



MÄK and TENA

How MÄK Products Support the Test and Training Enabling Architecture

What is TENA?

TENA (Test and Training ENabling Architecture) is an interoperability architecture that is similar in many ways to HLA (the High Level Architecture). It was developed by the Foundation Initiative 2010 (FI2010) program, which is managed by the Director of the Central Test and Evaluation Investment Program (CTEIP), which is part of the US Office of the Undersecretary of Defense for Test and Evaluation (DOTE-OSD). TENA was developed to serve as the interoperability architecture for the US Live Range community. Currently, the direction of TENA is managed by the TENA Architecture Management Team (AMT) – a group that meets about once a quarter. The TENA program began in the late 1990s, making it almost as old as HLA.

How does TENA work?

TENA is a publish-subscribe-based architecture that is designed to get data from many sources to many recipients over a network (or other transport mechanism). Like HLA, TENA defines an API to a distributed piece of infrastructure software that all applications use to exchange data. In TENA, that piece of infrastructure software is called the TENA Middleware (roughly analogous to the HLA RTI). Like HLA, TENA also has the concept of an Object Model – a document that defines the kinds of data that the various applications need to exchange. In TENA, this is called an LROM (Logical Range Object Model) – similar to the HLA FOM concept.

Unlike HLA, the TENA Middleware API is type-safe, and not data-independent. When someone develops an LROM, they submit that LROM to the TENA website. A code-generator sends back an LROM-specific API that is used to extend the core Middleware API. In effect you are rebuilding the Middleware so that it is aware of your LROM, and can ensure that you are correctly encoding and decoding your data. Also unlike HLA, TENA places much of the object management responsibility within the Middleware, rather than at the federate (or TENA Application) level. The TENA middleware maintains your list of reflected and published objects, and allows you to set or obtain their current state.

How does TENA differ from HLA?

There are many minor technical differences, but only a few significant ones. The most important is probably the one described above: In HLA, data marshalling and data distribution are provided by two separate layers of software. The RTI is responsible for distributing your data, but knows nothing about the structure of your data (data types, etc.) Data marshalling, object management, and a type-safe interface are typically provided by a piece of Middleware that sits on top of the RTI (e.g. VR-Link). In TENA, data management and data distribution both take place in the same piece of software – the TENA Middleware.

The TENA object model has two capabilities that HLA does not offer. The first (and more significant) is support for object composition. In HLA, representing a “HAS-A” relationship between objects requires including one object identifier as an attribute of another. The RTI itself is unaware of the relationship. In TENA, the HAS-A relationship can be represented more directly in the Object Model, in a way that is known to the Middleware. The second capability is support for Remote Method Invocation. In TENA, objects can have functions in addition to attributes. However, RMI can be easily approximated in HLA by sending Interactions that cause a remote function to be executed.

On the other hand, as of October 2008, TENA does not support the HLA concepts of Time Management, or Data Distribution Management (DDM), making it less appropriate for non-real-time applications, or for executions that require a high level of scalability. Support for Save and Restore is also missing from TENA.

Another difference is that while the HLA Standard does not include a default FOM (the RPR FOM and others exist as separate standards), TENA does provide a “default” object model (called the TENA Object Model). Various executions are free to use other LROMs if they wish, but the TENA Software Development Activity that maintains the TENA Middleware also maintains and distributes the TENA Object Model. The TENA Object Model does support some concepts that are familiar to DIS or HLA RPR FOM users, such as Platform, WeaponFire and Detonation. But unlike DIS or the RPR FOM, the TENA Object Model does not employ dead-reckoning or thresholding, so updates for fast-moving objects may be sent quite frequently. In addition, an attribute update in TENA includes values for *all* attributes of an object, as opposed to just the ones that have changed.

The most significant difference between HLA and TENA, however, is philosophical, as opposed to technical. HLA was designed to be a general-purpose Standard, meant to be applicable to a wide variety of Live, Virtual and Constructive simulation applications – both military and commercial. From the beginning, it was developed with an eye towards becoming an IEEE Standard – a stable interface that can be implemented in a number of different ways, so that each federation could choose the implementation that best suits the network topology and performance profile of that federation. By contrast, TENA was developed specifically to meet the needs of the US Range Community. Distribution is more restricted, and functionality that is not required by its relatively narrow target audience, is omitted from the architecture. There is no TENA “Interface Specification” that exists independent of a specific version of the US DoD’s implementation of the TENA Middleware. Each release of the TENA Middleware defines a version of the TENA API, and the API tends to change from release to release. The benefit of the TENA approach is that the architecture can evolve more rapidly than something like HLA. The costs are that TENA Applications must continually be modified to keep up with the changing interface, and that TENA users do not have various Middleware implementations to choose among, based on the requirements of their federations.

Do any of MÄK’s products support TENA?

Yes! Both VR-Link and VR-Exchange support TENA. VR-Link 3.10 was the first VR-Link version to support TENA, and VR-Exchange 1.2 was the first VR-Exchange version to include a TENA Broker. Since support was added in these products, new versions have been updated to reflect the latest version of the TENA middleware.

Additionally, products like VR-Vantage can be extended to support TENA with minimal work by writing a new connection. While we have done some work to prototype this in the past, it is not currently part of the VR-Vantage product. If you are interested in a version of VR-Vantage which supports TENA please let us know.

What LROMs (Logical Range Object Models) do you support?

Back when we developed our strategy for HLA support, we realized that it was important to provide customers with *both* built-in support for a common reference FOM (the RPR FOM), *and* the flexibility to tailor our tools to custom FOMs (our FOM-Agility concept). This strategy ensures instant, out-of-the-box interoperability for those customers who are able to use a Standard FOM, but also provides a clear path for those who cannot. We have followed the same approach with TENA - providing built-in support for several TENA LROMs, while also allowing the flexibility to map to custom LROMs through *LROM Mappers* (see LROM Agility below).

VR-Link includes built-in support for the TENA Object Model (Platform Objects, Application Management Objects, and WeaponFire and Detonation Messages only), as well as the MÄK LROM (a MÄK-developed LROM that is based on the TENA Object Model, but adds classes for a variety of DIS/RPR concepts). VR-Link also includes example source code for a JNTC (Joint National Training Capability) LROM Mapper. The JNTC LROM is maintained by the US Joint Forces Command, and has been used in a variety of TENA executions.

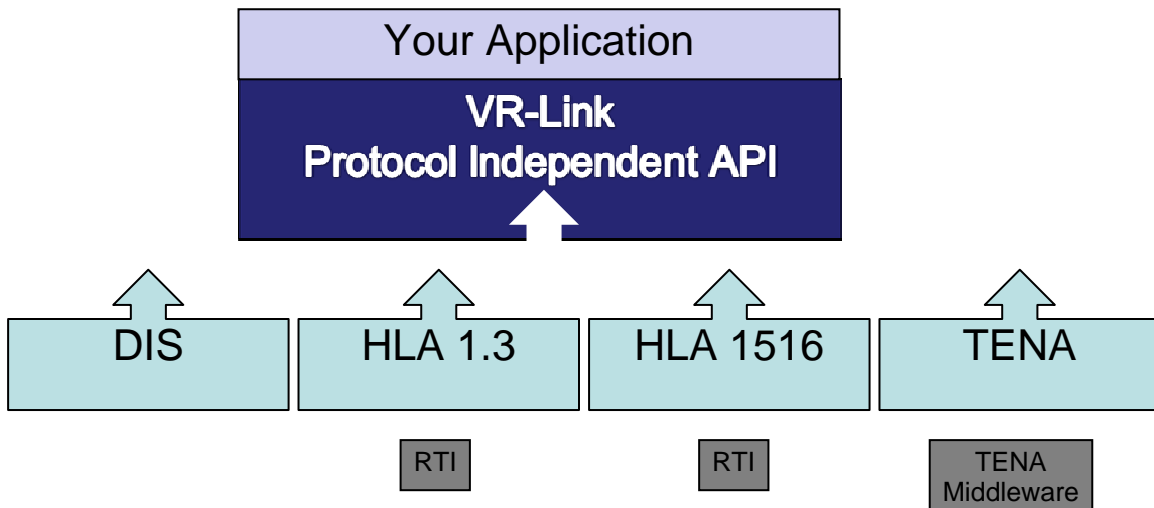
What is the relationship between VR-Link and the TENA Middleware?

VR-Link sits between your application code and the TENA Middleware. The TENA version of the VR-Link library *uses* the TENA Middleware by making calls to *its* API. This relationship is very similar to the

way things work in HLA, where VR-Link implements the interface that it provides to the application by making calls to the HLA RTI.

How does TENA support work in VR-Link?

Our strategy for supporting TENA is an extension of our strategy for DIS and HLA. VR-Link provides a top level API that abstracts away from the details of the networking protocols, along with separate libraries that implement the API for each protocol. This means that you can write code once, and switch between the various protocols largely just by recompiling and linking with the appropriate versions of the VR-Link libraries. Historically, we have provided implementations of the VR-Link API for DIS, HLA 1.3, and the IEEE 1516 version of HLA. When we added support for TENA we created an additional “leg” – a fourth implementation of VR-Link’s protocol-independent API.



What does the MÄK LROM look like?

In the case of TENA, a “RPR LROM” did not already exist, with representations of all of the object and messages types that the VR-Link API covers. So it was necessary for us to create one: the MÄK LROM. When developing the MÄK LROM, we did not start from scratch and we did not duplicate pieces that already existed. We first started with the TENA Object Model (TENA’s “default” LROM that is maintained by the TENA SDA), and used whatever we could from that LROM (the Platform, AMO, WeaponFire, and Detonation classes, and the EmbeddedSystem base class.). We then looked at the JNTC (Joint National Training Capability) LROM – an object model that is now commonly used in TENA executions in which the US Joint Forces Command is a participant. From there, we were able to leverage the AirTransponder class to represent the IFF concept in VR-Link. But there were many VR-Link concepts that were not yet represented in either of these common LROMs. For these concepts (including Aggregates, Simulation Management Messages, Gridded Data, Collision, Environmental Process, Radios, and others.), we developed our own object and message classes, and added them to the MÄK LROM. For the most part, we tried to use the same attribute names and data types that you would find in HLA’s RPR FOM.

Does MÄK consider the MÄK LROM to be a proprietary LROM?

No! In fact, we had hoped to call it the “RPR LROM” instead of the “MÄK LROM” in order to underscore that point, but the TENA SDA preferred to reserve that name until there was some consensus among users that our LROM is in fact consistent with what is implied by the RPR LROM label. So although it is currently named the MÄK LROM, we would like the community to think of it as an early version of the RPR LROM for TENA – something that is freely available to the community, and that can be supported by anyone to foster a-priori interoperability among TENA Applications. The MÄK LROM is currently publicly available on the TENA web site.

Consistent with our corporate philosophy of supporting and promoting interoperability standards, we are happy to work with both the TENA SDA (the keepers of the TENA LROM), and JFCOM (the keepers of the JNTC LROM) to merge our object and message classes into these common reference LROMs. Our hope is that these classes will eventually be pushed down into the TENA LROM itself, which would eliminate the need for a separate MÄK LROM.

How does LROM Agility work?

The concept of LROM Agility in the TENA version of VR-Link is very similar to the FOM Agility concept in the HLA version. VR-Link uses an “LROM Mapper” to map between the LROM-independent VR-Link API and the particular LROM being used in your TENA execution. An LROM Mapper is a DLL that contains a variety of classes and functions to achieve this mapping. For example, when you instantiate a VR-Link “Publisher” object to publish a locally simulated object, the Publisher will ask the LROM Mapper for the name of the TENA class to use to represent the type of object being published. Similarly, a VR-Link Reflected Object List will call functions in the LROM Mapper DLL to find out which LROM classes to subscribe to. When a Publisher needs to send an update, it will use the *Interface* object provided by the LROM Mapper for the class in question to map from VR-Link’s native data representations to those that are dictated by the LROM. Similarly, when an update is received, the Reflected Object will use the Interface object contained in the LROM Mapper DLL to decode the incoming data into one of VR-Link’s State Repositories.

VR-Link’s DLL-based object model mapping approach is particularly valuable in TENA, due to the fact that each TENA LROM requires a new LROM-specific middleware extension (the code that is auto-generated for you when you submit an LROM to the TENA web site). A typical TENA application needs to change significantly every time its target LROM changes: application code needs to be modified so that it uses the newly generated API, and then the application must be recompiled and re-linked. With VR-Link’s LROM Agility approach, the changes are limited to the LROM Mapper DLL. As long as your LROMs include data representations for the kinds of objects and messages contained in the VR-Link API, you can switch among various LROMs by configuring your application for the appropriate LROM Mapper DLL. It is not necessary to recompile or re-link your application. This is particularly important in cases when you are using COTS or other third-party applications for which you do not even have source code.

What if I want to directly access the TENA Middleware, or Tune Protocol-Specific Parameters?

No problem. Our goal is to insulate you from the networking details you don’t want to see, not to restrict you from getting to the ones you do. VR-Link allows you to directly access the TENA Middleware’s classes and functions if you want to, so that you can control and configure TENA-specific functionality. Of course, you are now operating below the level of VR-Link’s protocol-independent API, and you will need to use #ifdefs to conditionally compile your protocol-specific code only when compiling for that particular protocol.

Will I automatically get TENA support when I upgrade VR-Link?

Unfortunately, no. The TENA “leg” of VR-Link is being treated as a separately priced add-on module to VR-Link. Please contact info@mak.com, or your local MÄK reseller for pricing information.

Does MÄK support a TENA-to-HLA bridge?

Yes! VR-Exchange is an interoperability portal that can perform many-to-many translation among various interoperability protocols and object models. The VR-Exchange architecture consists of a protocol-independent Data Exchange, and a Broker for each supported protocol. Each Broker runs as a separate process, and trades data with the other Brokers through the Data Exchange. The Data Exchange is a shared memory interface that is created and maintained by a central “portal” process.

VR-Exchange includes Brokers for TENA as well as HLA 1.3 and the IEEE 1516 version of HLA. When used with these Brokers, VR-Exchange supports out-of-the-box translation between HLA and TENA. (Of course, your HLA FOM and your TENA LROM must represent similar enough concepts that a mapping between them is sensible.)

Does VR-Exchange also support DIS-TENA bridging?

Yes! VR-Exchange also includes a DIS Broker, which supports the most commonly used DIS PDUs. Configure VR-Exchange with a DIS Broker and TENA Broker for out-of-the-box translation between DIS and TENA.

Can I use VR-Exchange to translate among different TENA LROMs?

Yes. If you configure VR-Exchange with two TENA Brokers, each using a different LROM Mapper, you will be able to achieve interoperability between TENA Applications that were built to different LROMs (assuming, again, that the two LROMs represent similar concepts, and that there is sensible mapping between them.)

Can I use VR-Exchange to bridge different versions of the TENA Middleware?

Yes. As in HLA, the lack of a “wire standard” in TENA means that different versions of the TENA Middleware will not be able to interoperate on the network. And although the TENA SDA is the only “vendor” of TENA Middleware, different releases of that Middleware are not interoperable. Just like in HLA, where VR-Exchange can be used to perform RTI-to-RTI bridging, VR-Exchange can be used to allow TENA applications that are built to different versions of the TENA Middleware to interoperate. Just configure VR-Exchange with two TENA Brokers, each built against a different version of the TENA Middleware. (This may require a custom TENA Broker built against the desired version of the TENA Middleware, if you need to support a version of the Middleware for which we are not distributing a built-in Broker).

Is the TENA Broker for VR-Exchange a Separately Priced Option?

No – the TENA Broker comes standard with VR-Exchange. Since the role of VR-Exchange is to bridge and translate among various protocols, it makes sense for us to include Brokers for various protocols with the standard VR-Exchange license.

When will the other MÄK products support TENA?

For the most part, that depends on *you!* Because there are relatively few existing COTS products in the TENA market, it has been somewhat difficult for us to gauge the level of demand for products like ours in the TENA community. This is one reason we have decided to take a phased approach to TENA development. Feedback from you, our customers, will dictate how much time we invest in TENA, and which products we focus on in what order. We have already built custom TENA versions of various MÄK products for customers who required specific limited functionality, but have not released full-featured TENA versions of these products yet. If you are interested in a TENA version of VR-Vantage, MÄK Plan View Display, VR-Forces, MÄK Data Logger, please let us know by sending email to sales@mak.com.

Isn't MÄK an HLA Company? Does MÄK's TENA support represent a change in strategy?

Not really. Our strategy has always been to help our customers meet their interoperability requirements regardless of which protocol they have chosen - as long as there is enough demand to justify our investment. Many years ago, people used to think of us as the DIS company. Yet when HLA came about, we quickly developed expertise, and released HLA versions of all of our products. We have been involved in the RPR FOM from its inception, yet we have also been willing to support other FOMs when our customers have needed us to. Our decision to add TENA to our staple of supported architectures only reinforces our commitment to meeting the broad interoperability needs of our customers.

We of course continue to be heavily involved in the evolution of both the HLA and DIS standards, and will continue to maintain, improve and extend the HLA and DIS versions of our products as well.

What is MÄK's role in the TENA community?

MÄK started attending TENA AMT meetings in mid-2004. When we became TENA stakeholders by starting to invest in TENA products, we became members of the TENA AMT Advisory Group. We were given a slot on the AMT-32 agenda to present our TENA products, and have accepted an offer from the TENA SDA to include a page dedicated to our TENA tools on the TENA web site.

Does MÄK distribute the TENA Middleware?

No. Distribution of the TENA Middleware and other supporting utilities is managed by the TENA Software Development Activity (SDA) on behalf of the US DoD. This software must be requested from the SDA at <https://www.tena-sda.org>. In addition to the core Middleware libraries, you must also download the LROM-specific libraries that are generated by the TENA website for the LROM you will be using (even if it is the MÄK LROM). The TENA SDA restricts distribution of these libraries, and we are not allowed to distribute them ourselves.

Is the TENA Middleware exportable?

We understand that the US DoD has arrangements with various other countries that allow them to obtain and use the TENA Middleware. Please contact the TENA SDA through <http://www.tena-sda.org> for specific information.

Where can I find more information about TENA?

The TENA Software Development Activity's website is: <http://www.tena-sda.org>.

Should I switch from HLA to TENA?

HLA and TENA provide roughly the same functionality, and should provide roughly equivalent performance, given well-built implementations of both. So technical concerns are probably not what will drive your decision. Most HLA versus TENA decisions are based on programmatic or political factors. People that do move to TENA do so largely because they need to interoperate with US Ranges, or with someone else who has chosen TENA. One should also consider the availability of tool support for the various architectures. Although TENA versions of some visualization, logging, diagnostic, and simulation tools are available or coming soon (including MÄK products), there is currently a greater breadth and depth of GOTS and COTS tools available for HLA than for TENA.