



VR-Link 3.13 Release Notes

This release note provides the following release-specific information for VR-Link Release 3.13:

Systems Supported	2
Disk Space Requirements	2
Building VR-Link on Linux	2
Compiler Compatibility on Windows	2
TENA Compatibility	3
FLEXlm Support	3
Using Libraries and Binaries Built with Visual Studio 2005 and Later	3
Patch Required for AMD Dual-processor Windows PCs	4
Backward Compatibility	4
Network Compatibility	4
RPR FOM Versions Supported	5
RTI Support	5
New Features and Changes	6
New Socket Implementation	6
Documentation Updates	7
Bug Fixes	7
Known Problems	8

Copyright © 2009 VT MÄK, 68 Moulton St., Cambridge, MA 02138 All rights reserved.
MÄK Technologies®, VR-Forces®, RTIsPY®, and VR-Link® are registered trademarks of VT MÄK.
Document ID: VRL-3.13-3-090209

Systems Supported

Table 1 lists the platforms currently supported by MÄK VR-Link 3.13. Table 2 lists the platforms supported by the TENA version of VR-Link. Please contact MÄK if you are interested in purchasing VR-Link for other platforms. Application code must be built with the indicated compilers in order to link to VR-Link libraries.

Table 1: Platforms supported for HLA and DIS

Platform	Compiler
Red Hat Enterprise Linux Workstation 4.0, 5.0 Fedora 7	default compiler
PC with Windows XP/Vista	Microsoft Visual C++ 7.1 Microsoft Visual C++ 8.0 Microsoft Visual C++ 9.0 (32-bit and 64 bit)

Table 2: Platforms supported for TENA

Platform	Compiler
Red Hat Enterprise Linux Workstation 4.0, 5.0	default compiler
PC with Windows XP/Vista	Microsoft Visual C++ 7.1 Microsoft Visual C++ 8.0

Disk Space Requirements

Linux installations require approximately 80 MB for VR-Link files and an additional 120 MB for documentation files.

On PCs, you need 320 MB for VR-Link files and 120 MB for all documentation files.

Building VR-Link on Linux

VR-Link does not require you to use the MÄK build system, but if you choose to, you must use gmake 3.81 or later. You can download it from the following URL:

<http://savannah.gnu.org/projects/make/>

Compiler Compatibility on Windows

MÄK provides versions of product releases that have been compiled with Microsoft Visual C++ 7.1, 8.0, and 9.0. When you run MÄK products together, for example, the Logger 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.

TENA Compatibility

VR-Link 3.13 was built against TENA 5.2.2. The MÄK LROM version is 2.1.

FLEXlm Support

VR-Link 3.13 uses FLEXlm 11.6.

Using Libraries and Binaries Built with Visual Studio 2005 and Later

All MÄK products built with Microsoft 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 historically 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.

Unfortunately, with the release of the new C Runtime Libraries required by Microsoft Visual Studio 2005 (MSVC++8.0) and later, 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 and later. The 32-bit installer is named *vcredist_x86.exe*; the 64-bit installer (if supported) is named *vcredist_x64.exe*. They are in the base directory of any installed MÄK product that requires them.

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>



You must ensure that the preprocessor defines `_SECURE_SCL=0`, and `_HAS_ITERATOR_DEBUGGING=0` are set for MSVC++8.0 and MSVC++9.0 builds. If these are not set, random crashes and assertions may be encountered during runtime."

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.

Backward Compatibility

A high degree of compatability was maintained between 3.12 and 3.13. With the exception of specific networking classes all applications should be source compatible. If you are upgrading from VR-Link 3.11 or earlier, be aware that VR-Link's implementation of articulated parts changed in version 3.12. Please see *VR-Link Developer's Guide* for details about the current implementation of articulated parts.

Network Compatibility

HLA only

VR-Link 3.13 is compliant with:

- ♦ RPR-FOM 0.5, 0.7, 0.8, 1.0, and a subset of 2.0 (draft 6, 14, and 17)
- ♦ MÄK RTI 2.x, 3.x
- ♦ Pitch RTI 1.3 C++ interface.

Other RTIs that support the HLA 1.3 specification or the SISO DLC HLA API 1516 (SISO-STD-004.1-2004 version of the IEEE 1516 specification.)

DIS only

This release supports DIS 2.0.4, IEEE 1278.1, 2.1.4, and IEEE 1278.1a, and can therefore interoperate with DIS applications of any of these versions.

RPR FOM Versions Supported

VR-Link 3.13 has built-in support for versions 0.5, 0.7, 0.8, 1.0, and 2.0, drafts 6, 14, and 17, of the RPR FOM. By default, VR-Link 3.13 uses RPR FOM 1.0.

VR-Link does not officially support RPR FOM 2 Draft 18 at this time. However, the draft appears to be similar enough to RPR FOM 2 Draft 17 that the 2.0017 FOM Mapper may be used for federates wishing to interoperate with RPR FOM 2 Draft 18.

If you want to use a version of the RPR FOM other than 1.0, pass the version number (0.5, 0.7, 0.8, 2.0006, 2.0014, or 2.0017) to the *DtRprFomMapper* constructor and pass the resulting object to the *DtExerciseConn* constructor. Also, make sure you are using a federation execution name that corresponds to the right FED file. For example:

```
DtExerciseConn conn("VR-Link20017", "MyApp", new DtRprFomMapper(2.0017));
```

VR-Link examples like *f18* and *hlaNetdump* have a command line option, `--rprFomVersion`, that you can use to choose a RPR FOM version, using one of the version numbers listed in the previous paragraph.

RTI Support

VR-Link 3.13 has been tested with the Pitch RTI 1.3 and MÄK RTI 3.3.x.

If you use the MÄK RTI, remember to make sure that VR-Link can find your FED file or FDD file, and optional *rid.mtl*. Put the FED file or FDD file in the directory from which you are running, or set the environment variable `RTI_CONFIG` to the directory that contains it. See *MÄK RTI Reference Manual* for a list of the options for specifying the location of the *rid.mtl* file.

New Features and Changes

VR-Link 3.13 has the following new features and changes:

- ♦ New sockets that support IPv6 have been implemented. They replace the existing socket classes in VR-Link (*DtSocket* and its derivatives). The *DtSocket* classes are now *DtInetSockets*. The old *DtSocket* classes have been deprecated, but are still accessible in the *vlNetworkCompatibility* library.
- ♦ Support for MSVC++ 9.0. The MSVC++ 9.0 version includes 32-bit libraries and 64-bit libraries.
- ♦ You can now specify a device address by using the `--deviceAddress` command-line option, or in the application initializer. This is useful for machines with multiple network cards. If set, it uses the device's broadcast address to send DIS PDUs, and only receives PDUs from the specified network card.
- ♦ The DIS exercise connection now can suppress self reflection by using the `--suppressSelfReflect` command-line option, or be set in the application initializer.
- ♦ The DIS exercise connection now can use IPv6 by using the `--useIpv6` command-line option, or by setting a flag in the application initializer.
- ♦ You can now add and remove interest in DI-Guy PDUs (DIS only) after a reflected entity list has been created.

New Socket Implementation

The VR-Link socket classes have been reimplemented. The new network infrastructure provides the following features:

- ♦ Better error message handling (storage and retrieval on a per-socket basis)
- ♦ IPv6 support
- ♦ Synchronous and asynchronous operation from the same socket
- ♦ Scatter/gather
- ♦ Customizable payload parsers
- ♦ Customizable packet handlers (asynchronous mode)
- ♦ Improved support for multi-homed hosts.

Network code has been significantly refactored and is no longer source compatible with older versions of VR-Link. However, these changes have largely been abstracted away by the *DtExerciseConn* class, which has only minor changes. You will only be affected if you have written your own socket subclasses. In this case, you should be able to change the link line to use the old classes and will otherwise not be affected.

Documentation Updates

VR-Link Developer's Guide has been updated for this release. The chapter “FOM Agility Examples” has been removed from the manual. The information that was in that chapter is now part of the class documentation. Documentation of API examples has been added to the class documentation for each supported simulation standard.

Bug Fixes

VR-Link 3.13 fixes the following bugs:

- Comment interaction encoding and decoding was incorrect for RPR FOM 2, draft 17. This has been fixed, and can be accessed by setting the RPR FOM Mapper revision to 2 (using `--rprFomRevision` or in the *DtVrApplicationInitializer*) for RPR FOM 2, draft 17.
- The *DtConfigVar<DtVector>* members of the initializer class (as used in the F-18 example) did not work properly.
- On Linux, *DtDynamicLibrary* opens dynamic libraries with `RTLD_LOCAL` by default now, rather than with `RTLD_GLOBAL`. You can still use `RTLD_GLOBAL` by calling *DtDynamicLibrary::create()*, and passing in `true` as the `useGlobal` parameter.
- Using the copy constructor for any of the derivatives of *DtPduWithData* caused a crash with MSVC++8.0.
- The *GridDataRecordType0/1/2* structures were not properly encoded or decoded for RPR FOM 2 draft 17. This has been fixed and can be accessed by setting the RPR FOM Mapper revision to 3 (using `--rprFomRevision` or in the *DtVrApplicationInitializer*) for RPR FOM 2, draft 17.
- The *DtReflectedEntityList* class did not remove interest in DI-Guy PDUs (for DIS) on destruction.
- The *DtReflectedEntityList* class did not remove interest in Entity State Update PDUs (for DIS) on destruction.
- IFF PDUs were not sent out if the Host ID or Event ID changed.
- A *newReflectedDesignator()* method has been added to the DIS *DtReflectedDesignatorList* class. It can be overridden to change the creation of designators in DIS.
- The *HostEntityID* in the *IsPartOf* struct for entities (in RPR FOM 2, draft 17) was not filled in properly.
- All DIS encoders and decoders now have base classes.
- Thread safety has been improved for the classes *DtOutputStream*, *DtOutputStream-Buffer*, and all supplied *DtPrinter* classes.
- The *DtArtPartThresholder* and *DtThresholder* classes have been moved to the *vlpi* (protocol independent) library. They were in the *vl* (protocol dependent) library.

- The `RTIObjectIdArrayStruct` was not properly encoded or decoded for RPR FOM 2, draft 17. This has been fixed, and can be accessed by setting the RPR FOM Mapper revision to 2 (using `--rprFomRevision` or in the *DtVrApplicationInitializer*) for RPR FOM 2, draft 17.
- The `operator<()` method in the *DtSimulationAddress* class has been made `const`.
- Instead of taking a *DtDDMRegionSP*, the HLA *DtReflectedObjectList* classes and *DtObjectPublisher* classes now take a *DtDDMRegionSet* pointer. This is simply an STL set of *DtDDMRegionSP*'s, so that you can specify more than one region to subscribe to or register with.

Known Problems

This release of VR-Link has the following known problems:

- VR-Link correctly encodes and decodes *EnvironmentalProcess* objects for RPR FOM 2, Draft 14. However, the RPR FOM specifies a way of encoding which prevents VR-Link from correctly decoding modified or custom *EnvironmentalRecords*. Unfortunately, many MÄK products built with VR-Link use customized *EnvironmentalRecords*. If you use these products or if you use customized records in your applications, please do not use the RPR FOM 2 Draft 14 FOM Mapper. This is not a problem in other FOM versions, and it has been resolved in future versions of the FOM.
- The RTI 1516 API specifies that all strings passed to and from the RTI are handled as wide strings. VR-Link (and the MÄK RTI) store strings internally as narrow strings. This means that true multibyte wide strings used by RTI Federates may not be handled correctly when received by VR-Link. VR-Link handles name reservation wide strings correctly, however, as this is a 1516-only function. VR-Link provides conversion functions from Wide To Narrow/Narrow to Wide string conversion in *vlStringUtil.h*.
- On non-windows platforms, the classes *DtSharedMemoryPoolManager* and *DtSharedMemoryPoolClient* (*vlShmPool.h*) sometimes have difficulty creating a large memory pool (greater than 1 MB) if the pool name is longer than two or three characters. The classes refuse to create the memory pool even if the requested pool is smaller than the system resource SHMMAX. Setting the pool name to fewer than three characters will work around this problem.
- Section 6.10 in *VR-Link Developer's Guide* describes interoperability issues that arise when you try to run an HLA 1.3 federate with an IEEE 1516 federate. In particular it describes the requirement to use an identical FED or FDD file for all federates. 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.