# VR-Forces

## Getting Started Guide

VT MÄK
A company of VT Systems

# VR-Forces

## Getting Started Guide

VT MÄK
A company of VT Systems

VT MÄK
150 Cambridge Park Drive, 3rd Floor
Cambridge, MA 02140 USA

Voice: 617-876-8085
Fax: 617-876-9208

info@mak.com

www.mak.com

Revision VRF-4.2-14-130909

# Contents

**1. Introduction** ..... 1

Guide to the VR-Forces Documentation Set ..... 1

**2. Installing VR-Forces** ..... 5

Make Sure You Have the Correct Application Versions ..... 6
Install Applications in the Correct Order ..... 7
Uninstall and Reinstall Applications in the Correct Order ..... 7
Set Up Licensing ..... 7

**3. Running a Scenario** ..... 9

Start VR-Forces ..... 10
Load a Scenario ..... 13
Run the Scenario ..... 16
Understand the Scenario ..... 16
Getting Information about Individual Entities ..... 16
Viewing Plans ..... 19
Viewing Force Hierarchies and Relationships ..... 20
Scenario Objects ..... 21

**4. Creating a Scenario** ..... 23

Create a Scenario ..... 23
Create an Entity ..... 26
Create Tactical Graphics ..... 29
Save the Scenario ..... 30
Tell the Entity What To Do ..... 30

**5. Assigning Tasks** ..... 31

Independent Tasks ..... 31
Concurrent Tasks ..... 31
Assigning an Independent Task ..... 32
Running the Simulation ..... 33
Set Data Requests ..... 34

**6. Writing Plans** ..... 35

Choosing Between Independent Tasks and Plans ..... 35
Entity Plans and Global Plans ..... 36
Write an Entity Plan ..... 37
Using Conditions in Plans ..... 40
Global Plans ..... 41
The Bad Global Plan ..... 42
The Good Global Plan ..... 44

**7. VR-Forces Performance Issues** ..... 47

Terrain Database Size ..... 47
Scenario Size ..... 48
Simulation Protocol and Network Considerations ..... 48

**8. VR-Forces Utility Applications** ..... 49

The Entity Editor ..... 49
The OPD Editor ..... 50
Scenario Merge ..... 50
MÄK Terrain Database Tool ..... 50

**9. Tutorials and Example Scenarios** ..... 51

Example Scenarios ..... 51
Example Scenarios for B-HAVE ..... 53

**Index** ..... 55

# 1. Introduction

VR-Forces is a robust application with many features. We think it is pretty easy to use – once you get to know it a bit. However, because it has so many features, which are documented in great detail in several manuals, you may be unsure where to start. And like any application, VR-Forces has its quirks, which can trip up new users. So in this Getting Started Guide we provide a simplified introduction to VR-Forces. We get you up and running quickly and we point out some of the key things to do and to avoid. Once you have gotten started, you can delve more deeply into the rest of the documentation to learn more – or just explore the application on your own.

## Guide to the VR-Forces Documentation Set

The VR-Forces documentation set is provided to all customers in PDF format. Documentation is in the *./doc* directory and, in Windows, is accessible from the VR-Forces shortcuts on the Start menu. Table 1 tells you which manual to use to find various kinds of information about VR-Forces. The VR-Forces Documentation Center is a central resource for accessing all of the VR-Forces manuals. It contains tables of contents for each manual and a master index that combines the indexes of all of the manuals. All of the entries in the tables of contents and index are live links to the manuals. The documentation center also lets you open any of the individual manuals directly.

API documentation is provided in HTML format and is accessible from the Windows Start menu or, for Linux and Windows, by opening *./doc/classdoc/index.html*.

Table 1: Documentation roadmap

| If you want to know: | Read this: |
|---|---|
| How to navigate the 2D and 3D views, attach the observer to entities, change GUI settings and so on. Everything about the GUI except how to create and run scenarios. | *VR-Forces Users Guide*, online help. |
| How to create scenarios, assign tasks and sets, and write plans. | *VR-Forces Scenario Management Guide*, online help, and video tutorials at www.mak.com/products/vrforces_tutorials.php. |
| How to use the SDK (APIs). | *VR-Forces Developers Guide*, class documentation, header files. |
| How to add, change, delete, or configure entity and object models and behaviors without writing code. | *VR-Forces Configuration Guide*. |
| How to optimize performance. | *VR-Forces Configuration Guide*. |
| How to use the Entity Editor, OPD Editor, or Scenario Merge utilities. | *VR-Forces Configuration Guide*, online help in each utility. |
| How to compose terrains using elevation data, imagery, and feature data and save the terrains to the MTF format. | *VR-Forces Configuration Guide* and online help. |
| How to create GDB terrains from other formats, associate image data with terrains, and create MTD files. | *MAK Terrain Database Tool Users Guide*, TDB Tool online help. |
| Add feature data to a GDB file. | *MAK Terrain Database Tool Users Guide*, TDB Tool online help. |
| How to upgrade from older versions of VR-Forces. | *VR-Forces Release Notes*. |
| High level information about component models. | *VR-Forces Developers Guide* (HTML documentation). |
| What the API examples do and how to build them. | *VR-Forces Developers Guide* (HTML documentation). |

Table 1: Documentation roadmap

| If you want to know: | Read this: |
|---|---|
| Answers to frequently asked questions. | http://www.mak.com/community-forum/9-simulate-vr-forces.html. |
| Which manual to look in to find specific information. | *VR-Forces Documentation Center*. |

# 2. Installing VR-Forces

A full VR-Forces installation requires installation of two modules, the VR-Forces application (which includes the SDK) and the VR-Forces application data. You do not have to install the SDK if you only want to use the VR-Forces application. You must install the data. The application will not run without it.

The Windows version of VR-Forces and its plug-in applications install from a typical installation wizard. The Linux version installs by uncompressing and untarring the application and data files.

As you proceed through the installation process, be aware of the following issues:

- Make sure you install the correct versions of the VR-Forces applications.
- Install the applications in the correct order.
- Uninstall and reinstall applications in the correct order.
- Set up licensing correctly.
- If you are a developer, install the correct version of VR-Link and other required libraries.

For details, please see Chapter 2 of *VR-Forces Users Guide* and the MÄK web site at: http://http://www.mak.com/install-guide.html. You can download an installation guide from: http://www.mak.com/doc/installguide.pdf.

## Make Sure You Have the Correct Application Versions

VR-Forces has separate installers for VR-Forces, VR-Forces data, and the B-HAVE Module for VR-Forces. (The B-HAVE Module is an optional, extra-cost plug-in for VR-Forces.) Developers must also install VR-Link. HLA users must install an RTI. You might also have other MÄK products such as VR-Vantage and MÄK Logger. Mismatched applications will not run correctly, or at all.

- Make sure that all installers were built with the same compiler.
- Make sure that the version of VR-Link you have installed is the same version used to build VR-Forces.
- Make sure that the B-HAVE Module was built for your version of VR-Forces. (For example, B-HAVE Module 4.2 for VR-Forces 4.2.)

If you are not sure which versions of the various MÄK products go together, please review the release notes for your version of VR-Forces and the product version and build compatibility pages on the MÄK web site: http://http://www.mak.com/support/product-versions.html and http://http://www.mak.com/support/build-compatibility.html

## Install Applications in the Correct Order

To install the VR-Forces SDK, you must enter an installation code during the installation process. Your MÄK salesperson will provide the code when you are given the download links or it will be included with your installation DVD. Be sure that you have it before you start installing VR-Forces.

The correct order for installing VR-Forces is:

1. Install the VR-Forces application.
2. Install the VR-Forces data.
3. Install the B-HAVE Module for VR-Forces (if purchased).

## Uninstall and Reinstall Applications in the Correct Order

If you must reinstall any of the VR-Forces applications (for example, because you have received a patched version), then you must uninstall the applications in reverse order before reinstalling the updated version.

## Set Up Licensing

All MÄK products require that you run a license server and have the appropriate product licenses. The MÄK License Manager installer should have been provided with your VR-Forces installers. If not, you can download them at:

- Windows: ftp://ftp.mak.com/out/MAKLicenseManager-win32-setup.exe
- Windows: ftp://ftp.mak.com/out/MAKLicenseManager-win64-setup.exe
- Linux: ftp://ftp.mak.com/out/MAKLicenseManager-linux-setup.tar.gz

Complete instructions are provided with the License Manager software.

Plug-ins may be licensed separately from VR-Forces. Read their documentation for licensing details. If you are installing the B-HAVE Module, you must read section 1.2, “Installing the B-HAVE Module”, in *B-HAVE Module for VR-Forces Users Guide* for licensing information.

# 3. Running a Scenario

This chapter explains the basic steps for loading a scenario, explores what is going on in VR-Forces, and points you to the sections in *VR-Forces Scenario Management Guide* that provide all the details. Chapter 4, *Creating a Scenario* shows you how to create a scenario.

**To simulate entities in VR-Forces, you have to:**

1. Start VR-Forces.
2. Load a scenario or create a scenario.
3. Run the scenario.



Some new users start VR-Forces and load a terrain database. Then they wonder why most of the menu options are unavailable. To create or manipulate entities, you must have a scenario loaded. Although there are some cases in which you would just load a terrain database, they are the exception for typical use of VR-Forces.

## Start VR-Forces

To use VR-Forces, you start two executables: a front-end (the graphical user interface, or GUI) and a back-end (the simulation engine). You can start them separately (independent mode) or with one command or menu option (combined mode). In this guide, we just use combined mode.

If you want to run VR-Forces using HLA, you may need to start your RTI before you start VR-Forces. If you are using the MÄK RTI 3.4 or later, it will prompt you to choose an RTI connection.

**To start VR-Forces on Windows:**

1. On the Start menu, choose **Programs** → **MAK Technologies** → **VR-Forces 4.2** → **VR-Forces GUI + Simulation Engine.**

   If this is the first time that you are running VR-Forces, or if you have not previously selected a default startup connection, the Simulation Connections Configuration dialog box opens (Figure 1).

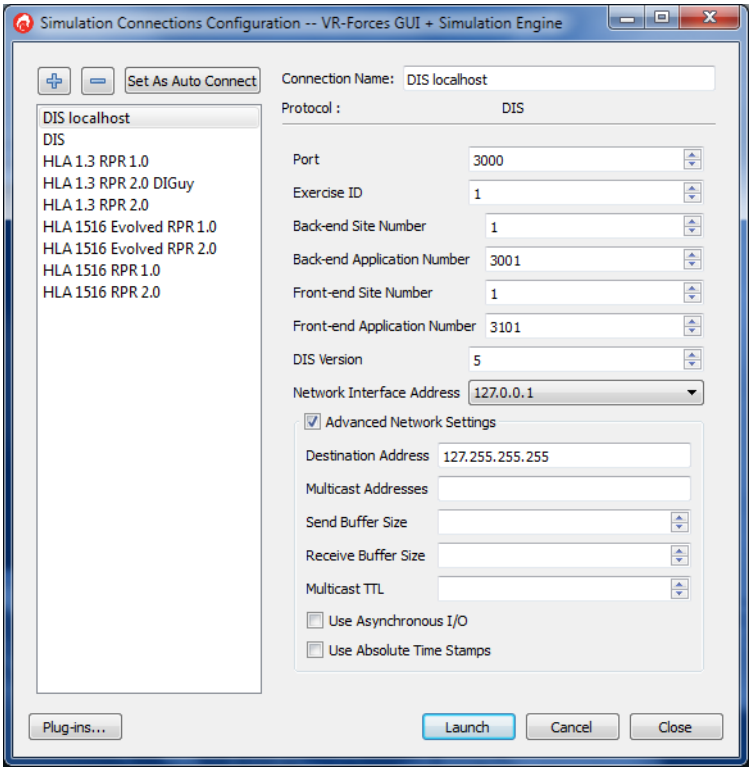


Figure 1. Simulation Connections Configuration dialog box

2. In the Simulation Connections Configuration dialog box, select a connection configuration. (You can create your own configurations, but for now use DIS or HLA 1.3 RPR 1.0. If you use HLA, you must have separately installed an RTI.)
3. Click Launch. VR-Forces starts up (Figure 2). The Scenario Startup screen provides shortcuts for starting a scenario or loading a scenario. If you do not want to use the screen, click Close. That closes the Scenario Startup screen, not VR-Forces.

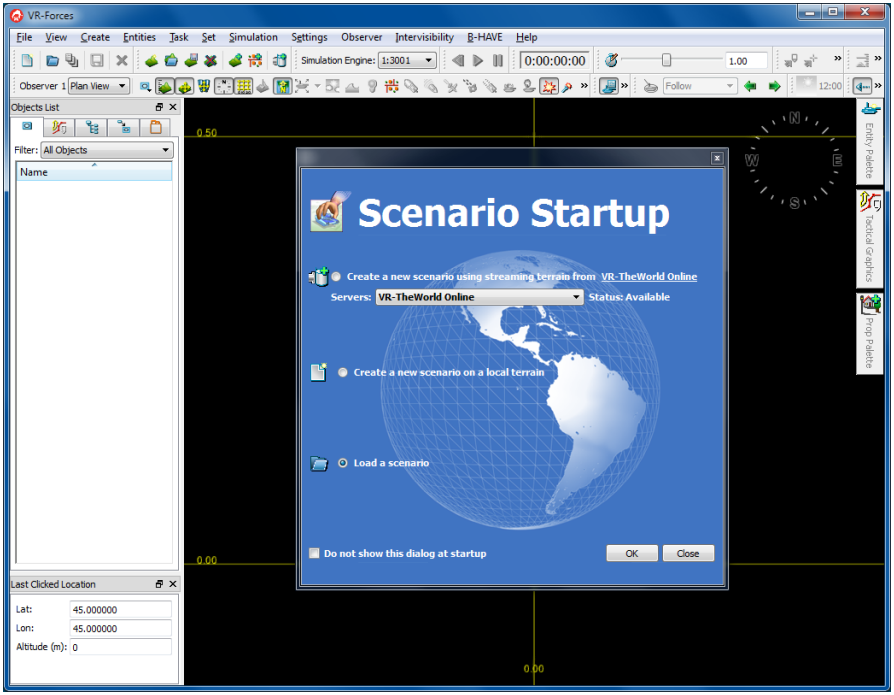


Figure 2. VR-Forces window at startup

VR-Forces has a 2D view and a 3D view. The 2D view (called Plan View observer mode) is the default view.

**To start VR-Forces in combined mode on Windows or Linux from the command line:**

1. Open a console window.
2. Change to the *./bin* directory for the protocol you are using.
3. Enter the following command:

```
vrfLauncher --app vrfGui -z vrfSim
```

For details, please see the following sections in *VR-Forces Users Guide*:

- Section 3.1, "The VR-Forces Program"
- Section 3.2, "Front-end and Back-end Concepts"
- Section 5.1, "Starting VR-Forces".

## Load a Scenario

To get a quick feel for VR-Forces, load one of the example scenarios and run it.

**To load a scenario:**

1. On the Scenario Startup screen, select Load a Scenario, or in the VR-Forces window, choose **File** → **Load Scenario**. The Load Scenario dialog box opens. It defaults to the *./userData/scenarios* directory. Each example scenario is in its own directory.
2. Select the *makland* directory.
3. Select *maklanddemo.scn* (Figure 3).

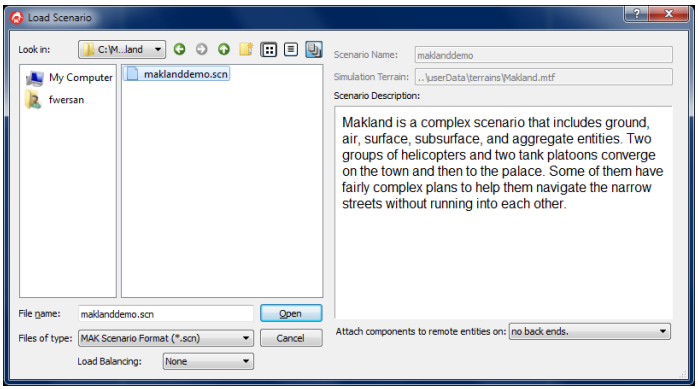


Figure 3. Makland scenario selected

4. Click Open. The scenario is loaded and the Scenario Information dialog box is displayed. It has a description of the scenario (Figure 4).

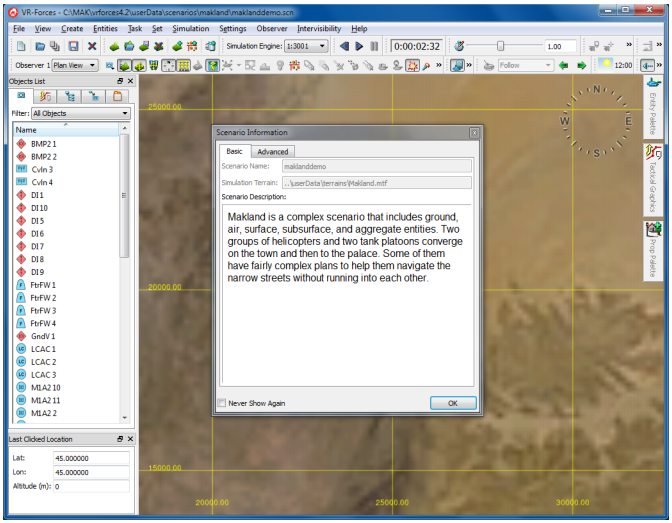


Figure 4. Makland scenario information

5. Close the Scenario Information dialog box. Now you can see the terrain, but you do not see any entities, because they are in a different area of the terrain.
6. Press the period key. The observer (the viewpoint in the display) attaches to the first entity in the list and you can now see entity icons (Figure 5).
7. Press **z**. This detaches the observer from any specific entity.

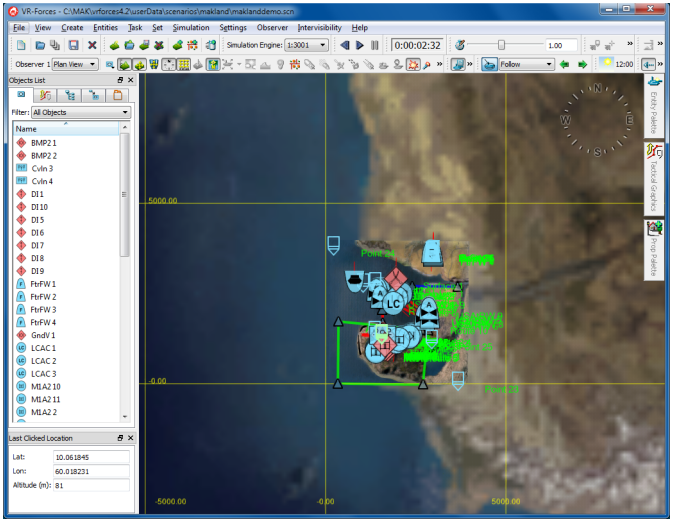

Figure 5. maklandDemo scenario

8. Press **e** to zoom in until you can see the entities in the scenario a bit more clearly (Figure 6).

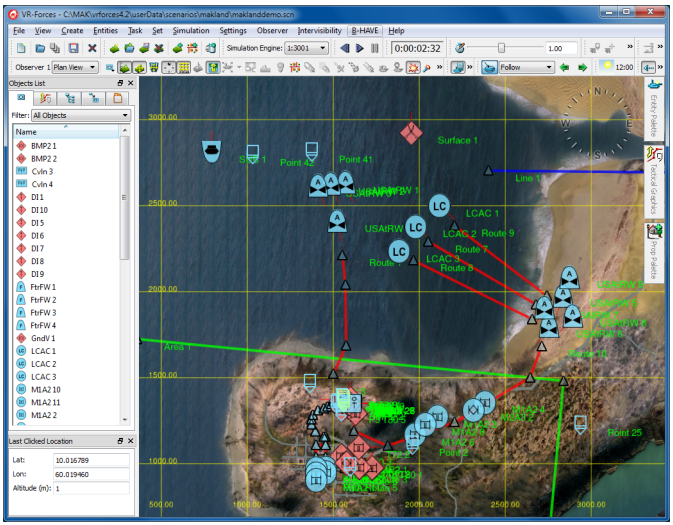

Figure 6. maklandDemo scenario (zoomed in)

## Run the Scenario

When the scenario is finished loading, you can run it.

➤ To run a scenario, choose **Simulation** → **Run Scenario**, or click the Run arrow on the Simulation toolbar (Figure 7).

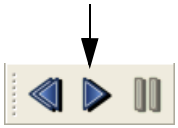

Figure 7. Simulation toolbar

The entity icons begin moving. You will see transient icons that represent missiles, lines representing munition fire and detonation, and other graphics.

## Understand the Scenario

In addition to displaying entities, objects, and graphics, VR-Forces provides a lot of information about what is going on as a scenario unfolds. In this section we will cover some of the ways to learn about a scenario.

### Getting Information about Individual Entities

On the left side of the VR-Forces window is a toolbar with a set of panels for controlling various aspects of the GUI. The top panel is the Objects List panel. It has tabs that show different views of the objects in the scenario.

1. Select the leftmost tab. It lists all entities.
2. Select (click) the entity named M1A2 4. In Figure 8, it is highlighted. (It is just to the east of the town, advancing along a road towards the town.)

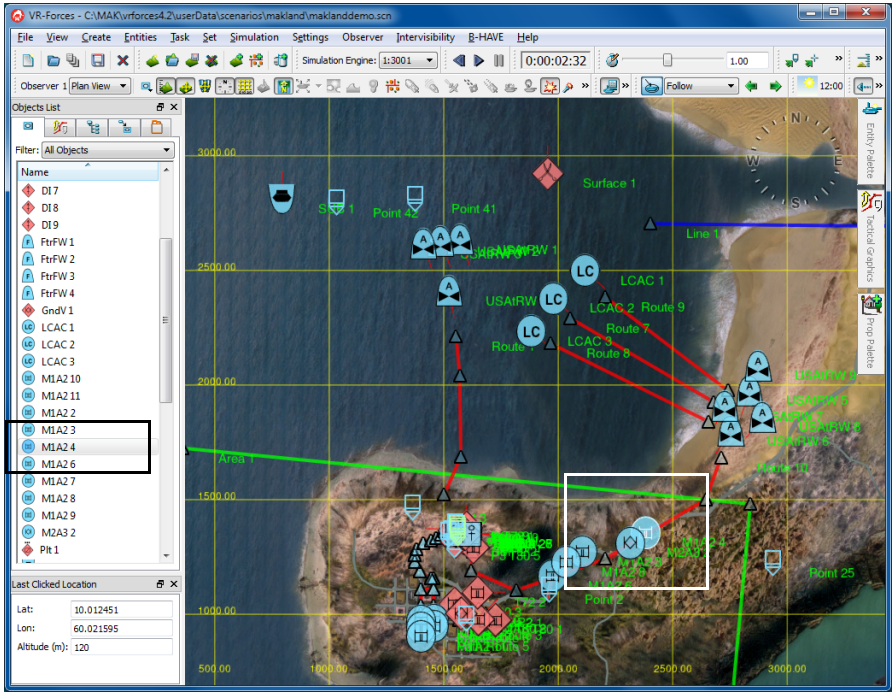


Figure 8. M1A2 4 advancing towards the town

3. Choose **Entities** → **Information**. A dialog box opens (Figure 9). It has pages that describe the entity's state and what it is doing. The M1A2 is an individual entity. If it were an aggregate entity, the dialog box would list its subordinates. At the bottom of the dialog box is a console window. It displays messages that the entity has received.

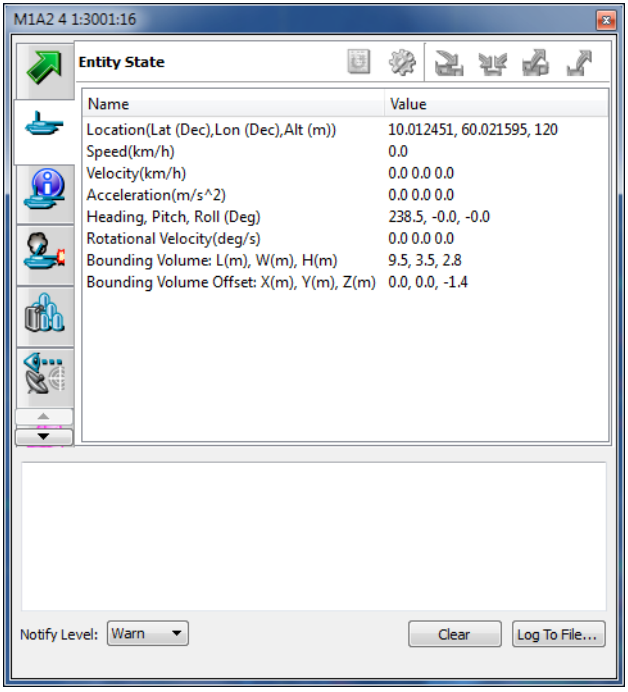


Figure 9. Entity Information dialog box

You can display information dialog boxes like this for most objects.

For more information about displaying information about entities and objects, please see Chapter 6, *Managing Objects* and Chapter 7, *Viewing Entities and Entity Information*, in *VR-Forces Users Guide*.

## Viewing Plans

Most of the entities in the scenario are moving. Entities move in response to independent tasks or tasks that are part of a plan. A plan is a list of tasks and set data requests that an entity executes in sequence. A plan can also contain tasks that are executed only under certain conditions.

**To view the plan for M1A2 4:**

1. Select M1A2 4.
2. Choose **Entities** → **Plan**. The Plan window opens (Figure 10).

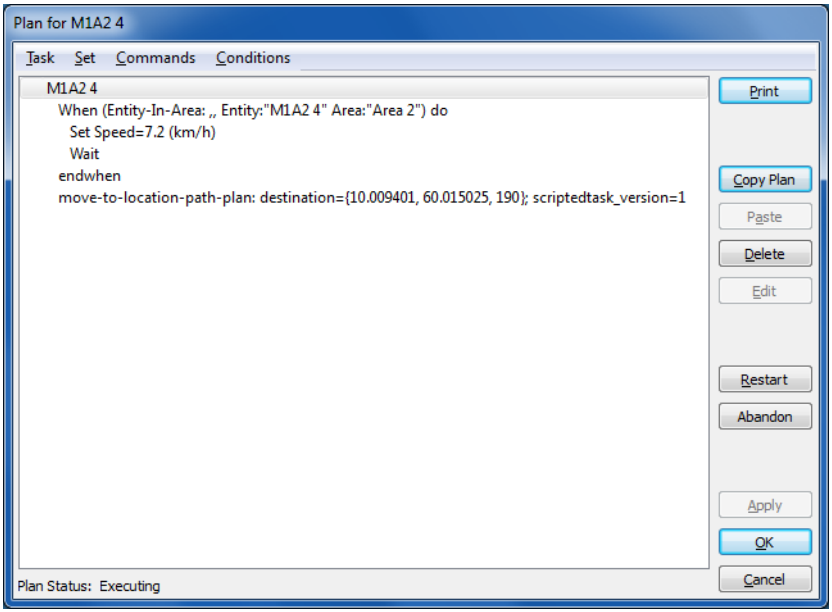


Figure 10. Entity plan

This plan is fairly simple. It tells the entity to move to a location using path planning. It also includes a condition block. The condition is that when the entity enters Area 2, it should change its speed and stop (Wait task).

For more information about tasks and plans, please see Chapter 5, *Assigning Tasks* and Chapter 6, *Writing Plans*.

## Viewing Force Hierarchies and Relationships

All entities exist within the context of a force, usually just called Friendly and Opposing. Some entities may be Neutral. (You can create up to 255 different forces.) The forces emulate military hierarchies.

➤ On the Objects List panel, select the Echelon View tab (Figure 11). This tab shows the hierarchical arrangement of the entities by force. (The makland scenario has a fairly flat hierarchy, so the figure also includes an example of a more detailed hierarchy.)

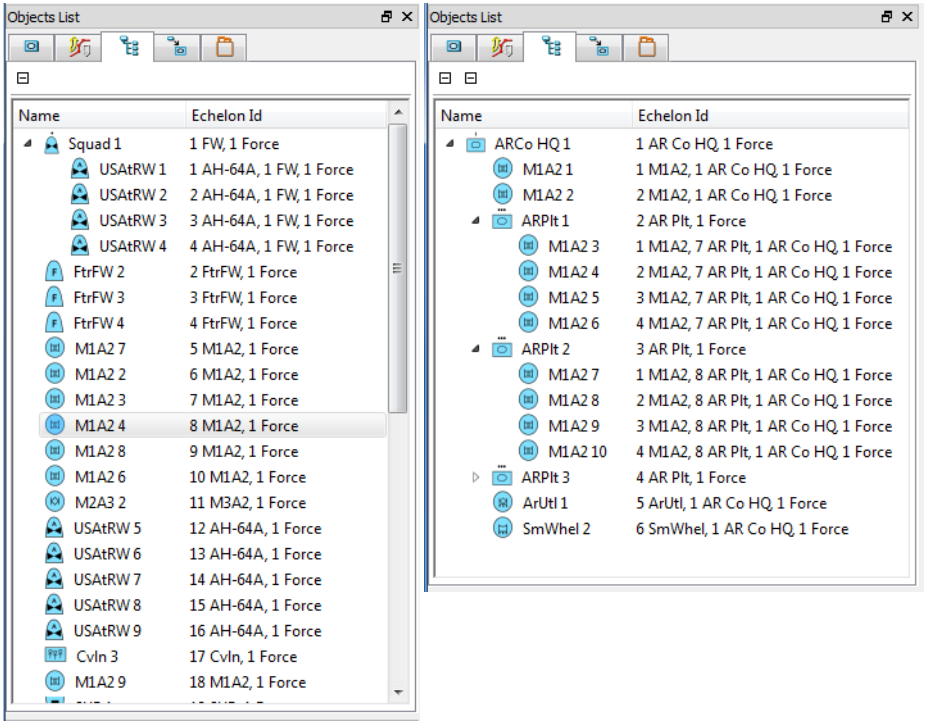


Figure 11. Echelon View

## Scenario Objects

In addition to entities, a scenario can contain tactical graphics. A specific subset of tactical graphics – waypoints, routes, areas, phase lines, and obstacles – are also called control objects. Entities can interact with control objects. For example, an entity can move to a point, or follow a route.

Figure 12 shows the list of objects on the Environmentals tab of the Objects List panel and identifies some objects on the terrain.

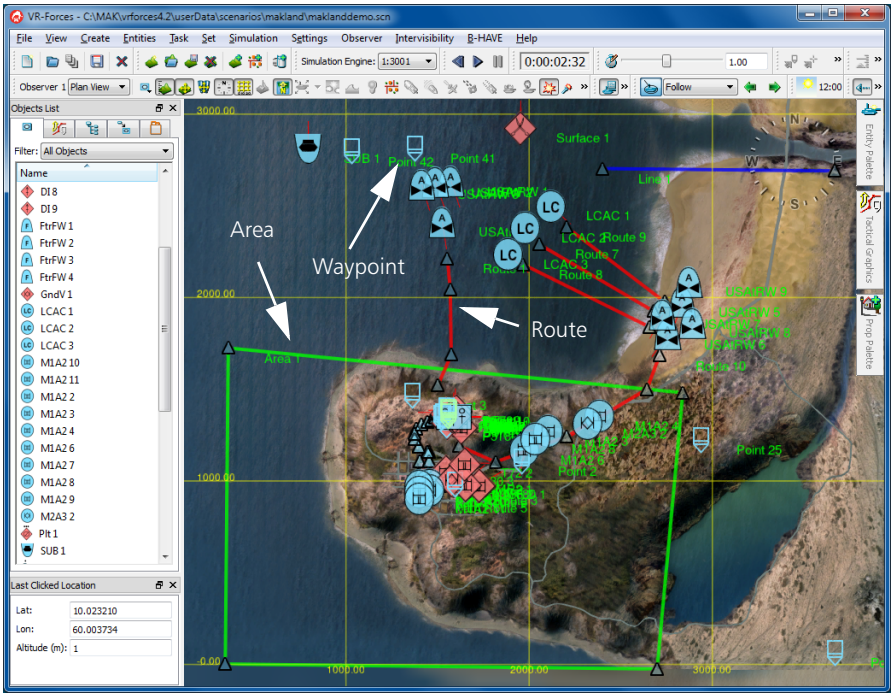


Figure 12. Tactical graphics

For more information about objects, please see:

- Chapter 6, *Managing Objects*, in *VR-Forces Users Guide*
- Chapter 5, *Overlays and Tactical Graphics*, in *VR-Forces Scenario Management Guide* .

# 4. Creating a Scenario

This chapter and the next two are essentially an extended tutorial for creating a simple scenario. The finished scenario, GettingStartedExample, is distributed with VR-Forces (*./userData/scenarios/GettingStarted/GettingStartedExample*).

**To create a scenario, you must:**

1. Choose a terrain.
2. Create entities.
3. Optionally, create control objects and overlay objects.
4. Tell the entities what to do.
5. Save the scenario.

## Create a Scenario

**To create a scenario:**

1. Choose **File** → **New Scenario**, or click the New Scenario button (  ) on the File toolbar. The Choose Simulation Terrain dialog box opens. By default, it opens in *./userData/terrains*, the directory that contains the terrain databases shipped with VR-Forces.
2. In the file list, select *Makland.mtf*. (A MÄK terrain file (MTF) packages together information about a terrain and related imagery for convenient loading in VR-Forces.)
3. Click Open. The New Scenario dialog box opens.

4. Give the scenario a name, such as myGettingStartedExample (Figure 13).

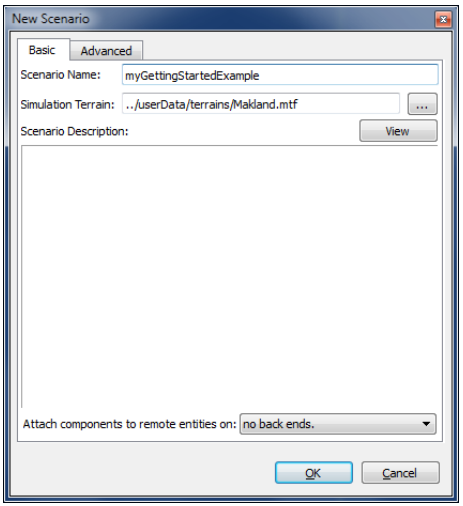


Figure 13. New Scenario dialog box

5. We will skip the scenario description. The Advanced tab specifies a variety of default scenario parameters, which we will accept. Click OK.

   The window displays the terrain database that you chose (Figure 14).

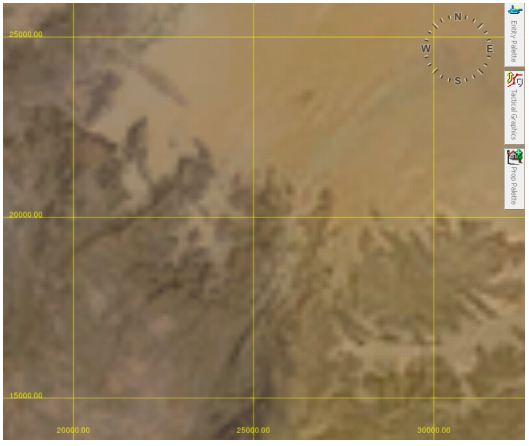

Figure 14. Makland terrain, Plan View mode

6. Press **q** to zoom out until you can see all of Makland (Figure 15).

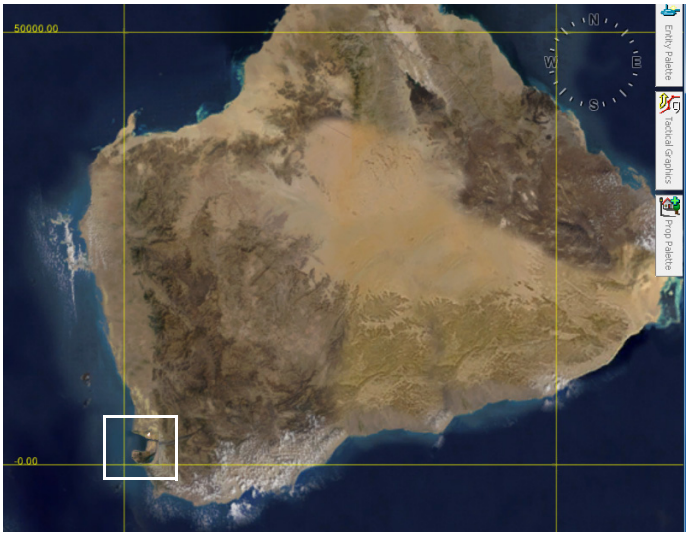


Figure 15. Makland zoomed out

7. Zoom in on the area outlined in Figure 15.
8. Save the scenario. (It is a good idea to save your scenario frequently as you develop it.)
   a. Choose **File** → **Save Scenario**. A Save dialog box opens.
   b. Click the create folder button.
   c. Type a name for the folder, for example, myGettingStarted.
   d. Select the folder.
   e. Click Open.
   f. By default, VR-Forces uses the name that you gave the scenario when you created it as the name when you save it. Accept the default name or type a different one.
   g. Click Save.

For details about creating scenarios, please see Chapter 1, *Creating and Running Scenarios*, in *VR-Forces Scenario Management Guide*.

For details about MTF files, please see *VR-Forces Configuration Guide*.

## Create an Entity

After you create a new scenario, you populate it with the entities you need to achieve the goals of your simulation project.

**To create a new entity:**

1. If you are not in Plan View observer mode, select it in the Observer Settings toolbar (Figure 16).


Figure 16. Observer Settings toolbar

2. In the Makland terrain there is a town in the center of the peninsula. Press **e** to zoom in on it (Figure 17).

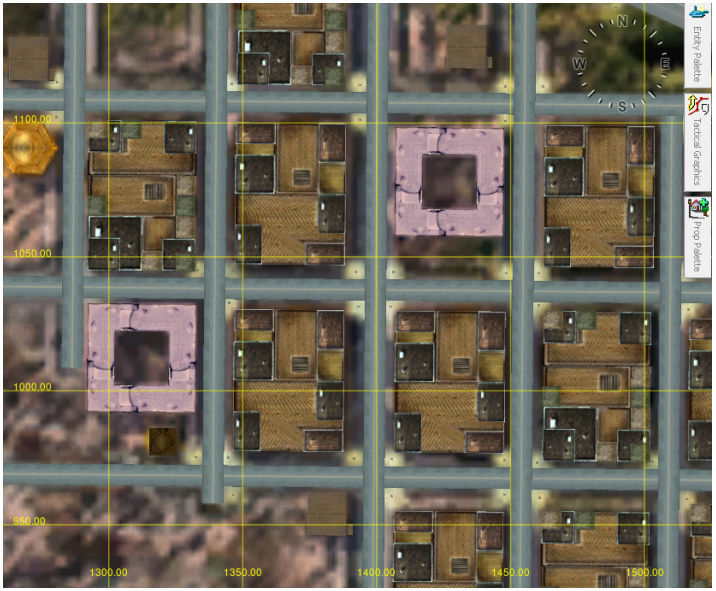


Figure 17. VR-Village

3. On the right side of the window, click the Entity Palette button. It expands to show a list of entities that you can create (Figure 18).
4. In the Categories list, select Ground.

5. In the Force list, select Friendly.

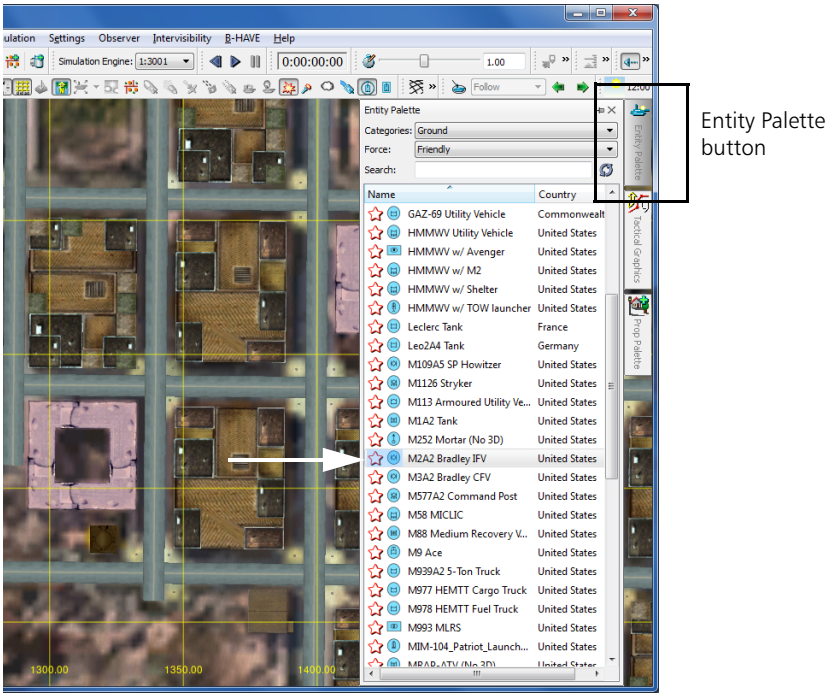


Figure 18. Entity Palette

6. In the list of entities, select M2A2 Bradley IFV. The cursor changes to show the icon for an M2A2 Bradley.
7. Click on the terrain on the North/South road in the center of the town. An icon is placed at that location. (When you are in create entity mode, you can continue clicking to create multiple instances of the selected entity type.)
8. Right-click to exit create entity mode. The scenario now looks like Figure 19.

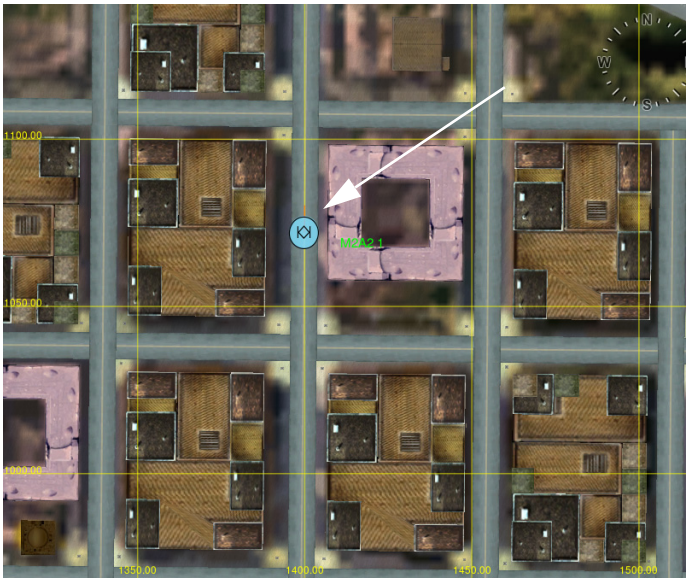


Figure 19. New entity

For details about creating objects, please see Section 2.1, "Creating Objects," in *VR-Forces Scenario Management Guide*.

## Create Tactical Graphics

Tactical graphics are points, lines, and areas that identify locations on the terrain. An entity can interact with these objects in various ways. One common use for tactical graphics is to have an entity move to a point or along a route.

**To create a route:**

1. Choose **Create** → **Waypoint**. The cursor changes to show the waypoint sysmbol. A Create Waypoint tab gets added to the right side of the window. We do not need to use it for this exercise.
2. Click on the road at the intersection in front (North) of the Bradley (Figure 20). Note that the cursor stays in create mode.


Figure 20. Vertex locations

3. Click on the intersection South of the Bradley, as shown in Figure 20.
4. Right-click to exit create mode.
5. Save the scenario.

For details about creating tactical graphics, please see Section 2.1, "Creating Objects," in *VR-Forces Scenario Management Guide*.

## Save the Scenario

The biggest mistake that new VR-Forces users make is to run a simulation before they save it. VR-Forces prompts you to save new or changed scenarios before you run them, but it does not automatically save them.

If you have not saved the scenario yet, save it now.

**To save a scenario:**

1. Choose **File** → **Save Scenario** or click the Save button on the file toolbar. The Save Scenario dialog box opens.
2. Optionally, create a directory in which to save the scenario.
3. Select the directory in which you want to save the scenario.
4. Give the scenario a name.
5. Click Save.

For details, please see Section 1.3, "Saving a Scenario," in *VR-Forces Scenario Management Guide*.

## Tell the Entity What To Do

Entities know how to do some things without being told. If an entity detects an enemy entity, it fires automatically (unless you have told it not to). If an entity is moving and it detects another entity in its way, or an obstacle of some sort, it tries to avoid it. However, to do anything really useful, you have to give an entity a task. You can give an entity tasks by writing plans, or by assigning independent tasks.

This is an important enough subject that we will devote the next two chapters to it. Please read on.

# 5. Assigning Tasks

This chapter is an introduction to tasks and how to assign them. You can give an entity a task by including it in a plan or by assigning it to the entity from the Task menu. We call tasks that are assigned from the Task menu "independent tasks".

The process of assigning a task is largely the same however you do it. We provide examples in this guide, but we really want to focus on the strategy for assigning tasks and plans and alert you to common problems that new users have.

## Independent Tasks

You can give an entity an independent task at any time. You can do it during scenario creation and you can do it while a scenario is running. The most important things to remember about independent tasks are that when you give an entity an independent task:

- It stops whatever it is doing and immediately begins to execute the new task (unless the new task is a concurrent task).
- If the entity is executing a plan, the plan is abandoned and the entity executes the independent task. When it is finished with the task, it does not return to the plan.

### Concurrent Tasks

Most independent tasks immediately override whatever an entity is doing. Concurrent tasks are the exceptions to this rule. A concurrent task does not cancel an entity's current task. An entity executes the concurrent task while it continues to execute the task it was working on. The primary purpose of concurrent tasks is to let an entity send a radio message while it is executing a movement task. (All movement tasks are mutually exclusive (that is, not concurrent).)

## Assigning an Independent Task

In our example scenario, the M2A2 Bradley vehicle is sitting on the road, doing nothing. Now we will tell it to move to Waypoint 1.

1. Select the Bradley.
2. Choose **Task** → **Movement** → **Move to Waypoint (Direct)**. The Move To Waypoint (Direct) dialog box opens (Figure 21). It lists the waypoints in the scenario.

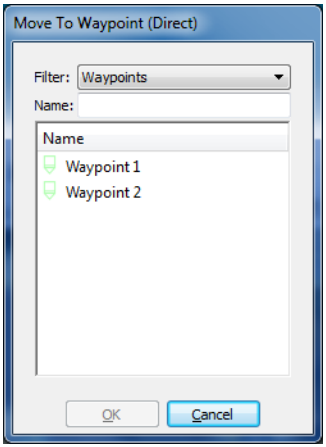


Figure 21. Move To Waypoint (Direct) dialog box

3. Select Waypoint 1.
4. Click OK.
5. Choose **File** → **Save**.

At this point nothing is happening, because the scenario is not running. However, you can check the Entity Information dialog box for the Bradley (select the Bradley and press **i**) to see that the task has been assigned (Figure 22).

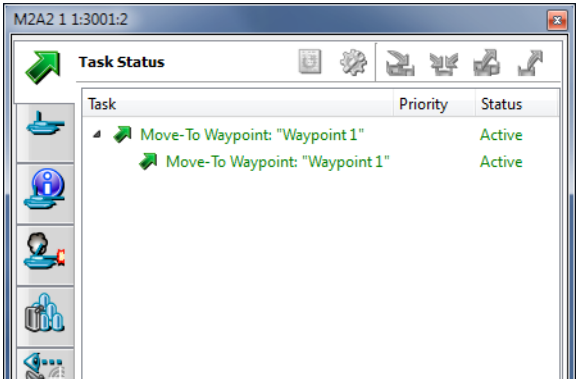


Figure 22. Entity Information dialog box

For complete details about task concepts and independent tasks, please see Chapter 6, *Assigning Tasks*, in *VR-Forces Scenario Management Guide*.

## Running the Simulation

The example scenario is still quite simple and does not do much. However, run it now to verify that everything you did works as expected.

➤ To run the simulation, choose **Simulation** → **Run Scenario**. (If you have not saved the scenario, VR-Forces prompts you to save it.)

The Bradley should move to the waypoint.

For details, please see Section 1.4, "Starting a Simulation," in *VR-Forces Scenario Management Guide*.

## Set Data Requests

Set data requests (sets for short) change the state of an entity. For example, you could set an entity to the destroyed state, or restore an entity from a destroyed state to a healthy state. Like tasks, you can issue sets from the Set menu or in plans. The main distinction between a task and a set is that a set always takes place immediately and sets do not interrupt tasks, except where logically expected, such as Destroyed or if you set a needed resource to zero. For a quick example of using a set data request, do the following:

1. If the scenario is running, choose **Simulation** → **Pause Scenario**.
2. Choose **Simulation** → **Rewind Scenario**.
3. Select the entity.
4. Choose **Set** → **Location**. The Location dialog box opens.
5. Click someplace on the terrain.
6. Click OK. The entity jumps to the selected location.
7. Choose **Simulation** → **Rewind Scenario**. If you are prompted to save changes, click No.

# 6. Writing Plans

In Chapter 5, *Assigning Tasks*, we gave the entity in our simple scenario a single, independent task. When we ran the simulation, it executed that task. What if you wanted the Bradley to move to Waypoint 1 and then move to Waypoint 2? Could you give it two independent tasks in the order you wanted them executed? No, because the second task would override the previous one. You would have to watch the scenario and when the entity completed the first task, give it the next task. What if there were many entities in the scenario? Could you keep track of them all and give the appropriate task at the appropriate time? Probably not.

If you want entities to execute tasks without direct human oversight, you need to give them plans. In this chapter we talk about the types of plans, add a plan to our example scenario, and then discuss using conditions in plans.

## Choosing Between Independent Tasks and Plans

Here are some guidelines for choosing between independent tasks and plans:

- For simple scenarios with few entities or quick proof of concept actions: use plans or independent tasks.
- For moderately to very complex scenarios: use plans.
- To script assignment of independent tasks: use global plans.
- While running a scenario: abandon an entity's planned actions by assigning independent tasks, by writing a new plan, or by the Abandon Plan command.

## Entity Plans and Global Plans

VR-Forces has two kinds of plans – entity-specific plans (entity plans) and global plans. The two types of plans look similar. They both can include tasks, set data requests, and simulation commands. They both can include conditional blocks. However, the way that tasks are assigned and carried out is very different. It is very important that you understand the differences.

In an entity plan:

- The entity owns the plan and it stays with the entity regardless of name or force changes.
- All of the tasks and sets apply implicitly to the entity.
- Tasks are executed sequentially as they are completed. In other words, task 2 is not issued until task 1 is complete.

In a global plan:

- There is no ownership. Global plans automate entity-independent scenario management.
- Tasks and sets are explicitly assigned to the entity that will execute them.
- All tasks are essentially independent tasks. They are sent to the assigned entities sequentially, but without regard to what the entity is doing. If a global plan has successive tasks for the same entity, they are sent in order, but the second task immediately overrides the first task, just as if you assigned two successive tasks from the Task menu.



If you want an entity to execute a series of tasks, one after the other, write an entity plan.

For details about the ins and outs of using tasks, set data requests, and plans, please see the following chapters in *VR-Forces Scenario Management Guide*:

- Chapter 6, *Assigning Tasks*
- Chapter 7, *Task Procedures*
- Chapter 8, *Setting Entity State*
- Chapter 9, *Writing Plans*.

## Write an Entity Plan

Previously, we assigned the Bradley vehicle a Move To Waypoint (Direct) independent task. Now we will do the same thing using a plan. And to highlight the difference between independent tasks and plans, we will give it a set data request and another task in the plan.

**To write the plan:**

1. Select M2A2 1.
2. Choose **Entities** → **Plan**. The Plan dialog box opens (Figure 23).

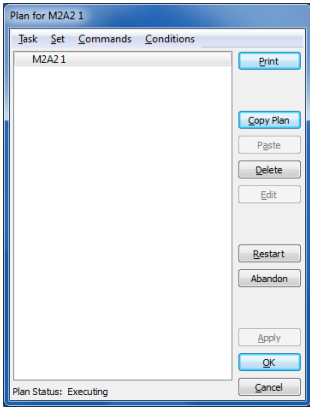


Figure 23. Plan dialog box

3. In the plan window, choose **Task** → **Movement** → **Move to Waypoint (Direct)**. The Move To Waypoint (Direct) dialog box opens. It lists the waypoints in the scenario. (Figure 24).

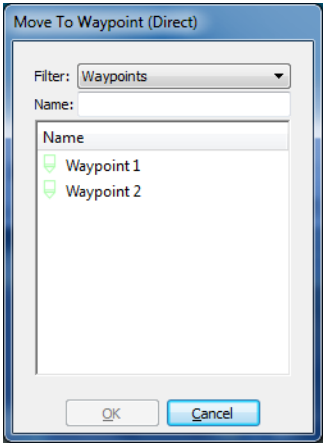


Figure 24. Move To Waypoint (Direct) dialog box

4. Select Waypoint 1.
5. Click OK. The task is added to the plan (Figure 25).

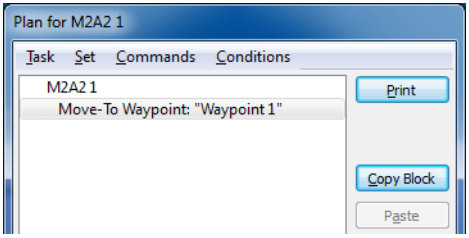


Figure 25. Plan with task

6. In the plan window, choose **Set** → **Position** → **Heading**. The Heading dialog box opens.
7. In the Heading text box, type 180.0.
8. Click OK. The set data request is added to the plan (Figure 26).

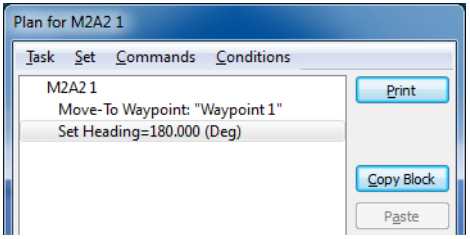


Figure 26. Plan with task and set data request

9. In the plan window, choose **Task** → **Movement** → **Move to Waypoint (Direct)**. The Move To Waypoint (Direct) dialog box opens.
10. Select Waypoint 2.
11. Click OK. The task is added to the plan. Figure 27 shows the final plan.

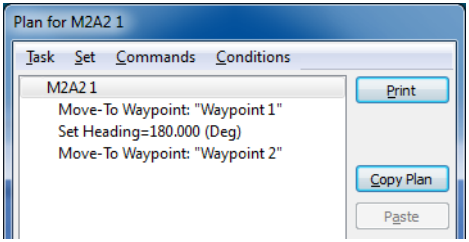


Figure 27. Plan with task and set data request

12. In the Plan window, click OK.
13. Save the scenario.

Run the scenario. The Bradley moves to Waypoint 1. Then it changes its heading to 180° and moves to Waypoint 2.

## Using Conditions in Plans

If you have a simple scenario in which you know everything that will happen and when it will happen, you can write plans that script this process exactly. However, in many simulations you do not know exactly what will happen and when it will happen. You probably know that some things might happen and you know what you want entities to do if certain events occur. You can plan for these events by putting conditional blocks in plans.

The three types of conditions supported by plans are:

- `If`: An `If` condition tests whether something is true at precisely the point when the test is made. If the test is true, the tasks in the `If` block get executed. Use an `If` condition if you want to test a condition and respond to it only at a certain point in a plan's execution.
- `When`: A `When` condition is tested repeatedly during a scenario. If at any time the condition being tested is true, the tasks in the `When` block get executed. Use a `When` condition if you want to respond to an event, but you do not know when that event will occur. The first time that a `When` condition is encountered in a plan, VR-Forces registers it. Then VR-Forces begins to test it.
- `While`: A `While` is not really a condition. It is a loop. VR-Forces enters the loop if a condition is true and repeats the tasks in the `While` block over and over as long as the condition stays true.

The most important things to remember about using conditions are:

- `If` conditions get tested once – when the plan gets to the condition as it sequentially moves through the plan. `If` conditions often do not work as expected because of timing issues: they are tested too soon, or too late. If a condition is time-dependent, you need to control for that issue in your planning – perhaps by using a `When`.
- `When` conditions do not get tested until they are registered. You should almost always put `When` conditions at the beginning of a plan. This ensures that they get registered before the entity starts executing its tasks. The only time you would put a `When` condition in the middle of a plan is if you do not want VR-Forces to start testing until the entity has done some other things first. (And even then you could probably nest the `When` inside another `When` and put it at the beginning of the plan.)
- `While` loops get tested at the start of each loop. For a `While` loop to be of value, you must ensure that the task sequence in the loop allows it to complete and test the loop's condition. For example, suppose you put a Patrol Between task inside a `While` loop. You want the entity to keep executing this task until the condition in the loop is satisfied. However, since a Patrol Between task does not end on its own, unless you have some other logic in the plan that can end the Patrol Between task, the entity will just stay on this task, the plan will not progress to the end of the loop, and the condition will never be tested.

Remember, the VR-Forces plan editor ensures that the syntax of plans is correct, but it cannot ensure that the logic of your plan is what you intend.

## Global Plans

Earlier in this chapter, we explained the differences between entity plans and global plans ("Entity Plans and Global Plans," on page 6-36). Adding a task or set to a global plan is slightly different from adding a task to an entity plan, but overall the mechanics of creating global plans are similar to those for entity plans. In this section we focus on the behavioral differences between global plans and entity plans.

The GettingStartedExample scenario (*./userData/scenarios/GettingStarted/GettingStarted-Example*) has two global plans (Good Global Plan and Bad Global Plan) that demonstrate how to write a global plan that does what you want and how to write one that does not do what you want. (You can write as many global plans as you want to.)

In each plan, we create a second M2A2 entity and give it the same tasks as M2A2 1 – move to Waypoint 1, set heading, and move to Waypoint 2.

## The Bad Global Plan

Let's start by taking a look at the bad global plan.

1. Load the GettingStartedExample scenario.
2. Choose **Simulation** → **Global Plan**. The Global Plan dialog box opens (Figure 28).

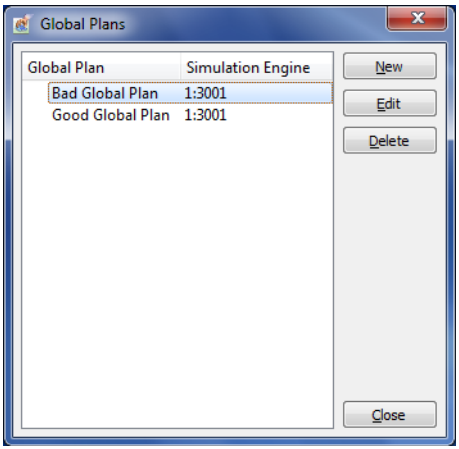


Figure 28. Global Plan dialog box

3. Select Bad global plan.
4. Click Edit. The plan is displayed (Figure 29).

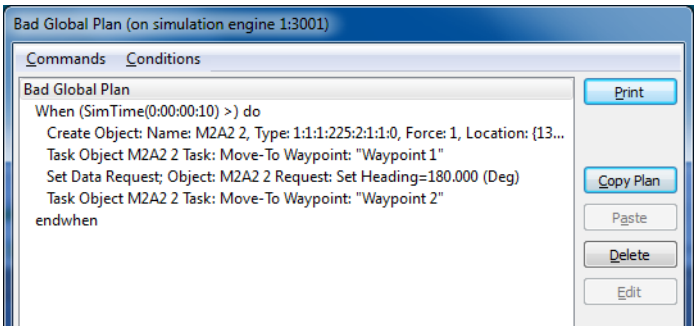


Figure 29. Bad global plan

In Bad Global Plan, we wait until 10 seconds of simulation time have elapsed (so we put everything in a `When` block), then create the entity and give it three tasks, just like in the plan for M2A2 1. The plan looks like this:

```
1    When (SimTime(0:00:00:10) >) do
2      Create Object: Name: M2A2 2, Type: 1:1:1:225:2:1:1:0, Force:
         1, Location: {10.009850, 60.012666, 190}, Heading(Deg):
         0.0, Clamped,
3      Task Object M2A2 2 Task: Move-To Waypoint: "Waypoint 1"
4      Set Data Request; Object: M2A2 2 Request: Set Heading=180.0
         (Deg)
5      Task Object M2A2 2 Task: Move-To Waypoint: "Waypoint 2"
6    endwhen
```

Run the GettingStartedExample scenario. The following happens:

1. At the start of the scenario, the `When` block is registered.
2. After 10 seconds, the entity gets created.
3. It immediately changes heading (line 4), then immediately starts moving to Waypoint 2 (line 5).
4. Stop the scenario.
5. Rewind the scenario.

The entity never moved to Waypoint 1 because a global plan sends tasks without regard to completion of prior tasks. The last task sent overrides all previous tasks for a given entity. In this case, the Move To Waypoint 2 task overrides the Move To Waypoint 1task. If you want an entity to complete each task before starting the next one, you must have the plan evaluate task completion before issuing a new task.

## The Good Global Plan

Now let's look at the good global plan.

1. In the Global Plan dialog box, select good global plan.
2. Click Edit. The plan is displayed.

Here is the plan:

```
1     When (SimTime(0:00:01:00) >) do
2       Delete Object: Name: M2A2 2
3       Create Object: Name: M2A2 2, Type: 1:1:1:225:2:1:1:0, Force:
          1, Location: {10.009795, 60.012667, 190}, Heading(Deg):
          0.0, Clamped,
4       Create Object: Name: Area 1, Type: 1:18:0:0:1:0:0:0, Force: 3,
          First Point (of 4): {10.010156, 60.012632, 190},
          Heading(Deg): 0.0,
5       When (Entity-In-Area: ,, Entity:"M2A2 2" Area:"Area 1") do
6         Set Data Request; Object: M2A2 2 Request: Set Heading=180.0
            (Deg)
7         Task Object M2A2 2 Task: Move-To Waypoint: "Waypoint 2"
8       endwhen
9       Task Object M2A2 2 Task: Move-To Waypoint: "Waypoint 1"
10    endwhen
```

The Good Global Plan is designed to wait until the entity finishes moving to Waypoint 1 before it sends the next task. We do this by drawing an area around Waypoint 1 because we can test to see if an entity is in an area. When the entity enters the area, we can assume that it has finished moving to the waypoint. The plan is structured as follows:

1. Everything is in a `When` block that does not get triggered until one minute of simulation time has passed (line 1). (This is so the good global plan does not interfere with the bad global plan.)
2. After one minute, we delete the entity created by the Bad Global Plan (line 2) and then create a new entity for this plan (line 3).
3. We create the area around the waypoint (line 4).
4. We create a `When` block to test for when the entity enters the area (line 5). Inside the block we set the heading (line 6) and add the Move to Waypoint 2 task (line 7).

The When block is not at the very start of the plan because we do not want to create it until the entity and area that it references already exist.

5. After the nested `When` block, but within the main `When` block, we put the Move To Waypoint 1 task (line 9).

Run the scenario. The bad global plan runs, as you have already seen. After one minute, the good global plan starts. The execution sequence is as follows:

1. When the scenario starts, the main `When` block is registered.
2. After one minute, M2A2 2 is deleted and a new M2A2 2 is created.
3. Area 1 is created.
4. The Entity In Area condition is registered.
5. The entity is assigned the Move To Waypoint 1 task. It moves to the waypoint.
6. When the entity enters Area 1, the Entity-In-Area `When` condition is satisfied. The entity's heading is set and it is given the Move To Waypoint 2 task.
7. The entity moves to the waypoint.

There is actually a much easier way to give an entity a series of tasks from a global plan. You can use the Issue Plan command to send an entire plan to an entity. It will execute the plan just as if you had written a regular entity plan. However, we did things the hard way in this section to emphasize how global plans process commands and how important it is to use the correct logic in a global plan.

You could also create complex task management using Lua scripts. However, scripting is beyond the scope of this manual. For details, please see *VR-Forces Scenario Management Guide*.

While the global plans are executing, M2A2 1 is also executing its plan. You may notice that when it gets to Waypoint 2, it drives a bit past the waypoint and displays an alert icon. It drives past the waypoint because M2A2 2 is already there, so it cannot move precisely to the waypoint. It displays an alert icon to indicate that a message has been sent to its console (Figure 30). The message says that it could not complete the task because there was an obstruction (M2A2 2). When you give entities tasks, whether as independent tasks or in plans, be aware that if multiple entities are moving to the same location, they will have to avoid each other and may not be able to complete the task as expected.

If you have B-HAVE installed, you might not see an alert icon. M2A2 1 will still avoid M2A2 2 and if you open the Entity Information dialog box and set the notification level to Info, you will see messages when M2A2 1 starts avoiding M2A2 2.

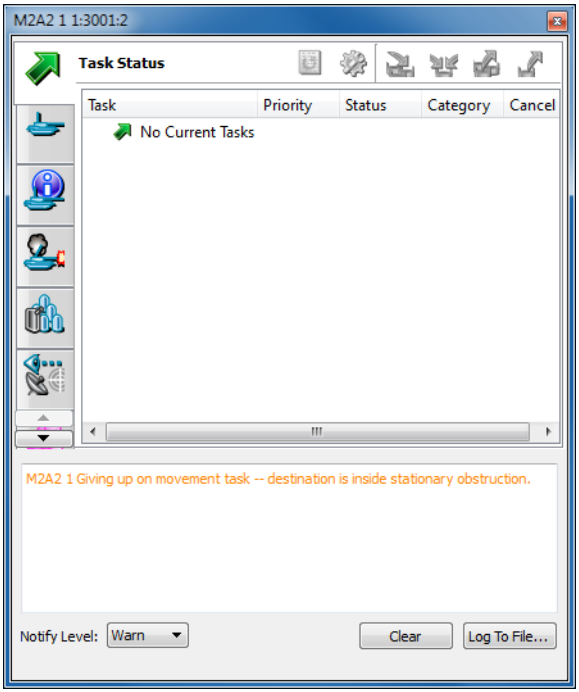


Figure 30. Entity console message

For more information about global plans, please see Section 9.5, "Creating Global Plans," in *VR-Forces Scenario Management Guide*.

# 7. VR-Forces Performance Issues

This chapter is a brief introduction to the major factors that affect performance. Chapter 1, *Optimizing Performance*, in *VR-Forces Configuration Guide* discusses performance issues in detail.

## Terrain Database Size

VR-Forces loads a copy of the terrain database for each front-end and back-end executable. Therefore, in a combined mode session it loads two copies of the database. The larger the database, as measured by the number of polygons and features, the more memory it will take, resulting in less memory available for simulation and display tasks. You can alleviate this problem somewhat by running the front-end and back-end on different computers.

If the terrain that you want to load is too large to fit into memory, you will have to reduce the number of polygons or features, or both to a manageable level. Depending on how the terrain was created, you may be able to optimize it to improve performance.

You might also be able to use a paged or streaming terrain instead of a static terrain.

For information about optimizing terrain, please see *MAK Terrain Database Tool Users Guide*. For information about paged and streaming terrains, please see Section 3.11.5, "Using Large Terrain Databases," in *VR-Forces Users Guide*, and Chapter 10, *Composing Terrains*, in *VR-Forces Configuration Guide*.

## Scenario Size

The size of a scenario affects performance in several ways. The more entities and objects in the scenario, the harder the back-end has to work to simulate them and the longer it takes for the front-end to update the display as entities change their state.

The effect of large numbers of entities is highly dependent on what they are doing. If many entities are moving at the same time, it places a greater load on VR-Forces than if only some of them are. If many entities are moving in proximity to each other, the increased calculations required for obstacle avoidance affects performance.

You can improve performance for large scenarios by spreading the simulation load over several back-ends.

## Simulation Protocol and Network Considerations

VR-Forces supports DIS and HLA. Each has its pros and cons.

DIS does not require any additional software, so it is easy to use out-of-the-box. However, DIS requires that entities send a complete state update (heartbeat) at regular intervals (typically every 5 seconds) even if their state has not changed. In large scenarios, this can flood the network with update messages, which can result in dropped packets.

HLA is more efficient than DIS. It does not require a heartbeat. Unless an application asks it to send a complete state update, HLA only sends data when an entity's state changes and it only sends the data that has changed. To use HLA, you must install an HLA RTI, which is available from various software vendors (including VT MÄK). This requirement introduces additional installation and configuration effort.

VR-Forces supports the use of HLA Data Distribution Management (DDM) as a means of managing large numbers of entities dispersed over wide areas. For details, please see Section 1.4, "Filtering Entities Using Interest Management," in *VR-Forces Configuration Guide*.

VR-Forces supports the HLA 1.3 specification, the SISO DLC version of the IEEE 1516 specification, and HLA Evolved (IEEE 1516-2010).

# 8. VR-Forces Utility Applications

VR-Forces includes the following utility applications to help you with common tasks:

- Entity Editor
- OPD Editor
- Scenario Merge
- MÄK Terrain Database Tool.

The utilities are in the *./bin* directory and, on Windows, can be started from the VR-Forces/Tools submenu on the Start menu.

## The Entity Editor

The Entity Editor helps you create new entity types and edit existing entities. Using the Entity Editor is the simplest way to add new systems to entities, such as weapons and sensors. It lets you:

- Add forces and specify force assignment.
- Assign categories for the Create menu and Entity Creation Palette.
- Specify the icon an entity uses.
- Specify core entity attributes.
- Edit DI-Guy character type, appearance, and animation.
- Create and edit aggregates.
- Edit aggregate formations.
- Create new simulation model sets.

For complete instructions for using the Entity Editor, please see Chapter 5, *Editing Entity Simulation Models*, in *VR-Forces Configuration Guide*.

## The OPD Editor

The OPD Editor lets you edit all entity platforms and systems. When you want to edit entity models, always start with the Entity Editor, it is much easier to use than the OPD Editor. But if you need to edit parameters that the Entity Editor does not cover, then use the OPD Editor.

For details about using the OPD Editor, please see Chapter 6, *Advanced Entity Model Editing*, in *VR-Forces Configuration Guide*.

## Scenario Merge

By default, VR-Forces assumes that a scenario is being built by a single user, or several users at different front-ends that are sharing the same back-ends. However, in some use environments, creation of a scenario is distributed to different groups working asynchronously, with the intention of merging the resulting scenarios into one large scenario. Merging scenarios by hand is a very difficult process, which is prone to error. The Scenario Merge tool lets you easily merge scenarios together.

For details about using the Scenario Merge tool, please see Appendix B, *Merging Scenarios*, in *VR-Forces Scenario Management Guide*.

## MÄK Terrain Database Tool

The MÄK Terrain Database Tool (TDB Tool) lets you create GDB terrains for use with VR-Forces.

The terrain agility features in VR-Forces provide a lot of flexibility for loading elevation data, imagery, and feature data. However, there are cases where VR-Forces cannot load some terrain data or cannot load it in an optimal manner. Using the TDB Tool, you may be able to load this terrain data and convert it to the GDB format, which can then be loaded by VR-Forces.

The TDB Tool also lets you add point and area feature data to a GDB terrain, which can then be imported into VR-Forces.

For complete details about the TDB Tool, please see *MAK Terrain Database Tool Users Guide*.

# 9. Tutorials and Example Scenarios

*VR-Forces Scenario Management Guide* includes two tutorials that go into greater detail than the example in this guide. We also have some brief videos that illustrate the tutorials. You can view the video tutorials at *./doc/vrforces_tutorialvideos.htm*.

## Example Scenarios

VR-Forces includes many example scenarios that demonstrate its features. Here is an annotated list of the example scenarios provided with VR-Forces:

- apache_groundDB.Three helicopters attack a convoy. Demonstrates movement tasks, use of routes and waypoints, and a conditional task.
- breaching. Three tank platoons engage opposing forces. Utility vehicles breach a mine field. This scenario contains aggregate entities. It also demonstrates use of tactical graphics.
- BombMission. The scenario mission drops a bomb on a high priority target on the island of Oahu, Hawaii. To accomplish the mission, several F-16 decoys are used to distract enemy forces while a B-2 drops a high altitude JDAM on the primary target.
- CounterMeasuresDemo. This scenario demonstrates the chaff and flare feature for fixed-wing aircraft.
- embarkdemo. A helicopter picks up passengers and takes them to an aircraft carrier. An LCAC lands on a beach and discharges a HMMWV and truck. A firefight takes place near the airport. This scenario demonstrates the VR-Forces embarkation feature. It also shows a Patriot missile shooting down a helicopter. It includes fixed-wing, rotary-wing, ground, surface, and lifeform entities. This scenario is the basis for the embarkdemo recording distributed with MÄK Data Logger.

- embarkexample. This scenario is a subset of the embarkdemo. It only demonstrates the embarkation activities. The fire fight at the airport and the LCAC landing are not part of this scenario. This scenario is fully explained in the Embarkexample tutorial in *VR-Forces Scenario Management Guide*.
- firstexperience. This scenario demonstrates the scenario that users of *VR-Forces First Experience Guide* create as they work through the guide.
- GettingStartedExample. This scenario demonstrates the scenario that users of this manual create.
- maklanddemo. Makland is a complex scenario that includes ground, air, surface, subsurface, and aggregate entities. Two groups of helicopters and two tank platoons converge on the town and then to the palace. Some of them have fairly complex plans to help them navigate the narrow streets without running into each other. This scenario is the basis for the maklanddemo recording distributed with MÄK Data Logger.
- marineMissileDefense. This scenario demonstrates the ability of surface vessels to survive multiple hits. An opposing force ship fires cruise missiles at a friendly force ship. The blue ship uses its missile defense system to try to shoot down the incoming missiles. If the blue ship is hit, it does not sink after the first successful strike; it takes multiple hits to successfully sink it.
- CommsDemo. This scenario is an example for use with Network Centric Forces (Qualnet and VR-Forces).
- remoteAttachment: This scenario demonstrates how to attach components to remote entities. See the readme.txt file for instructions for running the scenario.
- takeoffandlanding: Two jets take off and then land at an airport. This scenario demonstrates the fixed-wing take off and fixed-wing landing tasks.
- TrafficDemo. This scenario demonstrates use of the road following feature. It shows vehicles moving along roads in opposite directions and passing each other.
- WeatherDemo-Raining and WeatherDemo-Clear. The weather scenarios demonstrate how VR-Forces models weather effects and how they affect entities. In WeatherDemo-Raining, the opposing force entities are able to travel much closer to the friendly force entities before they are detected than they do in WeatherDemo-Clear.

These scenarios are provided to demonstrate the features of VR-Forces. They do not pretend to demonstrate realistic military tactics or strategy.

## Example Scenarios for B-HAVE

If you have the B-HAVE Module for VR-Forces installed, the following example scenarios are included:

The B-HAVE Module includes the following example scenarios:

- CarBomb: Entity behavior is controlled through B-HAVE tasks in entity plans. Civilians wander around the business district of a small town. Enemy combatants move through tunnels and buildings to set up positions while a car drives to a location and explodes. Surviving civilians flee. Dismounted infantry search for the enemy combatants. Reinforcements arrive by helicopter, disembark, and help search for enemy combatants or take up positions to secure the town. Civilians return to the bomb site. This scenario uses embarkation. If you are using HLA, you must use RPR FOM 2.0 draft 17.
- CarBomb_NoEmbarkation: This version of the CarBomb scenario does not use embarkation, so you can use it with RPR FOM 1.0. (Embarkation requires use of RPR FOM 2.0, draft 17.)
- Crowd: 100 civilians wander randomly around the city in the TerraSim_SampleUrban terrain. They go in and out of buildings and to multiple floors of buildings.
- FirstExperienceBurstTraffic, FirstExperienceSimpleTraffic, and FirstExperienceCustomPlan. Three scenarios that match the traffic tutorials in B-HAVE First Experience Guide. They demonstrate progressively more complex use of the pattern of life feature.
- Interior Movement: Demonstrates the movement of lifeform entities within buildings. Some civilians wander randomly in and out of buildings, while some DI and other civilians follow pre-determined plans to reach points on different floors of some of the buildings in TerraSim_SampleUrban. DI 1 has no plan. You can task him to move to some of the waypoints to demonstrate dynamic path planning.
- Makland B-HAVE: Mixed vehicle and life form activity on the Makland terrain. Some civilian traffic moves along the roads. Some DIs walk in and out of the palace, changing guard duty, and some military vehicles leave the palace to drive along the roads to locations in the town.
- RaidDemo: This scenario simulates a warehouse raid in Honolulu, Hawaii. Three groups of soldiers approach the raid location by helicopter and boat. They disembark, and four two-man teams move through cover to the doors of the warehouse, while the rest of the soldiers stay back to cover the approach. After the soldiers disembark, the helicopters leave and fly a route nearby, waiting for the signal to come back.

  Once the raid teams are in position, two teams perform a room clear action, while the other two stay covering the exits.

After the room is cleared, the helicopters are called back. The raid teams fall back, the soldiers embark by groups, and the helicopters and boat make their way back to the carrier they were launched from.

This scenario uses the following scripted tasks:

- Move in Cover. This task moves an aggregate of soldiers through phase lines representing cover.
- Synchronize. This task listens for a specific message from a selected group of entities, and broadcasts a response.
- Move In Parallel. This aggregate task re-tasks each subordinate with a move to, bypassing the aggregate's formation movement.
- Clear Room. This aggregate task moves each subordinate simultaneously along each of two selected routes.

- ReactToExplosion: In this scenario, a bomb in a duffle bag explodes. All civilians in the area react by running from the explosion. A nearby officer radios to all emergency vehicles in the area. The responders then rush to the area, with civilian vehicles pulling over for them as they go.

  This scenario uses the following reactive tasks:

  - Flee from Explosion. Civilians react to the nearby explosion by fleeing in the opposite direction.
  - Make Way for Emergency. Civilian vehicles react to nearby emergency vehicles by pulling to the side of the road. They merge back into the road when the vehicle has passed.
- Scatter. An opposing force entity walks through town. Civilians hide from it.
- SubwayPOL. This scenario generates civilian POL (Pattern of Life) traffic entering and leaving a subway station. The goal is to generate a scene which can be used to stimulate a security camera. The location of the camera is on the rooftop of a building across the road from the subway station, it is marked by the Point-of-Interest tactical graphic.

  In this scenario several patterns are used:

  - Bursts of people leaving the station every 5 minuts when a train comes.
  - Continuous, but slow stream of people entering the station from multiple directions.
  - Pedestrians bypassing the station.

  The scenario requires accces to VR-TheWorld (VR-TheWorld.com) to load streaming terrain.

# Index


## A

apache_groundDB example scenario 51
area 21
assigning
  task, independent 32

## B

Bad Global Plan 42
behavior, entity 30
B-HAVE Module 6
breaching example scenario 51
build compatibility 6

## C

CarBomb scenario 53
CarBomb_NoEmbarkation scenario 53
changing
  entity, state 34
Choose Simulation Terrain dialog box 23
combined mode, starting VR-Forces in 12
concurrent task 31
condition 40
control object 21
  creating 29
creating
  control object 29
  entity 26
  graphical object 29
  new scenario 23
  route 29
  scenario 23
  tactical graphics 29
Crowd scenario 53

## D

dialog box
  Choose Simulation Terrain 23
  Save Scenario 30
DIS, HLA 48
documentation, guide to 1

## E

Echelon View 20
embarkdemo example scenario 51
embarkexample example scenario 52
entity
  behavior 30
  creating 26
  force 20
  information 16

entity (continued)
plan 36
state, set data request 34
Entity Editor 49
example
apache_groundDB 51
breaching 51
embarkdemo 51
embarkexample 52
FirstExperienceBurstTraffic 53
FirstExperienceCustomPlan 53
FirstExperienceSimpleTraffic 53
makland 52
takeoffandlanding 52
example scenario 51

## F

file, MTF 23
FirstExperienceBurstTraffic example scenario 53
FirstExperienceCustomPlan example scenario 53
FirstExperienceSimpleTraffic example scenario 53
force 20

## G

global command, Issue Plan 45
global plan 36, 41
Good Global Plan 44
graphical object, creating 29

## H

hierarchy 20

## I

If condition 40
independent task 31, 35
information, entity 16
installation guide 5
installing
reinstalling 7
VR-Forces 5
Interior Movement scenario 53
Issue Plan command 45

## L

license server, management of 7
line 21
loading, scenario 13

## M

MÄK Terrain Database Tool 50
MAK web site, product and support pages 6
Makland B-HAVE scenario 53
makland example scenario 52
MTF file 23

## N

New Scenario dialog box 23

## O

object, control 21
object parameter database 50
obstacle 21
OPD Editor 50
overlay 21

## P

performance
effect of terrain size 48
optimizing 47
terrain, size 47

plan 30, 35
  condition 40
  entity 36
  global 36, 41
  viewing 19
  writing 35, 37
point 21
product, uninstalling and reinstalling 7
product version 6
protocol 48

## R

RaidDemo, scenario 53
ReactToExplosion, scenario 54
reinstalling, products 7
route, creating 29
running
  scenario 16, 33
  VR-Forces 10

## S

Save Scenario dialog box 30
saving, scenario 25, 30
Scatter, scenario 54
scenario
  apache_groundDB 51
  breaching 51
  creating 23
    entities 26
  creating new 23
  embarkdemo 51
  embarkexample 52
  example 51
  FirstExperienceBurstTraffic 53
  FirstExperienceCustomPlan 53
  FirstExperienceSimpleTraffic 53
  loading 13
  makland 52
  RaidDemo 53
scenario (continued)
  ReactToExplosion 54
  running 16, 33
  saving 25, 30
  Scatter 54
  size, affect on performance 48
  SubwayPOL 54
  takeoffandlanding 52
Scenario Merge 50
set 34
set data request 34
simulation
  creating 23
  protocol 48
  running 16, 33
Simulation Connections Configuration dialog box 10
starting
  combined mode 12
  VR-Forces 10
state
  entity, changing 34
SubwayPOL, scenario 54
support
  product, web site 6

## T

tactical graphic, creating 29
takeoffandlanding example scenario 52
task 30
  concurrent 31
  independent 31
    assigning 32
TDB Tool 50
terrain, affect on performance 47
terrain database 23
toolbar 16
tutorial 51
  video 51

## U

uninstalling 7
utility
  Entity Editor 49
  MÄK Terrain Database Tool 50
  OPD Editor 50
  Scenario Merge 50

## V

version
  product 6
    web page 6
video tutorial 51
viewing, plan 19
VR-Forces, starting 10

## W

When condition, While loop 40
writing, plan 35, 37


Link - Simulate - Visualize

VRF-4.2-14-130909

150 CAMBRIDGE PARK DRIVE, 3RD FLOOR CAMBRIDGE, MA 02140 617.876.8085 www.mak.com