MÄK VR-Forces 2.1.2-ngc Release Notes

This release note contains the following release-specific information for VR-Forces Release 2.1.2-ngc:

Document ID: VRF-2.1.2-3-020627

Systems Supported

VR-Forces 2.1.2-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
SGI IRIX6.5	MipsPro7.2.1 C++ compiler (available from SGI)
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 3.7.2-ngc. If you are building for HLA and want to link with the MÄK RTI, use MÄK RTI 1.3.6-ngc.

System Requirements for PCs

Minimum system requirements are:

- Microsoft Windows NT 4.0 SP6/Windows 2000/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 160 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 59 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 Java and Javascript in your browser.

Network Compatibility

HLA only

VR-Forces 2.1.2-ngc was built against VR-Link 3.7.2-ngc and is compliant with:

- RPR-FOM 1.0 and 2.0
- MÄK RTI 1.3.6-ngc
- DMSO RTI NG v1, v2, v3.0, v3.1, v3.2, and 4.0.

VR-Forces 2.1.2-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 2.1.2-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.

RPR FOM Versions Supported

VR-Forces 2.1.2-ngc has built-in support for versions 1.0 and 2.0 of the RPR FOM. It also supports VR-Link's ability to support alternative FOMs through the FOM Mapper. By default, VR-Forces 2.1.2-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 2.1.2-ngc has been tested with DMSO RTI NG v3.2, and 4.0.

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.

RTI Library Names

VR-Forces 2.1.2-ngc expects library names that match those of DMSO RTI NG v4. If you run VR-Forces 2.1.2-ngc with DMSO RTI NG v4, both the UNIX and PC versions work without any modification. If you want to run VR-Forces 2.1.2-ngc with DMSO RTI NG v1 or v2, you must make the following changes:

- For UNIX, you must use the libraries *libRTI-NGv2.so* or *libRTI-NGv1.so*, which are supplied by MÄK in the *./lib* directory. These libraries correspond to DMSO RTI NG v1 and v2. To use VR-Forces 2.1.2-ngc with RTI NG v1 or v2, copy the appropriate library to the DMSO RTI NG *lib* directory and rename it *libRTI-NG.so*. (You might want to rename the v3.x *libRTI-NG.so* library first.)
- For PCs, rename the libraries that come with RTI NG v1 or v2 to the names used by the RTI NG v4 libraries (*libRTI-NG.lib* and *libfedtime.lib*).

New Features and Updates

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

- Support for session IDs that determine which front-ends can control which back-ends, and allow you to load different terrain databases for the same exercise.
- Support for a new emitter component that publishes electromagnetic emissions to the network.

VR-Forces Sessions

When you start VR-Forces applications, you can specify a session ID. Session IDs provide the following benefits:

- They allow you to specify which front-ends control which back-ends. This feature could be useful if you want one set of VR-Forces applications to simulate friendly forces and the other set to simulate opposing forces. If all applications were in the same session, participants could send orders to the entities in their opposing force. By using separate sessions, this is not possible.

  In Figure 1, the front-end with session ID 1 can control the back-ends with that same session ID. The front-end with session ID 2, controls the back-end with session ID 2.

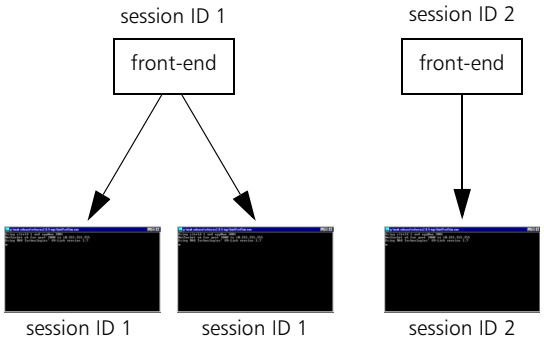

Figure 1. Using session ID to control back-ends

- They allow you to load different terrain databases in different applications that are part of the same scenario and exercise. This feature may be helpful if you are participating in an exercise that covers a large geographical area, but only need to observe portions of that total terrain. Loading databases that cover a subset of the total exercise zone can improve performance of your computer.

  In Figure 2, the applications with session ID 1 have loaded the monterey database, while the applications with session ID 2 have loaded the hunter database. These two databases cover different portions of California, USA. The entire exercise would cover the terrain covered by two databases.

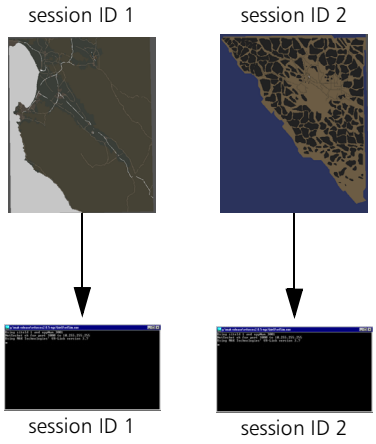

Figure 2. Using session IDs to load different terrain databases

Specifying the Session ID

To specify the session ID, use the `-i` *`session_ID`* startup option, for example:

```
vrfGui.exe -s 1 -a 2 -i 25
vrfSim.exe -s 1 -a 1 -i 25
```

The default session ID is 1.

The Emitter Component

The Emitter Component is a sensor component that mounts on an entity and models basic electromagnetic emissions. An entity can have multiple emitter components. Each emitter component can have multiple beams. This section describes the emitter component from the end-user viewpoint (that is, in the front-end.)

Emission properties (azimuth, elevation, effective radiated power, and so on) for each beam are specified in the emitter component descriptor in the object parameters database. Emission properties are published by the emitter component.

You can turn emissions for a given emitter on and off with the Set Emitter command. When you set an emitter on or off, you must specify its system ID. The system ID is specified by the **system-id** parameter in the object parameters database. The Set Emitter dialog box (Figure 3) does not have a list of system IDs. You must know this information before you set emissions on or off.

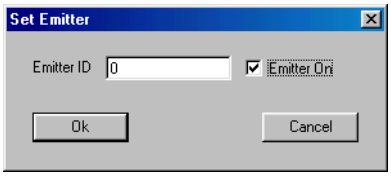

Figure 3. Set Emitter dialog box

An emitter component can publish itself as a standard beam type or a tracking beam type. The type is determined by the setting for the **targeting-capable** parameter in the object parameters database. It applies to all beams emitted by the specific component. If **targeting-capable** is False, the beam is a standard beam. If **targeting-capable** is True, it is a tracking beam. If an emitter component has **targeting-capable** set to True, it registers for set-target messages. If you set a target for the entity on which the emitter is mounted, it publishes the target entity as its tracking target for all of the components beams.

Notes

- Setting a target does not cause the beam to actually track the target. The beams behavior is determined by the parameters in its description in the object parameters database.
- The emitter will continue to publish the targeted entity as its tracking target until you set another target, regardless of the status of that entity in the exercise. There is no way to set the target to “none”.

The emitter component description is as follows:

component-type:	emitter-component
component-descriptor-type:	emitter-component-descriptor
Attaches to:	None.
Mounts on:	Any entity that specifies a vrf-moving-object-state-repository or a state repository derived from vrf-moving-object-state-repository.
Port interfaces:	None.
Model:	Simple multi-beam emitter system.
Parameters:	Used to set beam parameters. See Table 2. (The beam list parameters are as described in IEEE 1278.1, section 5.2.22)

Table 2: Emitter component parameters

Parameter	Description
system-id	Integer ID of this system. Multiple systems on one entity must have unique system-ids.
system-name-enum	System name enumeration. (See the DIS Enumerations Document.)
system-function-enum	System function enumeration. (See the DIS Enumerations Document.)
mount-location	Emitter system offset from entity in meters. (X Y Z)
on	Emitter system on at startup. (True/False)
targeting-capable	Emitter system will set target to entities specified in "Set Target" messages. (True/False)
beam-list	List of parameters for each beam.
frequency	Center frequency of emission, in Hz.
frequency-range	Frequency range around center, in Hz.
azimuth-center	Azimuth center of emission, in radians.
azimuth-sweep	Azimuth range around center, in radians.
elevation-center	Elevation center of emission, in radians.
elevation-sweep	Elevation range around center, in radians.
pulse-repetition-frequency	Pulse repetition frequency, in Hz.
pulse-width	Pulse width, in milliseconds.

Table 2: Emitter component parameters

Parameter	Description
sweep-sync	Sweep sync, as a percentage.
power	Effective radiated power, in dBm.
beam-function-enum	Beam function enumeration. (See the DIS Enumerations Document.)
beam-param-index	Beam parameter index.

A new entity, Emitting F/A-18 Hornet, has been added to the object parameters database. The emitter component description is as follows:

```
;; Entry for emitting F18 -- provided as an example of the use
;; of the emitter-component.
;;
(US-fighter-emitting-air-defense-fixed-wing-platform
   (parameter-type "fixed-wing-entity-param")
   (object-type 1 (1 2 225 1 9 1 1))
   (sensors
      (emitter-component
         (component-descriptor-type "emitter-component-descriptor")
         (component-type "emitter-component")
         (system-id 0)
         (system-name-enum 1)
         (system-function-enum 2)
         (mount-location 0.0 0.0 0.0)
         (on True)
         (targeting-capable True)
         (beam-list
            (beam-0
               (frequency 8999999488.0)
               (frequency-range 5000.)
               (azimuth-center 0.0)
               (azimuth-sweep 0.1047000)
               (elevation-center 0.349066)
               (elevation-sweep 0.523599)
               (pulse-repetition-frequency 0)
               (pulse-width 0)
               (sweep-sync 0)
               (power 70.)
               (beam-function-enum 0)
               (beam-param-index 0)
            )
         )
      )
   )
)
```

To configure multiple emitters for an entity, add additional emitter-component blocks. Each component block must have a unique name and each **system-id** parameter must be unique. For example:

```
(emitter-component-1
   (component-descriptor-type "emitter-component-descriptor")
   (component-type "emitter-component")
   (system-id 0)
   ...
)
(emitter-component-2
   (component-descriptor-type "emitter-component-descriptor")
   (component-type "emitter-component")
   (system-id 1)
   ...
)
```

To add multiple beams on an emitter, add additional beam-list blocks and change the name of each block. For example:

```
(US-fighter-emitting-air-defense-fixed-wing-platform
   (parameter-type "fixed-wing-entity-param")
   (object-type 1 (1 2 225 1 9 1 1))
   (sensors
      (emitter-component
         ...
         (beam-list
            (beam-0
               ...
               (beam-param-index 0)
            )
            (beam-1
               ...
               (beam-param-index 0)
            )
         )
      )
   )
)
```

DtEmitterComponent

This section and the following section describe the *DtEmitterComponent* for VR-Forces toolkit users.

The *DtEmitterComponent* models electromagnetic emissions from a sensor mounted on an entity. Each emitter component represents a single emitter system which can have one or more emitter beams associated with it. When the emitter component is turned on, all data describing the emitter beams is published to the network. When the emitter component is turned off, the existence of the emitter system is still published, however no beam data is sent.

The *DtEmitterComponentDescriptor* specifies all the emitter system and emitter beam parameters including frequency, azimuth, elevation, and so on. See the "The Emitter Component" on page 6 for a complete list of parameters.

DtEmitterComponent provides a simple targeting capability, which can be optionally turned on in the component descriptor. When enabled, the emitter component registers for *DtSetTargetRequest* messages. When the component receives a *DtSetTargetRequest* message, it sets each beam in the emitter system to target the specified entity by filling in the track/jam field in the beam parameters.

DtEmitterComponent registers for *DtSetEmitterRequest* messages, which allow turning on and off the emitter system at runtime.

The *DtEmitterComponent* represents the emissions from generic sensor systems. Therefore it does not do any calculations required to sense other entities in the environment, because those calculations are specific to the sensor type. This component provides an API so that a component derived from it can do the appropriate calculations and make use of this component for providing emissions data.

DtSetEmitterRequest

The *DtSetEmitterRequest* is a request for an entity with an emitter component to turn the specified emitter either on or off. Since an entity can have multiple emitter components, the emitter system ID must be specified in the message.

set-data-request-type:	DtSetEmitterRequestType
parameters:	emitter-id – Integer value specifying which emitter component on this entity should receive the message.
	system-on - True/False indicating whether the emitter is to be turned on or off.

Bug Fixes

This release fixes the following problems:

- The missile icon flashed briefly at the origin of the database when a missile was fired.
- Larger scenarios, such as Berkeley scenario provided with VR-Forces, no longer exceed the capacity of the buffers on the network socket when running the HLA version of VR-Forces.

Known Problems

VR-Forces Release 2.1.2-ngc has the following known problems:

- The Go To Next and Go To Previous options on the Map Navigation menu do not work with aggregates. If you have a mixture of aggregates and individual entities, using these menu options works fine if there is a series of contiguously listed individual entities. Once the selection moves to an aggregate (11:1:0 ...) the entity list window loses a selection and the view in the Simulation View window doesn't change.

- Aggregates composed of fixed-wing entities do not carry out tasks as an aggregate.
- The Windows and IRIX versions of VR-Forces do not work together.
- If you are using VR-Forces on a multi-processor computer, set the affinity to single-processor. VR-Forces does not currently take advantage of multiple processors.
- 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.
- Batch scenarios do not automatically record to MÄK Logger files.
- 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.
- Phase line labels are displayed in the middle of the phase line rather than the beginning. To determine which side of a phase line is "left" for the purpose of the EntLeftOfLine condition, you will need to remember where you started each phase line.
- Surface and lifeform entities are not configured to fire weapons or be fired upon. You can customize the object parameters database entries to enable this.
- Repeated attempts to toggle the intervisibility fan may result in multiple intervisibility fans displayed on top of one another. It will not turn them off.
- The waypoint icon is currently being displayed centered on its location. According to MIL-STD-2525B it should be positioned with the lower tip of the positioned at the waypoint's location.
- Stealth control using the Stealth Control dialog box from VR-Forces does not work well. (The Stealth Attach and Teleport Stealth commands work.)
- Joysticks do not work correctly with DirectInput.
- If a front-end or back-end tries to load a FLT file more than once, it will crash.
- If you run multiple front-ends, the "extra" front-ends do not show the run or pause buttons depressed when the simulation state changes.
- The Berkeley Demo recording provided with the MÄK Logger does not work with this release of VR-Forces.
- 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.

- The list of parameters for *DtIfCreateVrfObject* on page C-8 of MÄK VR-Forces Developer's Guide is incorrect. The correct list of parameters is in the Documentation section of this release note.
- Front-ends that did not initiate a scenario cannot use canvas selections to complete dialog boxes, for example, selecting a waypoint for a move to waypoint task.
- You cannot add a right-click menu item with the plug-in interface.
- The online help may not work with Netscape 6 or later. If the contents does not display, try clicking the contents button after the introductory topic finishes loading.