VR-Link 5.4.1 Release Notes

This release note provides the following release-specific information for VR-Link Release 5.4.1:

Document ID: VRL-5.4.1-3-190318

Systems Supported

Table 1 lists the platforms currently supported by MAK VR-Link 5.4.1. Please contact MAK 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

Platform	Compiler
Red Hat Enterprise Linux Workstation 6.0 (32 bit and 64 bit libraries)	default compiler
Red Hat Enterprise Linux Workstation 7.0 (64 bit libraries)	default compiler
PC with Windows 10	Microsoft Visual C++ 10.0 (32-bit and 64 bit)
	Microsoft Visual C++ 12.0 (64 bit)
	Microsoft Visual C++ 14.0 (64 bit)

The VC 14 build is binary compatible with all MAK VC 15 products.

VR-Link for C# supports all Windows compilers.

VR-Link for Java supports all compilers and operating systems.

VR-Link for Java requires Java 8.

Disk Space Requirements

Linux installations require approximately 2 GB of disk space. On Windows, you need approximately 1.5 GB of disk space.

Compiler Compatibility on Windows

MAK provides versions of product releases that have been compiled with different versions of Microsoft Visual C++. When you run MAK products together on the same computer, for example, the Logger and a VR-Vantage application, we strongly recommend that you run versions compiled with the same compiler. Mixing products compiled with different versions of the compiler on the same computer can result in program instability. The exception to this is that products built with VC 14 and VC 15 are binary compatible.

License Manager

To run the licensed version of VR-Link, you must install license management software. VR-Link 5.4.1 uses FLEXlm 11.13.0.2. If you are upgrading from a version of the VR-Link that used an older version of FLEXlm, you must upgrade your license management files. You do not need a new license. Licenses are forward compatible.

The License Manager files are not part of the VR-Link installer. You can download them at:

- Windows:
  http://ftp.mak.com/out/MAKLicenseManager-win-setup.exe (for 32-bit systems)
  http://ftp.mak.com/out/MAKLicenseManager-win64-setup.exe
- Linux: http://ftp.mak.com/out/MAKLicenseManager-linux-setup.tar.gz (for 32-bit systems)
  http://ftp.mak.com/out/MAKLicenseManager-linux64-setup.tar.gz

License manager support is available at:

http://www.mak.com/support/licenses

Network Compatibility

VR-Link 5.4.1 is compliant with RPR FOM 1.0 and RPR FOM 2.0 and includes FOM mappers for many draft versions of the RPR FOM. It is compliant with RTIs that support the HLA 1.3 specification, the SISO DLC HLA API 1516 version of the IEEE 1516 specification (SISO-STD-004.1-2004), or HLA Evolved.

To use an RTI with VR-Link it must use the same operating system and be built with the same compiler.

VR-Link 5.4.1 supports DIS 4, 5, 6, and 7, and can therefore interoperate with DIS applications of any of these versions.

Backward Compatibility

VR-Link 5.4.1 has the following backwards compatibility issues:

- VR-Link versions 5.0-5.1.3 encoded environmental process DIS Version 7 PDUs incorrectly. This has been corrected. Unfortunately this means that VR-Link 5.3.1 will not inter-operate with those versions of VR-Link for DIS 7 Environmental Process PDUs. We recommend upgrading all VR-Link 5.0-5.1.3 federates if you are using that specific configuration.
- VR-Link 5.2 encoded data interactions incorrectly for RPR FOM 2.0. This is fixed in VR-Link 5.3.1. However, this incompatibility causes applications that were built with VR-Link 5.2 to crash when trying to decode VR-Link 5.3.1 interactions.

  For backwards compatibility, the incorrect encoding is now provided as RPR 2.0 Revision 1. The correct encoding is RPR 2.0 Revision 2. Use Revision 1 to maintain compatibility with previous versions of VR-Link. Use Revision 2 to be compliant with RPR FOM 2.0.
- VR-Link 5.0 renamed header files to conform to MAK file naming conventions. If you have not already migrated to these new file names, the next two sections describe the changes and the migration script.

Using the Compatibility Directory

VR-Link has a *compatibility* directory in its *include* directory. This directory has the same directory structure as the main *include* directory with header files that have the same names as previous versions of VR-Link. The header files in this directory do not duplicate the declarations in the renamed header files, they include the renamed header files themselves. To use the *./include/compatibility* directory, add it to your project's include directory list. You will not have to update the include statements in your project files.

Using the Migration Script

VR-Link provides a script (*./compatibility/updateIncludes.sh* on Linux, *./compatibility/updateIncludes.exe* on Windows) that will go through all of your source files and modify the include statements to reference the correct header files.

The update script uses the mapping files located in *./compatibility/mappings* to update include statements for the renamed VR-Link header files in a given set of source directories. The script searches all input directories for header (*.h and *.hpp), inline (*.inl), and source files (*.c, *.cpp, and *.cxx), and checks the include statements in those files against the mapping files. For the script to correctly map the header file names, VR-Link include statements must specify the VR-Link library path. This is so the scripts can identify which mapping file to use to map the include statement. Therefore, the base VR-Link include directory must be set as the include path for projects, instead of adding the individual library directories (*include/vl*, *include/mtl*, and so on.).

The script does the following:

- If a header file has been renamed, the header file in the include statement is renamed.
- If a header file has been removed, the include statement is removed.

The command syntax for the script is as follows:

```
updateIncludes [options] directory1 ... directoryn
```

where:

- `directory1 ... directoryn` is a list of directories in which the script will search for source and header files for updating.
- `-d` displays debug level information about the files and include statements as they are processed
- `-h` provides information about the various flags.
- `-v` displays slightly more verbose output than `-d`.
- `-x` tells the script to perform the replacements. If the script runs without the `-x` flag, it outputs the commands that will be run if the `-x` flag is specified. This is so you can verify the changes that are going to be made before the script changes all the files.

RTI Support

VR-Link 5.4.1 has been tested with the MAK RTI 4.5 for HLA 1.3, HLA 1516, and HLA Evolved.

If you use the MAK 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 MAK RTI Reference Manual for a list of the options for specifying the location of the rid.mtl file.

Applications built with VR-Link 5.4.1 for HLA can use any RTI that conforms to the relevant dynamic link compatible HLA standard (1.3, SISO DLC 1516, HLA Evolved) and was built using the same operating system and compiler.

New Features and Changes

VR-Link 5.4.1 has the following new features:

- DIS PDU relative timestamps are now adjusted by using the time the PDU is received, not the time it is processed. This avoids problems with time PDUs might spend sitting in the receive queue waiting for processing (e.g., under heavy PDU loads). Intuitively, the relative timestamp should be closer to the sent time, assuming the CPU clocks are synchronized between sender and receiver and that the network latency is very low.
- C# only receives an EntityStateMessage when a reflectedEntity post update callback is received. (Previously, an EntityStateMessage would be received for each entity every time drainInput was called.)
- The DtExerciseInitializer setting allowCoupledPdus has been removed and replaced by attributePduStrategy. These values are set:
  - DtAttributePduStrategy_Unspecified = 0
  - DtAttributePduStrategy_NoAttributePdu = 1
  - DtAttributePduStrategy_Couple = 2
  - DtAttributePduStrategy_DoNotCouple = 3
  - DtAttributePduStrategy_DoNotCoupleAndSendAttributeFirst = 4

  Unspecified is the default. If using DIS7, it sends and couples the Attribute PDU (DtAttributePduStrategy_Couple) when necessary. If using DIS earlier than 7, it will not use the Attribute PDU (DtAttributePduStrategy_oAttributePdu). The other settings will have their behavior fixed regardless of DIS version (overrides).

  - No Attribute PDU setting causes publishers to never send the Attribute PDU.
  - Couple setting causes publishers to couple the Attribute PDU regardless of DIS version (so using DIS6 with this can cause a non-standard packet to be bundled at the end of a PDU).
  - Do Not Couple setting causes PDUs that would normally be coupled under setting 2 to be sent as two separate packets (not bundled) regardless of DIS version.
  - Do Not Couple and Send Attribute First settings will send the same packets as setting 3, but the attribute PDU is sent first, then the other.
- RPR2 draft 17 FOMs will no longer be remade for newer versions of VR-Forces.

  The table lists the FOMs that are replaced by newer versions:

Table 1-1: Deprecated and Replacement FOMs

Old	New
MAK-RPR1-1-3.fed	MAK-RPR1-1-4.fed
MAK-RPR1-1-3.omt	MAK-RPR1-1-4.omt
MAK-RPR1-1-3.xml	MAK-RPR1-1-4.xml
MAK-RPR1-1-3_evolved.xml	MAK-RPR1-1-4_evolved.xml
MAK-RPR2-1-3.fed	MAK-RPR2-1-4.fed
MAK-RPR2-1-3.omt	MAK-RPR2-1-4.omt
MAK-RPR2-1-3.xml	MAK-RPR2-1-4.xml
MAK-VRFAggregate-2_evolved.xml	MAK-VRFAggregate-3_evolved.xml
MAK-VRFExt-3_evolved.xml	MAK-VRFExt-4_evolved.xml
MAK-DIGuy-3_evolved.xml	MAK-DIGuy-4_evolved.xml

Fixed Bugs

This release fixes the following issues that were problems in previous releases:

- C# geocToTopoTransform() function now returns correct coordinate transform. 65250
- The PDU Type value for the Record-R and RecordQuery-R PDUs were swapped. They are now corrected to match the standard. 63459
- The recordCount and recordLength fields are now checked to ensure that they do not exceed the total length of the record data. This prevents a memory access violation. 63554
- The IFF simulation bit in the PDU status is now also set in the PDU header. 63584
- The Java version of VR-Link now receives fire interactions. 63598
- The wrong hashCode was used for the Java entity identifier. 63599
- When using the code generator for C#, objects were not correctly published. 63720
- C# generated code sometimes (depending on the FOM definition used for generation) published interaction parameters in little endian byte order. 63721
- DtMultiConfigVariable<DtString> could not handle quoted arguments containing spaces. The code was changed to use DtCmdLine::MultiArg<std::string> to resolve the problem. 64544