MAK TECHNOLOGIES

VR-Vantage Migration Guide

Version 3.1

ST Engineering North America

MAK Technologies
10 Fawcett Street, Suite 204
Cambridge, MA 02138 USA

Voice: +1 617.876.8085
Fax: +1 617.876.9208

info@mak.com

www.mak.com

Revision VRV-3.1-8-240129

Contents

Preface

This manual is for developers who must migrate applications from previous versions of VR-Vantage to the current version. It is not intended to be fully comprehensive, but it does cover many of the significant items that must be migrated.

VR-Vantage Documentation

VR-Vantage documentation is provided as manuals in PDF format, online help, and HTML class documentation. The PDF files are in the *./doc* directory. The VR-Vantage documentation set is as follows:

- *VR-Vantage Users Guide* contains all documentation for persons who will install VR-Vantage and use it to view simulations. The manual assumes that you are familiar with basic administrative tasks and the graphical window environment for your operating system.
- *VR-Vantage Migration Guide* collates the most substantial migration details for recent releases, targeted at developers.
- Online help. VR-Vantage has online help accessible from the Help menu.
- *VR-Vantage Developers Guide* and API documentation. Class documentation and developers guides in linked HTML pages.
- *Adding Content to MAK Applications Reference Manual* explains how to add visual models and terrain to VR-Vantage.
- *VR-Vantage Release Notes*.
- *VR-Vantage First Experience Guide* is a brief introduction to the most basic features of VR-Vantage.

MAK Products

The MAK Technologies product line includes the following:

- VR-Link® Network Toolkit. VR-Link is an object-oriented library of C++ functions and definitions that implement the High Level Architecture (HLA) and the Distributed Interactive Simulation (DIS) protocol. VR-Link has built-in support for the NETN FOM and RPR FOM and allows you to map to other FOMs. This library minimizes the time and effort required to build and maintain new HLA or DIS-compliant applications, and to integrate such compliance into existing applications.

  VR-Link includes a set of sample debugging applications and their source code. The source code serves as an example of how to use the VR-Link Toolkit to write applications. The executables provide valuable debugging services such as generating a predictable stream of HLA or DIS messages, and displaying the contents of messages transmitted on the network.
- MAK RTI. An RTI (Run-Time Infrastructure) is required to run applications using the High Level Architecture (HLA). The MAK RTI is optimized for high performance. It has an API, RTIspy®, that allows you to extend the RTI using plug-in modules. It also has a graphical user interface (the RTI Assistant) that helps users with configuration tasks and managing federates and federations.

- VR-Forces®. VR-Forces is a computer generated forces application and toolkit. It provides an application with a GUI, that gives you 2D and 3D views of a simulated environment.

  You can create and view entities, aggregate them into hierarchical units, assign tasks, set state parameters, and create plans that have tasks, set statements, and conditional statements. You can simulate using entity-level modeling, which focuses on the actions of individual people and vehicles, and aggregate-level modeling, which focuses on the interaction of large hierarchical units.

  VR-Forces also functions as a plan view display for viewing remote simulation objects taking part in an exercise. Using the toolkit, you can extend the VR-Forces application or create your own application for use with another user interface.

- VR-Vantage™. VR-Vantage is a line of products designed to meet your simulation visualization needs. It includes VR-Vantage and the VR-Vantage Toolkit.

  - VR-Vantage displays a realistic, 3D view of your virtual world, a 2D plan view, and an exaggerated reality (XR) view. Together these views provide both situational awareness and the big picture of the simulated world. You can move your viewpoint to any location in the 3D world and can attach it to simulation objects so that it moves as they do.
  - SensorFX. SensorFX is an enhanced version of VR-Vantage that uses physics-based sensors to view terrain and simulation object models that have been materially classified. It is built in partnership with JRM Technologies.
  - The VR-Vantage Toolkit is a 3D visual application development toolkit. Use it to customize or extend MAK's VR-Vantage applications, or to integrate VR-Vantage capabilities into your custom applications. VR-Vantage is built on top of OpenSceneGraph (OSG). The toolkit includes the OSG version used to build VR-Vantage.

- MAK Data Logger. The Data Logger, also called the Logger, can record HLA and DIS exercises and play them back for after-action review. You can play a recorded file at speeds above or below normal and can quickly jump to areas of interest. The Logger has a GUI and a text interface. The Logger API allows you to extend the Logger using plug-in modules or embed the Logger into your own application. The Logger editing features let you merge, trim, and offset Logger recordings.

- VR-Exchange™. VR-Exchange allows simulations that use incompatible communications protocols to interoperate. For example, within the HLA world, using VR-Exchange, federations using the HLA RPR FOM 1.0 can interoperate with simulations using RPR FOM 2.0, or federations using different RTIs can interoperate. VR-Exchange supports HLA, TENA, and DIS translation.

- VR-TheWorld™ Server. VR-TheWorld Server is a simple, yet powerful, web-based streaming terrain server, developed in conjunction with Pelican Mapping. Delivered with a global base map, you can also easily populate it with your own

custom source data through a web-based interface. The server can be deployed on private, classified networks to provide streaming terrain data to a variety of simulation and visualization applications behind your firewall.

- DI-Guy™. DI-Guy is an SDK that allows you to embed the DI-Guy library in your real-time application and populate your world with lifelike human characters. DI-Guy provides high-fidelity human characters that move realistically, respond to simple high-level commands, and travel throughout the environment as directed by the hosting visual application. It comes with thousands of ready-to-use characters, appearances, and motions. DI-Guy character simulation is an integral part of all MAK's visual applications, including VR-Engage, VR-Forces, and VR-Vantage.
- RadarFX. RadarFX is a client-server application that simulates synthetic-aperture radar (SAR). The server application, which is based on VR-Vantage and SensorFX, loads a terrain database and, optionally, connects to simulations. A client application requests SAR images from the server. RadarFX includes a sample client application.
- VR-Engage. VR-Engage is a multi-role virtual simulator that lets users play the role of a first person human character, a ground vehicle driver, gunner or commander, a sensor operator, or the pilot of a fixed wing aircraft or helicopter. It incorporates the VR-Force simulation engine and the VR-Vantage graphics rendering capabilities.
- WebLVC Server. WebLVC Server implements the server side of the WebLVC protocol so that web-based simulation federates can participate in a distributed simulation. It is part of the WebLVC Suite, which includes the server and several sample applications that work with VR-Forces and VR-Vantage.
- MAK Legion. MAK Legion is a collection of applications and an SDK designed to support simulations with millions of simulation objects.

How to Contact Us

For VR-Vantage technical support, information about upgrades, and information about other MAK products, you can contact us in the following ways:

Telephone

Call or fax us at:	Voice:	617-876-8085
	Fax:	617-876-9208

E-mail

Sales and upgrade information:	info@mak.com
Technical support:	support@mak.com

Internet

MAK website home page:	www.mak.com
License key requests:	www.mak.com/support/get-licenses
Technical support:	www.mak.com/mak-one/support/portal

Product version and platform information:	www.mak.com/support/product-versions
For the free, unlicensed MAK RTI:	www.mak.com/support/bonus-material

Post

Send postal correspondence to:	MAK Technologies 10 Fawcett Street, Suite 204 Cambridge, MA 02138 USA

When requesting support, tell us the product you are using, the version, and the platform on which you are running. If you are using VR-Vantage, VR-Forces, VR-Engage, or DI-Guy, please also include your data compatibility version and date.

Document Conventions

This manual uses the following typographic conventions:

Sans Serif Bold	Indicates a key on the keyboard or menu options. Also indicates parameters.
`Monospaced`	Indicates values you enter such as commands or arguments.
`Monospaced Italic`	Indicates command variables that you replace with appropriate values.
{}	Indicates required arguments.
[]	Indicates optional arguments.
|	Separates options in a command where only one option may be chosen at a time.
`(	)`	In command syntax, indicates equivalent alternatives for a command-line option, for example, `(-h | --help)`.
/	Indicates a directory. Since MAK products run on both Linux and Windows PC platforms, we use the / (slash) for generic discussions of pathnames.
Italic	Indicates a file name, pathname, or a class name.
➤	Indicates a procedure.
Menu > Option	Indicates a menu choice. For example, an instruction to select the Save option from the File menu appears as: Choose **File > Save**.
	Indicates supplemental or clarifying information.
	Indicates additional information that you must observe to ensure the success of a procedure or other task.

Directory names are preceded with dot and slash characters that show their position with respect to the VR-Vantage home directory. For example, the directory *./vrvantage3.1/doc* appears in the text as *./doc*.

For anything in the data directory, the path is shown starting with *./SharedData/##/latest* to be clear that it is not in the VR-Vantage home directory. That is the default installation path. If you installed the data in a different directory, substitute your path.

Mouse Button Naming Conventions

An instruction to click the mouse button refers to clicking the primary mouse button, usually the left button for right-handed mice and the right button for left-handed mice. The context-sensitive menu, also called a popup menu or right-click menu, refers to the menu displayed when you click the secondary mouse button, usually the right button on right-handed mice and the left button on left-handed mice.

1. Migration to VR-Vantage 2.6

This chapter describes migration issues from previous versions of VR-Vantage to 2.6.

1.1. osgEarth File Updates

VR-Vantage nows supports osgEarth 3.0 and all earth files included with MAK products have been updated to the new format. osgEarth 3.0 still loads most osgEarth 2.0 earth files through its backwards-compatibility code.

Some osgEarth 2.0-style data does not load in osgEarth 3.0, in particular anything relating to the Procedural Terrain (osgEarthSplat) subsystem.

The earth file (.earth) format has evolved and there are new standards for working with them. This section describes the changes. If you have your own earth files, then you may need to make these edits.

1.1.1 Remove max_level from Image Layers

The `max_level` property on an image layer dictates the highest terrain level of detail (LOD). At LODs above this value, the layer should not render at all. There was a bug in earlier versions of osgEarth that caused the `max_level` property on an image layer to be ignored sometimes.

In earth files that use the `agglite` driver, if the `max_level` property is set, the layer disappears when the observer zooms in. The solution is to replace the `max_level` property in image layers with the `max_data_level` property.

The `FeatureImage` layer type replaces the `agglite` driver. See "1.1.8 Image and Elevation Layers" on page 14.

1.1.2 Remove attenuation_distance and Add attenuation_range

Attenuation controls the distance over which to blend an image layer from opaque to transparent. VR-Vantage uses this to blend imagery with procedural splatting over a camera distance. In older versions of osgEarth, the `attenuation_distance` property existed in the global terrain options and applied to all image layers. In osgEarth 3.0, this property is gone, and there is a new `attenuation_range` property on the image layer.

Remove the `attenuation_distance` property from the global terrain options, and then add the `attenuation_range` property to the image layer(s) you want to blend.

1.1.3 Replace high_resolution_first with progressive

By default, the terrain engine prioritizes the paging of high-resolution tiles. Earlier versions of osgEarth have a `high_resolution_first` property to control this to some degree. That property has been replaced with the simpler `progressive` property, which forces terrain to page in one LOD at a time.

In your files, replace the `high_resolution_first` property with the `progressive` property and set the value to “true” to force low resolution terrain tiles to page in first.

1.1.4 Disable Instancing on Light Point Layers

For point-model substitution layers, the `instancing` property used to default to false, and now it defaults to true. The VR-Vantage lighting framework does not work when instancing is enabled on these model layers.

In your light point layers, add the property: `<instancing>false</instancing>`.

1.1.5 Replace marker with model (Optional)

The `marker` symbol no longer exists in osgEarth 3.0. It was replaced by the separate `model` and `icon` symbols.

For completeness, you can replace any `marker` symbols with `model` symbols. VR-Vantage has been updated to automatically convert `marker` symbols to `model` symbols on load, so it should work without any manual edits.

1.1.6 Add disable_tiling to WFS features block and Set to true

Previously, WFS feature sources allowed you to access them as pre-tiled or untiled sources transparently. This functionality is no longer available by default.

If you have a feature layer that points to a pre-tiled WFS resource on VR-TheWorld and describes a custom tiling scheme (layout and levels), then you need to add the backwards-compatibility property `disable_tiling` to the WFS features block and set it to “true”, for example:

```
 <model name="model-forests"
            driver="feature_custom"
            cacheid="model-forests"
            vrfsim:enabled="true">
        <feature_indexing enabled="false"/>
        <clustering>true</clustering>
        <lighting>true</lighting>
        <features name="Vegetation" driver="wfs">
                <url>http://vr-theworld.com/vr-theworld/features/wfs</url>
                <typename>7</typename>
            <disable_tiling>true</disable_tiling>
        </features>
...
 </model>
```

This enables the legacy behavior, allowing you to specify your own tiling scheme by telling the server not to give you the pre-tiled data.

As a long-term solution, when you require an untiled layer, you should create a non-tiled layer on the server and point to that (instead of a pre-tiled WFS layer).

1.1.7 Effectmap Extension Replaced Lightmap Extension

The lightmap osgEarth layer extension is no longer supported. The effectmap extension is able to create the same type of layer using the following:

```
 <effectmap effect_map_type="LIGHT_MAP">
    ...
 </effectmap>
```

where the ... is replaced with an image layer (same as the way that the old lightmap layer used to work).

1.1.8 Image and Elevation Layers

Layers now all have exact Layer types. Instead of coupling a generic image layer with a driver property, you now specify only the exact layer type, for example:

Old: `<image name="Hawaii" driver="tms">`

New: `<TMSImage name="Hawaii">`

Image types include:

- TMSImage
- GDALImage
- WMSImage
- XYZImage
- FeatureImage
- CompositeImage

Elevation types include:

- TMSElevation
- GDALElevation
- MBTilesElevation
- CompositeElevation

Other layers:

- FeatureModel
- GroundCover
- SplatImage
- LandCover
- CompositeLandCover

You can still use the image/driver construct for osgEarth 2.0-style TileSource drivers. It works for `feature_custom` and the various MCOD and CDB plug-ins.

1.1.9 Composite Layers

osgEarth 3.0 composite layers have a new syntax:

```
<CompositeImageLayer name="my layer">
   <layers>
       <TMSImage ...>
       <GDALImage ...>
       ...
```

`CompositeElevationLayer` and `CompositeLandCoverLayer` follow the same pattern.

1.1.10 Feature Sources

A feature source provides access to shapefile and feature data. Feature sources are first-class layers in osgEarth 3.0. They also have explicit names without drivers, for example:

Old: `<features driver="ogr">`

New: `<OGRFeatures>`

Other feature source layers include:

- TFSFeatures
- XYZFeatures
- WFSFeatures

You can still embed a feature source in a Layer as before.

Since feature sources are layers in osgEarth 3.0, you can put them anywhere in the earth file and then reference them by name using the features property, for example:

```
<OGRFeatures name="road-data"> ...
   ...
<FeatureModel name="roads" features="road-data"> ...
```

Using this construct, more than one Layer can reference the same feature source without needing to repeat the declaration.

1.1.11 Composite LandCover Layers

osgEarth 2.0 supported multiple `LandCover` layers in an earth file. This is no longer the case. osgEarth 3.0 supports only one LandCover layer. It uses the first one it finds in the earth file.

Use the new `CompositeLandCover` Layer to combine multiple sources into one Layer.

```
<CompositeLandCover>
   <layers>
       <LandCover ...>
       <LandCover ...>
```

> The layout is a little different than osgEarth 2.0's representation of a multi-source LandCover layer. Please refer to *./userData/servers/landcover.worldwide.Combined.online.xml* for an example.

1.1.12 Fractal Refinement of LandCover Data

The `LandCover warp` property was an attempt to make low-resolution land cover data look a little more natural from a distance in osgEarth 2.0. Unfortunately, it destroyed coastlines and mangled the integrity of the source data. For these reasons, it is no longer used in osgEarth 3.0.

osgEarth 3.0 `LandCover` supports fractal refinement to up-sample low resolution LandCover data. By setting the `max_data_level` higher than the maximum natural LOD of the source data, a `LandCover` layer automatically creates interpolated data up to the `max_data_level` LOD.

1.1.13 GroundCover Layers (Trees, Grass, and so on)

A new `spacing` property is measured in meters, and defines the average spacing between objects. This is more intuitive than the old `density` property, which was based on the hardware tessellation algorithm.

The density still works, but is now interpreted as “objects per square kilometer”. `spacing` is the preferred property.

`max_distance` still works, but avoid values that are greater than the maximum range of the LOD setting.

The new `Grass` layer type improves the visual quality of grass. It uses the same semantics as the `GroundCover` layer.

2. Migration to VR-Vantage 2.7

This chapter describes a migration issue from previous versions of VR-Vantage to 2.7.

2.1. File Locations

Entity configuration has been broken up into many files (one for each entity). The files are organized into a folder structure that matches the main data folder and, when VR-Vantage starts, it reads all configuration files in the sub-folders.

As of VR-Vantage 2.7:

- The model definition configuration file location changed from *./appData/definitions/ModelDefinitions* to *./data/Configuration/Definitions/Entity*.
- The element definition configuration file location changed, as well:
  - *./appData/definitions/Entity/ElementDefinitions* to *./data/Configuration/Definitions/Entity*.
  - *./appData/definitions/Unit/ElementDefinitions* to *./data/Configuration/Definitions/Unit*.
  - *./appData/definitions/TacticalGraphics/ElementDefinitions* to *./data/Configuration/Definitions/TacticalGraphics*.

If you are upgrading from a previous version of VR-Vantage, you can copy your *.ommx* files into these new locations and they will be picked up automatically.

3. Migration to VR-Vantage 2.8

There are no significant changes to the VR-Vantage API or source files that affect migrating from VR-Vantage 2.7. For occasional updates to migration information, please check https://www.mak.com/mak-one/support/help/migration-support.

4. Migration to VR-Vantage 3.0

This chapter describes migration issues from previous versions of VR-Vantage to 3.0.

4.1. Shared Data

Code that accessed files formerly found in *./data* (or `$(DATA_DIR)`) may need to look for them in the shared data package instead. Generally, you can use the `$(SHARED_DATA)` prefix and the `DtDe`'s `pathConfiguration().resolvePath()` to ensure that your code does not need to know exactly where the shared data is stored.

The data directory has been reorganized, as illustrated in Figure 1.

Figure 1. Shared data directory structure (partial)

```
$(SHARED_DATA) /
    Fonts
    Icons
    ModelData /
        Configuration
        Lifeforms
        Structures
        Vehicles
        etc.
    TerrainData /
        Terrain /
            California
            Massachusetts
            etc.
        TerrainConfiguaration /
            Ala Moana.mtf
            Ground_DB II.mtf
            etc.
    etc.
```

As an example, what you would previously access with:

```
myDe.dePathConfiguration().resolvePath("$(DATA_DIR)/Other/Numbers_rgb.meif")
```

is now found at:

```
myDe.dePathConfiguration().resolvePath("$(SHARED_DATA_DIR)/ModelData/Other/Numbers_rgb.meif")
```

4.2. Listener/Visualizer Refactor

Many of the classes used for configuring, creating, and managing the entities, aggregates, and tactical graphics received by the VR-Link-based drivers have been refactored to make more of the core reusable in other non-VR-Link drivers.

4.2.1 State Listeners

All listener signals for entities, aggregates, and tactical graphics have been moved to the state listeners (in `vrvCore`). The VR-Link-specific listeners overrides now only convert values from protocol-specific values to protocol-independent values and pass

them down to the base listeners. This means that all state visualizers can potentially discover all changes to the visualized object's state without depending on protocol-specific classes (from the `vrvVrl` library). The APIs for the base listeners have also been expanded considerably.

4.2.2 State Visualizers

Because of the changes to state listeners outlined above, almost all protocol-dependent visualizers have been removed from `vrvVrl` and the `vrvCore`-resident base classes refactored to use the data and signals provided by the state listeners in `vrvCore`. This makes it easier to extend visualizers as well, since you do not need to use the `DT_PROTOCOL_NAMESPACE` macro and set up multiple builds for each protocol.

A new additional constructor of the form:

```
DtStateVisualizer( DtBaseConnection& connection, DtStateListener&
  listener, DtSceneObjectAgent* sceneObjectAgent, int modelSet )
```

has been added. The previous constructor is still available for only backwards compatibility. Deriving classes should fully implement the new form because that it what the visualizer factory uses when it is created in a VR-Link connection.

With the new constructor, a corresponding creator method was also necessary:

```
virtual DtStateVisualizer* create( DtBaseConnection& simulation,
  DtStateListener& listener, DtSceneObjectAgent* sceneObjectAgent,
  int modelSet ) const
```

The visualizer factory can create either, based on whether or not a scene object agent is provided. When using a VR-Link connection, however, the scene object agent will be provided and therefore this form is used.

The `DtKinStateVisualizer` class has been replaced by `DtKinematicState` class. However, configuration for this class is still done using the `DtKinStateVisualizerDefinition`.

4.2.3 Element Managers

The element processors continue to handle element discovery from the VR-Link connection; however, the actual creation and management of those elements has moved to new element managers for entities, aggregates, and tactical graphics. These new managers also process the event signals from the render thread, such as selection and the enabling/disabling of various visual types.

4.2.4 Show/Hide Contexts

A new show/hide context enumeration, `DtShowHideVisualizerContext`, has been added. Many visualizers now use this mechanism to implement visibility. This makes it easier to manage the multiple ways in which show whether some visual element might shown or hidden. For example, an emitter volume would not be shown if:

- The entity that it is a part of was hidden or if emitter volumes in general were disabled.

- The API had requested that the emitter volume for that particular entity be hidden.

The show/hide context list allows us to track all of the reasons that a visualizer might be hidden, and show it only when all reasons have been removed.

4.2.5 Tactical Graphics Facades

There were several facade classes for tactical graphics that were used in tactical graphics visualizers. These were only wrappers around some of the components of the tactical graphics element and components of a visualizer from a given element. This mixing of visualizer components with element components made them incompatible with the new overall structure of Element objects, particularly the new `DtKinematicState` object. These have all been removed.

4.3. Other API Changes

There are other notable changes to the API:

- The `DtCoordinateSystem` class is not thread safe. As we have been moving some methods into multiple threads for better CPU utilization, users of the `DtCoordinateSystem` class should make a clone of `DtCoordinateSystem` for their own use and also listen for the signal that tells you when a coordinate system has changed so that you can update your copy. Several examples now use this methodology, such as exampleProjectile.
- The `DtCachedIntersectorManager` class was renamed to `DtCachedIntersectorContainer`.
- The `DtIntersectorCreator` declaration was moved out of `DtIntersector.h`. If you used this, you should now include `DtIntersectorCreator.h`.

5. Migration to VR-Vantage 3.1

There are no significant changes to the VR-Vantage API or source files that affect migrating from VR-Vantage 3.1. For occasional updates to migration information, please check https://www.mak.com/mak-one/support/help/migration-support.

Index

A

C

D

E

F

G

W

X

VRV-3.1-8-240129