTime*, which represents time as a double. (Similarly, the DMSO RTI includes a *libfedtime.so* shared library that contains a sample implementation of federation time.)

Known Problems

When you configure the Flexlm license manager on a PC, make sure that when you enter a pathname, the drive letter is uppercase.