MÄK VR-Forces 3.0-ngc Release Notes

This release note contains the following release-specific information for VR-Forces Release 3.0-ngc:

Document ID: VRF-3.0-3-020903

Systems Supported

VR-Forces 3.0-ngc is available for the platforms listed in Table 1. For the availability of other platforms, contact your MÄK salesperson. For toolkit users, application code must be built with the indicated compilers in order to link to VR-Forces libraries.

Table 1: Platforms supported

Platform	Compiler
Linux 7.2	GNU C++ (GCC) 3.0.2
PC with Windows NT 4.0 SP6 or Windows 2000 SP2	Microsoft Visual C++ 6.0, Service Pack 5

VR-Link and MÄK RTI Compatibility

To build VR-Forces applications, you must link with VR-Link 3.7.3-ngc. If you are building for HLA and want to link with the MÄK RTI, use MÄK RTI 1.3.7-ngc or later.

System Requirements for PCs

Minimum system requirements are:

- Microsoft Windows NT 4.0 SP6/Windows 2000 SP2/Windows XP
- Intel Pentium compatible CPU, 500 Mhz or better
- RAM: 128MB. 256-512 recommended. VR-Forces loads a copy of the terrain database for each front-end and back-end loaded.
- Hard disk space - Approximately 200 MB for complete installation of application and toolkit. Approximately 90 MB for application alone. Most of this is for the terrain databases shipped with VR-Forces. The class reference documentation for the toolkit uses 43 MB. You may need considerably more space depending on the databases you plan to use.
- Two or more multiples of the amount of RAM for virtual memory. Running VR-Forces in HLA may require increased amounts of virtual memory.

Online Help

The online help is in HTML format and uses the default browser for your computer. For all online help features to work, you must enable Javascript in your browser.

Network Compatibility

HLA only

VR-Forces 3.0-ngc was built against VR-Link 3.7.3-ngc and is compliant with:

- RPR-FOM 1.0 and 2.0
- MÄK RTI 1.3.7-ngc
- DMSO RTI NG v3.0, v3.1, v3.2, 4.0, and 6.0.

VR-Forces 3.0-ngc is compatible with applications that use earlier versions of VR-Link if they support versions of the RPR FOM listed above.

DIS only

VR-Forces 3.0-ngc 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.

Backward Compatibility

VR-Forces 3.0-ngc has made significant changes to the API. Programs built with the release 2.x API will need some porting to be compatible with release 3.0. Scenarios created with release 2.x are compatible with release 3.0-ngc.

RPR FOM Versions Supported

VR-Forces 3.0-ngc has built-in support for versions 1.0 and 2.0 (draft 6) of the RPR FOM. It also supports VR-Link's ability to support alternative FOMs through the FOM Mapper. By default, VR-Forces 3.0-ngc uses RPR FOM 1.0.

If you are building an application with the VR-Forces toolkit and you want to specify a version of the RPR FOM through code, pass the version number (2.0) 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-Link20", "MyApp", new DtRprFomMapper(2.0));
```

RTI Support

The MÄK RTI is included on all MÄK product CDs. It is link-compatible with the DMSO RTI NG. VR-Forces 3.0-ngc has been tested with DMSO RTI NG 4.0 and 6.0.

If you use the MÄK RTI, be sure that VR-Forces can find your FED file, and optional *rid.mtl* file, by putting them in the directory from which you are running, or by setting the environment variable RTI_CONFIG to the directory that contains them.

New Features and Updates

This release of MÄK VR-Forces has the following updates and new features:

- Completely new graphical user interface (front-end) that uses the Qt toolkit
- Significant improvements to the toolkit architecture, making it easier to:
  - Embed the simulation engine in applications
  - Extend the VR-Forces back-end
- Significant improvements in performance
- A new remote control API for controlling the back-end from control applications
- New models for individual lifeforms (dismounted infantry and civilians)
- Updated documentation.

New Graphical User Interface

VR-Forces release 3.0-ngc features a completely redesigned graphical user interface (GUI) with enhanced usability features, including:

- A Minimap for quickly selecting the portion of the terrain database you want to view.
- Removable and dockable toolbars.
- Tear-off menus for creating entities and assigning tasks and set commands.
- Support for localization using Qt Linguist.
- Ability to resize entity icons and labels
- Ability to configure the color of interface features such as grid lines, contour lines, and grid line labels.

All new features and changes are described in *MÄK VR-Forces User's Guide* and online help, which has been completely updated for this release.

This release does not include documentation or source files for the Graphical User Interface API. If you have immediate need of this material, please contact support@mak.com.

Changes in GUI Behavior from Previous Releases

This section lists some differences in behavior between the new VR-Forces GUI and previous releases:

- If you run multiple VR-Forces front-ends and back-ends you have some responsibility for coordinating their availability for creating, loading, and saving scenarios. See Chapter 3, "VR-Forces Concepts", in *MÄK VR-Forces User's Guide* for details.
- The behavior for creating control objects has changed slightly. In previous versions of VR-Forces (versions 2.x and older), when you created a new control object by right-clicking in the terrain and choosing `Create` *`control-object`* from a popup menu, the point at which you initially clicked in the terrain either became the location of the object (in the case of a waypoint), or became the location of the first point in a sequence of points (as in the case of routes, phase lines, and areas). In the new GUI, the point at which you right-clicked is no longer used in the positioning of the object. You must now explicitly left-click to start placing the one or more points that make up the control object.
- The `-B` command line option does not apply to the GUI. You use it only with the back-end.
- Aggregates and their subordinates are displayed simultaneously in the front-end (expand and collapse is not supported). When you select a group of entities to task in the GUI, do not include an aggregate and its subordinates in the same selection. Tasking an aggregate and one or more of its subordinates simultaneously may result in undefined behavior.

Updated Simulation API

VR-Forces 3.0 includes a variety of improvements to the top-level VR-Forces simulation API to make it easier to embed VR-Forces functionality into other applications, or customize or extend the simulation engine. Most of the changes are related to how applications initialize VR-Forces and instantiate the objects that manage the simulation as a whole. Chapter 1 of *MÄK VR-Forces Developer's Guide* and the updated examples in *./examples* provide details of the new classes you need to use to initialize and manage VR-Forces-based applications. Lower-level implementation classes like those that implement components, plans, parameters, messages, tasks, and so on, have remained largely unchanged, and there is a high degree of backwards compatibility with previous releases.

For those upgrading from pre-3.0 versions of VR-Forces, it is no longer necessary to instantiate *DtVrfCreator* or *DtSimManager* within your VR-Forces applications. Those classes still exist, but their importance has diminished. Their APIs and purposes have changed significantly, so changes to application code will be necessary. It is also no longer necessary to subclass *DtVrfCreator* to register custom subclasses of VR-Forces classes like *DtSimComponent*, *DtSimTask*, and so on, with VR-Forces.

Under the new scheme, a VR-Forces-based application instantiates the new *DtCgf* class to create a simulation engine object, and then ticks() that *DtCgf* object each frame to cause the simulation to occur. *DtVrfCreator* is now used only within *DtCgf* to initialize some top-level manager classes like *DtVrfObjectManager*, *DtPlanManager*, and so on. *DtSimManager* is still used by *DtCgf* to schedule events and timers, but many applications will never need to know that *DtSimManager* even exists. The new method for registering custom subclasses for components, tasks, messages, and so on, is to register a creator function with the appropriate factory. Factories are accessible through *DtCgf's DtFactoryManager*.

The code for the vrfSim application and other API examples has been greatly simplified. They now use a class called *DtVrfApp* to handle command-line options, and initialize and tick the *DtCgf* object for you. We include source code for *DtVrfApp* as a model for you in creating your own applications or extending vrfSim.

Again, see *MÄK VR-Forces Developer's Guide* and *./examples/** for details.

Changes to the GUI/Simulation Engine Communications

In VR-Forces 2.x, the vrfGui application played a larger role in loading and initializing scenarios. The front-end would load a scenario file, and send individual messages to relevant simulations engines asking them to instantiate each entity in turn, assign plans, and so on. In VR-Forces 3.0, the job of loading scenarios is handled mainly in the simulation engines. When you choose Load Scenario in the GUI (or call loadScenario() from the Remote Control API), the front-end merely breaks the order of battle and plan files into sub-files describing the simulation responsibilities of each simulation engine. It then sends those subfiles to the relevant back-ends, where they are read by the simulation engines. Saving scenarios works similarly. Simulation engines save individual OOB and plan files, and send those files to the control application (for example, vrfGui) that asked for the save. That control application merges the files sent by the various back-ends, and saves the resulting OOB and Plan files for the scenario as a whole.

Improved Performance

VR-Forces has made significant changes to the simulation engine to improve performance. It also now includes a variable tick rate parameter that you can set in the object parameters database on a per-entity or per-component basis. For information about setting the variable tick rate, see Chapter 10, "Advanced Configuration", in *MÄK VR-Forces User's Guide*.

VR-Forces' performance will vary based on the following:

- Size of the terrain database
- Amount of memory available on your computer
- Computer speed and processing power
- Number of objects (entities, control objects) in the scenario and how active their plans are
- The number of objects you try to simulate on a given back-end
- The frequency of the tick rate that you set
- The number of front-ends and back-ends you run on your computer.

Miscellaneous Toolkit Changes

This section highlights changes to the toolkit that you should be aware of:

- The Status Manager has been replaced by the Status Publisher in the back-end and the Status Listener in the front-end. VR-Forces back-ends no longer need to be aware of other back-ends. A back-end now publishes its presence to the network using the *DtVrfStatusPublisher* class. The GUI learns about VR-Forces back-ends through its *DtVrfStatusListener* class.
- *DtVrfMessageInterface* (*vrfMsgIf.h*) replaces the *DtSimManagerMessageInterface* class.

Remote Control API

The MÄK VR-Forces Remote Control API is a set of classes that allow an application to easily control one or more remote VR-Forces applications. The MÄK VR-Forces GUI (the vrfGui executable) uses the VR-Forces Remote Control API to control the vrfSim application or other VR-Forces applications. The Remote Control API can also be used in custom VR-Forces front-ends, simulation managers, or instructor/operator stations.

The Remote Control API contains no GUI-related code. You can use it as easily in an application that uses scripts or command-line input as you can in a GUI-based application. You can use it in any application that needs to control remote VR-Forces simulation engines.

MÄK VR-Forces Developer's Guide describes the remote control API.

New Lifeform Models

Civilians and dismounted infantry have new models that more accurately reflect the behavior and movement of people. The dismounted infantry entities have new weaponry available. You can set the posture of a lifeform. Posture controls the static position (standing, kneeling, prone) and the movement type (walking, running, crawling).

New Command Line Options

The following command line options are not documented in *MÄK VR-Forces User's Guide*:

- `-r`: If you start the back-end without the `-L` (load scenario file) or `-T` (load terrain file) options, the back-end starts in pause mode. To override this behavior and start in run mode, use the `-r` command line argument.
- `-e sleep_time`: The `-e sleep_time` argument lets you control the duration of the sleep interval (a call to VR-Link DtSleep()) that the back-end takes each tick in the main simulation loop. Calling DtSleep periodically in the back-end allows the process to share the CPU with any other processes currently running. By default, the sleep interval is 0.01 seconds. Under Windows, specifying a sleep interval of 0 on the command line results in a call to the Win32 sleep(0) command instead of DtSleep(). The result is that the CPU will be released to those processes with a priority equal to or greater than the back-end process.

Documentation

All MÄK documentation has been updated for this release.

Installing CGM Symbols

The CGM files in *./data/symbols/cgm* are in a zip file. You must unzip them before you can use them.

Bug Fixes

This release fixes the following problems:

- Late-joining front-ends show the list of back-ends.
- DIS and HLA now use the same default site and application numbers, so you can load example scenarios in either protocol when you are not in combined mode.
- Front-ends that did not create a scenario can manipulate entities and objects in it.
- When you run with multiple front-ends, they all update the state of the buttons in the simulation toolbar.

Known Problems

VR-Forces Release 3.0-ngc has the following known problems:

- This release does not support paper map (.map) files.
- This release does not support line-of-sight calculations in the GUI.
- If you load a FLT file, then close it and load another FLT file, the back-end will crash.
- Aggregates composed of fixed-wing entities do not carry out tasks as an aggregate.
- If you are using a DTED database and "Projected False" is specified in his PvDted-MetafileRecord, entities do not execute movement tasks.
- The *world.gdb* file is based on a DTED database. If you use this database, some entities may not execute tasks. Some entities will not display properly.
- When you aggregate a set of aggregates into a higher-echelon aggregate using the "Aggregate As" menu item, make sure that the aggregates being combined have been collapsed to their highest level. Failure to do so will aggregate the entities at the level they are currently displayed. For example, suppose that you want to aggregate two teams of two tanks each into a platoon. If at the time of the "Aggregate As" operation, one of the teams is expanded, displaying its constituent tanks, while the other team is collapsed, displaying a single team icon, the resulting platoon will be composed of two individual tanks and a team, instead of two teams.
- Surface entities are not configured to fire weapons or be fired upon. You can customize the object parameters database entries to enable this.
- Joysticks do not work correctly with DirectInput.
- The pseudo-aggregate formation controller ignores entities in a formation that are independently tasked when determining if the promotion order needs to be updated. Currently it has no way to determine if remote entities are independently tasked. One possible solution might be to listen to task and task complete messages for all the subordinates in order to maintain a local picture of what is going on.
- Uninstalling VR-Forces in Windows via InstallShield uninstalls the entity fonts. If you have another application that uses these fonts, such as the MÄK Plan View Display, the correct entity icons would not be displayed. You must manually install the fonts from their location in the other application that needs them.