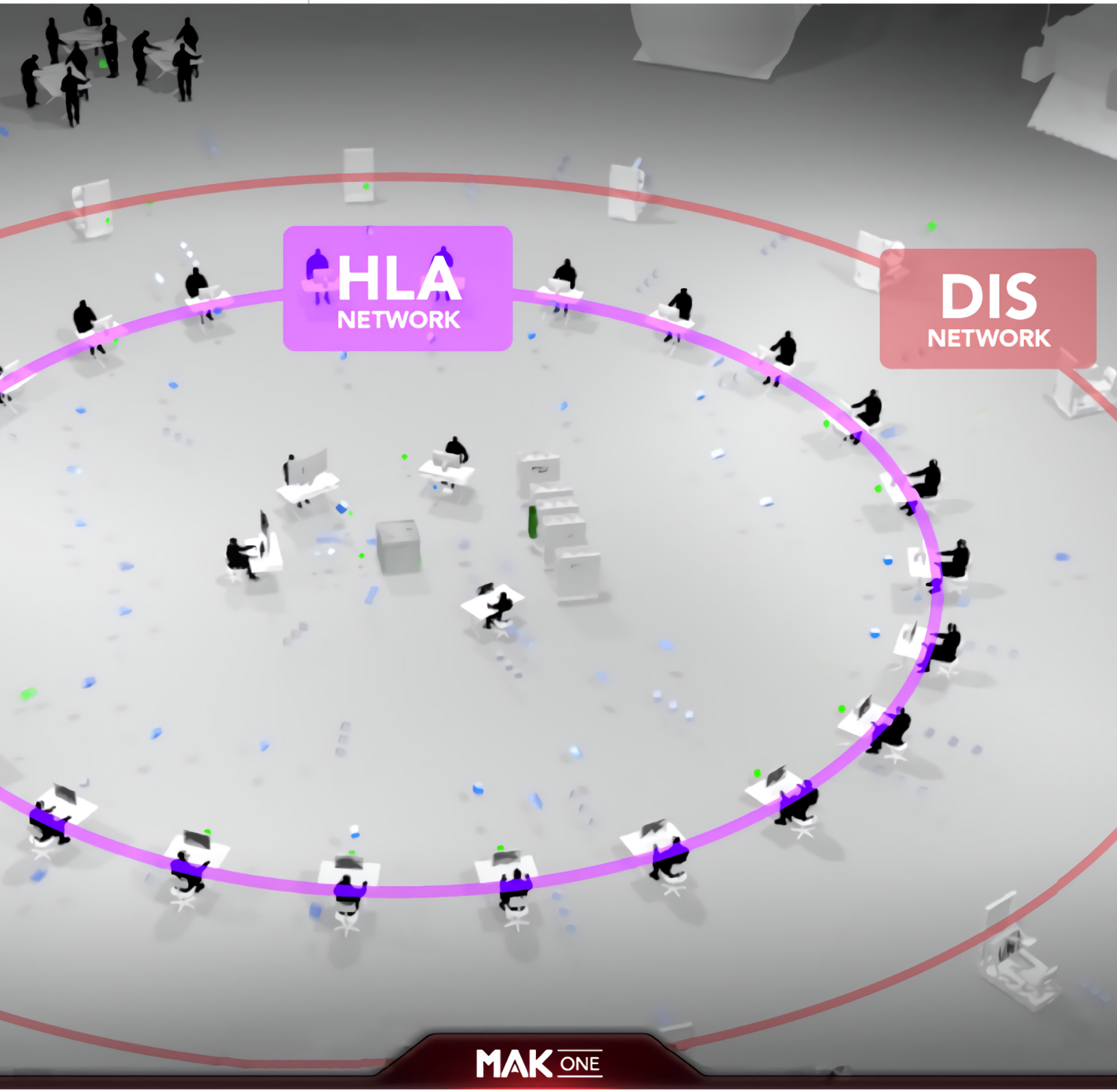
MAK TECHNOLOGIES

MAK RTI Capabilities



Version 5.0

ST Engineering
North America

MAK RTI Capabilities

The MAK High Performance RTI is a proven solution that enables HLA federations to rapidly and efficiently communicate. This document describes its capabilities.

An RTI, or Run-Time Infrastructure, is a required component of any HLA (High Level Architecture) federation. According to the rules of HLA, various simulation applications, known as federates, use the RTI to exchange simulation data.

The HLA Interface Specification, part of the HLA Standard, dictates the set of functionality that federates can expect of an RTI, and provides API mappings for various programming languages, including C++ and Java. An RTI consists of a set of libraries that implement this standard API, along with supporting tools required by the implementation (for example, a central server application known as an RTI Executive or rtiexec). Federates communicate by making calls to RTI functions through the standard API.

RTI functionality falls into eight categories: Federation Management, Declaration Management, Object Management, Ownership Management, Time Management, Data Distribution Management, Support Services, and the Management Object Model (MOM).

A Long History of Excellence

The MAK RTI was first released in June 1998, making it the first commercially developed RTI. It has been in continuous use ever since, giving it a longer history than any other RTI available.

The MAK RTI was originally developed in response to concerns expressed by the real-time simulation community about the performance of early RTIs. We initially focused on meeting the needs of that segment of the HLA community, providing an RTI that kept latency, bandwidth, and CPU usage to an absolute minimum. At the time, the MAK RTI implemented only those services that were typically required by real-time federations. But over time, DDM, Time Management, and MOM were added, and by early 2002, the MAK RTI was a full implementation of the entire HLA Interface Specification, Version 1.3.

The MAK RTI Supports HLA 1.3, IEEE 1516-2000, IEEE 1516-2010 (HLA Evolved), and IEEE 1516-2025 (HLA 4)

The MAK RTI has been verified for the HLA 1.3 and IEEE 1516-2000 versions of HLA. Full verification for HLA 1.3 and IEEE 1516-2000 required passing more than 3000 individual tests, each of which involved up to five federates making a series of RTI service calls in a particular order. The tests were designed to ensure that an RTI complies with every statement in the HLA Interface Specification.

The U.S. Department of Defense (DoD) has ceased funding the verification of RTIs. Prior to this cessation, the MAK RTI had passed all available verification tests for IEEE 1516-2010. While there is no current schedule for the U.S. DoD to verify IEEE 1516-2025, the MAK RTI will be submitted once testing is scheduled.

The MAK RTI is Link Compatible with Other RTIs

As long as different RTIs implement the exact header files as dictated by the HLA Interface Specification, it is often as easy as just swapping shared libraries (DLLs or DSOs). No recompiling or relinking is necessary, meaning that end-users (non-developers) can choose their desired RTI implementation at federation execution time. A tool vendor can create a single version of their executable that works with the MAK RTI and with any other RTI that is link-compatible. Occasionally, some upfront tuning is necessary to ensure that an executable will work well with several RTIs, due to differences in recommended tick rates, or possible inadvertent dependencies on implementation characteristics of one of the RTIs. Once an executable has been tested with both, however, choosing which implementation to run with is just a matter of pointing to the appropriate RTI libraries.

Specifically, we have tested the dynamic-link compatibility of the HLA 1.3 version of the MAK RTI with RTI NG Pro, the MATREX RTI, and with the pRTI 1.3 from Pitch. Applications that have been built against these RTIs can typically switch to the HLA 1.3 version of the MAK RTI without recompiling or re-linking.

For IEEE 1516, things are not as simple, because the original IEEE 1516 C++ API did not support Dynamic-Link compatibility among RTIs. For this reason, SISO developed the Dynamic-Link Compatible (DLC) C++ API for IEEE 1516. MAK was the first to implement the SISO Standard DLC API in our RTI, which enables us to be Dynamic-Link Compatible with other RTIs that implement this standard.

We have also tested compatibility of the HLA 1516-2010 (HLA Evolved) versions of MAK ONE applications with pRTI HLA Evolved. We will test the HLA 1516-2025 (HLA 4) versions of our applications with the pRTI HLA 4 when it becomes available.

The HLA Evolved and HLA 4 Versions Can Interoperate

We have attempted to retain as much wire compatibility as possible between the HLA evolved and HLA versions. In certain cases, changes in the semantics of the services between HLA versions made this impossible. However, for most services (including DDM, ownership transfer, and time management), federates built to the HLA Evolved API should be able to communicate with federates built to HLA 4. HLA Evolved or HLA 4 FOMs can be used with federates built against the HLA4 or HLA Evolved API, allowing you the flexibility to upgrade simulations to the latest IEEE 1516 version one at a time, rather than upgrading them all at once.

The MAK RTI Supports Any FOM

Like any RTI, the MAK RTI reads your FED file (or FDD file for IEEE 1516) to obtain the information it needs about your federation's FOM: names of classes, attributes and parameters; transport and order types; and region information necessary for DDM. There is no hard-coded FOM-specific data in the MAK RTI.

The MAK RTI Supports Non-Realtime Simulations

While the MAK RTI's low latencies and efficient use of network and computational resources make it particularly well-suited to real-time simulations, the MAK RTI is a complete implementation of the HLA Interface Specification. It includes all of the Time-Management services necessary for use in non-real-time environments, including time-stepped or event-based simulations. Further, our focus on performance has carried through to our implementation of time-management and other RTI services. After all, it's not just real-time simulations that care about the performance and scalability of an RTI.

The MAK RTI Supports Multiple Platforms

The full MAK RTI distribution—including fully-compliant support for all services, the rtiexec, the RTIspy API, RTI Forwarders, Federation Shadow, Federate Protocol Server, and the RTI Assistant—is supported on both Windows and Linux (several different compiler versions on each).

We are also willing to port the MAK RTI to just about any platform you want us to support. We have done demand-based porting of various releases to DEC Alpha, PC-Solaris, HP, AIX, PowerUX, VxWorks, and OpenVMS. Let us know what you need.

Verified Fully Compliant by US DoD

The MAK RTI has passed all RTI Verification tests administered by the US DoD for HLA 1.3, HLA 1516 (IEEE 1516.1-2000), and HLA Evolved (IEEE 1516.1-2010). (No verification tests have been scheduled for HLA 4 (IEEE 1516-2025).)

HLA 1.3

The MAK RTI was officially verified by the U.S. DoD as Fully Compliant with the HLA 1.3 version of the HLA Standard way back in November 2002. HLA 1.3 Verification required passing over 1,500 rigorous tests administered by the RTI Verification Team on behalf of DMSO (Defense Modeling and Simulation Office).

HLA 1516

The MAK RTI was officially verified by the US DoD as Fully Compliant with the HLA 1516 version of the HLA Standard (IEEE 1516.1-2000) in February 2006. It was the first RTI verified compliant with the SISO Standard Dynamic Link Compatible API for HLA 1516. HLA 1516 verification required passing 1,856 tests administered by the RTI Verification Team on behalf of DMSO.

HLA Evolved and HLA 4

Despite occasional rumors to the contrary, no RTI vendor has an actual verification certificate for HLA Evolved (IEEE 1516.1-2010). The US DoD RTI Verification activity was suspended due to budget constraints in January 2015, before declaring any RTIs officially verified. However, at the time that the RTI verification activity was suspended, the MAK RTI with HLA Evolved had successfully passed 100% of the available tests – 1,981 in all – with just a handful of additional tests yet to be administered, because these last tests were still under development by the RTI Verification Team. While it is unfortunate that the US DoD suspended RTI Verification within days of being able to declare the MAK RTI officially verified, MAK customers can proceed with confidence that the MAK RTI is Fully Compliant with the HLA Evolved Standard.

The MAK RTI is Robust and Versatile

The MAK RTI is fully implemented in C++, allowing it to be used directly by C++ federates without going through any kind of wrapper. This choice not only allowed us to make the MAK RTI as efficient as possible, but also allowed us to create the RTIspy® plug-in API, which allows C++ developers to customize or extend the MAK RTI. The MAK RTI also includes a Java "cap" that allows Java federates to use the MAK RTI.

The MAK RTI is Fault Tolerant

First, we can automatically recover from temporarily broken TCP connections, without the federate even knowing that we have disconnected and reconnected. Second, if a federate crashes or permanently loses its connection, the RTI and the rest of the federation can continue gracefully. The lost federate is removed from any Time Management or Synchronization Point calculations so that the other federates are not held up. Orphaned objects can be cleaned up, deleted, or tracked for later ownership transfer purposes.

The MAK RTI Supports Multithreading

A configuration parameter dictates whether the job of reading and writing packets to the network should be offloaded onto a separate asynchronous thread. Doing so has several advantages: First, it means that you are less likely to drop packets due to not ticking the RTI often enough. The asynchronous thread is always reading and queuing packets as fast as possible, whereas without it, the RTI must wait for the federate to give it the necessary processor time. The other advantage is that when there is a lot of data to be read from the network, your federate is not hung up waiting for the read operation to finish. You can move on to other simulation tasks while the network-reading thread listens for incoming data. Obviously, the advantages of running in multithreaded mode are more pronounced on a multi-core or multi-processor machine.

The MAK RTI Can Deliver Callbacks Asynchronously

If you enable asynchronous callbacks in the configuration file, FederateAmbassador services will be called from an asynchronous thread, without the federate having to call tick(). Of course, this requires that your federate be written in a thread-safe manner, so that RTI callbacks and the main federate thread do not access the same piece of data concurrently.

The RTIspy® Lets You Extend the MAK RTI

RTIspy® is a plug-in API that allows C++ developers to customize or extend the MAK RTI. The RTIspy plug-in API is a C++ interface you can use to alter, extend, or query the RTI's functionality programmatically from a dynamic library you create. So, if you want to use special radio communications instead of our standard TCP/IP infrastructure, you can change it! Or, if you simply want to collect detailed information about what is going on in your federation, that is possible, too.

The RTIspy plug-in API is a feature that is truly unique to the MAK RTI. It allows you to build plug-ins to alter, extend, and query any aspect of the MAK RTI's functionality.

Through the API, you can change the RTI's "wire format" by implementing new network messages, perform encryption or compression on network messages, register callbacks to be executed whenever certain services are invoked, tune performance based on program-specific requirements, and override default implementations of key services. The RTI Assistant API lets you tailor a proven RTI to meet your program's needs.

In part to demonstrate just some of the power of the RTIspy plug-in API, we chose to implement the RTIspy Diagnostic GUI entirely as a plug-in, using the RTIspy plug-in API.

The MAK RTI Is Fast

To us, performance is not an afterthought; performance is part of the design. Configuration file parameters allow you to disable services that you do not need. If you do not have Time Management or DDM enabled, for example, we automatically use a more compact message format. Obviously, though, if you do use the additional services, extra bookkeeping information needs to be exchanged. In general, latencies are not affected by additional services, since our communications strategies have not significantly changed since the original subset RTI was released. We have also taken great pains to make sure that you do not pay a penalty for unused services in terms of computation time. In general, we check whether the advanced services are enabled, and revert to our original, compact algorithms if they are not. For example, if MOM is not enabled, we avoid collecting the information that would be necessary to send MOM updates.

For further information about the performance of the MAK RTI, please ask for a copy of a paper we have published, entitled The MAK High-Performance RTI: Performance By Design.

The MAK RTI Is Easy to Use

Historically, most RTIs have required that you hand-edit a configuration file, just to do something as simple as moving the rtiexec from one machine to another, or to make sure that two different exercises on your network didn't conflict. The MAK RTI makes this simple. It is no longer necessary to edit configuration files to configure the most commonly used options: You can switch between running with an rtiexec and Lightweight Mode, choose from a list of available rtiexecs running on your network, and even launch and configure a new rtiexec, all from a GUI that comes up automatically the first time you run a federate. In addition to choosing among pre-configured RTI Connections, you can add new Connections by entering a port number and multicast address. You can also "force full compliance" from the GUI to ensure that all of the RTI services are enabled. If you do not want to be prompted to select an RTI Connection next time you run a federate, check the "Always attempt to use this connection" box to make a particular connection the default.

For convenience, the MAK RTI provides an icon on the Windows Tray. From here, you can change the default RTI Connection, force resigns of local or remote federates, launch the RTI Assistant, run latency tests, enable logging, and view the RTI notification history. It also allows you to bring up the Federations View and Network Components View, which will show you the current state or network configuration of any federation on your network.

RTI Assistant

The RTI Assistant visually displays information gleaned directly from each LRC and the rtiexec – a much more intimate view than can be provided by MOM-based tools. You can easily see the world from the perspective of the LRC – scanning the list of known objects, the interactions it has sent and received, and the current state of all FOM subscriptions and publications. For example, the RTI Assistant makes it easy to determine whether a federate is not seeing an object because it is not subscribed to an appropriate class, because the object has never been registered, or because the federate failed to properly handle the object discovery callback.

The tool also helps to solve tricky timing problems by keeping a log of all federate-invoked and RTI-invoked services, and by displaying Time-Management parameters like current logical time, requested time, and LBTS for each federate. The network-monitoring panel allows you to examine bandwidth usage by object, both local and remote. The display also provides information on what percentage of CPU time is being used by the RTI and by FederateAmbassador callbacks. The service instrumentation panel shows exactly how many microseconds each RTI call takes. A graphical representation of your federation's topology helps you understand how your federates are distributed among various LANs, and how the RTI uses RTI Forwarders to efficiently route data across a WAN.

The RTI Assistant Manages RTI Connections

The HLA specification defines the information that a federate must provide to an RTI to create or join a federation. This information might be built into your application or you might have to specify it when you start the application. For example, applications based on VR-Link have to specify, at a minimum, a federation name and a federate name.

The MAK RTI uses the concept of an RTI connection. An RTI connection specifies a port and multicast address. All of the federates in a federation must use the same RTI connection. If a federation is using the rtiexec, an RTI connection includes the address of the RTI Forwarder being used by the rtiexec.

When you install the MAK RTI, it is configured with a default lightweight connection and a default rtiexec connection that uses the computer's name. You can create additional connection configurations to meet the needs of your federations.

By default, when you start a federate, the MAK RTI prompts you to select an RTI connection configuration. If you always want to use the same connection, you can configure the MAK RTI to always try to use that connection without prompting you to select it.

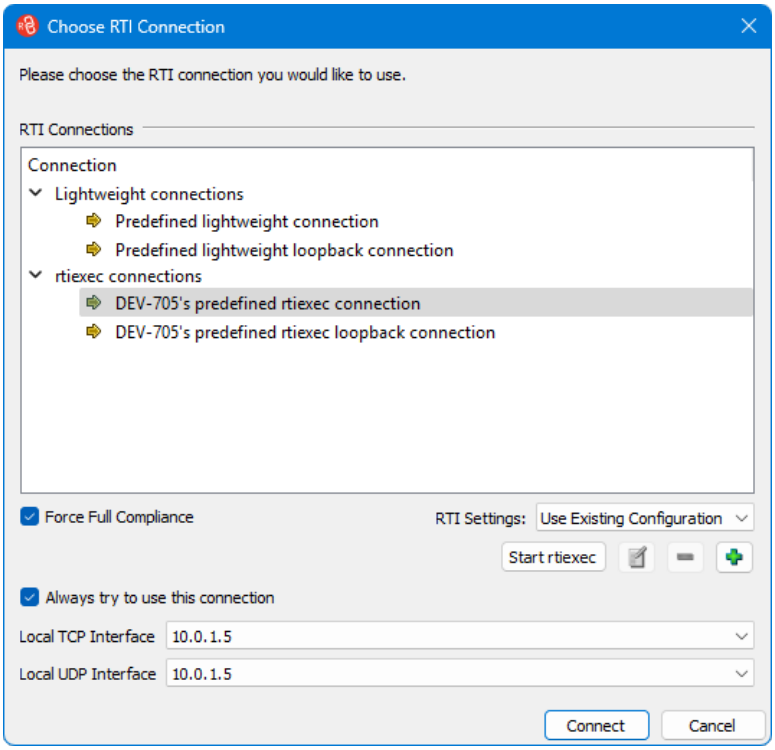


Figure 1. Choose RTI Connection dialog box

The RTI Assistant Federations View

The Federations View displays information about federations in three panels:

- Network Map. The Network Map displays icons for the RTI Forwarder, the rtiexec, and the federates in the selected connection. The links between components represent the reliable connections (TCP) between them.
- Selected Node. The Selected Node panel displays details about the node that is selected in the Network Map.
- Federations. The Federations panel lists the federations that are active on the selected connection and the federates in each federation.

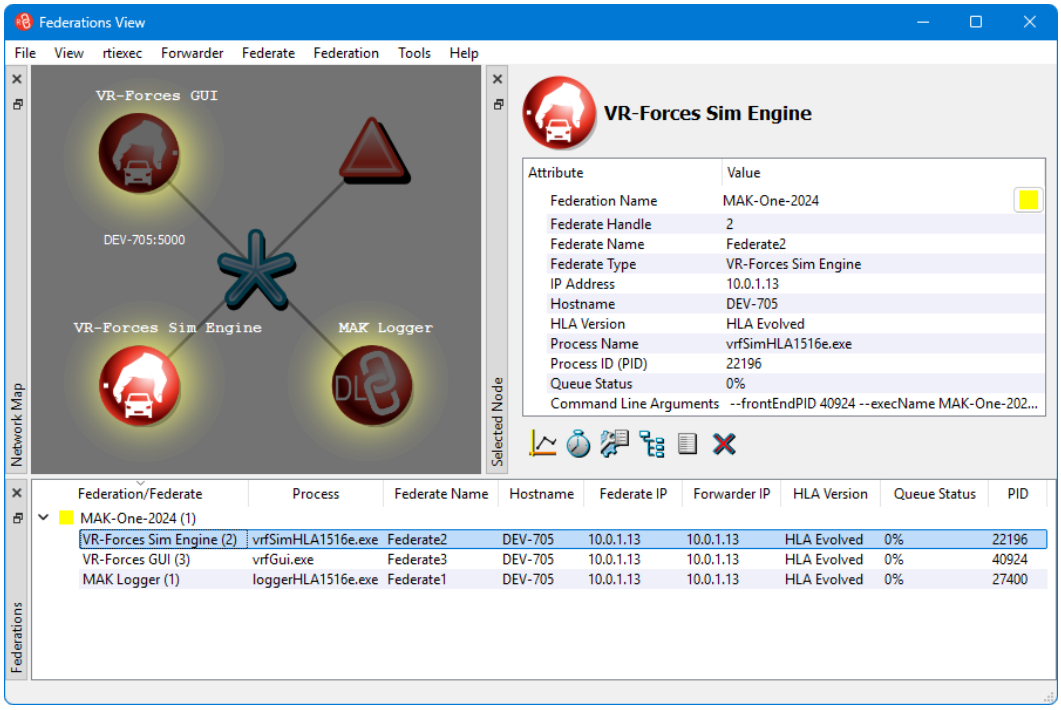


Figure 2. Federations View

In addition to providing information, the Federations View lets you resign federates, destroy federations, and shut down local RTI components (LRCs).

The Network Monitoring Tool

The RTIspy Network Monitoring Tool is a window into the "black box" of the RTI. It records each message sent and each API call made. You can use this information to help weigh the costs and benefits of different RTI features. It can also help you locate the bottlenecks in your simulations, whether they are due to the underlying network, processing in the RTI, processing in Federate Ambassador call-backs, or the simulation execution itself.

The RTIspy Network Monitoring Tool exposes the internal processing of the RTI in the following ways:

- The Callbacks page can show you which RTI Ambassador or Federate Ambassador calls might be affecting execution.
- You can observe how much data is sent across the network to represent the simulation's objects and interactions. This can help identify possible problems and optimizations.
- You can log the statistics to a file, allowing deeper analysis after a simulation run.

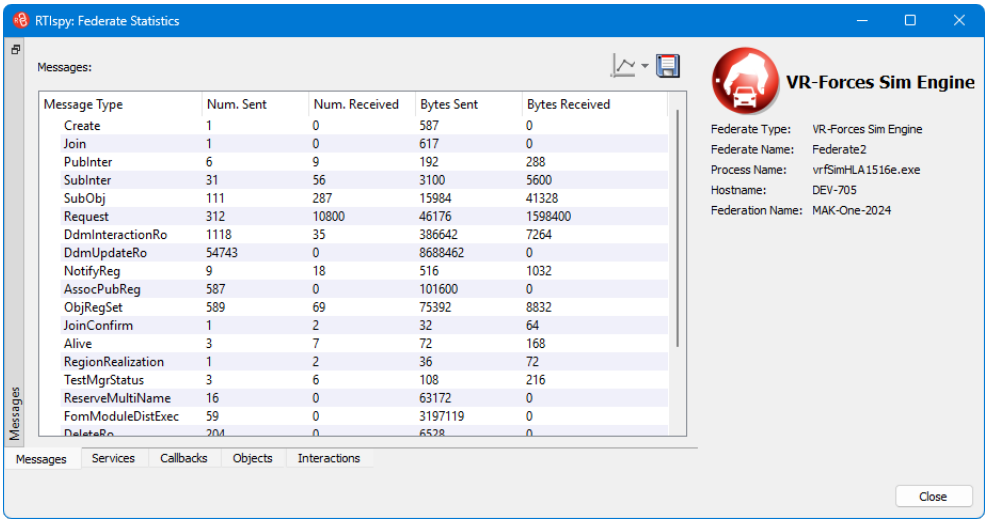


Figure 3. RTIspy Federate Statistics, Messages

Monitoring Network Latency

The MAK RTI can run latency tests among federates. You must have at least two federates in the federation to run latency tests. When you start latency testing for a federate, the RTI automatically runs a latency test from this initiating federate to all other federates in the federation. The results are displayed in the Latency Test window. If you select a federate in the Latency Test window, information about the federate is displayed in the Details panel.

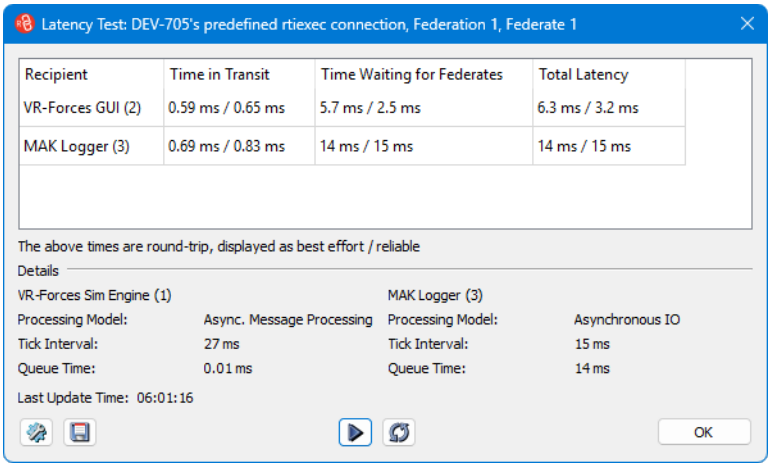


Figure 4. Latency Test window

The MAK RTI Includes Examples

The MAK RTI includes multiple examples of using the HLA API, from basic services to more advanced services and concepts.

- The rtisimple example provides a starting point for federates, demonstrating federation management and object management services.
- The simpleDdm and simpleTime examples build off of the rtisimple example but demonstrate more advanced families of services such as Data Distribution Management and Time Management respectively.
- The simpleFedPro example, which is an example of a Federate Protocol client, using the Federate Protocol introduced in IEEE 1516-2025 (HLA 4).
- The HLA Bounce example is an HLA federate that demonstrates subscription/unsubscription, ownership management, and DDM in the context of a whimsical graphical user interface.
- The MAK Shooter example is an HLA federate that demonstrates object publishing, subscribing, updating of attributes, and sending of interactions in the form of a simple arcade style shooter game.

The MAK RTI Includes an RTI Executive

There are many aspects of an RTI's functionality that require some form of centralized knowledge. For example, someone needs to insure that federate and object identifiers are unique, someone needs to decide when a federation is synchronized, and someone needs to keep track of which federate is the laggard in time-management calculations. There are basically two strategies that RTI developers can take in managing this centralized knowledge: Make one federate's Local RTI Component (LRC) responsible, or require a central server application commonly known as an RTI Executive or rtiexec. The MAK RTI chooses the rtiexec approach because it is simpler (no need to transfer central-knowledge to another federate when the designated federate resigns), and usually more robust (rtiexec crashes are usually less frequent than federate crashes). When using the set of services that require centralized knowledge, it is necessary to run the rtiexec before any federates try to create or join a federation.

It is important to realize that even when you are using the rtiexec, most of your data still goes "peer-to-peer" between the federates. Specifically, attribute updates and interactions are typically sent over UDP multicast, without the rtiexec even getting involved. Reliable traffic goes through an RTI Forwarder, which may or may not live in the rtiexec, depending on configuration.

Lightweight Mode Supports Rapid Development and Real-Time Federations

Although some RTI services do require centralized knowledge, we have found that a significant fraction of HLA federations do not rely on those services. For these federations, the MAK RTI provides a Lightweight Mode, where no rtiexec is necessary. In this connectionless, fully-distributed mode, all network messages are communicated directly between the federates' Local RTI Components (LRCs). A configuration setting indicates to a federate's LRC that it is running in Lightweight Mode, and should not attempt to connect to an rtiexec.

Lightweight Mode is well-suited for many real-time federations that do not use Time-Management, DDM, MOM, reliable transport (TCP), or synchronization points. It is particularly helpful during the development and debugging stage of a federation, when federates are unstable, and reinitializing the rtiexec between runs would be a hassle.

Obviously, the MAK RTI is not fully compliant when configured for Lightweight Mode, since not all services can be supported.

The MAK RTI Supports Operation Over the Internet or Other WANs

If you are using reliable transport, no special configuration is necessary to run an HLA federation over the internet or other WAN, as long as there is a network route from each federate to the RTI Forwarder. When using best-effort transport, however, the default configuration is not appropriate, since UDP multicast packets typically cannot be sent over a WAN. Instead, you can take advantage of Distributed Forwarding to run an RTI Forwarder on each side of the WAN. With this configuration, each forwarder can listen for local UDP multicast packets and send them via TCP across the WAN to one or more remote RTI Forwarders.

MAK RTI users can run as many or as few forwarders as they want at no extra cost.

In order to make the most efficient use of WAN bandwidth, you will want to use multiple, distributed copies of the MAK RTI's RTI Forwarder. When you run an RTI Forwarder on each LAN, you save bandwidth by minimizing the number of copies of each packet that needs to be sent over the WAN. Instead of sending a separate copy of each packet to each remote federate, only one copy of each packet traverses each WAN link, and the remote RTI Forwarder sends the message on to each recipient on its LAN. The RTI Forwarders support sender-side filtering, so that only interested federates' LRCs will receive each message, and messages are not sent over WAN links at all when no one on the other side needs the data. Data received locally by an RTI Forwarder via UDP multicast is automatically sent to remote RTI Forwarders (if needed) via TCP.

The MAK RTI supports several different topologies for using distributed forwarders. Figure 6 shows the distributed singleton forwarder network. Other topologies use forwarder groups to create single-portal forwarder networks and load balanced forwarder networks.

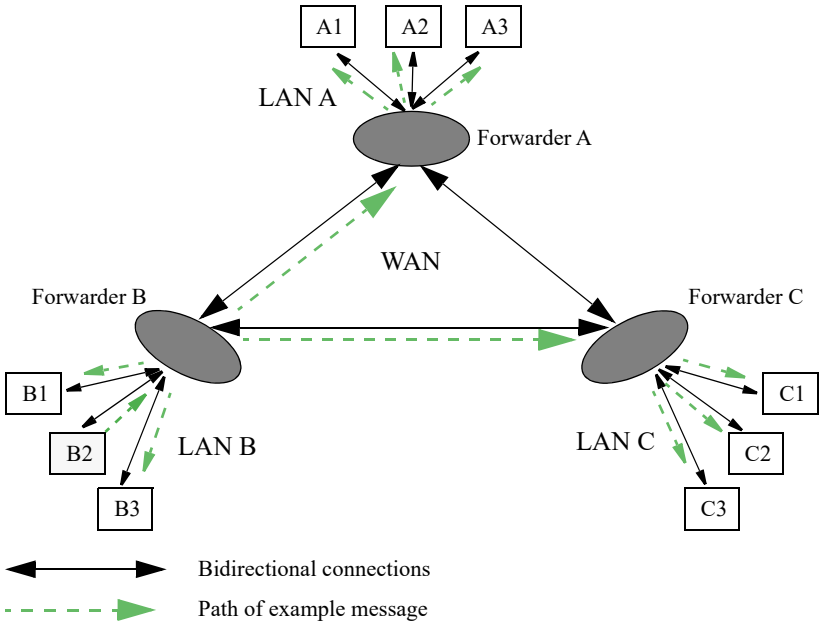


Figure 5. Distributed Singleton Forwarder Network on a WAN

The MAK RTI Can Run Behind a Firewall

We realize that many of today's HLA federations span wide-area networks, and often each LAN is protected by a firewall. Many of our customers use the MAK RTI in this environment. The MAK RTI's RTI Forwarder running on each LAN serves as a single point of entry for all RTI data coming to each LAN. This makes it very easy to configure your firewall to allow HLA traffic to enter, without needing to update it when federates move from machine to machine. If you use a Virtual Network scheme such as VPN, even this minimal firewall configuration becomes unnecessary.

The MAK RTI Unlicensed Mode Allows Limited Free Use

You can download the MAK RTI from https://www.mak.com/mak-one/support/help/bonus-material, and start using it immediately in unlicensed mode, without requiring a license key. In unlicensed mode, all RTI services are available, but the RTI only allows two federates to join each federation execution. Unlicensed mode is useful for developers who need to run and test their federate (along with one other federate) without consuming one of their lab's RTI licenses. It is useful for people who want to take their federate on the road for demonstrations, without requiring an RTI license on their laptop. It is useful for developers who sometimes work at home, and cannot justify purchasing an extra RTI license for remote development. And, of course, it is useful for evaluating and becoming familiar with the MAK RTI before purchasing.

When you want to upgrade to a full license, in order to run in “real” federations with more than two federates, all you need is a license key. No new RTI download is necessary.

The RTI Program Protection Plan Supports Large Programs

We recognize that large programs have special RTI needs. Through our RTI Program Protection Plan, we can put together a custom package of licenses, support, and consulting tailored for your program. In addition to a site license good for an unlimited number of federates, a typical plan might include a dedicated support engineer, on-demand porting to new compiler versions or operating systems, on-site consulting, or custom features.

The RTI Program Protection Plan allows program managers to lock in a price up front, so that they can budget for the future, without fear of unexpected costs creeping in later.

The MAK RTI - A de facto Standard

These days, it seems like almost everyone needs a way to implement HLA. We've sold many thousands of licenses of the MAK RTI around the world. Including site licenses and title licenses, well over ten thousand federates use the MAK RTI. Many HLA product companies distribute the MAK RTI with their HLA products, and many more specifically test their products against the MAK RTI before each release.

It is rapidly becoming the de facto standard in the industry for HLA 1.3 and IEEE 1516 federations, both large and small. Some key programs that have chosen the MAK RTI:

- US Air Force Joint Strike Fighter program
- NATO Mission Training Through Distributed Simulation exercise
- US Navy DDG 1000 program
- US Navy PEO IWS Simulation Network
- Turkish Air Defense Training Centre
- NASA Air Traffic Operations Laboratory (ATOL)
- Single European Sky Air Traffic Management simulation network
- European Union Ocean 2020 Program
- Canada's Department of National Defense Canadian Advanced Synthetic Environment (CASE)
- US Marine Corps Tactical Environment Network
- Netherlands' TACTIS program
- Australian Defence Simulation Office Joint Simulation Capability program

In Canada, the Department of National Defense (DND) chose the IEEE 1516 version of the MAK RTI (after a formal evaluation and selection process) for the War in a Box project – part of the Canadian Advanced Synthetic Environment (CASE) project.

In Australia, the Australian Defence Simulation Office (ADSO) chose to standardize on the MAK RTI for High Level Architecture (HLA) networking for the Joint Simulation Capability, in conjunction with the Australian Army, Navy, Air Force, and Defence Science Technology Organization (DSTO).

In Europe, the MAK RTI was used in the EADS GeneSyS project. Various divisions of European companies such as Thales, Alenia Marconi, BAE, Electronica SpA, Sopra Steria, and Saab are all using the MAK RTI.

In Asia, the MAK RTI is being used by NEC on the JDSS (Japan Defense Simulation System) Program.

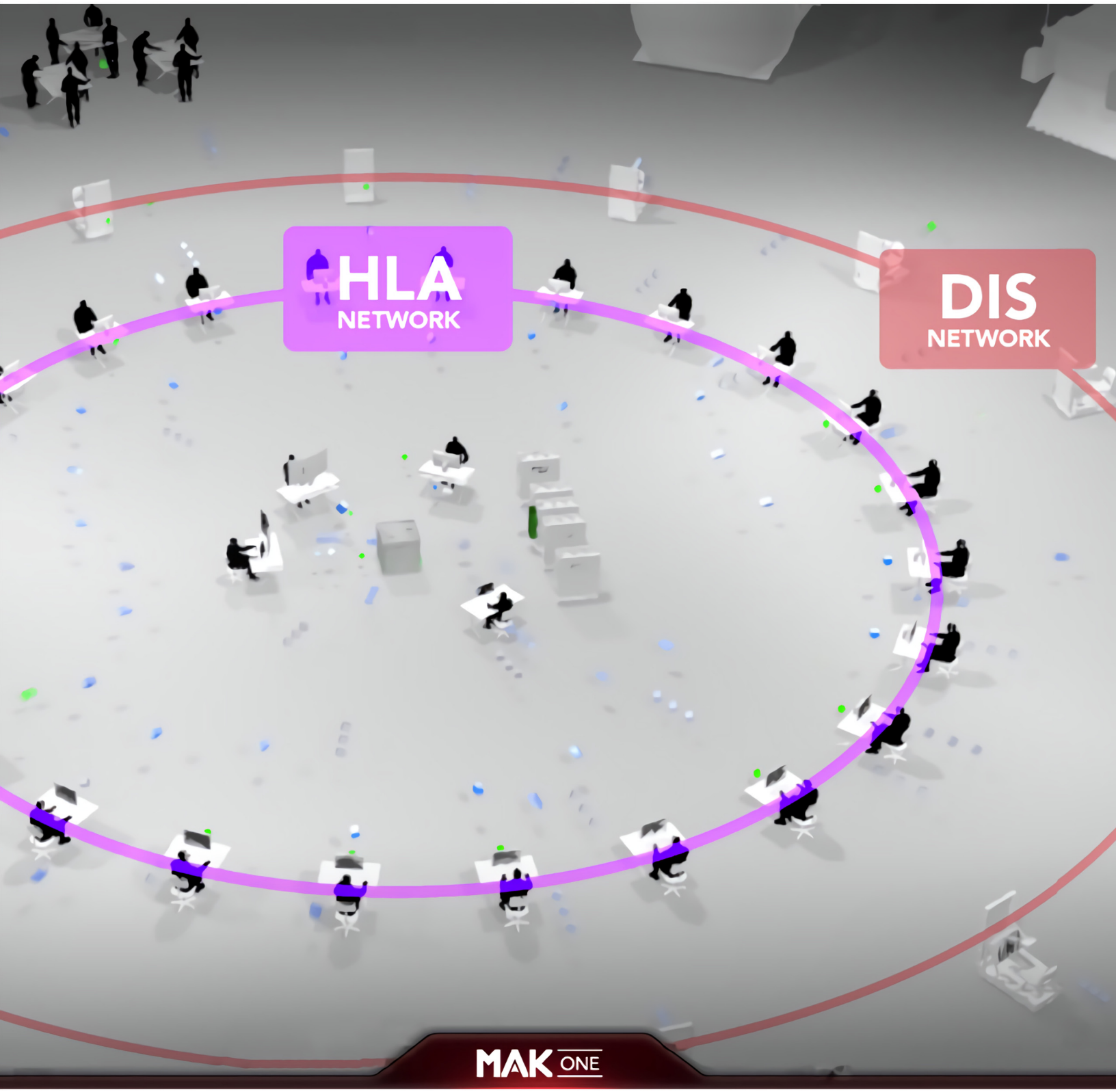
In fact, the MAK RTI is used by all of the top ten defense contractors in the world.

Superior Technical Support

At MAK, technical support is not just an afterthought. Our reputation for supporting our customers is one of the key reasons that people choose our products. When you submit a support ticket with questions, you work directly with our product developers who know the software inside and out. When you buy MAK's products, you can be sure that MAK will be in your corner as you work towards the successful completion of your HLA project.

If you need assistance that goes beyond the scope of technical support, our engineering services group is available to do customization, federation-specific performance tuning, or consulting on demand. We have had many MAK RTI customers buy our engineers' time to help them use it to its best advantage.

With maintenance, you are entitled to upgrades when they are released. We are constantly working to improve our products, and the fact that our developers provide customer support directly means that we are able to quickly react to customer feedback and integrate desired features in a timely manner.



RTI-5.0-13-250626