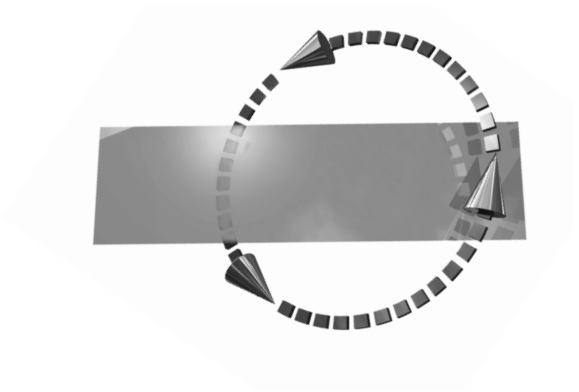

VR-Link 3.9.4 Release Notes

This release note provides the following release-specific information for VR-Link Release 3.9.4:

Document ID: VRL-3.9.4-3-050318

Systems Supported

Table 1 lists the platforms currently supported by MÄK VR-Link 3.9.4. 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

Platform	Compiler
SGI IRIX6.5	MipsPro7.3 C++ compiler (available from SGI)
Sun Sparcstation with Solaris2.7 or later	SPARCompiler C++ version 5.2 (available from Sun)
PC with Red Hat Linux 8.0 and 9.0	GNU C++ (GCC) 3.2
PC with Windows 2000/XP	Microsoft Visual C++ 6.0 SP5 Microsoft .NET

You must use *libXml2* to build any HLA application.

Disk Space Requirements

On SGI IRIX, you need approximately 270 MB for VR-Link files and 60 MB for all documentation files. Other UNIX installations require approximately 50 MB for VR-Link files and an additional 60 MB for documentation files.

On PCs, you need 50 MB for VR-Link files and 60 MB for all documentation files.

Compiler Compatibility on Windows

MÄK provides versions of product releases that have been compiled with Microsoft Visual C++ 6.0 and 7.1. When you run MÄK products together, for example, the VR-Link 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.

Patching the Microsoft Visual C++ 6.0 Compiler

If you use Microsoft Visual C++ version 6.0, you must install a patch to a standard header file, in order to successfully build applications for HLA 1.3 or 1516. The patch comes from Dinkumware, Ltd (the company that developed the Standard C++ Library header files for Microsoft), but for convenience, we are distributing it with VR-Link releases.

The patch fixes a bug that would cause applications to crash if they pass std::sets or std::maps from an application to a DLL or vice-versa. For details about the patch go to http://www.dinkumware.com/vc_fixes.html.

To install the patch:

1. In the top level VR-Link directory, is a file called *XTREE-msvc++patch*. Rename this file *XTREE*.
2. Make a back-up version of the original *XTREE*.
3. In the Visual Studio *include* directory (for example, *C:\Program Files\Microsoft Visual Studio\VC98\Include*), replace the original version of *XTREE* with the patched version.

Backward Compatibility

VR-Link 3.9.4 is highly compatible with VR-Link 3.9.1 and 3.9.3. For example, the *talk* and *listen* example programs from VR-Link 3.9.3 and previous versions compile and link unmodified against VR-Link 3.9.4.

Network Compatibility

HLA only

VR-Link 3.9.4 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 2.1
- DMSO RTI NG 4.0, and 6.0 (but not 5.0). The Linux version is not compatible with DMSO RTI NG.

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 3.9.4 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 3.9.4 uses RPR FOM 1.0.

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-Link20006", "MyApp", new DtRprFomMapper(2.0006));
```

VR-Link examples like *f18* and *hlaNetdump* have a command line option, `-f`, that you can use to choose an RPR FOM version by passing a RPR FOM keyword as the argument to `-f`. The keywords for specifying different versions of the RPR FOM with the `-f` startup option are:

- RPR FOM 0.5: `vlRPR05`
- RPR FOM 0.7: `vlRPR07`
- RPR FOM 0.8: `vlRPR08`
- RPR FOM 1.0: `vlRPR10`
- RPR FOM 2.0 draft 6: `vlRPR20006`
- RPR FOM 2.0 draft 14: `vlRPR20014`
- RPR FOM 2.0 draft 17: `vlRPR20017`.

RTI Support

VR-Link 3.9.4 has been tested with DMSO RTI NG v6, Pitch RTI 1.3, and MÄK RTI 2.2.

VR-Link 3.9.4 may work with older versions of the MÄK RTI, but we recommend using it with release 2.2.

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 3.9.4 has the following changes:

- The matrix library has been cleaned up and documentation improved
- Many preprocessor flags have changed
- The format for conditional compilation code has changed
- Environment objects and environment object transactions (interactions) implemented
- Change to the DtExerciseConn error status type.

Changes to the Matrix Library

- *mathTypes.h* has been deprecated. It is replaced with *vlMath.h*. The *mathTypes.h* header file is still included for backwards compatibility, but it may be removed in a future release.
- The following functions, which were never implemented, have been removed:
  - DtElrCopy ()
  - DtElrElrCat ()
  - DtElrForm ()
  - DtElrIdent ()
  - DtElrToMat ()
  - DtElrTranspose(t)
  - DtDcmToElr ()
  - DtVecElrTransform ()
- The following function has been removed from the matrix library:

```
#define ADD 1.0
#define SUBTRACT -1.0
DT_DLL_MATRIX void DtSum(const DtVector& a,const DtVector& b,DtReal
  s,DtVector& ans);
```

This has been replaced by various operators on *DtVector* like:

```
DtVector operator+(DtVector, DtVector);
DtVector operator-(DtVector, DtVector);

DtVector::operator+=(DtVector);
DtVector::operator-=(DtVector);
```

- The public data members in *DtCoordTransform* offset and rotation have been made protected. Code which previously used them should now use:

```
DtCoordTransform::rotation();
DtCoordTransform::offset();
DtCoordTransform::setOffset();
DtCoordTransform::setRotation();
```

- The following member functions, which were note referenced by any header files or source files, have been removed:
  - elr_form()
  - elr_copy()
  - elr_ident()
  - elr_to_mat()
  - elr_elr_cat()
  - elr_transpose().

The Header File *vlMath.h* has been Added to the Matrix Library

The file *vlMath.h* has been added to the matrix library. It replaces *mathTypes.h*. In a future release, the matrix library will be renamed *vlMath*. Some math constants have been moved from vlutil.h to *vlMath*. Previous versions of VR-Link removed duplicate definitions for mathematical constants. *vlMath* will be the central repository for mathematical constants and basic mathematical functions.

The following macros have been removed from *LibMatrix.h* and *vlutil.h*:

- SUM()
- MIN()
- MAX()
- ABS()
- LIMIT()
- SQUARE()
- SQR()
- DtSUM()
- DtMIN()
- DtMAX()
- DtABS()
- DtLIMIT()
- DtSQR.

The listed macros have been replaced by the following templated functions in *vlMath.h*:

- DtMin()
- DtMax()
- DtAbs()
- DtLimit()
- DtSquare().

Changes to Preprocessor Flags

The CPreProcessor Flags DtRPR=1 and DtDISVERS=5 are deprecated. In a future release, they will no longer be valid.

The FLAG DtDIS=1 has been added.

To build for DIS use DtDIS=1.

To build for HLA 1.3, use DtHLA=1.

To build for HLA 1516, use DtHLA=1 DtHLA_1516=1.

Changes to Conditional Compilation Code

In the past, code for conditional compilations has been written as follows:

```
#if DtHLA
// HLA Specific Code
#else
// DIS Specific Code
#endif
```

or:

```
#if !DtHLA
// DIS Specific Code
#endif
```

This code will continue to work after this release. However, in the future, MAK may choose to add a new protocol, which will cause this to break since the statement `!DtHLA` may not always refer to DIS. To ensure future compatibility with conditional compilation options, code should be written as follows:

```
#if DtHLA
// HLA specific code
#elif DtDIS
// DIS specific code
#else
#error "Please Choose  a protocol!"
#endif
```

or, for DIS specific code:

```
#if DtDIS
// stuff here
#endif
```

Environment Objects and Environment Object Transactions

Environment objects and environment object transactions (interactions), as defined in the RPR-FOM Draft 17, have been implemented.

The following objects and derived objects have been implemented:

- EnvironmentObject
- ArealObject
  - MinefieldObject
  - OtherArealObject.
- LinearObject
  - BreachableLinearObject
  - BreachObject
  - ExhaustSmokeObject
  - MinefieldLaneMarkerObject
  - OtherLinearObject
- PointObject
  - BreachablePointObject
  - BurstPointObject
  - CraterObject
  - OtherPointObject
  - RibbonBridgeObject
  - StructureObject

The following interactions and derived interactions were implemented:

- EnvironmentObjectTransaction
- ArealObjectTransaction
  - OtherArealObject
- LinearObjectTransaction
  - BreachableLinearObjectTransaction
  - BreachObjectTransaction
  - ExhaustSmokeObjectTransaction
  - OtherLinearObjectTransaction
- PointObjectTransaction
  - BreachablePointObjectTransaction
  - BurstPointObjectTransaction
  - CraterObjectTransaction
  - OtherPointObjectTransaction

– RibbonBridgeObjectTransaction
– StructureObjectTransaction

DtExerciseConn Error Status Type

The error status type, used to pass error status during construction of the *DtExerciseConn* has been changed from an int to DtExerciseConn::InitializationStatus, for example:

```
DtExerciseConn(const char* execName,
  const char* federateName,
  DtFomMapper* mapper = DtRprFomMapper::create(0),
  const char* fedFileName = 0,
  InitializationStatus* status = 0);
```

On successful construction of a *DtExerciseConn*, status will be assigned the value DtINIT_SUCCESS, for both HLA and DIS. The error codes have not changed, only how they are typed.

In previous versions of VR-Link DtINIT_SUCCESS was a macro, this has been changed to an enumerated value owned by the exercise connection.

Because this parameter defaults to null (meaning no error status is requested) only code which uses this status will be affected. This is a source compatability change, which might require minor code adjustments.

In the future we plan to add an exception mechanism to *DtExerciseConn*.

Miscellaneous Changes

- *DtlFileInputStream* no longer deletes the file pointer passed to it in the CTOR. This is consistent with the “you allocate it, you deallocate it” policy followed throughout VR-Link code.

Documentation Updates

The PDF version of *VR-Link Developer’s Guide* has been updated for this release.

Section 4.1.3 in *VR-Link Developer’s Guide* did not include link flags for HLA. The link flags are as follows:

Link flags for HLA 1.3:

```
-lvrhome/lib -lvlHLA13 -lmatrix -lmtl -lvlutil -lxml2
```

Link flags for HLA 1516:

```
-Lvrhome/lib -lvlHLA1516 -lmatrix -lmtl -lvlutil -lxml2
```

Bug Fixes

VR-Link 3.9.4 fixes the following bugs:

- In DIS only, when the ForceId is changed on an entityState object, the publisher forces an update. Previously, changes in ForceId were not made until the next heartbeat.

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 Environmental-Records. 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*.