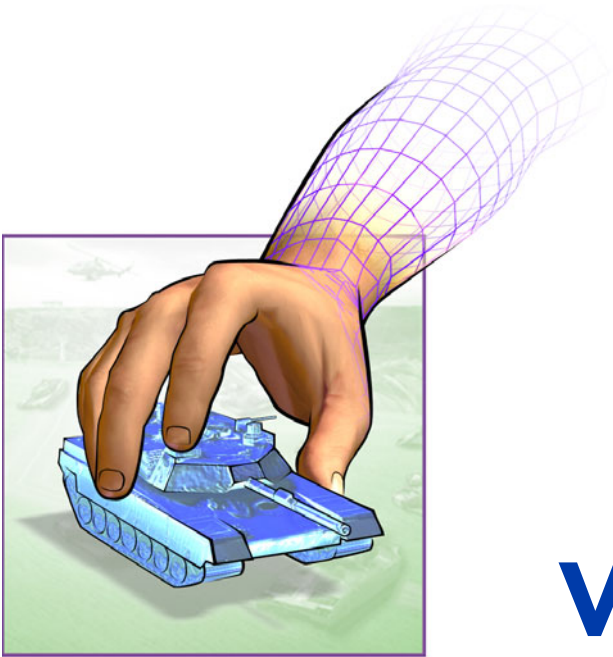
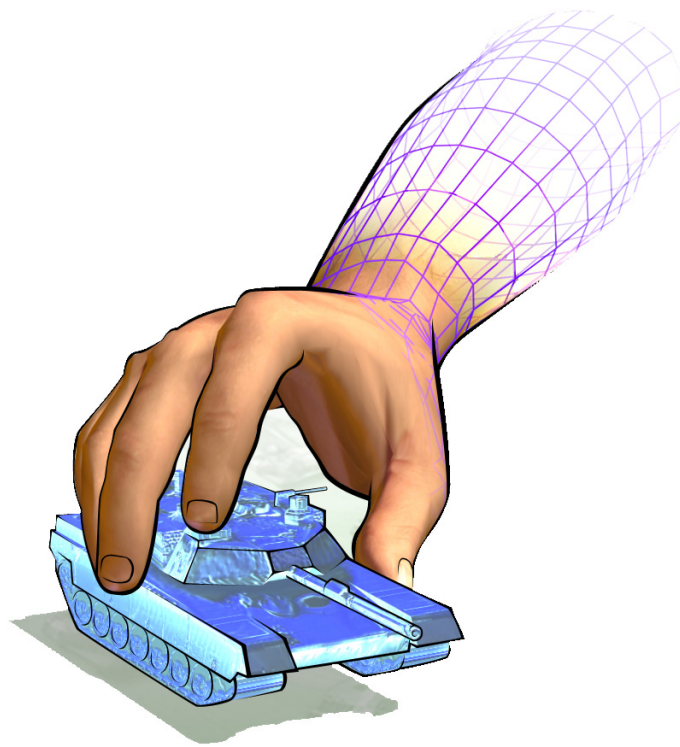




# VR-Forces

## Configuration Guide





# VR-Forces



## Configuration Guide

Copyright © 2009 VT MÄK  
All rights Reserved. Printed in the United States.

Under copyright laws, no part of this document may be copied or reproduced in any form without prior written consent of VT MÄK, Inc.

VR-Exchange™ and VR-Vantage™ are trademarks of VT MÄK.  
MÄK Technologies®, VR-Forces®, RTIspy®, B-HAVE®, and VR-Link® are registered trademarks of VT MÄK.

All other trademarks are owned by their respective companies.

For third-party license information, please see [“Third Party Licenses,”](#) on page xix.

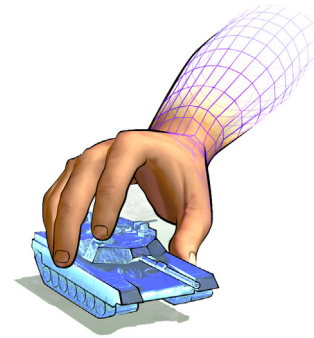
VT MÄK  
68 Moulton St.  
Cambridge, MA 02138 USA

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

info@mak.com  
www.mak.com

Revision VRF-3.12-7-090707





# Contents

---

## Preface

---

|                                                        |       |
|--------------------------------------------------------|-------|
| How This Manual is Organized .....                     | xiii  |
| VR-Forces Documentation.....                           | xiv   |
| MÄK Products .....                                     | xv    |
| How to Contact Us .....                                | xvii  |
| Document Conventions .....                             | xviii |
| Hardware and Operating System Naming Conventions ..... | xviii |
| Mouse Button Naming Conventions.....                   | xix   |
| Third Party Licenses .....                             | xix   |
| Boost License.....                                     | xix   |
| libXML and libICONV .....                              | xx    |
| pThreads Library.....                                  | xx    |
| EHS, PCRE, and PME .....                               | xx    |

## Chapter 1 Optimizing Performance

---

|                                                            |      |
|------------------------------------------------------------|------|
| 1.1. Introduction .....                                    | 1-3  |
| 1.2. Adjusting Refresh Rates .....                         | 1-4  |
| 1.2.1. Changing the Frame Rate for the Display .....       | 1-4  |
| 1.2.2. Changing the Network Frame Rate .....               | 1-5  |
| 1.3. Monitoring Front-end and Back-end Frame Rates .....   | 1-5  |
| 1.3.1. Viewing Frame Rate Statistics .....                 | 1-5  |
| 1.3.2. Configuring Statistics Reporting .....              | 1-7  |
| 1.4. Tuning Front-End Performance .....                    | 1-8  |
| 1.4.1. The Meeting Frame Rate Targets Level .....          | 1-8  |
| 1.4.2. The Not Meeting Frame Rate Targets Level .....      | 1-8  |
| 1.4.3. The Heavy Network Traffic Level .....               | 1-9  |
| 1.5. Miscellaneous Performance Configuration Options ..... | 1-10 |
| 1.5.1. Turn Off Track Histories .....                      | 1-10 |
| 1.5.2. Turn Off the Display of Polygons .....              | 1-10 |

|                                                                                 |      |
|---------------------------------------------------------------------------------|------|
| 1.5.3. Limit Use of Object Transparency .....                                   | 1-10 |
| 1.5.4. Limit Use of Spot Reports .....                                          | 1-10 |
| 1.5.5. Configure Hazard Cloud Graphics .....                                    | 1-11 |
| 1.5.6. Configure Intervisibility Intersection Testing .....                     | 1-11 |
| 1.5.7. Using Asynchronous I/O for DIS Exercises .....                           | 1-11 |
| 1.6. Setting the Tick Rate .....                                                | 1-12 |
| 1.6.1. Specifying the Tick Rate for Components .....                            | 1-12 |
| 1.6.2. Tuning the Network Interface .....                                       | 1-13 |
| 1.6.3. Tuning the State Repository Tick Rate .....                              | 1-13 |
| 1.7. Balancing Visual Quality Against Network Performance .....                 | 1-14 |
| 1.7.1. Balancing Visual Quality Versus Performance in the<br>3D Front End ..... | 1-15 |
| 1.7.2. Configuring Performance Options .....                                    | 1-15 |
| 1.8. Tuning the Target Frame Rate .....                                         | 1-16 |
| 1.9. Optimizing Performance for the 3D Front End .....                          | 1-16 |
| 1.9.1. Displaying Performance Statistics .....                                  | 1-17 |

## Chapter 2 VR-Forces Configuration Files

---

|                                                                               |      |
|-------------------------------------------------------------------------------|------|
| 2.1. MÄK Technologies Lisp (MTL) Files .....                                  | 2-2  |
| 2.1.1. Using Environment Variables in an MTL File .....                       | 2-2  |
| 2.1.2. Specifying Lists of Lists .....                                        | 2-3  |
| 2.1.3. Using Conditional Statements .....                                     | 2-3  |
| 2.2. Configuration File Descriptions .....                                    | 2-4  |
| 2.3. The <i>vrGui.mtl</i> Configuration File .....                            | 2-5  |
| 2.4. The <i>vrSim.mtl</i> Configuration File .....                            | 2-14 |
| 2.5. Symbol Mapping Files .....                                               | 2-18 |
| 2.5.1. Symbol Mapping Syntax for Animated Images .....                        | 2-19 |
| 2.5.2. Symbol Mapping Syntax for TrueType and Unicode<br>TrueType Fonts ..... | 2-20 |
| 2.5.3. Changing the Symbol Map File During Runtime .....                      | 2-20 |
| 2.6. Other Configuration Files .....                                          | 2-20 |
| 2.6.1. The <i>plugin.mtl</i> File .....                                       | 2-20 |
| 2.6.2. The <i>commonObjects.mtl</i> File .....                                | 2-20 |
| 2.6.3. The <i>rangeRings.mtl</i> Configuration File .....                     | 2-21 |
| 2.6.4. The <i>boundingVolume.mtl</i> Configuration File .....                 | 2-21 |
| 2.7. Configuring Overlay Objects .....                                        | 2-21 |
| 2.8. Configuring DI-Guy Integration .....                                     | 2-22 |
| 2.9. Configuring Communication Models .....                                   | 2-23 |
| 2.9.1. Simple Radio Communication Model .....                                 | 2-23 |
| 2.9.2. External Communication Model .....                                     | 2-23 |
| 2.9.3. Custom Communication Models .....                                      | 2-24 |

## Chapter 3 Scenario Files

---

|                                                                              |      |
|------------------------------------------------------------------------------|------|
| 3.1. Introduction .....                                                      | 3-2  |
| 3.2. The Scenario File .....                                                 | 3-3  |
| 3.2.1. Scenario Parameters .....                                             | 3-4  |
| 3.2.2. Specifying Pathnames for Scenario Files .....                         | 3-6  |
| 3.2.3. Changing the Terrain Database for a Scenario .....                    | 3-7  |
| 3.3. Editing the Plan Files .....                                            | 3-8  |
| 3.3.1. The Structure of an Entity Plan File .....                            | 3-9  |
| 3.4. Editing the Object Map File .....                                       | 3-11 |
| 3.5. Editing the Order of Battle File .....                                  | 3-11 |
| 3.5.1. Saving Default Entity Parameters to the Order of<br>Battle File ..... | 3-12 |
| 3.6. The Overlay File .....                                                  | 3-12 |
| 3.7. The Selection Groups File .....                                         | 3-12 |
| 3.8. Temporary Scenario Directories .....                                    | 3-12 |

## Chapter 4 Merging Scenarios

---

|                                                            |     |
|------------------------------------------------------------|-----|
| 4.1. The Scenario Merge Tool .....                         | 4-2 |
| 4.2. Starting Scenario Merge .....                         | 4-3 |
| 4.3. Creating a Scenario Merge Project .....               | 4-3 |
| 4.3.1. Adding Scenarios to a Project .....                 | 4-4 |
| 4.3.2. Removing Scenarios from a Project .....             | 4-4 |
| 4.3.3. Specifying an Ignore List File .....                | 4-5 |
| 4.3.4. Saving a Project .....                              | 4-6 |
| 4.3.5. Loading a Project .....                             | 4-6 |
| 4.4. Merging Scenarios .....                               | 4-7 |
| 4.5. Displaying Scenario Files .....                       | 4-7 |
| 4.5.1. Editing Scenario Files .....                        | 4-8 |
| 4.5.2. Closing Scenario Files .....                        | 4-8 |
| 4.6. Specifying the Output Notification Level .....        | 4-8 |
| 4.7. Using the Scenario Merge Command-Line Interface ..... | 4-9 |

## Chapter 5 Editing Entity Models

---

|                                                                       |     |
|-----------------------------------------------------------------------|-----|
| 5.1. Introduction .....                                               | 5-3 |
| 5.1.1. Simulation Model Sets .....                                    | 5-4 |
| 5.1.2. Starting the Entity Editor .....                               | 5-4 |
| 5.2. Creating a New Simulation Model Set .....                        | 5-5 |
| 5.2.1. Specifying the Default Simulation Model Set for Scenarios .... | 5-6 |
| 5.3. Opening a Simulation Model Set .....                             | 5-6 |

|                                                                                  |      |
|----------------------------------------------------------------------------------|------|
| 5.4. Editing an Entity Model .....                                               | 5-7  |
| 5.4.1. Changing an Entity's Icon .....                                           | 5-8  |
| 5.4.2. Editing an Entity's Category .....                                        | 5-10 |
| 5.4.3. Editing an Entity's Force .....                                           | 5-11 |
| 5.4.4. Changing an Entity's Entity Enumeration .....                             | 5-12 |
| 5.4.5. Changing an Entity's Name .....                                           | 5-12 |
| 5.4.6. Specifying Whether an Entity can be Created .....                         | 5-13 |
| 5.4.7. Specifying a 3D Model for an Entity .....                                 | 5-14 |
| 5.4.8. Editing Behavioral Parameters .....                                       | 5-14 |
| 5.4.9. Editing a Lifeform's DI-Guy Character, Appearance,<br>and Animation ..... | 5-15 |
| 5.4.10. Undoing (Reverting) Changes to an Entity .....                           | 5-16 |
| 5.5. Editing Aggregates .....                                                    | 5-17 |
| 5.5.1. Adding Subordinates .....                                                 | 5-19 |
| 5.5.2. Removing Subordinates .....                                               | 5-20 |
| 5.5.3. Replacing Subordinates .....                                              | 5-20 |
| 5.5.4. Changing the Order of Subordinates .....                                  | 5-20 |
| 5.5.5. Editing a Subordinate Aggregate .....                                     | 5-21 |
| 5.5.6. Adding Formations .....                                                   | 5-21 |
| 5.5.7. Removing Formations .....                                                 | 5-22 |
| 5.5.8. Renaming Formations .....                                                 | 5-22 |
| 5.5.9. Specifying the Default Formation .....                                    | 5-22 |
| 5.5.10. Expanding and Collapsing the Display of Formations .....                 | 5-22 |
| 5.5.11. Displaying Bounding Boxes .....                                          | 5-22 |
| 5.5.12. Copying Formations .....                                                 | 5-23 |
| 5.5.13. Laying Out a Formation Automatically .....                               | 5-24 |
| 5.5.14. Changing a Formation's Layout by Hand .....                              | 5-24 |
| 5.6. Editing an Entity's Systems .....                                           | 5-25 |
| 5.6.1. Adding A System .....                                                     | 5-26 |
| 5.6.2. Selecting a Damage Model or Movement Model .....                          | 5-26 |
| 5.6.3. Editing a System's Properties .....                                       | 5-27 |
| 5.6.4. Editing a System in the OPD Editor .....                                  | 5-27 |
| 5.6.5. Removing a System .....                                                   | 5-27 |
| 5.7. Creating a New Entity .....                                                 | 5-28 |
| 5.7.1. Creating a New Entity from an Existing Entity .....                       | 5-28 |
| 5.8. Deleting an Entity Model from a Model Set .....                             | 5-29 |
| 5.9. Navigating Among Edited Entities .....                                      | 5-29 |
| 5.10. Editing the Create Menu .....                                              | 5-29 |
| 5.10.1. Generating the Create Menu Automatically .....                           | 5-30 |
| 5.10.2. Editing the Create Menu Manually .....                                   | 5-31 |
| 5.10.3. Undoing (Reverting) Changes to the Create Menu .....                     | 5-33 |
| 5.11. Editing the Category List .....                                            | 5-34 |
| 5.11.1. Adding a New Category .....                                              | 5-34 |
| 5.11.2. Removing a Category .....                                                | 5-34 |
| 5.11.3. Renaming a Category .....                                                | 5-35 |
| 5.11.4. Undoing Category Changes .....                                           | 5-35 |

|                                      |      |
|--------------------------------------|------|
| 5.12. Managing the Forces List ..... | 5-35 |
| 5.12.1. Adding a Force .....         | 5-36 |
| 5.12.2. Editing a Force .....        | 5-37 |
| 5.12.3. Removing a Force .....       | 5-37 |

## **Chapter 6     Advanced Entity Model Editing**

|                                                            |      |
|------------------------------------------------------------|------|
| 6.1. The Object Parameter Database Editor .....            | 6-2  |
| 6.2. Starting the OPD Editor .....                         | 6-2  |
| 6.3. Loading an Object Parameter Database .....            | 6-3  |
| 6.3.1. Changing the Object List View .....                 | 6-4  |
| 6.4. Editing an Object's Parameters .....                  | 6-4  |
| 6.4.1. Parameter Descriptions .....                        | 6-4  |
| 6.4.2. Editing Individual Parameters .....                 | 6-5  |
| 6.4.3. Editing Parameters for Multiple Objects .....       | 6-6  |
| 6.4.4. Deleting Parameters and Components .....            | 6-6  |
| 6.4.5. Restoring Parameters .....                          | 6-6  |
| 6.5. Creating a New Object from an Existing Object .....   | 6-7  |
| 6.6. Adding New Components and Parameters .....            | 6-7  |
| 6.6.1. Adding New Components .....                         | 6-8  |
| 6.6.2. Editing Components .....                            | 6-8  |
| 6.6.3. Adding New Resource Parameters .....                | 6-10 |
| 6.6.4. Adding New Subordinate Object Parameters .....      | 6-10 |
| 6.6.5. Adding New Soil Type Parameters .....               | 6-11 |
| 6.7. Editing Variables .....                               | 6-11 |
| 6.8. Editing Systems .....                                 | 6-11 |
| 6.9. Saving Changes to the Object Parameter Database ..... | 6-13 |

## **Chapter 7     The Object Parameter Database**

|                                                                    |      |
|--------------------------------------------------------------------|------|
| 7.1. The Object Parameter Database .....                           | 7-2  |
| 7.1.1. VR-Forces Classes and Object Parameters .....               | 7-6  |
| 7.1.2. Object Types .....                                          | 7-6  |
| 7.1.3. Parameter Types .....                                       | 7-9  |
| 7.1.4. Local and Remote Subcomponents .....                        | 7-10 |
| 7.2. Component Descriptors and Parameters .....                    | 7-13 |
| 7.2.1. Common Elements of Component Descriptors .....              | 7-14 |
| 7.2.2. Configuring the Priority of Components .....                | 7-16 |
| 7.3. Adding a New Entity to the Object Parameter Database .....    | 7-17 |
| 7.4. Systems Definition Files .....                                | 7-18 |
| 7.4.1. Creating a System Definition File .....                     | 7-19 |
| 7.4.2. Configuring System Connections .....                        | 7-19 |
| 7.4.3. Adding Variable Bindings .....                              | 7-22 |
| 7.5. Configuring Aggregate Entities .....                          | 7-23 |
| 7.5.1. Adding an Aggregate to the Object Parameters Database ..... | 7-23 |
| 7.6. Changing the Echelon ID Category and Echelon ID Level .....   | 7-26 |
| 7.7. Configuring Graphical Objects .....                           | 7-27 |

|                                                                      |      |
|----------------------------------------------------------------------|------|
| 7.8. Pathname Interpretation in Simulation Model Set Files .....     | 7-27 |
| 7.9. Updating Object Parameter Database Files for New Releases ..... | 7-28 |

## **Chapter 8     Configuring Object Behaviors**

---

|                                                                 |      |
|-----------------------------------------------------------------|------|
| 8.1. Configuring Sensors .....                                  | 8-3  |
| 8.1.1. Configuring a Sensor in an OPE File .....                | 8-3  |
| 8.1.2. Editing a Sensor System Definition File .....            | 8-4  |
| 8.1.3. Specifying a Sensor Signature .....                      | 8-5  |
| 8.1.4. Adding a New Sensor Type .....                           | 8-5  |
| 8.2. Detection Probability Tables .....                         | 8-8  |
| 8.3. Probability and Selection Tables for Munitions .....       | 8-9  |
| 8.3.1. Ammo Select Tables .....                                 | 8-9  |
| 8.3.2. Hit Probability Tables .....                             | 8-10 |
| 8.3.3. Damage Probability Tables .....                          | 8-12 |
| 8.4. Configuring an Entity to Take Damage from a Munition ..... | 8-13 |
| 8.5. Configuring Formations .....                               | 8-14 |
| 8.5.1. Formation Files .....                                    | 8-14 |
| 8.5.2. Creating a User-Defined Formation .....                  | 8-15 |
| 8.6. Configuring Emitters .....                                 | 8-16 |
| 8.7. Configuring Radios .....                                   | 8-18 |
| 8.7.1. Configuring Transmitters .....                           | 8-18 |
| 8.8. Configuring Task Execution Rules .....                     | 8-19 |
| 8.9. Configuring Joysticks .....                                | 8-20 |
| 8.10. Configuring Obstruction Avoidance .....                   | 8-21 |
| 8.10.1. Configuring Object Types .....                          | 8-21 |
| 8.10.2. Configuring Vector Feature Types .....                  | 8-22 |
| 8.11. Configuring Embarkation .....                             | 8-23 |
| 8.11.1. Editing Embarkation Parameters .....                    | 8-23 |
| 8.12. Configuring VR-Forces Components on Remote Entities ..... | 8-25 |

## **Chapter 9     Using VR-Forces with an External Comms Effects Server**

---

|                                                            |     |
|------------------------------------------------------------|-----|
| 9.1. Introduction .....                                    | 9-2 |
| 9.2. Configuring VR-Forces .....                           | 9-2 |
| 9.2.1. Enabling Radio Transmitter Publishing .....         | 9-3 |
| 9.2.2. Enabling the External Communication Model .....     | 9-4 |
| 9.2.3. Configuring the External Communications Model ..... | 9-5 |
| 9.3. Using VR-Forces with Qualnet .....                    | 9-5 |
| 9.3.1. Running VR-Forces with Qualnet .....                | 9-6 |
| 9.4. Example Scenario .....                                | 9-6 |
| 9.4.1. Running the Scenario .....                          | 9-7 |

## Chapter 10 Configuring 3D Models

|                                                                 |       |
|-----------------------------------------------------------------|-------|
| 10.1. Introduction to VR-Forces 3D Object Modeling .....        | 10-3  |
| 10.1.1. Entity, Object, and Effects Models .....                | 10-3  |
| 10.2. Mapping Models to Entities and Objects .....              | 10-4  |
| 10.2.1. Adding a Model Mapping .....                            | 10-5  |
| 10.2.2. Editing a Model Mapping .....                           | 10-6  |
| 10.2.3. Deleting a Model Mapping .....                          | 10-7  |
| 10.3. Creating and Editing Model Definitions .....              | 10-8  |
| 10.3.1. Creating a Model Definition .....                       | 10-9  |
| 10.3.2. Deleting a Model Definition .....                       | 10-9  |
| 10.3.3. Copying a Model Definition .....                        | 10-10 |
| 10.3.4. Filtering the List of Model Definitions .....           | 10-10 |
| 10.3.5. Editing a Model Definition .....                        | 10-11 |
| 10.3.6. Saving Model Definitions .....                          | 10-13 |
| 10.4. Configuring Model Instancing .....                        | 10-14 |
| 10.4.1. Clearing the Model Instancing Cache .....               | 10-14 |
| 10.5. Creating and Editing Model Definition Schemata .....      | 10-15 |
| 10.5.1. Creating Model Definition Schemas .....                 | 10-15 |
| 10.5.2. Deleting Model Definition Schemas .....                 | 10-16 |
| 10.5.3. Copying Model Definition Schemas .....                  | 10-17 |
| 10.5.4. Adding a Parameter to a Model Definition Schema .....   | 10-17 |
| 10.5.5. Deleting a Parameter from a Model Definition Schema ... | 10-18 |
| 10.5.6. Editing a Model Definition Schema Parameter .....       | 10-18 |
| 10.6. Configuring SpeedTree Trees .....                         | 10-19 |
| 10.6.1. Specifying the SpeedTree Performance Profile .....      | 10-20 |
| 10.7. Adding a Model to VR-Forces - A Tutorial .....            | 10-20 |
| 10.7.1. Create a Model Definition .....                         | 10-21 |
| 10.7.2. Add a Model Mapping .....                               | 10-24 |
| 10.7.3. Test the Model Mapping .....                            | 10-26 |

## Chapter 11 Configuring 3D Windows and Channels

|                                                              |       |
|--------------------------------------------------------------|-------|
| 11.1. The VR-Vantage Display Engine .....                    | 11-2  |
| 11.1.1. How VR-Forces Responds to Display Engine Connections | 11-2  |
| 11.1.2. The Display Engine's Connection States .....         | 11-3  |
| 11.2. Windows, Channels, and the Observer .....              | 11-3  |
| 11.3. Managing Display Engine Configurations .....           | 11-3  |
| 11.3.1. Adding a Window .....                                | 11-4  |
| 11.3.2. Adding a Channel to a Window .....                   | 11-6  |
| 11.3.3. Removing a Window .....                              | 11-6  |
| 11.3.4. Removing a Channel .....                             | 11-6  |
| 11.3.5. Saving a Display Engine Configuration .....          | 11-7  |
| 11.3.6. Loading a Display Engine Configuration .....         | 11-7  |
| 11.4. Changing a Window's Properties .....                   | 11-7  |
| 11.5. Changing a Channel's Properties .....                  | 11-9  |
| 11.5.1. Setting the Clipping Planes .....                    | 11-11 |

|                                                                  |       |
|------------------------------------------------------------------|-------|
| 11.5.2. Specifying the Projection Resize Policy Property .....   | 11-12 |
| 11.5.3. Changing a Channel's Frustum (Field of View) .....       | 11-14 |
| 11.5.4. Changing the Viewport .....                              | 11-15 |
| 11.6. Starting a VR-Vantage Display Engine .....                 | 11-16 |
| 11.7. Connecting to a Display Engine .....                       | 11-17 |
| 11.7.1. Disconnecting from a Display Engine .....                | 11-18 |
| 11.8. Configuring Multi-Channel Displays .....                   | 11-19 |
| 11.8.1. Changing the Camera's Position and Orientation Offset .. | 11-21 |

## Appendix A Editing the Create Menu

---

|                                                                      |     |
|----------------------------------------------------------------------|-----|
| A.1. The Create Menu .....                                           | A-2 |
| A.2. The <i>vrf.ent</i> Configuration File .....                     | A-2 |
| A.2.1. Object Blocks in <i>vrf.ent</i> .....                         | A-4 |
| A.2.2. Reorganizing the Create Menu .....                            | A-4 |
| A.2.3. Editing Menu Items .....                                      | A-4 |
| A.3. Adding New Entity Types to VR-Forces .....                      | A-5 |
| A.3.1. Adding a New Entity or Control Object to <i>vrf.ent</i> ..... | A-5 |
| A.4. Editing the Overlay Object Menus .....                          | A-5 |

## Appendix B Object Parameters

---

|                                                   |      |
|---------------------------------------------------|------|
| B.1. The Structure of an Entity Parameter .....   | B-2  |
| B.2. The Parameter Type Hierarchy .....           | B-3  |
| B.2.1. Vrf Base Object Parameters .....           | B-3  |
| B.2.2. Vrf Object Parameters .....                | B-7  |
| B.2.3. Moving Object Parameters .....             | B-8  |
| B.2.4. Parameter Types for Entity Platforms ..... | B-8  |
| B.3. Entity Parameters .....                      | B-8  |
| B.4. Object Type Enumerations .....               | B-9  |
| B.4.1. Object Super-type .....                    | B-9  |
| B.4.2. AggregateKind .....                        | B-9  |
| B.4.3. ObjectKind .....                           | B-10 |
| B.4.4. Aggregate Category .....                   | B-10 |
| B.4.5. Aggregate Specific .....                   | B-11 |
| B.4.6. Point Object Category .....                | B-12 |
| B.4.7. LinearObjectKind .....                     | B-12 |
| B.4.8. LinearObjectCategory .....                 | B-12 |
| B.4.9. Interaction .....                          | B-12 |

## Appendix C Creating and Editing Key Maps

---

|                                     |     |
|-------------------------------------|-----|
| C.1. The Key Map Editor .....       | C-2 |
| C.1.1. Binary Key Mappings .....    | C-2 |
| C.2. Editing a Key Map .....        | C-3 |
| C.2.1. Adding a Key Mapping .....   | C-4 |
| C.2.2. Changing a Key Mapping ..... | C-5 |

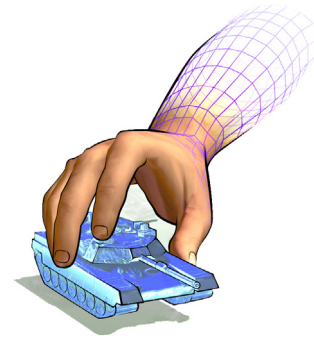


---

|                                             |      |
|---------------------------------------------|------|
| C.2.3. Deleting a Key Mapping .....         | C-5  |
| C.3. Using Combined Key Mappings .....      | C-6  |
| C.4. Filtering the Function List .....      | C-6  |
| C.5. Creating a Key Map .....               | C-7  |
| C.6. Deleting a Key Map .....               | C-7  |
| C.7. Loading a Key Map .....                | C-7  |
| C.8. Keyboard Mappings .....                | C-7  |
| C.9. Mouse Navigation Button Mappings ..... | C-11 |

## **Index**





# Preface

---

This manual is written for persons who want to configure or customize VR-Forces® without using the API. The manual assumes that you are familiar with the use of VR-Forces, as described in *VR-Forces Users Guide*.

Please see release documentation for information about changes and updates to VR-Forces since this manual went to press.

## ***How This Manual is Organized***

The chapters are organized as follows:

Chapter 1, *Optimizing Performance* describes how to monitor VR-Forces's performance in the Frame Rate Monitor and how to configure VR-Forces to improve performance.

Chapter 2, *VR-Forces Configuration Files* explains the syntax of configuration files and the parameters in the configuration files provided with VR-Forces.

Chapter 3, *Scenario Files* describes the files used to define a scenario and how to edit them.

Chapter 4, *Merging Scenarios* explains how to use the Scenario Merge tool to merge scenarios.

Chapter 5, *Editing Entity Models* explains how to use the Entity Editor to edit entity parameters, systems, and symbol mappings. It also explains how to create custom Create menus.

Chapter 6, *Advanced Entity Model Editing* explains how to use the OPD Editor to edit entity and system parameters. The OPD Editor lets you edit all entity parameters, while the Entity Editor only lets you edit some parameters.

Chapter 7, *The Object Parameter Database* describes the object parameter database, which contains parameters for entities and control objects, and how to edit it.

Chapter 8, *Configuring Object Behaviors* describes additional configuration files that VR-Forces uses to manage munitions, formations, and joystick use.

Chapter 9, *Using VR-Forces with an External Comms Effects Server* explains how to configure VR-Forces to work with external communications effects servers, such as QualNet.

Chapter 10, *Configuring 3D Models* explains how to map models to objects in the VR-Forces 3D Front End.

Chapter 11, *Configuring 3D Windows and Channels* explains how to configure the windows and channels in a display engine in the VR-Forces 3D Front End.

Appendix A, *Editing the Create Menu* explains the format of the *vr.ent* file and how to edit it to change the listings on the Create menu.

Appendix B, *Object Parameters* describes the high level parameters used in OPE files.

Appendix C, *Creating and Editing Key Maps* explains how to edit key mappings and create new key mappings in the VR-Forces 3D Front End.

## VR-Forces Documentation

The VR-Forces documentation set is as follows:

- *VR-Forces Getting Started Guide* is a quick introduction to VR-Forces. It covers the basics of installing VR-Forces, running a scenario, and creating a scenario. It focuses on helping new users avoid common mistakes.
- *VR-Forces Users Guide* describes how to install and use the VR-Forces program. It is distributed in printed format and as an online manual in Adobe Portable Document Format (PDF).
- *VR-Forces 3D Front End Users Guide* explains how to use the 3D features in the VR-Forces 3D Front End. Configuration of the 3D Front End is covered in *VR-Forces Configuration Guide*. It is distributed in printed format and in PDF format.
- *VR-Forces Configuration Guide* explains advanced features for configuring performance and how to edit VR-Forces configuration files. It includes documentation of the Entity Editor, OPD Editor, and Scenario Merge tools. It is distributed in printed format and in PDF format.
- *VR-Forces Developers Guide* explains how to use the VR-Forces APIs. It is distributed in printed format and in PDF format.
- *VR-Vantage Developers Guide* describes the VR-Vantage API. This is the API used to build the VR-Forces 3D Front End. Developers who want to customize or extend the 3D Front End will need to refer to this manual.
- *VR-Forces Master Index* combines the indexes from the VR-Forces manuals, including TDB Tool, to help you find references to subjects that are covered in two or more manuals or locate the manual in which a particular topic is discussed. It is distributed only in PDF format.

- ♦ *VR-Forces Migration Guide* collates API migration information for each release in the VR-Forces 3.x series. It is distributed only in PDF format.
- ♦ Online help. The VR-Forces front-end, the OPD Editor, and the TDB Tool have online help accessible from the Help menu.
- ♦ Class documentation. Classes are documented in HTML.
- ♦ *VR-Forces Release Notes*. Release documentation consists of release notes provided as a PDF file and in printed format.

The PDF files are in the *./doc* directory.

### Online Manuals

The online manuals (PDF) use hypertext links for the entries in the table of contents, for all index entries, and for cross references. PDF documentation also supports full text searching.

To view and print PDF documents, you need a copy of the free Acrobat Reader software (version 5.0 or later). To obtain a copy, visit [www.adobe.com](http://www.adobe.com).

## MÄK Products

VR-Forces is a member of the VT MÄK line of software products designed to streamline the process of developing and using networked simulated environments. The VT MÄK 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), the Distributed Interactive Simulation (DIS) protocol, and the Test and Training Enabling Architecture (TENA). VR-Link has built-in support for the 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.

- ♦ MÄK RTI. An RTI (Run-Time Infrastructure) is required to run applications using the High Level Architecture (HLA). The MÄK RTI is optimized for high performance. It has an API, RTIsby®, that allows you to extend the RTI using plug-in modules. It also has a graphical user interface (GUI) for the MÄK RTI *rtiexec*.

- ♦ **VR-Forces®.** VR-Forces is a computer generated forces application and toolkit. It provides an application with a GUI, that gives you a two dimensional view of a terrain database.

You can create and view local entities, aggregate them into hierarchical units, assign tasks, set state parameters, and create plans that have tasks, set statements, and conditional statements. VR-Forces also functions as a plan view display for viewing remote entities 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 your 3D visualization solution. It includes two end-user applications, MÄK Stealth and Vantage IG, and the VR-Vantage Toolkit. The Stealth displays a realistic, 3D view of your virtual world. You can view this world from the inside of a simulated moving vehicle, or place the eyepoint at another moving or stationary location. The Stealth lets you switch rapidly among several predefined viewpoints while the simulation is underway.

Vantage IG is a configurable desktop image generator for out the window (OTW) scenes and remote camera views. It has most of the features of the Stealth, but is optimized for its IG function.

The VR-Vantage Toolkit is a 3D visual application development toolkit. Use it to customize or extend Stealth and Vantage IG 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.

- ♦ **MÄK 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.
- ♦ **MÄK Plan View Display.** The Plan View Display (PVD) provides a 2D, “big picture” of a virtual environment. It shows simulated entities, such as vehicles, soldiers, and tanks moving over a 2D-map display.
- ♦ **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 limited DIS translation.
- ♦ **MÄK Gateway.** The Gateway is a translator application that allows DIS exercises to operate as members of HLA federation executions that use the RPR FOM, thereby preserving the value of simulation code written to support DIS. The MÄK Gateway is a standalone application and does not require you to write additional code for its use.

## ***How to Contact Us***

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

### **Telephone**

|                    |        |                                        |
|--------------------|--------|----------------------------------------|
| Call or fax us at: | Voice: | 617-876-8085 (extension 3 for support) |
|                    | Fax:   | 617-876-9208                           |

### **E-mail**

|                                |                                                              |
|--------------------------------|--------------------------------------------------------------|
| Sales and upgrade information: | <a href="mailto:info@mak.com">info@mak.com</a>               |
| Technical support:             | <a href="mailto:support@mak.com">support@mak.com</a>         |
| VR-Vantage support:            | <a href="mailto:vrv-support@mak.com">vrv-support@mak.com</a> |

### **Internet**

|                                           |                                                                                                      |
|-------------------------------------------|------------------------------------------------------------------------------------------------------|
| MÄK web site home page:                   | <a href="http://www.mak.com">www.mak.com</a>                                                         |
| License key requests:                     | <a href="http://www.mak.com/support/licenses.php">www.mak.com/support/licenses.php</a>               |
| Product version and platform information: | <a href="http://www.mak.com/support/productversions.php">www.mak.com/support/productversions.php</a> |
| For the free, unlicensed MÄK RTI:         | <a href="http://www.mak.com/products/rti.php">www.mak.com/products/rti.php</a>                       |
| Support FAQ:                              | <a href="http://www.mak.com/support/faq.php">www.mak.com/support/faq.php</a>                         |

### **Post**

|                                |                                                      |
|--------------------------------|------------------------------------------------------|
| Send postal correspondence to: | VT MÄK<br>68 Moulton St.<br>Cambridge, MA, USA 02138 |
|--------------------------------|------------------------------------------------------|

When requesting support, please tell us the product you are using, the version, and the platform on which you are running.

## Document Conventions

This manual uses the following typographic conventions:

|                          |                                                                                                                                                                                                                                 |
|--------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Monospaced</b>        | Indicates commands or values you enter.                                                                                                                                                                                         |
| <b>Monospaced Bold</b>   | Indicates a key on the keyboard.                                                                                                                                                                                                |
| <i>Monospaced Italic</i> | 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 MÄK products run on both UNIX and Windows PC platforms, we use the / (slash) for generic discussions of pathnames. If you are running on a PC, substitute a \ (backslash) when you type pathnames. |
| <i>Italic</i>            | Indicates a file name, pathname, or a class name.                                                                                                                                                                               |
| sans Serif               | Indicates a parameter or argument.                                                                                                                                                                                              |
| ➤                        | Indicates a one-step procedure.                                                                                                                                                                                                 |
| Menu → Option            | Indicates a menu choice. For example, an instruction to select the Save option from the File menu appears as:<br>Choose <b>File</b> → <b>Save</b> .                                                                             |
| <b>i</b>                 | Indicates supplemental or clarifying information.                                                                                                                                                                               |
| <b>!</b>                 | 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-Forces home directory. For example, the directory *vrforces/doc* appears in the text as *./doc*.

## Hardware and Operating System Naming Conventions

Unless otherwise specified, references to UNIX include Linux, regardless of the type of computer it is running on. References to PCs refer to Intel processor-compatible computers running a version of Microsoft Windows.



## **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 popup 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.

## ***Third Party Licenses***

MÄK software products may use code from third parties. This section contains the license documentation required by these third parties.

### **Boost License**

VR-Link, and all MÄK software that uses VR-Link uses some code which is distributed under the Boost License. All header files that contain Boost code are properly attributed. The boost web site is: [www.boost.org](http://www.boost.org).

Boost Software License - Version 1.0 - August 17th, 2003

Permission is hereby granted, free of charge, to any person or organization obtaining a copy of the software and accompanying documentation covered by this license (the “Software”) to use, reproduce, display, distribute, execute, and transmit the Software, and to prepare derivative works of the Software, and to permit third-parties to whom the Software is furnished to do so, all subject to the following:

The copyright notices in the Software and this entire statement, including the above license grant, this restriction and the following disclaimer, must be included in all copies of the Software, in whole or in part, and all derivative works of the Software, unless such copies or derivative works are solely in the form of machine-executable object code generated by a source language processor.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

## **libXML and libICONV**

VR-Link and all MÄK software that uses VR-Link, links in libXML and libICONV. On some platforms the compiled libraries and header files are distributed with MÄK Products. MÄK has made no modifications to these libraries. For more information about these libraries please see the following web sites:

- The LGPL license is available at: <http://www.gnu.org/licenses/lgpl.html>
- Information about IconV is at: <http://www.gnu.org/software/libiconv/>
- Information about LibXML is at: <http://xmlsoft.org/>

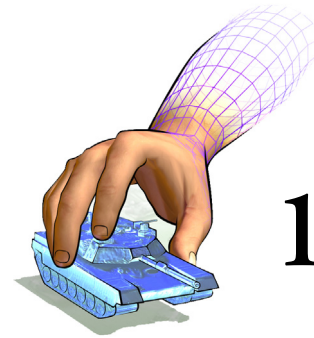
## **pThreads Library**

VR-Exchange links with the pThreads win32 library. The library is distributed under the GNU Lesser General Public License (LGPL). MÄK has made no modification to this library. For information about the pThreads win32 library please see: <http://sourceware.org/pthreads-win32/>

## **EHS, PCRE, and PME**

The MÄK RTI links with the EHS, PCRE, and PME libraries. These libraries are distributed under the GNU Lesser General Public License (LGPL). MÄK has made modification to these libraries only to allow them to compile on the supported platforms. For more information about these libraries please see the following web sites:

- EHS: <http://xaxxon.slackworks.com/ehs/>
- PCRE: <http://www.pcre.org/>
- PME: <http://xaxxon.slackworks.com/pme/index.html>



# *Optimizing Performance*

---

This chapter explains how you can view performance statistics for the front-end and back-end and how you can optimize GUI performance.

|                                                            |      |
|------------------------------------------------------------|------|
| Introduction .....                                         | 1-3  |
| Adjusting Refresh Rates .....                              | 1-4  |
| Changing the Frame Rate for the Display .....              | 1-4  |
| Changing the Network Frame Rate .....                      | 1-5  |
| Monitoring Front-end and Back-end Frame Rates .....        | 1-5  |
| Viewing Frame Rate Statistics.....                         | 1-5  |
| Configuring Statistics Reporting.....                      | 1-7  |
| Tuning Front-End Performance .....                         | 1-8  |
| The Meeting Frame Rate Targets Level.....                  | 1-8  |
| The Not Meeting Frame Rate Targets Level.....              | 1-8  |
| The Heavy Network Traffic Level.....                       | 1-9  |
| Miscellaneous Performance Configuration Options .....      | 1-10 |
| Turn Off Track Histories .....                             | 1-10 |
| Turn Off the Display of Polygons .....                     | 1-10 |
| Limit Use of Object Transparency .....                     | 1-10 |
| Limit Use of Spot Reports .....                            | 1-10 |
| Configure Hazard Cloud Graphics .....                      | 1-11 |
| Configure Intervisibility Intersection Testing.....        | 1-11 |
| Using Asynchronous I/O for DIS Exercises .....             | 1-11 |
| Setting the Tick Rate .....                                | 1-12 |
| Specifying the Tick Rate for Components .....              | 1-12 |
| Tuning the Network Interface .....                         | 1-13 |
| Tuning the State Repository Tick Rate .....                | 1-13 |
| Balancing Visual Quality Against Network Performance ..... | 1-14 |

## **Optimizing Performance**

---

|                                                                       |      |
|-----------------------------------------------------------------------|------|
| Balancing Visual Quality Versus Performance in the 3D Front End ..... | 1-15 |
| Configuring Performance Options .....                                 | 1-15 |
| Tuning the Target Frame Rate.....                                     | 1-16 |
| Optimizing Performance for the 3D Front End .....                     | 1-16 |
| Displaying Performance Statistics .....                               | 1-17 |

## 1.1. Introduction

This chapter describes ways that you can optimize performance through the GUI, the object parameter database, and command line options. You can also optimize performance programmatically. Please see Section 13.12, “[Optimizing GUI Performance](#)” in *VR-Forces Developers Guide* for details.

VR-Forces’s performance varies based on the following factors:

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

VR-Forces supports the following methods for optimizing performance:

- ♦ Adjusting the refresh rates
- ♦ Turning off processor-intensive features
- ♦ Adjusting the network processing time
- ♦ Specifying the tick rate for an entity or component
- ♦ Specifying a tick rate for the network interface
- ♦ Specifying the VR-Forces sleep time
- ♦ Specifying the number of objects that you expect to simulate in the exercise
- ♦ Optimizing the terrain. (For details, please see *MÄK Terrain Database Tool Users Guide*.)

## 1.2. Adjusting Refresh Rates

VR-Forces supports two types of refresh rates, the network frame rate and the display frame rate. The network frame rate determines how many times per second the network is polled for new information. The display frame rate determines how many times per second the display is updated with information received from the network. You can set these frame rates on the General tab of the Display Options dialog box.

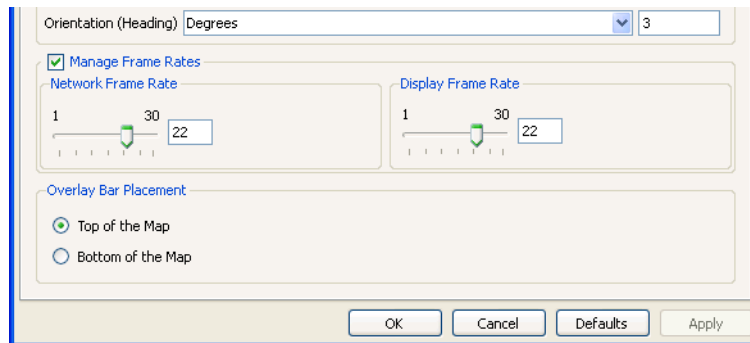


Figure 1-1. Display Options dialog box, General tab

### 1.2.1. Changing the Frame Rate for the Display

The display frame rate specifies the frequency with which VR-Forces updates the display. Updating the display less frequently may improve overall performance. However, depending on the speed at which entities are moving, it could introduce some jerkiness to their motion.

To change the display frame rate:

1. Choose **Options** → **Display**. The Display Options dialog box opens.
2. On the General tab (Figure 1-1), adjust the Display Frame Rate slider or enter a value from 1 to 30 in the text box.
3. Click OK.

The display frame rate does not affect the frame rate of the simulation source.

### 1.2.2. Changing the Network Frame Rate

The network frame rate specifies the frequency with which VR-Forces reads network input. Less frequent network reads may improve performance, but may result in lost information.

To change the network frame rate:

1. Choose **Options** → **Display**. The Display Options dialog box opens.
2. On the General tab ([Figure 1-1](#)), adjust the Network Frame Rate slider or enter a value from 1 to 30 in the text box.
3. Click OK.

## 1.3. Monitoring Front-end and Back-end Frame Rates

You can monitor the network and display frame rates in the Frame Rate Monitor. You can also monitor frame rate statistics for each back-end in your session, and you can specify:

- ♦ How frequently statistics are reported
- ♦ The sample size used to calculate statistics.

### 1.3.1. Viewing Frame Rate Statistics

The Statistics tab on the Frame Rate Monitor displays:

- ♦ The desired display and network frame rates, as specified in the Display Options dialog box (see [“Adjusting Refresh Rates,”](#) on page 1-4)
  - ♦ The actual frame rates
  - ♦ The GUI performance level
  - ♦ Frame rate statistics for each back-end in the session.
- To view frame rate statistics, choose **Options** → **Frame Rate**. The Frame Rate Monitor opens ([Figure 1-2](#)). The frame rate statistics are displayed in the window on the Statistics tab.

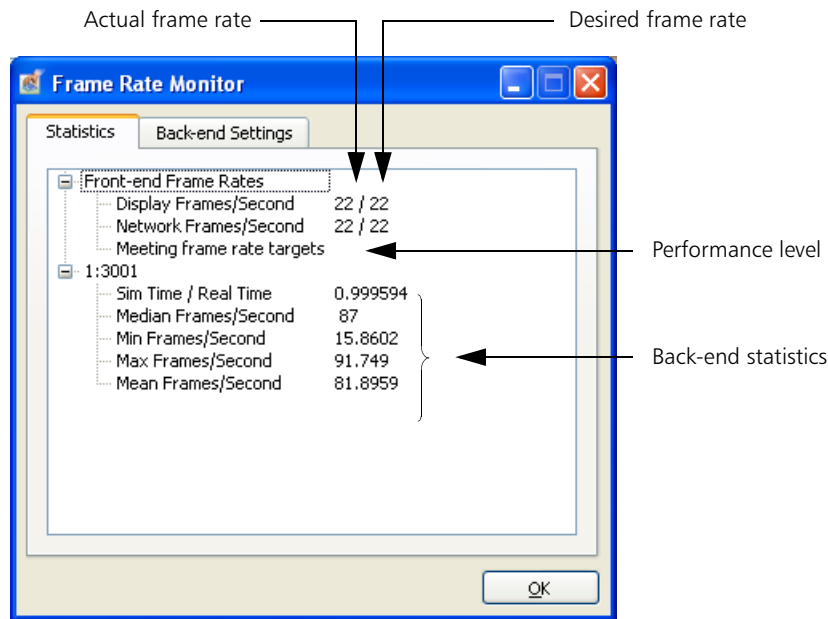


Figure 1-2. The Frame Rate Monitor

- To view the statistics for a back-end, click the plus sign next to its site:application number.

Table 1-1 describes the statistics displayed in the Frame Rate Monitor. For details about back-end statistics, please see [“Configuring Statistics Reporting,”](#) on page 1-7.

Table 1-1: Frame Rate Monitor statistics

| Statistic             | Description                                                                                       |
|-----------------------|---------------------------------------------------------------------------------------------------|
| Display Frames/Second | Actual frames displayed per second / the target display frame rate.                               |
| Network Frames/Second | Actual network frame rate / the target network frame rate.                                        |
| Performance status    | Indicates whether the GUI is meeting frame rate, not meeting the frame rate, or under heavy load. |



### 1.3.2. Configuring Statistics Reporting

On the Back-end Settings tab of the Frame Rate Monitor, you can specify:

- How frequently a back-end reports frame rate statistics
- The sample size to use for frame rate statistics.

You can also configure logging of frame rate statistics with the `logFrameRateStatistics` parameter in `vrfSim.mtl`.

To specify the back-end statistics frequency and sampling size:

1. Choose **Options** → **Frame Rate**. The Frame Rate Monitor dialog box opens (Figure 1-2).
2. Select the Back-end Settings tab (Figure 1-3).

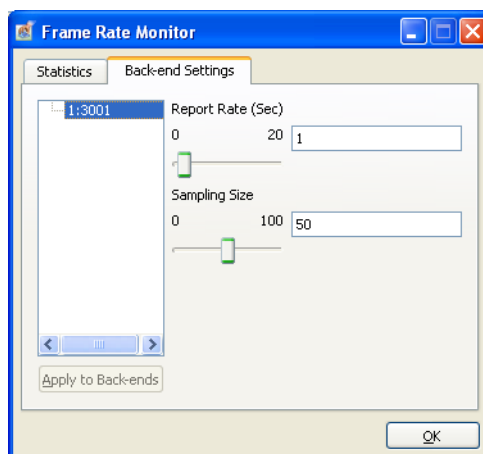


Figure 1-3. Frame Rate Monitor, Back-end Settings tab

3. In the list, select the back-end for which you want to configure the statistics display.
4. To specify the frequency, in seconds, that the back-end reports statistics, adjust the Report Rate slider.
5. To specify the number of frames that the back-end uses to calculate statistics, adjust the Sampling Size slider.
6. Click Apply to Back-ends.



When you apply changes to a back-end, the change affects reporting of statistics to all front-ends that are part of the session.

7. Repeat steps 3 through 6 for each back-end that you want to configure.

## 1.4. Tuning Front-End Performance

The Frame Rate Monitor provides feedback about VR-Forces's performance ([Figure 1-2](#)). Performance is characterized as being at one of the following levels:

- ♦ Meeting frame rate targets
- ♦ Not meeting the frame rate targets
- ♦ Heavy traffic.

### 1.4.1. The Meeting Frame Rate Targets Level

When VR-Forces is meeting frame rate targets, it is polling the network and updating the display at the frame rates specified in the Display Options dialog box.

### 1.4.2. The Not Meeting Frame Rate Targets Level

When VR-Forces cannot meet the frame rate targets set in the Display Options dialog box, it is in the Not Meeting Frame Rate Targets Level. Performance might still be acceptable. At this level you might want to consider changing some of the performance parameters described in [“The Heavy Network Traffic Level,”](#) on page 1-9.

### 1.4.3. The Heavy Network Traffic Level

VR-Forces enters the heavy network traffic level when it determines that reading and processing information from the network is at a level that would make it difficult to use VR-Forces. The front-end may become less responsive to user input. In heavy network traffic level, VR-Forces reduces the frequency with which it polls the network for new information and displays it on the screen. This allows you to interact through the front-end. Heavy traffic mode is disabled by default, and, if enabled, should only happen rarely, if at all.

When VR-Forces is in heavy network traffic mode, it monitors its performance. If the performance improves, it enters the “not meeting target” or “meeting target” level. [Table 1-2](#) lists configuration parameters in *vrGui.mtl* that support performance tuning.

Table 1-2: Performance tuning parameters

| Parameter                                  | Description                                                                                                                                                                                                                                                                                                                                      |
|--------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>consecutiveTimesOverThreshold</code> | Specifies the number of consecutive frames the system must exceed <code>networkTimeThreshold</code> before switching to heavy traffic mode. Enforced only if <code>tickManagingEnabled</code> is set to 1.                                                                                                                                       |
| <code>networkTimeThreshold</code>          | Specifies the maximum length of time, in milliseconds, that a network frame should take to execute. Enforced only if <code>tickManagingEnabled</code> is set to 1.                                                                                                                                                                               |
| <code>networkFrameRateBuffer</code>        | Specifies a value to use in determining whether or not to switch from “meeting targets” mode to “not meeting targets” mode. It multiplies <code>networkFrameRateBuffer</code> by the number of milliseconds the network frame took to execute. If the result is greater than the frame rate target, VR-Forces enters “not meeting targets” mode. |
| <code>tickManagingEnabled</code>           | Enables (1) or disables (0) checking for heavy network traffic.                                                                                                                                                                                                                                                                                  |

## 1.5. Miscellaneous Performance Configuration Options

Disabling the following features may improve 2D front-end performance. Some are discussed in greater detail in the rest of this section.

- ♦ Hide the Minimap
- ♦ Turn off track histories
- ♦ Turn off spot reports
- ♦ Turn off the display of polygons
- ♦ Limit use of object transparency
- ♦ Do not display ellipses or bounding volumes for hazard clouds
- ♦ Reduce the types of intersections tested for intervisibility
- ♦ Use asynchronous I/O for DIS.

### 1.5.1. Turn Off Track Histories

If you turn off track histories, performance may improve.

- To turn off track histories, clear the Track Histories check box on the General tab of the Display Options dialog box, or set the `enableHistoryTracking` parameter in `vrfGui.mtl` to 0.

### 1.5.2. Turn Off the Display of Polygons

If you are using a terrain database that has a raster map and you do not need to view the underlying polygons, turning off the display of polygons can increase performance of panning and zooming, particularly when you are zoomed in on the database.

To turn off the display of polygons:

1. Choose **Options** → **Terrain View**. The Terrain View Options dialog box opens.
2. On the Drawing Mode tab, in the Terrain Drawing group box, clear the Polygons check box.
3. Click OK.

### 1.5.3. Limit Use of Object Transparency

Setting transparency for display objects, such as areas and range rings, can degrade performance.

### 1.5.4. Limit Use of Spot Reports

Use of spot reports can degrade performance. You may want to limit the number of entities for which they are generated or disable spot reports entirely.

### 1.5.5. Configure Hazard Cloud Graphics

If you are displaying hazard clouds, displaying images instead of ellipses or bounding volumes can dramatically decrease performance. Therefore, do not use images if performance is an issue.

### 1.5.6. Configure Intervisibility Intersection Testing

When it calculates intervisibility, VR-Forces can check for intersections of polygons, features, and entities. Depending on the complexity of the terrain database and the number of entities in a simulation, increasing the types of intersections for which VR-Forces tests can degrade performance. Therefore, if performance is an issue, minimize the intersection testing being done.

### 1.5.7. Using Asynchronous I/O for DIS Exercises

By its nature, DIS generates a lot of network traffic. Therefore, you may experience problems with dropped packets and entities may time out. You can mitigate this problem by using asynchronous I/O. To use asynchronous I/O, start with the `--useAsyncIO` command line option or set `useAsyncIO` to 1 in *vrfGui.mtl*.

## 1.6. Setting the Tick Rate

VR-Forces lets you tune performance by varying the tick rate of components, state repositories, and network interfaces. By decreasing the frequency with which some entity components are ticked, you make more CPU time available to other entities. For example you could decrease the tick rate for slow moving entities such as dismounted infantry entity or a command post, making more CPU available for entities such as aircraft or missiles.

Variable tick rates are determined by a minimum tick period and a minimum tick period variance. The minimum tick period parameter specifies the shortest period of time, in seconds, that can elapse between ticks of an entity's components. VR-Forces may tick the components less frequently (due to heavy CPU load), but it will never tick the entity's components more frequently than specified.

To reduce the frequency that a component can be ticked, increase the minimum tick period value. To increase the frequency that a component can be ticked, reduce the minimum tick period value.

The minimum tick period variance argument specifies an amount of variance allowed around the minimum tick period. Specifying a minimum tick period variance value helps prevent the spikes in CPU usage that would result if all components tried to tick at the same time. Calculation of the time to tick a component using minimum tick period variance is as follows (the example uses the entity component arguments):

If the current time is  $t_0$  and the next scheduled tick is  $t_{\text{next}}$  then:

$$t_{\text{next}} = t_0 + \text{min-tick-period} +/\text{- rand}(\text{min-tick-period-variance}/2)$$

If you insert some sample values (in milliseconds), you get:

$$t_{\text{next}} = 100 + 50 +/\text{- rand}(50/2)$$

This means that the next tick will be no sooner than the current time plus 150 ms plus or minus a random value between 0 and 12.5 ms.

### 1.6.1. Specifying the Tick Rate for Components

In the object parameter database, you can specify a **min-tick-period** and **min-tick-period-variance** for each entity and for individual entity components. The component-level settings, if any, override the entity-level settings. You can update these parameters in the OPD Editor.

### 1.6.2. Tuning the Network Interface

You can tune the local and remote network interface subcomponents. Specify the `net-interface-min-tick-period` and `net-interface-min-tick-period-variance` parameters in the local and remote subcomponents blocks. If you reduce the rate at which the network interfaces are ticked, you can limit how often VR-Forces needs to do coordinate system conversion from network (geocentric) to database coordinates.



Reducing the frequency which which you tick network interface components could delay the discovery of objects in remote front-ends and back-ends.

---

### 1.6.3. Tuning the State Repository Tick Rate

You can tune the frequency with which state repositories get ticked. Specify the `state-repository-min-tick-period` and `state-repository-min-tick-period-variance` parameters in the local and remote subcomponents blocks. These parameters control how often VR-Forces recomputes the bounding volume of an object based on object geometry.

## 1.7. Balancing Visual Quality Against Network Performance

VR-Forces allows you to tune performance by adjusting dead-reckoning thresholds. The primary benefit of this feature is to improve the appearance of entities in 3D visualization programs such as the MÄK Stealth and the VR-Forces 3D Front End. Reducing thresholds improves the accuracy of entity movement and reduces jumpy movement. However it does so at the expense of network performance, because a lower dead-reckoning threshold results in more packets being sent for each entity.

When you change thresholds at runtime, VR-Forces sends a message to all back-ends in the scenario to change their default thresholds for translation and rotation. The new thresholds remain in effect until the back-ends are closed. They apply to any scenario run during this time and they override thresholds set in *vrfSimSettings.xml*. Changes to the default thresholds get saved in *vrfSimSettings.xml*.

If your exercise sends periodic updates, such as a DIS heartbeat, the thresholds evaluate changes since the most recent heartbeat.

Changes made at runtime have no effect in the following cases:

- If a back-end joins the scenario after the thresholds are sent. It uses the default VR-Link thresholds, or the values in the *vrfSimSettings.xml* file, if specified.
- If an object was instantiated with a customized VR-Link *DtThresholder*.

To configure performance options in the 2D front-end:

1. Choose **Options** → **Performance Options**. The Performance Options dialog box opens ([Figure 1-4](#)).

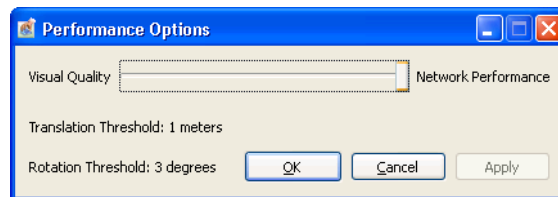


Figure 1-4. Performance Options dialog box

2. Adjust the slider to your desired preference for visual quality versus network performance.
3. Click OK.



### 1.7.1. Balancing Visual Quality Versus Performance in the 3D Front End

The 3D Front End has performance options similar to those in the 2D front-end.

To configure performance options in the 3D Front End:

1. Choose **Settings** → **Performance Options**. The Application Settings dialog box opens to the Performance Options page (Figure 1-5).

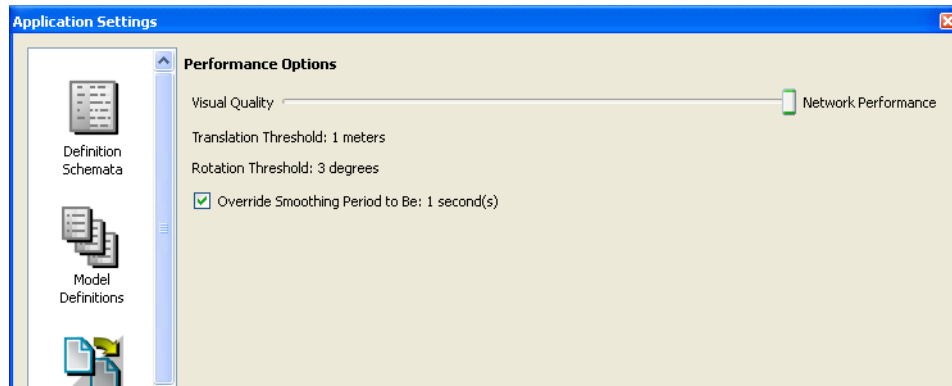


Figure 1-5. Performance Options page

2. Adjust the slider to your desired preference for visual quality versus network performance.
3. Click OK.

### 1.7.2. Configuring Performance Options

The following configuration parameters affect use of the visual quality performance option:

- ♦ `maxDRTranslationThreshold`, specified in *vrGui.mtl*, sets the maximum translation threshold value on the Visual Quality slider in the Performance Options dialog box
- ♦ `maxDRRotationThreshold`, specified in *vrGui.mtl*, sets the maximum rotation-threshold value on the Visual Quality slider in the Performance Options dialog box
- ♦ `dr-allow-gui-overrides`, specified in each OPE file, determines whether a given entity type can have its translation and rotation thresholds overridden by the value set by the Performance Options dialog box.

## 1.8. Tuning the Target Frame Rate

The `--targetFrameRate` command line option and `vrSim.mtl` parameter allow you to specify the mean frame rate above which the VR-Forces back-end should begin to sleep in order to give up CPU time to other processes. This parameter is only applicable to variable frame mode; the fixed frame modes already have a built in sleep that gives up unused time. Increasing the target frame rate will only prevent the back-end from sleeping if it cannot achieve that rate – ultimately it is up to the operating system to determine how much CPU time each process receives.

## 1.9. Optimizing Performance for the 3D Front End

Performance of a visualization product such as the 3D Front End depends on:

- ♦ Processor speed
- ♦ Video card
- ♦ Available memory
- ♦ Size of the terrain database
- ♦ The number and complexity of models loaded
- ♦ Number of objects in the simulation and how active they are.

You can display statistics to help you evaluate the 3D Front End's performance. For details, please see [“Displaying Performance Statistics,”](#) on page 1-17.

The following sections describe options for optimizing performance:

- ♦ [“Specifying the SpeedTree Performance Profile,”](#) on page 10-20
- ♦ [“Configuring File Caching,”](#) on page 10-25, in *VR-Forces 3D Front End Users Guide*
- ♦ [“Considerations and Limitations for Building Terrains,”](#) on page 10-5, in *VR-Forces 3D Front End Users Guide*.

### 1.9.1. Displaying Performance Statistics

The 3D Front End can display an overlay of performance statistics.



Displaying performance statistics may affect performance.

To display performance statistics:

1. Choose **Settings** → **Render Settings**. The Application Settings dialog box opens to the Render Settings page (Figure 1-6).

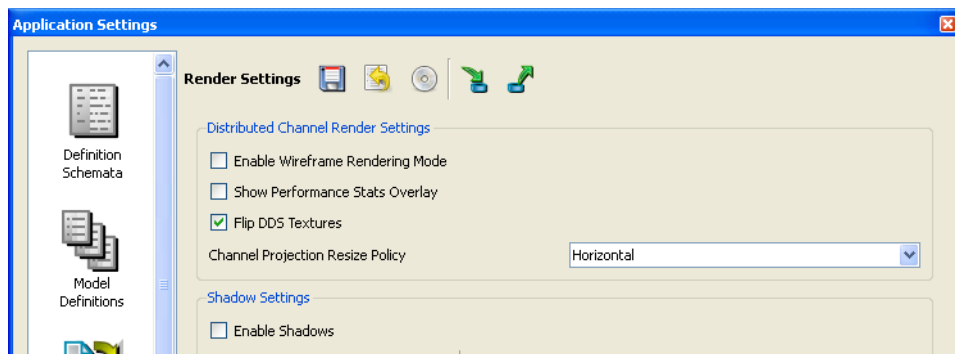


Figure 1-6. Render Settings page

2. Select the Show Performance Stats Overlay check box. A statistics display is added to the window (Figure 1-7).

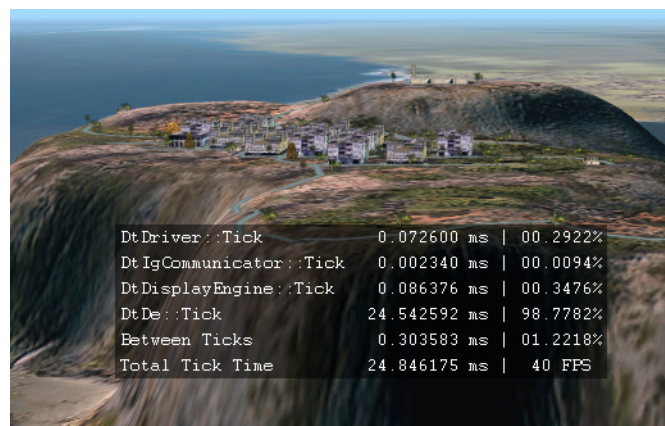
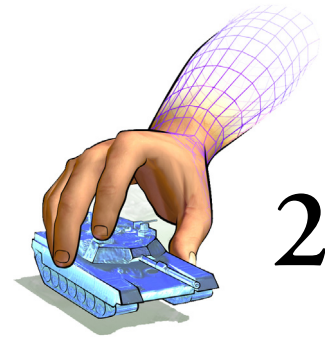


Figure 1-7. Performance statistics overlay





## ***VR-Forces Configuration Files***

---

This chapter describes the syntax of VR-Forces configuration (MTL) files and introduces the configuration files used by VR-Forces.

|                                                                   |      |
|-------------------------------------------------------------------|------|
| MÄK Technologies Lisp (MTL) Files .....                           | 2-2  |
| Using Environment Variables in an MTL File .....                  | 2-2  |
| Specifying Lists of Lists .....                                   | 2-3  |
| Using Conditional Statements .....                                | 2-3  |
| Configuration File Descriptions .....                             | 2-4  |
| The vrfGui.mtl Configuration File .....                           | 2-5  |
| The vrfSim.mtl Configuration File .....                           | 2-14 |
| Symbol Mapping Files .....                                        | 2-18 |
| Symbol Mapping Syntax for Animated Images .....                   | 2-19 |
| Symbol Mapping Syntax for TrueType and Unicode TrueType Fonts ... | 2-20 |
| Changing the Symbol Map File During Runtime .....                 | 2-20 |
| Other Configuration Files .....                                   | 2-20 |
| The plugin.mtl File .....                                         | 2-20 |
| The commonObjects.mtl File .....                                  | 2-20 |
| The rangeRings.mtl Configuration File .....                       | 2-21 |
| The boundingVolume.mtl Configuration File .....                   | 2-21 |
| Configuring Overlay Objects .....                                 | 2-21 |
| Configuring DI-Guy Integration .....                              | 2-22 |
| Configuring Communication Models .....                            | 2-23 |
| Simple Radio Communication Model .....                            | 2-23 |
| External Communication Model .....                                | 2-23 |
| Custom Communication Models .....                                 | 2-24 |

## 2.1. MÄK Technologies Lisp (MTL) Files

Many of the configuration files for MÄK products are ASCII text files that use MÄK Technologies LISP (MTL) to encode configuration information. MTL files have the extension *.mtl*. You can edit them in any text editor. Be sure that your text editor uses the proper end-of-line characters for your platform.

Most parameter entries take the format:

```
(setq parameter_name value)
```

For example:

```
(setq EnableWireFrame 1)
```

Or:

```
(setqb parameter_name value)
```

The `setqb` syntax indicates a bound variable.

To comment out a line, precede the text with two semi-colons (`::`). You can also add a comment at the end of a line by preceding it with two semi-colons.

In most cases, the parameter names are known to the MÄK application and the MTL commands simply assign values to them. However, you can use the `setq` command to create arbitrary symbolic names or aliases that are then used in later commands. For example, in the following two commands, the symbolic name `VaporTrail` is assigned to an OpenFlight file. Then the symbolic name is used in the `trail-map` command. The name `VaporTrail` has no intrinsic meaning. It is given meaning by the `setq` command, which allows it to be used later in the `trail-map` command.

```
(setq VaporTrail (list "../data/models/makTrails/Vapor-Trail" "VaporTrail.flr" 6.0 1.5 1.5 4.0 1.5 0.5))
(trail-map (list 1 2 -1 0 -1 -1 -1)(list VaporTrail 0.0 0.0 0.0))
```

In addition to the `setq` command, a commonly used command is the `load` command. This command instructs the application to load the file specified, for example:

```
(load mtl-path "params.mtl")
```

### 2.1.1. Using Environment Variables in an MTL File

If you want to use an environment variable in an MTL file, use the following syntax:

```
(setqb parameter_name (getenv (quote env_var)))
```

For example:

```
(setqb disDestAddr (getenv (quote DEST_ADDR)))
```

### 2.1.2. Specifying Lists of Lists

The example of setting a symbolic name in a previous section uses a list. You can also specify a list of lists, for example:

```
(trail-map (list 1 2 -1 -1 -1 -1 -1)
  (list vaporTrail -8.0 -3.0 0.0)
  (list vaporTrail -8.0 3.0 0.0)
)
```

A parameter entry can also have multiple individual lists:

```
(trail-map (list 1 2 -1 0 -1 -1 -1) (list VaporTrail 0.0 0.0
  0.0))
```

### 2.1.3. Using Conditional Statements

MTL commands can be conditional. Conditional statements are used frequently when parameters are different for DIS and HLA. A simple conditional statement has the format:

```
(if (condition) (statement_to_evaluate))
```

If you have more than one statement to evaluate, use a sequence block:

```
;; HLA specific parameters
(if (equal HLA 1)
  (sequence
    (setqb fomMapperLib "")
    (setqb fomMapperInitData "")
    (setqb rprFomVersion 1.0)
    (setqb pathToSublistFile "sublist.mtl")
    (setqb ignoreAdvisories 0)
    (setqb fedLookahead 1.0)
  )
)
```

## 2.2. Configuration File Descriptions

VR-Forces (*vrfSim* and *vrfGui*) uses the following configuration files. Unless otherwise noted, they are in *./config*.

- *appearance.mtl* specifies the appearance attributes that can be set in the Set Appearance dialog box.
- *boundingVolume.mtl* specifies bounding volumes of entities for use when calculating intervisibility intersections.
- *capabilities.mtl* specifies capabilities that can be set in the Set Capabilities dialog box.
- *character\_animations\_table.mtl* configures how VR-Forces implements the actions that DI-Guy characters can carry out.
- *character\_digest.xml* specifies the DI-Guy characters that VR-Forces supports.
- *commonObjects.mtl* configures various graphical objects that do not fit logically into the other configuration files.
- *configurationsSettings.xml* stores settings for the Simulation Connections Configuration dialog box.
- *entityEditor.mtl* stores settings for the Entity Editor.
- *entityEditorUi.mtl* configures the Entity Editor.
- *opdEditor.mtl* stores settings for the OPD Editor.
- *overlayMenu.mtl* specifies the overlay objects listed in the overlay objects menus and the Overlay Palette.
- *overlayPalette.mtl* (in *./data/simulationModelSets/<modelset>/gui*) configure how overlay objects are drawn and which symbols are used for them.
- *plugin.mtl* configures which plug-ins get loaded, if any.
- *pvd.mtl* specifies generic configuration information and protocol-specific configuration values.
- *rangeRings.mtl* configures the size and style of range rings.
- *symbolmap-bmp.mtl* (in *./data/simulationModelSets/<modelset>/gui*) maps entity types to bitmap files.
- *symbolmap-images.mtl* and *symbolmap-images-perspective.mtl* (in *./data/simulationModelSets/<modelset>/gui*) map entity types to representational images.
- *symbolmap-ttf.mtl* (in *./data/simulationModelSets/<modelset>/gui*) maps entity types to TrueType fonts.
- *vrf.ent* (in *./data/simulationModelSets/<modelset>/gui*) specifies the entities, aggregates, and control objects listed on the Create menu if you use a custom Create menu. For information about this file, please see [“The vrf.ent Configuration File,”](#) on page A-2.



- ♦ *vrf3DGui.mtl* specifies generic configuration information and protocol-specific configuration values for the 3D front-end.
- ♦ *vrfGui.mtl* specifies generic configuration information and protocol-specific configuration values for the front-end.
- ♦ *vrfGuiSettings.mtl* saves display settings for the front-end.
- ♦ *vrfSim.mtl* specifies configuration information for the back-end.
- ♦ *vrfSimSettings.xml* stores back-end settings.

If you want to use an alternative GUI configuration file, you can replace the default *vrfGui.mtl* file or you can start VR-Forces with the `-l` option and specify a different file, for example:

```
vrfGui -l "../config/myconfigfile.mtl"
```

## 2.3. The *vrfGui.mtl* Configuration File

The *vrfGui.mtl* file configures parameters for the 2D front-end. You can override the configuration file using command-line options.

[Table 2-1](#) describes the parameters and lists the equivalent command-line option, where applicable.

Table 2-1: *vrfGui.mtl* parameters (Sheet 1 of 10)

| Parameter                             | Description                                                                                                                     | Option          |
|---------------------------------------|---------------------------------------------------------------------------------------------------------------------------------|-----------------|
| <code>addSymbolMapFile</code>         | Specifies a symbol map file to use. You can have multiple instances of this parameter.                                          | <code>-m</code> |
| <code>aggregatesFilename</code>       | Specifies the aggregates configuration file to load for the Aggregate As dialog box.<br>Default: <i>aggregates.mtl</i> .        |                 |
| <code>appearanceFile</code>           | Specifies the configuration file for entity appearance attributes.                                                              |                 |
| <code>appNumber</code>                | Specifies the application number.                                                                                               | <code>-a</code> |
| <code>askSaveScenarioIfChanged</code> | Specifies whether (1) or not (0) to prompt a user to save a scenario when they run it for the first time or when they close it. |                 |

Table 2-1: *vrfGui.mtl* parameters (Sheet 2 of 10)

| Parameter                       | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              | Option |
|---------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|
| availableTaskSetRetrievalScheme | <p>Specifies how context sensitivity is implemented for the Task and Set menus. The options are:</p> <p>0: Retrieve tasks and sets by name</p> <p>1: Retrieve tasks and sets by entity type</p> <p>2: Do not retrieve tasks and sets</p> <p>Default: 0.</p> <p>This option is designed to support exercises in which multiple instances of VR-Forces have different configurations of plug-ins that support different task and set commands. For example the B-HAVE Module for VR-Forces adds commands to the Task and Set menus. If some entities in a simulation are simulated by VR-Forces that is running B-HAVE and some are not, the lists of available tasks and sets returned from the back-end would be incorrect for some entities if they are treated the same way.</p> <p>If you set this parameter to 0, the front-end sends a separate request for each entity, using its name. In this case the context sensitive list is always correct.</p> <p>If you have a large number of entities, you can reduce the number of requests by using entity type (option 1). However, the list may be incorrect for some entities.</p> |        |
| backendApplicationName          | Specifies the name of the back-end executable. Default: <i>vrfsim.exe</i> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |        |
| backendAppNum                   | Specifies the default application number for the back-end. Default: 3001.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |        |
| backendSiteId                   | Specifies the default site ID for the back-end. Default: 1.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |        |
| backendTimeoutInterval          | The amount of time that the front-end keeps a back-end in its list of back-ends without getting a status message from the back-end.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |        |
| batchScenarioFilename           | Specifies the scenario to load when VR-Forces starts. If <i>scenarioFilename</i> and <i>batchScenarioFilename</i> are both specified, the batch scenario is loaded.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |        |
| boundingVolumesFile             | Specifies the path of the <i>boundingVolume.mtl</i> file.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                | -b     |
| capabilitiesFile                | Specifies the configuration file for capabilities.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |        |

Table 2-1: *vrfGui.mtl* parameters (Sheet 3 of 10)

| Parameter                   | Description                                                                                                                                                                                                                                                                                                                            | Option |
|-----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|
| commandLineDiscoveryTimeout | Specifies, in milliseconds, how long the front-end waits to discover a back-end when it loads a scenario from the command line.                                                                                                                                                                                                        |        |
| defaultFreehandSamplingRate | Specifies the number of points to sample when drawing a freehand line.                                                                                                                                                                                                                                                                 |        |
| defaultOverlayMenu          | Specifies the name of the <i>overlayMenu.mtl</i> file to use.                                                                                                                                                                                                                                                                          |        |
| defaultScenario             | Specifies the location of the <i>default.scn</i> file.                                                                                                                                                                                                                                                                                 |        |
| defaultSettingsFile         | Specifies the name of the settings file to load and save settings to.                                                                                                                                                                                                                                                                  | -d     |
| defaultSymbolMapFile        | Specifies the symbol map file for entity icons.                                                                                                                                                                                                                                                                                        |        |
| defaultTerrainDatabasePath  | Specifies the starting directory for the Open dialog box for loading terrain databases.                                                                                                                                                                                                                                                |        |
| doNotUseSettingsFile        | Specifies whether to load or save settings (0) or not to load or save settings (1).                                                                                                                                                                                                                                                    | -D     |
| DtNotifyLevel               | Specifies the notification level for application level console messages.                                                                                                                                                                                                                                                               |        |
| enableEmitterHandling       | Not implemented. Do not edit this parameter.                                                                                                                                                                                                                                                                                           |        |
| enableLogoOverlay           | Enables (1) or disables (0) display of a logo overlay in the map area.                                                                                                                                                                                                                                                                 |        |
| hostAddrString              | Specifies an ethernet address to receive files from the server during a save. This is for use on computers that have multiple network cards and on which you do not want to use the first network card for this purpose.                                                                                                               | -H     |
| language                    | Specifies the language to use in the VR-Forces GUI.                                                                                                                                                                                                                                                                                    | -G     |
| makePathsRelative           | Specifies that pathnames in the scenario file be saved using absolute paths (0) or relative paths (1). Default: 1.                                                                                                                                                                                                                     |        |
| maxDrainInputTime           | Sets the maximum time to wait in drain-Input() before returning. For the DMSO and MÄK RTIs, the default values will drain the entire input stream before returning. For the Pitch RTI, the default values read a single input from the input stream, so set to a value other than -1 to process more information. Use with DIS or HLA. |        |

Table 2-1: *vrfGui.mtl* parameters (Sheet 4 of 10)

| Parameter                   | Description                                                                                                                                                             | Option |
|-----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|
| maximumSignalTimeDifference | Specifies the maximum number of seconds between two signals on a transmitter frequency within which to assume that the second signal is responding to the first signal. |        |
| objectConsoleHistoryLimit   | Specifies the maximum number of console messages to be cached for each object.                                                                                          |        |
| objectInformationToolbar    | Enables (1) or disables (0) display of the Object Information Toolbar.                                                                                                  |        |
| overlayLogoFile             | Specifies the image to display if the overlay logo is enabled.                                                                                                          |        |
| overlayLogoPosition         | Specifies the position of the overlay logo on the map. Position is a value from 1 through 9 based on the following grid:<br>7 8 9<br>4 5 6<br>1 2 3<br>Default: 7.      |        |
| overlayLogoPositionX        | Specifies the logo position by X and Y pixel coordinates. If not -1, overrides the overlay-LogoPosition parameter.                                                      |        |
| overlayLogoPositionY        |                                                                                                                                                                         |        |
| overlayLogoScaleFactor      | The amount to scale the logo image. Default: 1.                                                                                                                         |        |
| patternsFile                | Specifies the entity type patterns file to load for the Pattern tab on the conditional expression dialog boxes.                                                         |        |
| pluginsDirectory            | Specifies the directory in which configuration (MTL) files for plug-ins are located. The directory must end with a slash, for example, <i>./plugins/</i> .              |        |
| preserveEmptyOverlayLayers  | Specifies whether (1) or not (0) to keep an overlay from which all objects have been deleted. Default: 0 (remove empty layers).                                         |        |
| processSignalInteractions   | Enables (1) or disables (0) processing of transmitter information. Default: 1.                                                                                          |        |
| qtAssistantLocation         | Specifies the location of the Qt Assistant executable.                                                                                                                  |        |
| qtHelpFile                  | Specifies the location of the help file for the Qt Assistant browser. This setting overrides the setting in <i>vrfGui.mtl</i> .                                         |        |
| qtHelpFile                  | Specifies the location of the online help table of contents file. The <i>qtHelpFile</i> parameter in <i>vrfGui.mtl</i> overrides this parameter.                        |        |

Table 2-1: vrfGui.mtl parameters (Sheet 5 of 10)

| Parameter                   | Description                                                                                                                                                                                                                                  | Option |
|-----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|
| rangeRingsFile              | Specifies the configuration file for range rings.                                                                                                                                                                                            |        |
| remoteControllerReceivePort | Specifies the port to be used to receive scenario files from a back-end in response to a save scenario request.                                                                                                                              |        |
| saveTimeoutInterval         | The amount of time the front-end waits for a scenario to be saved before it resends a save message. The default is 60 seconds, so if you know that your scenario takes longer than this to save, you must increase this value.               |        |
| scenarioFileName            | Specifies the scenario to load when VR-Forces starts. If scenarioFilename and batchScenarioFilename are both specified, the batch scenario is loaded.                                                                                        |        |
| scenarioTimeoutInterval     | The amount of time that the front-end waits for a scenario to load before it sends another load scenario message. The default is 60 seconds, so if you know that your scenario takes longer than this to load, you must increase this value. |        |
| separateObjectLists         | Enables (1) or disables (0) use of separate toolbars for the various object selection lists. Default: 0 (Objects tab has all object lists.)                                                                                                  |        |
| sessionId                   | Specifies the session ID to use.                                                                                                                                                                                                             | -i     |
| ShowGroundTruthForForce     | Specifies the forces for which you want VR-Forces to display ground truth.                                                                                                                                                                   |        |
| showSplashScreen            | Enables (1) or disables (0) display of the splash screen at startup.                                                                                                                                                                         |        |
| ShowSpotReportsForForce     | Specifies the forces for which you want to display spot reports.                                                                                                                                                                             |        |
| simulationModelSet          | Specifies the default simulation model set or sets. If specifying multiple SMSs, separate them by a semi-colon.                                                                                                                              |        |
| siteld                      | The site ID.                                                                                                                                                                                                                                 | -s     |
| SpotReportSystemEnabled     | Enables (1) or disables (0) VR-Forces' ability to display spot reports.                                                                                                                                                                      |        |

Table 2-1: vrfGui.mtl parameters (Sheet 6 of 10)

| Parameter                   | Description                                                                                                                                                                                                                                                                                                              | Option |
|-----------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|
| storeLineValuesMode         | Specifies whether or not to preserve line creation settings for color, width, sampling and force settings.<br>Possible settings are:<br>0 – Do not store any values<br>1 – Store for freehand lines only<br>2 – Store for all lines.<br>This only applies to lines and routes and is only valid for the current session. |        |
| terrainDatabase             | Specifies a default terrain database to load at startup. If you load a database at startup, you cannot create a scenario with it.                                                                                                                                                                                        |        |
| terrainDatabase             | Specifies the terrain database to load by default at startup.                                                                                                                                                                                                                                                            |        |
| timeMultiplierLowerBound    | Specifies the lower boundary of the range of the Time Multiplier toolbar. Default: 0.066 (1/15).                                                                                                                                                                                                                         |        |
| timeMultiplierUpperBound    | Specifies the upper boundary of the range of the Time Multiplier toolbar. Default: 15.                                                                                                                                                                                                                                   |        |
| useConsole                  | Enables (1) or disables (0) display of the console window.                                                                                                                                                                                                                                                               |        |
| useCustomMapDatum           | Allows modification of the reference ellipsoid used for geocentric to geodetic conversion. If you enable this parameter, there are several sub-parameters to configure. Please see the comments in the file.                                                                                                             |        |
| usingGlobalMiniMap          | Specifies whether to use (1) or not to use (0) a global Minimap.                                                                                                                                                                                                                                                         | -g     |
| viewControlMessages         | Enables (1) or disables (0) response to view control messages.                                                                                                                                                                                                                                                           |        |
| <b>Remapping Parameters</b> | For details about remapping, please see Section 3.2.5, “ <a href="#">Working with Multiple Back-ends</a> ” in <i>VR-Forces Users Guide</i> .                                                                                                                                                                             |        |
| autoRemapInCombinedMode     | Enables (1) or disables (0) automatic remapping of objects when you run a scenario in combined mode. VR-Forces remaps objects to the back-end specified by the backendSiteld and backendAppNum parameters (default: 1:3001). Default: 1.                                                                                 |        |

Table 2-1: vrfGui.mtl parameters (Sheet 7 of 10)

| Parameter                            | Description                                                                                                                                                                                                                          | Option |
|--------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|
| autoRemapInSeparateMode              | Enables (1) or disables (0) automatic remapping of objects when you run a scenario in separate mode. Objects are mapped to the back-end specified by the separateModeRemapSiteId and separateModeRemapAppNum parameters. Default: 0. |        |
| separateModeRemapSiteId              | Specifies the site ID of the back-end that you want to remap objects to if you set autoRemapInSeparateMode to 1. Default: 1.                                                                                                         |        |
| separateModeRemapAppNum              | Specifies the application number of the back-end that you want to remap objects to if you set autoRemapInSeparateMode to 1. Default: 3001.                                                                                           |        |
| batchModeRemapSiteId                 | Specifies the site ID of the back-end that you want to remap objects to in batch mode. Default: 1.                                                                                                                                   |        |
| <b>Trajectory History Parameters</b> |                                                                                                                                                                                                                                      |        |
| enableHistoryTracking                | Enables (1) or disables (0) recording of trajectory histories. Disabling trajectory histories may improve performance. You cannot override this setting in the GUI. Default: 1.                                                      |        |
| decayTrackHistory                    | Specifies that an entity's trajectory history decays even if it is not moving (1) or does not decay when it is not moving (0). Setting this parameter to 1 may affect performance. Default: 0.                                       |        |
| minTrackHistoryMovement              | Specifies the minimum distance, in meters, between points used to create a trajectory history. Default: 60.                                                                                                                          |        |
| trackHistoryTimeout                  | The length of time (simulation seconds) to keep any point in the trajectory history list. Set to -1 to never timeout the points on the line. Default: 60.                                                                            |        |
| <b>Performance Tuning Parameters</b> |                                                                                                                                                                                                                                      |        |
| consecutiveOverFrameRateThreshold    | Specifies the number of consecutive frames the system must exceed networkTimeThreshold before switching to heavy network traffic mode. Enforced only if tickManagingEnabled is set to 1.                                             |        |

Table 2-1: vrfGui.mtl parameters (Sheet 8 of 10)

| Parameter                | Description                                                                                                                                                                                                                                                                                                               | Option |
|--------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|
| hideOffScreenSymbols     | Specifies whether or not to hide symbols when they go off screen instead of destroying them. Setting this parameter to 1 may improve performance for large exercises.                                                                                                                                                     |        |
| networkTimeThreshold     | Specifies the maximum length of time, in milliseconds, that a network frame should take to execute. Enforced only if tickManagingEnabled is set to 1.                                                                                                                                                                     |        |
| networkFrameRateBuffer   | Specifies a value to use in determining whether or not to switch from “meets frame rate target” mode to “not meeting target” mode. Multiply networkFrameRateBuffer by the number of milliseconds the network frame took to execute. If the result is greater than the frame rate target, enter “not meeting target” mode. |        |
| slowNetworkFrameOverride | Specifies the number of milliseconds to wait between network frames before polling the network again. Enforced only if tickManagingEnabled is set to 1.<br><br>Default: Twice the length of the longest network frame or 5 seconds, whichever is less.                                                                    |        |
| tickManagingEnabled      | Enables (1) or disables (0) checking for heavy network traffic.                                                                                                                                                                                                                                                           | -e     |
| batchModeRemapAppNum     | Specifies the application number of the back-end that you want to remap objects to in batch mode. Default: 3001.                                                                                                                                                                                                          |        |
| <b>HLA Parameters</b>    | For details, please see Section 4.8.3, “ <a href="#">Command Line Options for HLA Federations</a> ” In <i>VR-Forces Users Guide</i> .                                                                                                                                                                                     |        |
| destroyFedExec           | Specifies whether (1) or not (0) the <i>DtExerciseConn</i> destructor should try to destroy the federation execution. Default: 1.                                                                                                                                                                                         |        |
| execName                 | The federation execution name.                                                                                                                                                                                                                                                                                            | -x     |
| federateName             | The federate name. The federateName parameter in <i>vrfGui.mtl</i> overrides this parameter.                                                                                                                                                                                                                              | -N     |
| fedFileName              | Specifies the FED file name.                                                                                                                                                                                                                                                                                              | -F     |
| fomMapperLib             | A FOM Mapper library.                                                                                                                                                                                                                                                                                                     | -f     |
| fomMapperInitData        | An initialization string.                                                                                                                                                                                                                                                                                                 |        |
| --fomMapperInitData      |                                                                                                                                                                                                                                                                                                                           |        |



Table 2-1: vrfGui.mtl parameters (Sheet 9 of 10)

| Parameter                 | Description                                                                                                                                                                             | Option          |
|---------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|
| lookahead                 | Specifies the lookahead value if time regulating mode is enabled.                                                                                                                       |                 |
| minDrainInputTime         | The minimum time to stay in drainInput() before returning.                                                                                                                              |                 |
| rprFomVersion             | The RPR FOM version.                                                                                                                                                                    | --rprFomVersion |
| runInTimeConstrainedMode  | Enables (1) or disables (0) time constrained mode for HLA time management.                                                                                                              |                 |
| runInTimeRegulatingMode   | Enables (1) or disables (0) time regulating mode for HLA time management.                                                                                                               |                 |
| sendFedTime               | Enables (1) or disables (0) sending and receipt of interactions in time stamp order. Default 0.                                                                                         |                 |
| simTimeScale              | Specifies the desired ratio of simulation time to real time if time constrained mode is enabled.                                                                                        |                 |
| useAbsoluteTimeStamps     | If set to 1, use absolute time stamps. If set to 0, use relative time stamps. Default: 0.                                                                                               |                 |
| useAdvisories             | Enables (1) or disables (0) HLA advisory messages.                                                                                                                                      |                 |
| <b>DIS Parameters</b>     | For details, please see Section 4.8.4, " <a href="#">Command Line Options for DIS Exercises</a> " in <i>VR-Forces Users Guide</i> .                                                     |                 |
| addMulticastAddr          | The multicast address.                                                                                                                                                                  | -S              |
| appldRange                | Specifies the application ID range to use to identify entities generated by VR-Forces.                                                                                                  |                 |
| defaultHeartbeatThreshold | Specifies the DIS heartbeat threshold, in seconds. A heartbeat is sent when the threshold is reached, even if no data has changed.                                                      |                 |
| defaultObjectTimeout      | Specifies the DIS object timeout threshold, in seconds. If VR-Forces does not receive a heartbeat for an object for the time period specified, it removes the object from the exercise. |                 |
| destAddrString            | Destination address for point-to-point mode.                                                                                                                                            | -A              |
| disPort                   | The port number.                                                                                                                                                                        | -P              |
| exerciseld                | The exercise ID.                                                                                                                                                                        | -x              |
| maxDrainInputReads        | The maximum number of reads to make in drain input before returning                                                                                                                     |                 |

Table 2-1: *vrfGui.mtl* parameters (Sheet 10 of 10)

| Parameter             | Description                                                                                                                      | Option                        |
|-----------------------|----------------------------------------------------------------------------------------------------------------------------------|-------------------------------|
| mcastTtl              | Specifies the multicast time to live.                                                                                            | <code>--mcastTtl</code>       |
| recvBufferSize        | Specifies the receive buffer size, as a positive value. A value of -1 means use the system default.                              | <code>--recvBufferSize</code> |
| sendBufferSize        | Specifies the send buffer size, as a positive value. A value of -1 means use the system default.                                 | <code>--sendBufferSize</code> |
| useAbsoluteTimeStamps | If set to 1, use absolute time stamps. If set to 0, use relative time stamps. Default: 0.                                        |                               |
| useAsyncIO            | Enables (1) or disables (0) use of asynchronous I/O, which can help reduce dropped packets under heavy network load. Default: 0. | <code>--useAsyncIO</code>     |

## 2.4. The *vrfSim.mtl* Configuration File

The *vrfSim.mtl* file configures back-ends.

Table 2-2: *vrfSim.mtl* parameters

| Parameter                | Description                                                                                                                                                                                                                                                     | Option                            |
|--------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------|
| appNumber                | Specifies the application number.<br>Default: 3001.                                                                                                                                                                                                             | <code>-a</code>   <code>-p</code> |
| batchFile                | Specifies a batch file to run.                                                                                                                                                                                                                                  | <code>-B</code>                   |
| cgfDispatcherReceivePort | Specifies the port to be used by the <i>DtCgfDispatcher</i> 's file transporter to receive order of battle and plan files from the <i>DtVrfRemoteController</i> .                                                                                               |                                   |
| daemon                   | Runs <i>vrfSim</i> as a daemon process (Linux only.) <i>vrfSim</i> will not attempt to read characters from a console, and will fork a background process on startup. It will exit cleanly on receipt of the SIGTERM, SIGQUIT, SIGINT signals.<br>Default: Off. | <code>--daemon</code>             |

Table 2-2: *vrfSim.mtl* parameters

| Parameter                                 | Description                                                                                                                                                                                                                                                                                                                                        | Option          |
|-------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|
| <code>defaultMainLoopSleepInterval</code> | Sets the sleep interval for <code>DtCgf::tick()</code> . By adjusting how long your application sleeps before it ticks the <i>DtCgf</i> , you can tune the performance of the main application loop.                                                                                                                                               |                 |
| <code>defaultParameterDatabasePath</code> | Specifies the location to search for the object parameter database if the full path is not specified in a scenario file.                                                                                                                                                                                                                           |                 |
| <code>defaultTerrainDatabasePath</code>   | Specifies the location to search for the terrain database if the full path is not specified in a scenario file.                                                                                                                                                                                                                                    |                 |
| <code>doNotUseConsole</code>              | Disables (1) or enables (0) writing of output to the console. Default: 0.                                                                                                                                                                                                                                                                          |                 |
| <code>DtNotifyLevel</code>                | Sets the notification level (0-4) for information and warning messages. Default: 2.                                                                                                                                                                                                                                                                | <code>-n</code> |
| <code>entVrfDataTimeThreshold</code>      | Specifies how often, in seconds, VR-Forces entity data is sent. If you specify -1.0, the update is sent only if some item in the <code>vrfDataObject</code> has changed.                                                                                                                                                                           |                 |
| <code>envVrfDataTimeThreshold</code>      | Specifies how often, in seconds, VR-Forces environmental data is sent. If you specify -1.0, the update is sent only if some item in the <code>vrfDataObject</code> has changed.                                                                                                                                                                    |                 |
| <code>forceParameterDbReload</code>       | If set to 1 (default), forces the back-end to reload the object parameter database (OPD) each time a scenario is loaded. If set to 0, if a scenario is started that uses the same OPD as the previous one (for example, in a rewind), the OPD is not reloaded.                                                                                     |                 |
| <code>ignoreAdvisories</code>             | Turns off HLA advisory messages.                                                                                                                                                                                                                                                                                                                   |                 |
| <code>loadPluginIfVersionMismatch</code>  | Specifies whether (1) or not (0) to load a plug-in if there is a mismatch between how the plug-in was built and how VR-Forces was built. Mismatches can be build type (release versus debug), operating system, or protocol (DIS, HLA 1.3, HLA 1516.)<br><br>Allowing a plug-in to load if there is a mismatch can cause VR-Forces to be unstable. |                 |
| <code>logFrameRateStatistics</code>       | Enables (1) or disables (0) logging of frame rate statistics to a file.                                                                                                                                                                                                                                                                            |                 |

Table 2-2: *vrfSim.mtl* parameters

| Parameter                                     | Description                                                                                                                                                                                                                                                                                                                                   | Option                            |
|-----------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------|
| <code>maxDrainInputTime</code>                | Sets the maximum time to wait in <code>drainInput()</code> before returning. For the DMSO and MÄK RTIs, the default value drains the entire input stream before returning. For the Pitch RTI, the default value reads a single input from the input stream, so set to a value other than -1 to process more information. Use with DIS or HLA. |                                   |
| <code>pluginsDirectory</code>                 | Specifies the directory in which configuration (MTL) files for plug-ins are located. The directory must end with a slash, for example, <code>./plugins/</code> .                                                                                                                                                                              |                                   |
| <code>publishSimulationTime-ForStealth</code> | Specifies whether or not to publish the simulation time to the network so that the MÄK Stealth can set its own simulation clock to work in conjunction with the simulation time.                                                                                                                                                              |                                   |
| <code>scenarioFileName</code>                 | Specifies the scenario file to load.                                                                                                                                                                                                                                                                                                          | <code>-I</code>                   |
| <code>sessionId</code>                        | Specifies the session ID.                                                                                                                                                                                                                                                                                                                     | <code>-i</code>                   |
| <code>simMgmtPduProcessLevel</code>           | Specifies how simulation management PDUs are processed:<br>1 – Processes PDUs sent to a specific back-end. Wildcards are ignored.<br>2 – Processes only wildcarded PDUs. Ignores those sent to specific back-ends.<br>3 – Processes all PDUs.                                                                                                 |                                   |
| <code>simulationModelSet</code>               | Specifies the default simulation model set or sets. If specifying multiple SMSs, separate them by a semi-colon.                                                                                                                                                                                                                               |                                   |
| <code>siteld</code>                           | Specifies the site ID. 0 = use internal default.                                                                                                                                                                                                                                                                                              | <code>-s</code>   <code>-t</code> |
| <code>sleepInterval</code>                    | Specifies how much time, in seconds, the <i>DtCgf</i> class should sleep between calls to <code>tick()</code> . Default: 0.01.                                                                                                                                                                                                                | <code>-e</code>                   |
| <code>startInRunMode</code>                   | Specifies that the simulation start in run mode (1), rather than pause mode (0). Default: 0.                                                                                                                                                                                                                                                  |                                   |
| <code>statusUpdateHeartbeat</code>            | Specifies how often, in seconds, the back-end status heartbeat is sent.                                                                                                                                                                                                                                                                       |                                   |
| <code>terrainFileName</code>                  | Specifies the terrain database to load. This parameter is overridden by the scenario file if one is specified in <code>scenarioFileName</code> .                                                                                                                                                                                              | <code>-T</code>                   |

Table 2-2: vrfSim.mtl parameters

| Parameter                 | Description                                                                                                                                                                                                               | Option              |
|---------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------|
| useCustomMapDatum         | Allows modification of the reference ellipsoid used for geocentric to geodetic conversion. If you enable this parameter, there are several sub-parameters to configure. For details, please see the comments in the file. |                     |
| <b>HLA Parameters</b>     | For details, please see Section 4.8.3, <a href="#">“Command Line Options for HLA Federations”</a> in <i>VR-Forces Users Guide</i> .                                                                                       |                     |
| destroyFedExec            | Specifies whether (1) or not (0) the <i>DtExerciseConn</i> destructor should try to destroy the federation execution. Default: 1.                                                                                         |                     |
| fomMapperLib              | A FOM Mapper library.                                                                                                                                                                                                     | -f                  |
| fomMapperInitData         | An initialization string.                                                                                                                                                                                                 | --fomMapperInitData |
| execName                  | The federation execution name.                                                                                                                                                                                            | -x                  |
| federateName              | The federate name.                                                                                                                                                                                                        | -N                  |
| fedFileName               | Specifies the FED file name.                                                                                                                                                                                              | -F                  |
| minDrainInputTime         | The minimum time to stay in drainInput() before returning.                                                                                                                                                                |                     |
| rprFomVersion             | The RPR FOM version. (0.5, 0.8, 1.0, 2.0)                                                                                                                                                                                 | --rprFomVersion     |
| runInTimeManagementMode   | Enables (1) or disables (0) use of time management.                                                                                                                                                                       |                     |
| sendFedTime               | Enables (1) or disables (0) sending and receipt of interactions in time stamp order. Default 0.                                                                                                                           |                     |
| useAbsoluteTimeStamps     | If set to 1, use absolute time stamps. If set to 0, use relative time stamps. Default: 0.                                                                                                                                 |                     |
| useAdvisories             | Enables (1) or disables (0) HLA advisory messages.                                                                                                                                                                        |                     |
| <b>DIS Parameters</b>     | Please see Section 4.8.4, <a href="#">“Command Line Options for DIS Exercises”</a> in <i>VR-Forces Users Guide</i> .                                                                                                      |                     |
| addMulticastAddr          | The multicast address.                                                                                                                                                                                                    | -S                  |
| appldRange                | Specifies the application ID range to use to identify entities generated by VR-Forces.                                                                                                                                    |                     |
| defaultHeartbeatThreshold | Specifies the DIS heartbeat threshold, in seconds. A heartbeat is sent when the threshold is reached, even if no data has changed.                                                                                        |                     |

Table 2-2: *vrfSim.mtl* parameters

| Parameter             | Description                                                                                                                                                                              | Option           |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|
| defaultObjectTimeout  | Specifies the DIS object timeout threshold, in seconds. If VR-Forces does not receive a heart-beat for an object for the time period specified, it removes the object from the exercise. |                  |
| destAddrString        | Destination address for point-to-point communications mode.                                                                                                                              | -A               |
| disPort               | The port number.                                                                                                                                                                         | -P               |
| exerciseld            | The exercise ID.                                                                                                                                                                         | -x               |
| maxDrainInputReads    | The maximum number of reads to make in drain input before returning                                                                                                                      |                  |
| mcastTtl              | Specifies the multicast time to live.                                                                                                                                                    | --mcastTtl       |
| recvBufferSize        | Specifies the receive buffer size, as a positive value. A value of -1 means use the system default.                                                                                      | --recvBufferSize |
| sendBufferSize        | Specifies the send buffer size, as a positive value. A value of -1 means use the system default.                                                                                         | --sendBufferSize |
| useAbsoluteTimeStamps | If set to 1, use absolute time stamps. If set to 0, use relative time stamps. Default: 0.                                                                                                |                  |
| useAsyncIO            | Enables (1) or disables (0) use of asynchronous I/O, which can help reduce dropped packets under heavy network load. Default: 0.                                                         | -I               |

## 2.5. Symbol Mapping Files

VR-Forces loads a symbol map file to map entity types to symbols for display purposes. Symbol map files are in *./data/simulationModelSets/<modelset>/gui*. The default symbol map file is specified in *vrfGui.mtl*.

If you want to map entities to different symbols, you have the following options:

- ♦ Edit the `addSymbolMapFile` parameter in *vrfGui.mtl*
- ♦ Start VR-Forces with the `-m` option and specify a symbol map file
- ♦ Change the symbol map file after you start VR-Forces.

Each entry in a symbol map file has the format:

```
(symbol-map (list object_enumeration) "file_type" "mapping")
```

where:

- ♦ *object\_enumeration* is an enumeration that identifies the entity type of the object (see Figure 2-1).
- ♦ *file\_type* identifies the type of graphic (bmp, ttf, utf)
- ♦ *mapping* identifies the graphic file or font mapping for the object.

For example:

```
(symbol-map (list 1 -1 1 1 -1 -1 -1 -1 -1) "bmp"
  "../data/symbols/bmp/groundF.bmp")
```

If you want to add additional mappings to the configuration files, add new symbol-map entries.

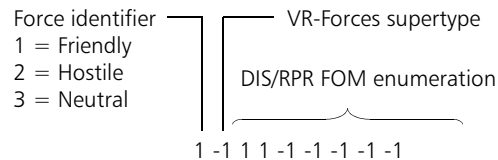


Figure 2-1. Symbol map enumeration syntax

### 2.5.1. Symbol Mapping Syntax for Animated Images

VR-Forces supports animated images for entities and effects such as explosions. The animated images are a sequence of raster images that are displayed rapidly to produce the effect of motion. The syntax for an animated image is as follows:

```
(symbol-map (list object_enumeration) "file_type" (list "mapping1" ...
  "mappingn") frame_interval loop)
```

where:

- ♦ *object\_enumeration* is an enumeration that identifies the entity type of the object (Figure 2-1). It includes the supertype value used in VR-Forces.
- ♦ *file\_type* identifies the type of graphic (anim-bmp).
- ♦ *mapping* identifies the graphic files for the animation.
- ♦ *frame\_interval* specifies the amount of time between each frame, in seconds.
- ♦ *loop* specifies whether or not to loop the animation. 1 = loop; 0 = do not loop.

For example:

```
(symbol-map (list 1 -1 1 2 225 4 1 -1 -1) "anim-bmp"
  (list "../data/symbols/images/C-130_Hercules0000.png"
    "../data/symbols/images/C-130_Hercules0001.png"
    "../data/symbols/images/C-130_Hercules0002.png") 0.1 1)
```

### 2.5.2. Symbol Mapping Syntax for TrueType and Unicode TrueType Fonts

The mapping portion of a symbol map entry for a TrueType font (ttf) is as follows:

```
"ttf" "font_name" "character"
```

For example, the mapping for a generic tracked entity is:

```
(symbol-map (list 0 -1 1 1 -1 1 -1 -1 -1) "ttf" "MapSym NK Land" "A")
```

The mapping portion of a symbol map entry for a unicode TrueType font (uttf) is as follows:

```
"uttf" "font_file" "display_character" "outline_character"  
"fill_character"
```

For example, the mapping for a generic tracked entity is:

```
(symbol-map (list 0 -1 1 1 -1 1 -1 -1 -1) "uttf" "MilStd 2525b" "0000"  
"4929" "0025")
```

You must have a viewer that can view unicode fonts to see which characters to use.

### 2.5.3. Changing the Symbol Map File During Runtime

You can change the symbol map file being used while VR-Forces is running.

To load a symbol map file:

1. Choose **Options** → **Display**. The Display Options dialog box opens.
2. Select the Symbols tab.
3. In the Symbol Mapping group box, select the top option button.
4. In the drop-down list, select the symbol set you want to use.
5. Click OK.

## 2.6. Other Configuration Files

This section describes some of the other configuration files.

### 2.6.1. The *plugin.mtl* File

The *plugin.mtl* file specifies plug-ins that you want to load automatically. VR-Forces does not include this file in the distribution. However, *./plugins/readme.txt* explains the syntax of a *plugin.mtl* file and includes sample text. If you want to load plug-ins automatically, create a *plugin.mtl* file and put it in the *./plugins* directory.

### 2.6.2. The *commonObjects.mtl* File

The *commonObjects.mtl* file has mappings for objects that use the same files regardless of the symbol map file in use. The syntax of entries is documented in the file.



### 2.6.3. The *rangeRings.mtl* Configuration File

The *rangerings.mtl* file configures how range rings are displayed. Range rings are configured for each supported weapon. The file documents the syntax of entries.

### 2.6.4. The *boundingVolume.mtl* Configuration File

The *boundingVolume.mtl* file specifies the bounding volumes that VR-Forces uses in the front-end when calculating intervisibility and entity intersections are enabled. The bounding volumes specified in the object parameter database are larger than the actual dimensions of the entities because the bounding volume is used for collision avoidance calculations. You can edit the file to set the bounding volumes for intervisibility to be the actual size of entities, thereby getting more accurate intervisibility information.

The *boundingVolume.mtl* file contains bounding volumes for all entities configured in the object parameter database, either using specific entity enumerations, or through wildcard matching. If you add entities to the object parameter database, you may want to add corresponding entries to the *boundingVolume.mtl* file. The bounding volume is specified as half of the length, width, and height of the actual bounding volume. The default half-size bounding volume is two meters long, 1.5 meters wide, and 1.5 meters high.

## 2.7. Configuring Overlay Objects

The list of overlay objects available on the Create menu and the Overlay Palette is determined by the *overlayMenu.mtl* file. You can edit them to change the organization of objects and to add or delete overlay objects.

The overlay objects are listed on each submenu in the order that they are listed in the configuration file. Please see the comments in *overlayMenu.mtl* for an explanation of the syntax of these files.

The *overlayPalette.mtl* file configures how overlay objects are drawn. Please see the comments in the file for details.

## 2.8. Configuring DI-Guy Integration

VR-Forces can specify the character, appearance, and animation values for DI-Guy models used in 3D visualization applications such as the VR-Forces 3D Front End and VR-Vantage. A model's character determines the kinds of movement it can make. The appearance determines what it looks like. The animation specifies the movements that the character makes.

In the VR-Forces front-end, you can specify the animation with the DI-Guy Animation task. You can set the appearance with the DI-Guy Appearance set data request. For details about specifying DI-Guy characteristics in the front-end, please see *VR-Forces Users Guide*.

You can specify the default character, appearance, and animation for a lifeform entity in the Entity Editor or OPD Editor. (For details, please see [“Editing a Lifeform’s DI-Guy Character, Appearance, and Animation,”](#) on page 5-15.) The DI-Guy mappings are stored in the entity's OPE file.

The DI-Guy characters, appearances, and animations available to lifeforms are controlled by two configuration files:

- ♦ `./config/character_animations_table.mtl` lists the DI-Guy characters available for VR-Forces entities and the appearances available for the character. This information is used to populate the drop-down lists in the Entity Editor. You can edit this file to add additional DI-Guy characters to those already configured for VR-Forces. The syntax of entries is described in the file.



Any characters you add to `character_animations_table.mtl` must be valid DI-Guy characters that are already in `character_digest.xml`. You cannot make up new types of characters.

---

- ♦ `./config/character_digest.xml` contains information that VR-Forces needs to synchronize an entity's simulated location with the movements generated by DI-Guy animations. For example, if a model's animation causes the model to step to the left, VR-Forces needs to move the simulated location of the entity the same distance to the left.

## 2.9. Configuring Communication Models

Communication models are configured in the *commModelParams.mtl* file in *./data/simulationModelSets/<model-set>*. Communication models determine how radio messages travel from one entity to another. They determine which entities are connected to each other and how long it takes a message to get from one entity to another.

Communication model classes are derived from *DtCommModel* and are generated from the communication model factory based on the string types specified in the *commModelParams.mtl* file. A communication model entry contains parameters for the type of communication model descriptor and communication model class being used.

### 2.9.1. Simple Radio Communication Model

VR-Forces includes a default communication model called *simple-radio-comm-model*. This communication model models radio traffic in a very simple way. Radio messages always reach their intended destination, as long as the destination is on a reachable network, and messages are all passed instantly. The simple radio communication model has one parameter, **connection-mode**, which determines how the radio networks are set up. There are three connection-modes available:

- ♦ **aggregate** mode specifies that each aggregate has its own radio network. Messages sent from an entity can reach any entity that is in the same organization, at the same level (the entity's peers in the organization), and can also reach the entity's direct superior. All other entities are outside the network, and are unable to receive messages sent by the entity.



If an entity is not in an aggregate, it is considered to be part of the force aggregate and its radio messages go to other entities that are part of the force aggregate.

---

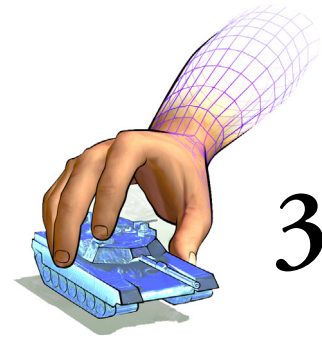
- ♦ **force** mode specifies that there is a radio network for each force in VR-Forces. Messages sent from an entity can reach any other entity of the same force.
- ♦ **all** mode specifies that all entities in VR-Forces are part of one large radio network. In this case, any entity can receive messages sent by any other entity.

### 2.9.2. External Communication Model

VR-Forces includes an external communication model that can communicate with an external communication effects server over the network to determine how to deliver radio messages. For details, please see Chapter 9, *Using VR-Forces with an External Comms Effects Server*.

### **2.9.3. Custom Communication Models**

You can create new communication models and install them in VR-Forces. This is useful if you want to define your own connectivity model, or write your own algorithms for determining if radio messages get delivered. For details, please see Section 9.5.1, “[Creating Custom Communication Models](#)” in *VR-Forces Developers Guide*.



## *Scenario Files*

---

This chapter describes the files that get created when you create a scenario.

|                                                                    |      |
|--------------------------------------------------------------------|------|
| Introduction .....                                                 | 3-2  |
| The Scenario File .....                                            | 3-3  |
| Scenario Parameters.....                                           | 3-4  |
| Specifying Pathnames for Scenario Files.....                       | 3-6  |
| Changing the Terrain Database for a Scenario .....                 | 3-7  |
| Editing the Plan Files .....                                       | 3-8  |
| The Structure of an Entity Plan File .....                         | 3-9  |
| Editing the Object Map File .....                                  | 3-11 |
| Editing the Order of Battle File.....                              | 3-11 |
| Saving Default Entity Parameters to the Order of Battle File ..... | 3-12 |
| The Overlay File .....                                             | 3-12 |
| The Selection Groups File.....                                     | 3-12 |
| Temporary Scenario Directories .....                               | 3-12 |

### 3.1. Introduction

When you create and save a scenario, VR-Forces stores information about the scenario in the following files:

- The scenario file, named *scenario\_name.scn*, lists the locations of the other scenario files, the terrain database, and the simulation model set file, and sets several parameters for the scenario
- The order of battle file, named *scenario\_name.oob*, stores identification and state information for entities, control objects, and overlay objects
- The object map file, named *scenario\_name.omp*, maps objects to the back-ends on which they were created
- The plan file, named *scenario\_name.pln*, stores entity plans
- The global plan file, named *scenario\_name.gpl*, stores the scenario's global plans
- The overlay file, named *scenario\_name.ovl*, stores information about unpublished overlay objects, if any
- The selection groups file, named *scenario\_name.sgr*, stores information about selection groups, if any
- The hostility relationships file, named *scenario\_name.hst*, lists the forces that each force is hostile to.

For the most part, you will not want to edit these files by hand. In some cases, we recommend that you not edit them. However, this chapter explains how to edit them if you need to.

## 3.2. The Scenario File

When you create a scenario, you can set its parameters in the New Scenario dialog box (Figure 3-1). Thereafter, you cannot edit scenario parameters in the front-end except for the option to attach components to remote entities. However, you can edit them in a text editor.

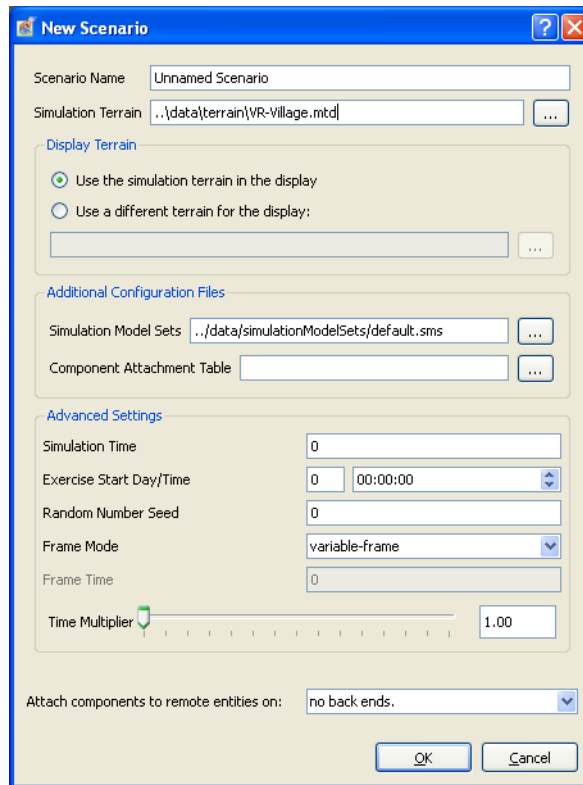


Figure 3-1. Choose Terrain dialog box

The default values for scenario settings in the New Scenario dialog box are drawn from the *default.scn* file. If you want to change the default settings, you can edit that file.

### 3.2.1. Scenario Parameters

A typical scenario file looks like the following:

```
(Scenario
  (Component-Attachment 0)
  (Component-Attachment-Table "")
  (Terrain-Database "..\data\terrain\Makland.mtd")
  (Gui-Terrain-Database "..\data\terrain\Makland.mtd")
  (Order-Of-Battle "example.oob")
  (Plan "example.pln")
  (Global-Plan "Makland.gpl")
  (Overlay "example.ovl")
  (SelectionGroups "example.sgr")
  (Object-Map "example.omp")
  (Force-Hostility "example.hst")
  (time-multiplier 1.000000)
  (auto-reorganize False)
  (random-number-seed 0)
  (frame-mode "variable-frame")
  (frame-time 0.000000)
  (sim-time 0.000000)
  (exercise-start-time 0.000000)
  (additional-opds "C:/MAK/vrf3.12/data/parameters/customOpd1.opd"
    "C:/MAK/vrf3.12/data/parameters/customOpd2.opd")
  (Simulation-Model-Set-Files "..\data\simulationModelSets\default.sms")
  (version 3.100000)
)
```

---

#### i

- ♦ Although you can edit the **sim-time** and **exercise-start-time** parameters in scenario files, we strongly recommend that you not change these parameters for scenarios that were saved with a simulation time other than 0. Some of the state information saved in the scenario may depend on these values and if you change them, the behavior of the scenario may not be what you expect.
  - ♦ Paths in the scenario files are relative to the application directory, for example, *vrforces3.11/binDIS/vrfSim.exe*.
- 

The parameters in a scenario file are as follows:

- ♦ **Component-Attachment** specifies whether or not attachment of components to remote entities is allowed in the scenario. The options are:
    - 0: no component attachment
    - 1: the first back-end manages all remote components
    - 2: remote component attachment is load-balanced on all back-ends.
- 

#### i

Although you can set this value in the New Scenario and Load Scenario dialog boxes, it is not saved in the scenario file. To force a specific value, you must edit the scenario file.

---



- ♦ **Component-Attachment-Table** specifies the component attachment table to use if attachment of components to remote entities is allowed.
- ♦ **Terrain-Database** specifies the terrain database to be loaded by the back-end.
- ♦ **Gui-Terrain-Database** specifies the terrain database loaded in the front-end.
- ♦ **Order-Of-Battle** specifies the order of battle file.
- ♦ **Plan** specifies the plan file.
- ♦ **Global-Plan** specifies the global plan file.
- ♦ **Overlay** specifies the overlay file.
- ♦ **SelectionGroups** specifies the selection groups file.
- ♦ **Object-Map** specifies the object mapping file.
- ♦ **Force-Hostility** specifies the force hostility file.
- ♦ **time-multiplier** specifies whether a scenario runs in real time or at a rate other than real time. A value greater than 1 runs the simulation faster than real time. A value between 0 and 1 runs the simulation slower than real time. You can set this value dynamically using the Time Multiplier toolbar.



If you are running a scenario using HLA Time Management, it is strongly recommended that you set **time-multiplier** to 1.

---

- ♦ **auto-reorganize** lets you specify the default behavior for aggregate reorganization. Set it to True or False. For information about reorganization, please see Section 3.6.1, “[How Aggregates Are Organized](#)” in *VR-Forces Users Guide*.
- ♦ **random-number-seed** sets the default random number seed. It provides a starting point for the generation of random numbers needed by VR-Forces. Since computer-generated random numbers are not truly random, if you run a scenario repeatedly with the same random number seed (and you run in fixed-frame-mode), you will get the same results. (For more information about repeatability, please see Section 3.13.2, “[Exercise Clock Modes](#)” in *VR-Forces Users Guide*.) Enter an integer between 0 and the largest integer permitted on your system.
- ♦ **frame-mode** sets the frame rate mode to one of the following:
  - Variable-Frame Run-To-Complete (**variable-frame**)
  - Fixed-Frame Best-Effort (**fixed-frame**)
  - Fixed-Frame Run-To-Complete (**fixed-frame-run-to-complete**.)

These frame rate modes are described in Section 3.13.2, “[Exercise Clock Modes](#)” in *VR-Forces Users Guide*.

- ♦ **frame-time** specifies the length of a frame, in seconds. If **frame-mode** is set to **fixed-frame** or **fixed-frame-run-to-complete**, you must set the frame time to a non-zero value. A value of zero for frame time will prevent simulation time from advancing in either of these modes.

- ♦ **sim-time** specifies the starting simulation time, in seconds. By default, a new scenario starts with simulation time set to 0.0. A scenario saved as a checkpoint saves the simulation time of the checkpoint.
- ♦ **exercise-start-time** specifies the starting exercise time for the scenario, in seconds. By default, an exercise starts at day 1, zero hours, minutes, and seconds. However, you can set the scenario to start running with a different exercise time.
- ♦ **additional-opds** specifies custom OPD files to load (for backwards compatibility.)
- ♦ **Simulation-Model-Set-Files** specifies a list of SMS files to load.
- ♦ **version** specifies versioning information for the scenario file. Do not edit this parameter.

### 3.2.2. Specifying Pathnames for Scenario Files

Pathnames in a scenario file can be absolute or relative. Relative pathnames are relative to the application directory. Section 5.3.3, “[Saving Scenarios with Relative or Absolute Pathnames](#)” in *VR-Forces Users Guide* explains how to configure the way that pathnames are saved. If you think you might need to move a scenario to another computer or a new directory structure, change the paths to be relative to the *./binx* directory.



If you plan to run multiple front-ends and back-ends on multiple machines, particularly if they use different operating systems, you must change the pathnames for the terrain database and object parameter database to relative paths or simple pathnames.

---

#### Configuring How Paths are Interpreted

The following parameters in *vrfsim.mtl* determine how pathnames for the object parameter database and terrain database are resolved:

- ♦ **defaultParameterDatabasePath** specifies the path to search for the *.opd* file when an explicit path is not specified in a scenario file
- ♦ **defaultTerrainDatabasePath** specifies the path to search for terrain database files when an explicit path is not specified in a scenario file.

When the back-end loads a scenario file, it looks for the object parameter database using the following search order:

1. If the path is absolute, load the file in the specified path.
2. If the path is relative, try to load the file relative to the application path (*vrforcesx.x/binx/vrfsim*).
3. Look in the directory specified by **defaultParameterDatabasePath**.

When the back-end loads a scenario file, it looks for the terrain database using the following search order:

1. If the path is absolute, load the file in the specified path.
2. If the path is relative, try to load the file relative to the application path.
3. Look in directory specified by `defaultTerrainDatabasePath`.
4. If defined, look in the user-defined environmental variable `VRF_TERRAIN_PATH`.

### 3.2.3. Changing the Terrain Database for a Scenario

When you create a new scenario, you must choose terrain databases for the front-end and back-end. By default, the back-end uses the same database as the front end. Please see Section 3.11.1, “[Using Large Terrain Databases](#)”, in *VR-Forces Users Guide* for additional details.

Once you have chosen the terrain database, you cannot change it from the graphical user interface. You can edit the pathname of a terrain database in the scenario file. However, unless the new terrain database covers exactly the same geographic area as the old one, the order of battle and overlay information associated with the scenario may be confusing or useless.

Although you would not typically change the terrain database used for a scenario, if you create a scenario in the 2D GUI and want to use it in the 3D GUI (and vice versa), you may need to change the terrain specified by the `Gui-Terrain-Database` parameter so that the front-end uses the appropriate type of database. For more information, please see the sections that describe creating scenarios in *VR-Forces Users Guide* and *VR-Forces 3D Front End Users Guide*.

### 3.3. Editing the Plan Files

VR-Forces saves two plan files for each scenario, an entity plan file (*.pln*), which stores the plans for all individual entities, and the global plans file (*.gpl*), which stores global plans.

Plan files are created interactively in the VR-Forces Plan window. The Plan window ensures that the plan is written using the correct syntax. We recommend that you not try to create plan files or edit plan files by hand. However, there may be occasions when it makes sense to edit a plan file by hand or programmatically. For example, if you change the name of an entity, updating each entity's plan in the Plan window to change all plan statements that refer to that entity could be tedious. It would be much quicker to open the plan file in a text editor and replace appropriate instances of the old entity name with the new entity name.



Before you edit a plan file, make a backup copy that you can revert to if necessary.

---

### 3.3.1. The Structure of an Entity Plan File

Figure 3-2 illustrates a simple entity plan that specifies one task for each of two entities (a global plan has a similar syntax):

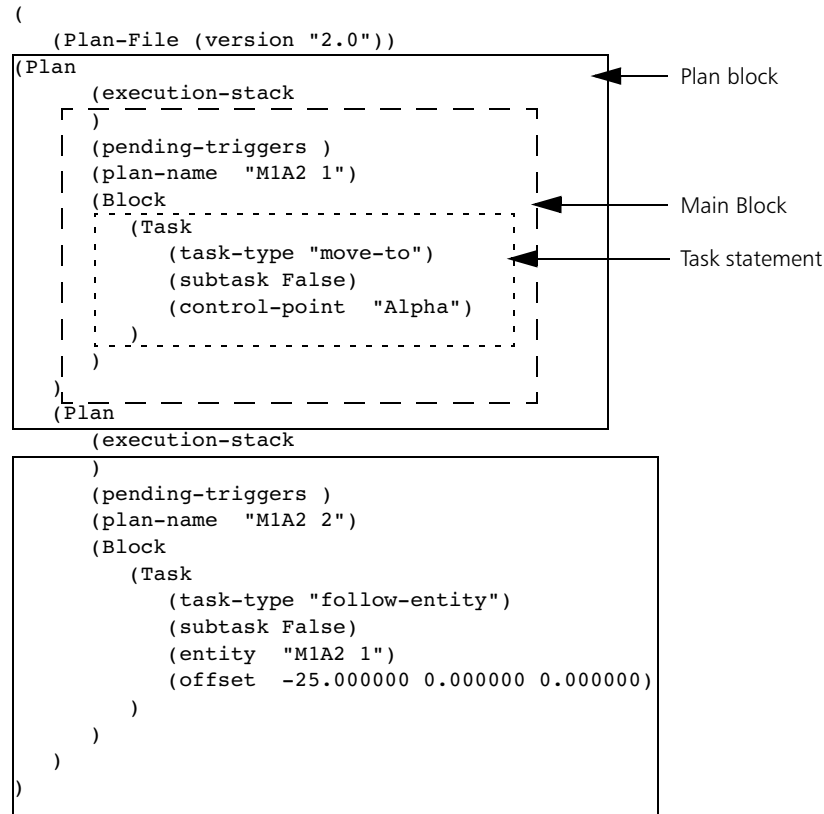


Figure 3-2. Simple plan

Plans are specified for entities by entity name. The plan is specified within a Plan block. The content of the plan is specified in the Main block. Tasks and set statements are specified in the Main block and in conditional statements. Conditional statements can be specified in the main block and can be nested in other conditional statements.

The basic syntax for a set statement in a plan is:

```
(Block
  (Set
    (set-data-request-type "set-heading")
    (heading 0.872665)
  )
)
```

Figure 3-3 illustrates the basic syntax for a conditional expression. It includes a task statement in the main block to show relative indentation.

```

(Block ← Main block
  (If ← If statement
    (Condition
      (EntInArea
        (Entity "M1A2 1")
        (Area "Blue")
        (Modifier ",")
      )
    )
    (Then-Block ← Then block
      (Task
        (task-type "move-to")
        (subtask False)
        (control-point "Bravo")
      )
    )
    (Else-Block ← Else block
      (Task
        (task-type "move-to")
        (subtask False)
        (control-point "Alpha")
      )
    )
  )
  (Task ← Task statement
    (task-type "follow-entity")
    (subtask False)
    (entity "M1A2 1")
    (offset -25.000000 0.000000 0.000000)
  )
)

```

Figure 3-3. Conditional expression with subblocks

To edit a plan file by hand, add Task, Set, If, While, and When sections as needed to the Block section for each entity. For tasks or set statements, you will always have a **task-type** or **set-data-request-type** parameter that needs to contain the type of task or set data request. The presence of additional parameters depends on the task or set data request that you created. You will need to know the names of the parameters and what type of value to specify.

Make sure that you match parentheses properly.

### 3.4. Editing the Object Map File

Mappings between objects and back-ends are saved to the object map file. If you have a scenario that uses multiple back-ends and you are uncertain about the site:application values you need to use to run the scenario, you can view the object map file to refresh your memory. The following example shows two entities and a control object mapped to two different back-ends:

```
(map-entry
  (address 1 3201)
  (name "M1A2 1")
)
(map-entry
  (address 2 3400)
  (name "T80 2")
)
(map-entry
  (address 1 3201)
  (name "route_1")
)
```

The **address** parameter lists the site and application number used for the back-end on which the object was simulated.

If you want to run an existing scenario using back-ends with different site and application values, you could edit the object map file and change the address field to match the new site and application values. However, if you make any mistakes in this process, your scenario might not run correctly. As an alternative to editing the object map file by hand, you could use the load balancing option to remap objects to back ends. For details about load balancing, please see Section 5.2.3, “[Load Balancing a Scenario](#)” in *VR-Forces Users Guide*.

### 3.5. Editing the Order of Battle File

VR-Forces saves entity state information and graphical object information to the order of battle file.

If you save a scenario after the simulation has run, for example, as a checkpoint, the order of battle file stores the state information for the entities at the time of the save. The next time you load this scenario, VR-Forces creates the entities using the settings in the object parameter database, but then overrides those settings with the information in the order of battle file. Therefore, changes you make to the object parameter database might not get picked up by existing scenarios. If you want to change a basic entity value, you might need to edit the order of battle file.

### 3.5.1. Saving Default Entity Parameters to the Order of Battle File

When an entity in a scenario is saved to the order of battle (OOB) file, parameters that are unchanged from the initial values in the object parameter database entry are written out in the form:

```
(parameter-name USE-DEFAULT)
```

The USE-DEFAULT keyword indicates that the next time this file is loaded (as part of a scenario) it should be initialized from data in the entity's object parameter database entry. The practical effect of the USE-DEFAULT keyword in OOB files is that if you modify the object parameter database, the next time you load a scenario those changes can affect the entities and objects in the scenario.

This lets you experiment with changes to entity parameters and view the effects of the changes to existing scenarios, without modifying the order of battle files.

## 3.6. The Overlay File

The Overlay file (*.ovl*) saves information about unpublished overlay objects and the order in which overlays and overlay objects should be drawn. (Published overlay objects are saved in the order of battle file.)

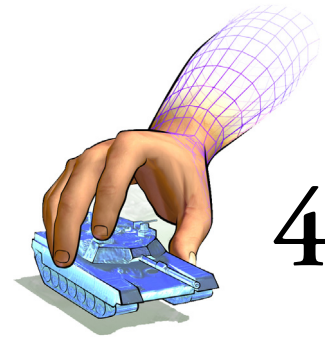
## 3.7. The Selection Groups File

The Selection Groups file (*.sgp*) saves information about selection groups. The information saved does not lend itself to editing by hand.

## 3.8. Temporary Scenario Directories

When you load a scenario, VR-Forces creates temporary directories in which it stores files that it needs to maintain and update the scenario. To prevent conflicts in the temporary directories when running multiple copies of VR-Forces from the same directory, VR-Forces creates unique temporary directories. The directories are created in the path specified by the TMP environment variable. If this variable is not specified on Linux, the */tmp* directory is used. The TMP variable must be specified on Windows. If VR-Forces cannot create a temporary directory, an error message is printed and the application exits. In the front-end, the temporary directory is created by *DtVrfRemoteController* and starts with the prefix *vrfte*. In the back-end, the temporary directory is created by *DtCgfDispatcher* and has the prefix *vrfbe*. In both cases, the temporary directory is removed when VR-Forces shuts down.





## *Merging Scenarios*

---

This chapter explains how to merge scenarios with the Scenario Merge tool.

|                                                      |     |
|------------------------------------------------------|-----|
| The Scenario Merge Tool .....                        | 4-2 |
| Starting Scenario Merge .....                        | 4-3 |
| Creating a Scenario Merge Project.....               | 4-3 |
| Adding Scenarios to a Project .....                  | 4-4 |
| Removing Scenarios from a Project.....               | 4-4 |
| Specifying an Ignore List File.....                  | 4-5 |
| Saving a Project .....                               | 4-6 |
| Loading a Project.....                               | 4-6 |
| Merging Scenarios.....                               | 4-7 |
| Displaying Scenario Files .....                      | 4-7 |
| Editing Scenario Files .....                         | 4-8 |
| Closing Scenario Files.....                          | 4-8 |
| Specifying the Output Notification Level.....        | 4-8 |
| Using the Scenario Merge Command-Line Interface..... | 4-9 |

## 4.1. The Scenario Merge Tool

VR-Forces users often assign scenario development to multiple persons or teams. Then they merge the separately-developed scenarios into a master scenario. Merging scenarios by hand can be a difficult task due to the requirement to reconcile entity and object names, plans, and other scenario elements that could be duplicated, particularly if objects are named using the default conventions.

The Scenario Merge tool allows you to combine two or more VR-Forces scenarios into a single scenario. Scenario Merge detects and resolves conflicts so that the simulated objects can co-exist in a single scenario, without losing their unique identity. Likewise, references to conflicting names are resolved, so that the plans from both scenarios work properly.

Scenario Merge has a graphical user interface and a command-line interface. It is an off-line tool for combining existing scenarios.

Scenario Merge has an API that lets you merge scenarios programmatically. For details, please see the *DtScenarioMergeDirector* class (*scenarioMergeDirector.h*) in the class documentation.

## 4.2. Starting Scenario Merge

The graphical user interface is the primary way to use the Scenario Merge tool.

- To start Scenario Merge in Windows, on the Start menu, choose **Programs** → **MAK Technologies** → **VR-Forces x.x** → **Tools** → **Scenario Merge**.

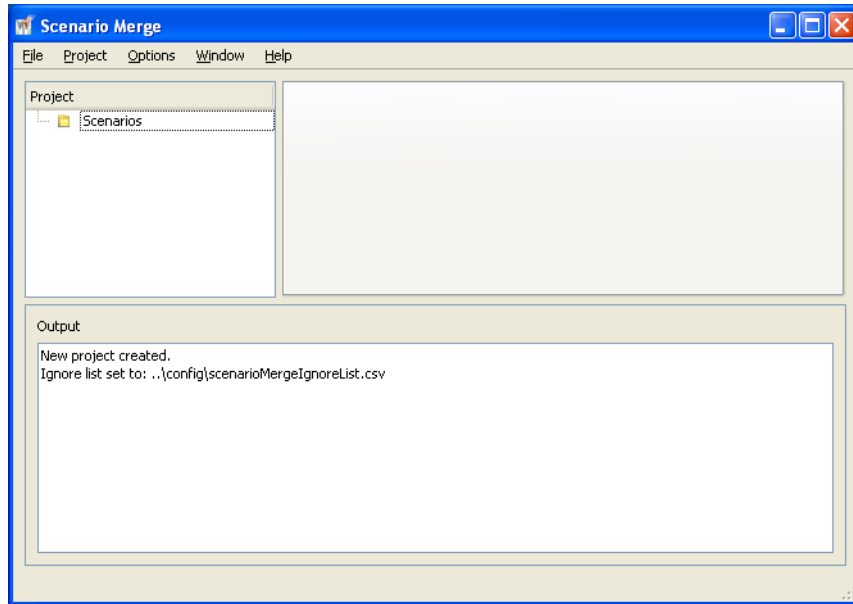


Figure 4-1. Scenario Merge window at startup

- To start the Scenario Merge from the command line, enter the following:  
`scenarioMerge [projectFile]`

If you start Scenario Merge from the command line and specify *projectFile*, Scenario Merge loads the project file. Otherwise, it opens a new project. Command-line syntax is described in [“Using the Scenario Merge Command-Line Interface,”](#) on page 4-9.

## 4.3. Creating a Scenario Merge Project

Before you can merge scenarios, you must create a Scenario Merge project. A project identifies the scenarios to merge, where to write the merged scenario, and optionally, a list of attributes to ignore during the merge process.

When you start Scenario Merge, it automatically creates a new, empty project. However, you can also start a new project manually.

- To open a new project, choose **File** → **New Project**.

### 4.3.1. Adding Scenarios to a Project

You can add as many scenarios as you want to a project. However, the first scenario you add is the key scenario. It determines the parameters of the final scenario, including, terrain database, object parameter database, exercise start time, and so on. If any of the scenarios that you merge into the first scenario have different scenario parameters, they are ignored and a diagnostic message is recorded in the project log file.

If you add scenarios from different terrains, Scenario Merge uses only geocentric coordinates for entity position data. This is accomplished using the Ignore List File, which is explained in [“Specifying an Ignore List File,”](#) on page 4-5.



Before you add a scenario created in VR-Forces 3.10.x or earlier to a project, open it in the current version of VR-Forces and save it.

---

To add a scenario to a project, use one of the following methods:

- ♦ Choose **Project** → **Add Scenario**, and select a scenario to add.
- ♦ Drag a scenario file (.scn) from Windows Explorer to the Scenario Merge Project tree view.
- ♦ Drag a scenario folder from Windows Explorer to the Scenario Merge Project tree view. If you drag a folder, Scenario Merge adds all scenario files in the folder to the project.

### 4.3.2. Removing Scenarios from a Project

To remove a scenario from a project:

1. In the Project window, select the scenario that you want to remove.
2. Choose **Project** → **Remove Scenario**.

### 4.3.3. Specifying an Ignore List File

Sometimes it is useful to suppress values when writing to the final scenario. For example, Scenario Merge suppresses any reference to a local coordinate system in the Order of Battle file (*.oob*) file. You can specify values to ignore in an ignore list file. The ignore list file is a comma-delimited text file, for example:

```
local-kinematics-state,IGNORE  
local-vertices,IGNORE
```

The first value is the string to search for during a merge operation. The second value is a text string that will replace the key in the merged scenario.

If a substitution is specified in the ignore list, it is applied to the following files in the scenario:

- ♦ Order of Battle (*.oob*)
- ♦ Plan (*.pln*)
- ♦ Global Plan (*.gpl*)
- ♦ Object Map (*.omp*).

To create an ignore list file, you must use a text editor. Scenario Merge just lets you specify which file to use. You must include the following lines from the default ignore list:

```
local-kinematics-state,IGNORE  
local-vertices,IGNORE
```

These entries are needed when merging scenarios that use different terrains.

To specify the Ignore List file:

1. Choose **Project** → **Specify Ignore List**. The Specify Ignore File dialog box opens.
2. Type a filename or click the Browse button and select the Ignore List file.
3. Click OK.

#### 4.3.4. Saving a Project

When you save a project, Scenario Merge creates a project file. You can load the project at a later date to merge the scenario again or change which scenarios you want to merge. You can also use a project file to merge files from the command-line. For details about running from the command-line, please see [“Using the Scenario Merge Command-Line Interface,”](#) on page 4-9.

To save a project:

1. Choose **File** → **Save Project**. The Scenario Merge dialog box opens.
2. Select a directory and type a name for the project.
3. Click Save. The project gets saved with the extension *.smp*.

#### The Scenario Merge Project File

A merge project is a comma-delimited text file that lists the scenarios to merge, the desired location for the final result, and merge options. The following is an example of a Scenario Merge project file:

```
OUT,..\\data\\scenarios\\demoMerge
IGN,..\\config\\scenarioMergeIgnoreList.csv
SCN,..\\data\\scenarios\\scenarioOne\\scenarioOne.scn
SCN,..\\data\\scenarios\\scenarioTwo\\scenarioTwo.scn
SCN,..\\data\\scenarios\\scenarioThree\\scenarioThree.scn
SCN,..\\data\\scenarios\\scenarioFlour\\scenarioFlour.scn
```

Each line starts with one of the following keywords:

- OUT specifies the location for the output scenario files.
- IGN specifies an Ignore List file. For details, please see [“Specifying an Ignore List File,”](#) on page 4-5.
- SCN specifies the path of a scenario to merge in.



Paths are relative to the location of *scenarioMerge.exe*.

---

#### 4.3.5. Loading a Project

If you have saved a project, you can load it, edit it as necessary, and regenerate the merged scenario.

To load a project:

1. Choose **File** → **Load Project**. An Open dialog box opens.
2. Select the Scenario Merge project file that you want to load.
3. Click Open.

## 4.4. Merging Scenarios

- To merge the scenarios in a project, choose **Project** → **Merge**. The scenarios are merged and informational messages are displayed in the console.

## 4.5. Displaying Scenario Files

You can display the text of one or more of the scenario files in a project.

- To display a scenario file, in the Project window, double click the file. It is displayed to the right of the Project window.

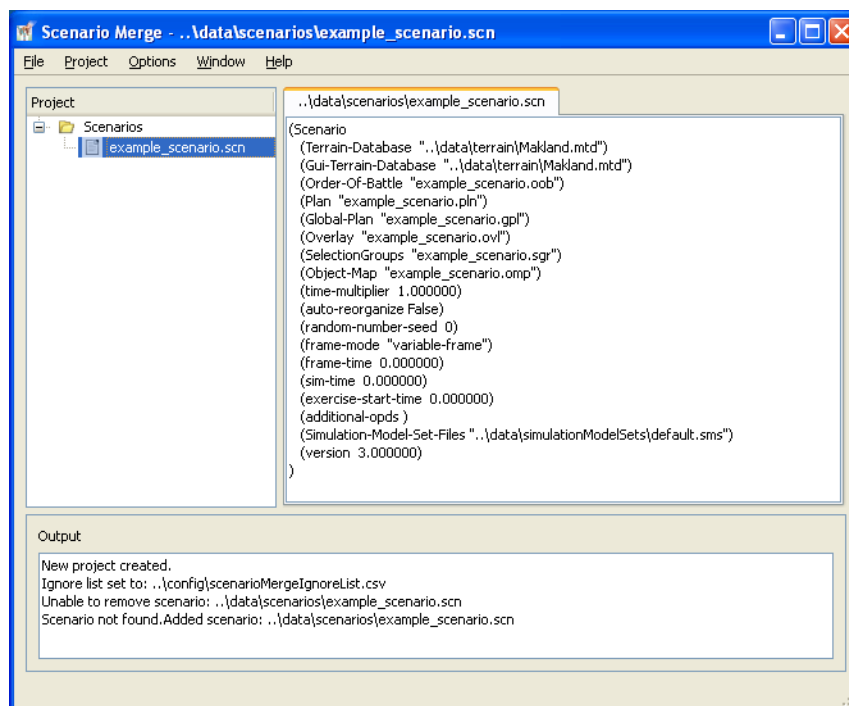


Figure 4-2. Scenario file display

### 4.5.1. Editing Scenario Files

If you display a scenario file, you can edit it. Any edits you make are saved back to the original scenario. You cannot edit the scenario file that gets created when you merge scenarios.



---

Editing scenario files is risky to scenarios and is advisable only for experienced VR-Forces users. Be sure you fully understand the contents of the file.

---

To edit a scenario file:

1. In the scenario file display window, make any edits desired.
2. Choose **File** → **Save Scenario** or **Save All**.

### 4.5.2. Closing Scenario Files

To close a scenario file that is displayed by Scenario Merge, use one of the following methods:

- ♦ Choose **Window** → **Close**. If multiple files are open, this closes the most recently opened one.
- ♦ Choose **Window** → **Close All**.
- ♦ Choose **File** → **Close Scenario**.

## 4.6. Specifying the Output Notification Level

You can specify the notification level used for sending messages to the output console. The notification levels correspond to the notification levels used throughout MÄK products and described in Chapter 2, [VR-Forces Configuration Files](#).

- To set the notification level, choose **Options** → **Set Notify Level** → *level*, where *level* is one of the following:
  - ♦ Fatal
  - ♦ Warning
  - ♦ Info
  - ♦ Verbose
  - ♦ Debug.



## 4.7. Using the Scenario Merge Command-Line Interface

To run the Scenario Merge tool from the command line, use the following syntax:

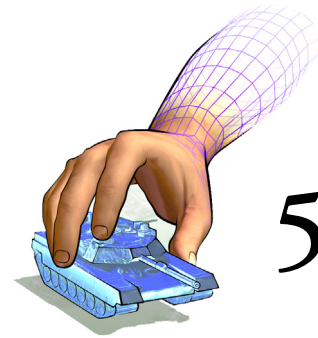
```
scenarioMerge -c projectFile
```

where:

**-c** specifies that Scenario Merge should run in command-line mode.

*projectFile* specifies a Scenario Merge project file to load. For details about creating project files, please see [“Adding Scenarios to a Project,”](#) on page 4-4.





# *Editing Entity Models*

---

The Entity Editor is a stand-alone tool that lets you configure entities, edit the Create menu, and create new simulation model sets.

|                                                                      |      |
|----------------------------------------------------------------------|------|
| Introduction .....                                                   | 5-3  |
| Simulation Model Sets.....                                           | 5-4  |
| Starting the Entity Editor .....                                     | 5-4  |
| Creating a New Simulation Model Set .....                            | 5-5  |
| Specifying the Default Simulation Model Set for Scenarios .....      | 5-6  |
| Opening a Simulation Model Set .....                                 | 5-6  |
| Editing an Entity Model .....                                        | 5-7  |
| Changing an Entity's Icon .....                                      | 5-8  |
| Editing an Entity's Category .....                                   | 5-10 |
| Editing an Entity's Force.....                                       | 5-11 |
| Changing an Entity's Entity Enumeration .....                        | 5-12 |
| Changing an Entity's Name .....                                      | 5-12 |
| Specifying Whether an Entity can be Created.....                     | 5-13 |
| Specifying a 3D Model for an Entity .....                            | 5-14 |
| Editing Behavioral Parameters .....                                  | 5-14 |
| Editing a Lifeform's DI-Guy Character, Appearance, and Animation.... | 5-15 |
| Undoing (Reverting) Changes to an Entity .....                       | 5-16 |
| Editing Aggregates .....                                             | 5-17 |
| Adding Subordinates .....                                            | 5-19 |
| Removing Subordinates .....                                          | 5-20 |
| Replacing Subordinates .....                                         | 5-20 |
| Changing the Order of Subordinates .....                             | 5-20 |
| Editing a Subordinate Aggregate.....                                 | 5-21 |
| Adding Formations.....                                               | 5-21 |

|                                                          |      |
|----------------------------------------------------------|------|
| Removing Formations .....                                | 5-22 |
| Renaming Formations .....                                | 5-22 |
| Specifying the Default Formation .....                   | 5-22 |
| Expanding and Collapsing the Display of Formations ..... | 5-22 |
| Displaying Bounding Boxes .....                          | 5-22 |
| Copying Formations .....                                 | 5-23 |
| Laying Out a Formation Automatically .....               | 5-24 |
| Changing a Formation's Layout by Hand .....              | 5-24 |
| Editing an Entity's Systems .....                        | 5-25 |
| Adding A System .....                                    | 5-26 |
| Selecting a Damage Model or Movement Model .....         | 5-26 |
| Editing a System's Properties .....                      | 5-27 |
| Editing a System in the OPD Editor .....                 | 5-27 |
| Removing a System .....                                  | 5-27 |
| Creating a New Entity .....                              | 5-28 |
| Creating a New Entity from an Existing Entity .....      | 5-28 |
| Deleting an Entity Model from a Model Set .....          | 5-29 |
| Navigating Among Edited Entities .....                   | 5-29 |
| Editing the Create Menu .....                            | 5-29 |
| Generating the Create Menu Automatically .....           | 5-30 |
| Editing the Create Menu Manually .....                   | 5-31 |
| Undoing (Reverting) Changes to the Create Menu .....     | 5-33 |
| Editing the Category List .....                          | 5-34 |
| Adding a New Category .....                              | 5-34 |
| Removing a Category .....                                | 5-34 |
| Renaming a Category .....                                | 5-35 |
| Undoing Category Changes .....                           | 5-35 |
| Managing the Forces List .....                           | 5-35 |
| Adding a Force .....                                     | 5-36 |
| Editing a Force .....                                    | 5-37 |
| Removing a Force .....                                   | 5-37 |

## 5.1. Introduction

Many VR-Forces users want to edit some or all of the following entity characteristics:

- ♦ Weapons systems
- ♦ Display image
- ♦ Basic parameters
- ♦ Sensors
- ♦ Force.

Users also want to add new entities and rearrange how entities are organized on the Create menu. The VR-Forces Entity Editor allows you to edit most of the aspects of an entity that a typical VR-Forces user will want to change. For example, you can change the force and icon associated with an entity type. You can easily add and remove sensors, weapons systems, and other major component systems, and can edit their properties.

The Entity Editor does not let you add or remove individual components or edit most of the more detailed entity parameters. For these advanced editing tasks, you can use the OPD Editor, which is described in Chapter 6, [Advanced Entity Model Editing](#), or edit specific configuration files, which are described throughout this manual.

If you want to add new entities or customize the entities supplied with VR-Forces, start with the Entity Editor. Use the OPD Editor or a text editor only if the Entity Editor does not let you make the changes you need to make.



Do not try to edit an entity model in the Entity Editor and the OPD Editor at the same time, except as described in [“Editing a System in the OPD Editor,”](#) on page 5-27. Use one tool, save the changes, and close the tool before opening the other.

---

### 5.1.1. Simulation Model Sets

A simulation model set (SMS) defines all of the entity and object models that you can create in a VR-Forces simulation. The simulation model set file tells VR-Forces where to find all of the configuration files it needs to create, simulate, and display entities and other simulation objects. These files include the object parameter database, damage files, formation files, and symbol mapping files. Every scenario must reference one or more simulation model sets.

If you use the Entity Editor to edit entity models, the only configuration file that you need to be aware of is the simulation model set file that you are using. All of the other files are handled by the Entity Editor.



When you use the Entity Editor, you are editing a simulation model set. The default SMS provided with VR-Forces is *default.sms*. We recommend that you keep the default SMS as a reference model set and not edit it. If you want to edit entity models, create a new SMS based on *default.sms* and use it as the default SMS for your scenarios. For details about creating a new SMS and changing the default SMS, please see [“Creating a New Simulation Model Set,”](#) on page 5-5.

---

### 5.1.2. Starting the Entity Editor

- To start the Entity Editor in Windows, choose **Start** → **Programs** → **MÄK Technologies** → **VR-Forces x.x** → **Tools** → **Entity Editor**. The Entity Editor opens ([Figure 5-1](#)).

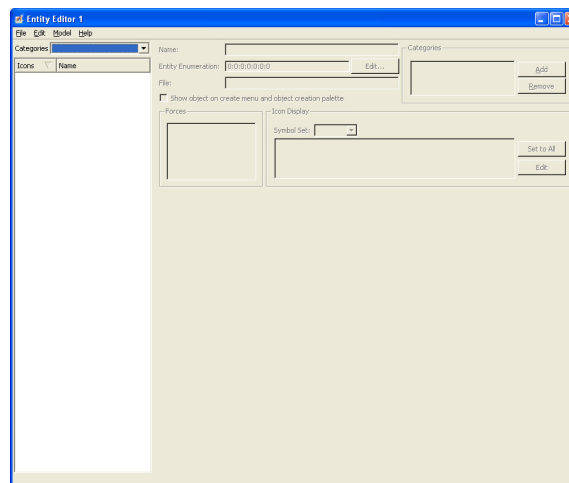


Figure 5-1. Entity Editor

To start the Entity Editor from the command-line in Windows or Linux:

1. Change directory to `./bin`.
2. Run:

```
entityEditor
```

## **5.2. Creating a New Simulation Model Set**

You can create new SMSs by copying and editing an existing model set or you can create a completely new model set and create the entities that you want to include in it.

To create a new simulation model set from an existing model set:

1. Choose **File** → **New Simulation Model Set**. The New Simulation Model Set dialog box opens (Figure 5-2).

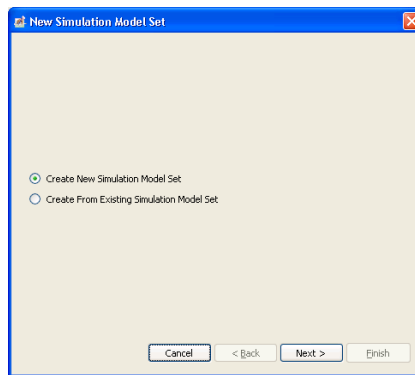


Figure 5-2. New Simulation Model Set dialog box

2. Select Create From Existing Simulation Model Set.
3. Click Next. The Select Model Set to Copy dialog box opens.
4. Select a model set and click Next. The New Model Set Name dialog box opens.
5. In the File name box, type a name for the new model set.
6. Click Finish. The Entity Editor loads information from the source model set.
7. Make any changes you want to the model set.
8. Choose **File** → **Save Simulation Model Set**.

To create a completely new simulation model set:

1. Choose **File** → **New Simulation Model Set**. The New Simulation Model Set dialog box opens (Figure 5-2).
2. Select Create New Simulation Model Set.
3. Click Next. The New Model Set Name dialog box opens.

4. In the File name box, type a name for the new model set.
5. Click Finish. The Entity Editor loads systems information from the default model set.
6. Create the entities that you want to have in the model set, as described in [“Creating a New Entity,”](#) on page 5-28.
7. Choose **File** → **Save Simulation Model Set**.

### 5.2.1. Specifying the Default Simulation Model Set for Scenarios

When you create a new scenario, the New Scenario dialog box lists *default.sms* as the SMS to use. You can change the default SMS by editing the `simulationModelSet` parameter in *vrfSim.mtl*.

## 5.3. Opening a Simulation Model Set

To edit entity models in the Entity Editor, you must open a simulation model set.

To open a simulation model set:

1. Choose **File** → **Open Simulation Model Set**. The Open Model Set dialog box opens.
2. Select the SMS you want to open.
3. Click Open. The SMS is loaded and a list of entities is displayed in the Entity Editor ([Figure 5-3](#)).



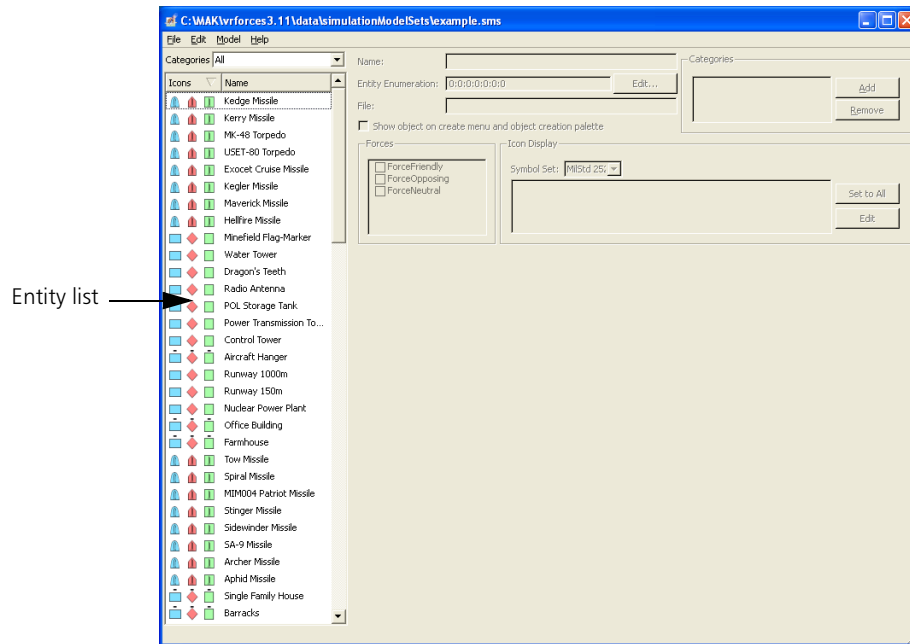


Figure 5-3. Entity Editor with model set loaded

## 5.4. Editing an Entity Model

This section explains how to edit an existing entity model. You can:

- ♦ Change an entity's icon
- ♦ Change the force assignments
- ♦ Add and remove systems
- ♦ Edit the properties of systems
- ♦ Edit some entity parameters
- ♦ Change the entity's category
- ♦ Change the entity's name
- ♦ Specify whether or not the entity can be created from the GUI.



When you edit an entity model in the Entity Editor, you are essentially editing the OPE file that it uses. In the default SMS, some OPE files are used by more than one entity. If you edit a particular entity in the Entity Editor, the tool creates a unique OPE file for that entity, so the changes do not affect other entities that might be using the original OPE file.

### 5.4.1. Changing an Entity's Icon

Each object can be created as a member of one or more forces. For each force that it can be created under, it has an image that is displayed in the map and various menus and lists. You can change the image associated with each object for each force it is associated with. You can change the image by selecting the symbol set to use or you can select a specific image. You can mix the types of images used from entity to entity and force-to-force for a particular entity.

To change the symbol set for an entity's image:

1. Select the entity whose icon you want to change. Its attributes are displayed (Figure 5-4).

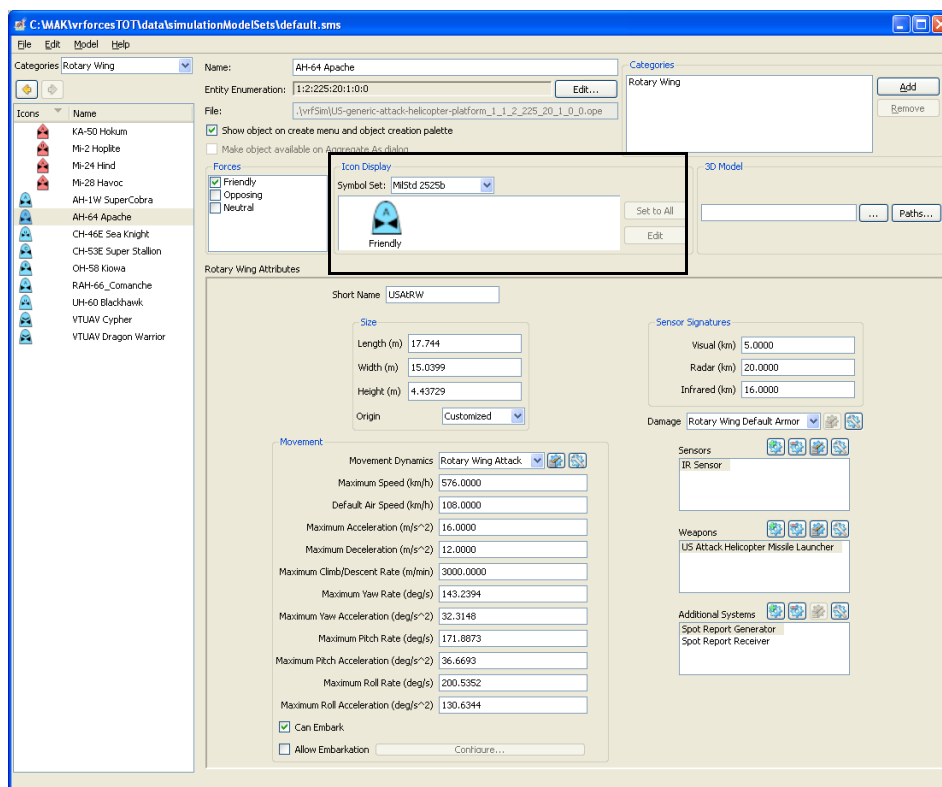


Figure 5-4. Attribute display for an AH-64 Apache

2. In the Icon Display group box, select an option from the Symbol Set drop-down list.



If you specify a specific image for an object, changing the symbol set does not affect the assigned image.

To specify an image for an entity:

1. Select the entity in the entity list.
2. In the Icon Display group box, click the image that you want to change.
3. Click the Edit. The Edit Icon dialog box opens.
4. Select an option from the Icon Type drop-down list. The Edit Icon dialog box redisplay to show the options appropriate to that icon type, as follows:
  - Animated Image: The Edit Icon dialog box displays a list of the images used to create the animated image. You can add to this list, remove files, change their order, or modify (replace) them. You can change how long each image is displayed (frame delay) as the animation is created.
  - Unicode True Type Fonts (2525B): You can change the echelon indicator, fill, and outline used for the image (Figure 5-5).

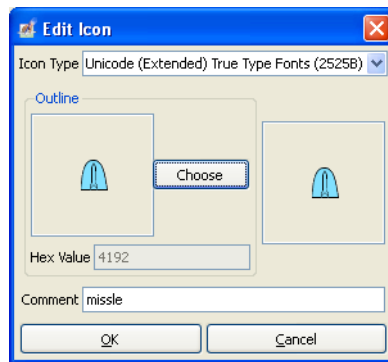


Figure 5-5. Edit Icon dialog box

- Image: You can select a raster image to use as the object icon.
  - Unicode (Extended) True Type Fonts (2525B): You can change the outline and the comment associated with the image.
5. Click OK. The new image is displayed in the Icon Display window.

### Using the Same Image for an Entity Regardless of Force

Ordinarily, the icons for entities in different forces are distinguished by color. For images, there may be differences in color or design. However, if you want to, you can use the same image for an entity regardless of its force.



This procedure is not valid for MilStd 2525b images.

To use the same image for an entity type regardless of its force:

1. In the Icon Display group box, select the image you want to use for all forces.
2. Click Set to All.



To use this procedure, you must select two or more forces for the entity.

---

### **Adding New Symbol Sets**

VR-Forces includes four symbol sets. You can add additional symbol sets and have them appear in the Entity Editor.

To add a symbol set to the Entity Editor:

1. Open the SMS file in a text editor. (SMS files are in *./data/simulationModelSets*.) The file contains a list of symbol map files similar to the following:

```
(symbol-map-files
  (symbol-file "MilStd 2525b" "$(model-set-directory)\gui\symbolmap-
    ttf.mtl")
  (symbol-file "MilStd 2525a" "$(model-set-directory)\gui\symbolmap-
    bmp.mtl")
  (symbol-file "Images" "$(model-set-directory)\gui\symbolmap-
    images.mtl")
  (symbol-file "Perspective Images" "$(model-set-
    directory)\gui\symbolmap-images-perspective.mtl")
)
```

2. Add a symbol-file entry for each symbol map file that you want to use.

### **5.4.2. Editing an Entity's Category**

Entities are organized by categories on the Create menu and the Entity Palette. You can change the category that an entity is listed in. You can also list it in more than one category.

To change an entity's category:

1. In the Categories group box, click the category that you want to change. The field becomes editable.
2. Select a different category from the drop-down list.

To add a creation category:

1. In the Categories group box, click Add. A new entry is added to the Categories list.

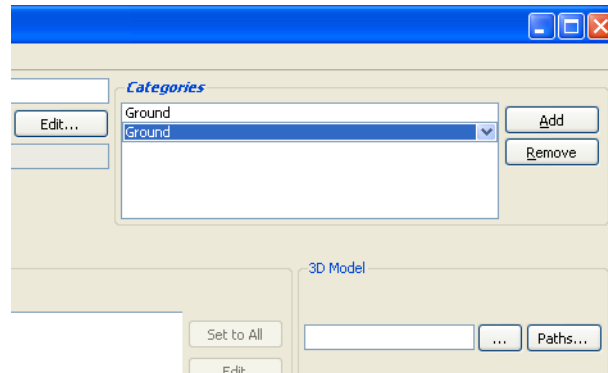


Figure 5-6. Adding a category

2. Select a category from the drop-down list.

### 5.4.3. Editing an Entity's Force

Each entity can be created as a member of one or more forces. You can specify which force an object can be created for. When you add an entity to a force, an icon for that force is assigned. If you remove an entity from a force, the icon is removed. You can quickly see which forces an entity is associated with by scanning the entity list in the Entity Editor (Figure 5-7).

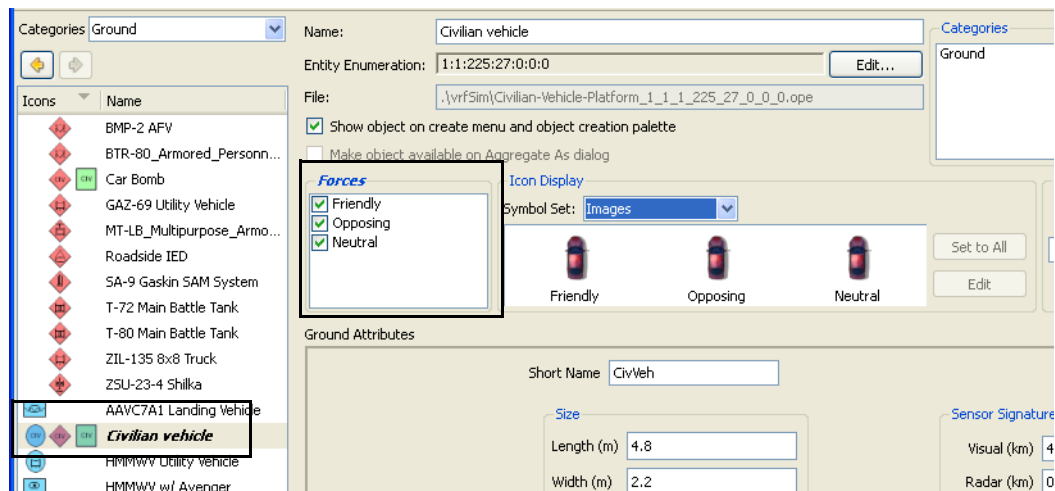


Figure 5-7. Entity (civilian vehicle) assigned to multiple forces

- To make or remove force assignments for an entity, in the Forces list, select or clear the check boxes for the forces that you want the entity to be assigned to.

#### 5.4.4. Changing an Entity's Entity Enumeration

The Entity Editor lets you change an entity's enumeration. You might want to or have to do this for any of the following reasons:

- ♦ If you create a new model, you must specify an entity enumeration.
- ♦ If you create a new model from an existing model, the Entity Editor automatically increments the existing model's enumeration for use with the new model. However, you might want to use a different enumeration than the one the Entity Editor generates.
- ♦ You might want to change an existing entity's enumeration.



When you change an entity's enumeration and save the change, the Entity Editor creates a new OPE file with the new enumeration in the file name. It does not delete or change the OPE file with the entity's previous enumeration.

---

To change an entity's entity enumeration:

1. Select the entity in the entity list.
2. Click Edit next to the Entity Enumeration text box. The Change Enumeration dialog box opens (Figure 5-8).

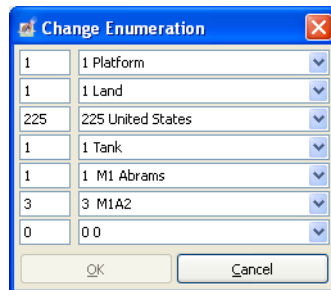


Figure 5-8. Change Enumeration dialog box

3. Edit the enumeration by typing new enumerations in the left column or selecting new coteries in the right column.
4. Click OK. The enumeration changes and the OPE filename changes.

#### 5.4.5. Changing an Entity's Name

To change an entity's name:

1. Select the entity in the entity list.
2. Type a new name in the Name text box.

### 5.4.6. Specifying Whether an Entity can be Created

You can control whether or not an entity is listed in the Create menu and Entity Palette. It may not make sense to create some types of entities, such as missiles, which are normally added to a scenario and removed dynamically.

- To enable or disable creation of an entity from the Create menu and Entity Palette, select or clear the Show Object on Create Menu and Object Creation Palette check box (Figure 5-9).

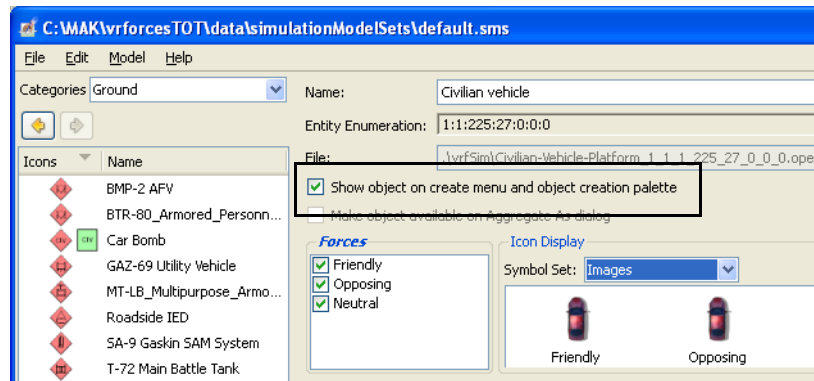


Figure 5-9. Show Object on Create Menu and Object Creation Palette check box

### 5.4.7. Specifying a 3D Model for an Entity

If you expect a scenario to be viewed in a 3D application, such as the MÄK Stealth or VR-Forces 3D Front End, you can specify the model to be used by the 3D application. It is your responsibility to make sure that the 3D application has that model and can load it.

To specify a 3D model:

1. Select the entity in the entity list.
2. In the 3D Model group box, click the browse button (Figure 5-10). The Select Model File dialog box opens.

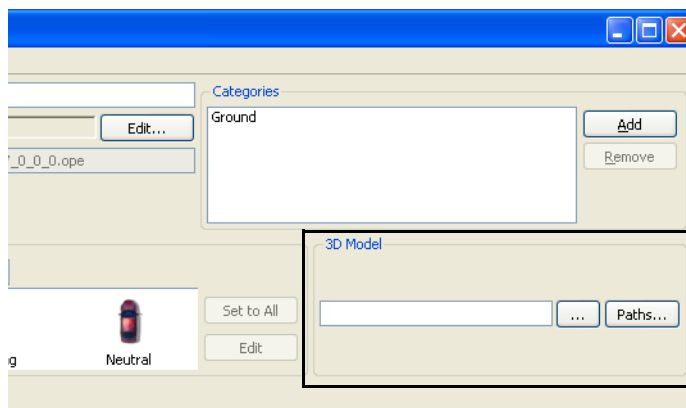


Figure 5-10. 3D Model group box

3. Browse to the model that you want to use and select the file.
4. Click Open.

### 5.4.8. Editing Behavioral Parameters

The Entity Editor lets you edit some of the entity parameters that affect behavior. For detailed information about entity parameters, please see the parameter descriptions in the OPD Editor online help.



The Sensor Signature parameters specify the distance at which other entities can sense the entity you are editing. By contrast, the sensor systems on the entity control its ability to sense other entities. For more information, please see “Editing an Entity’s Systems,” on page 5-25.

- To edit entity parameters, edit any of the parameters in the Attributes panel. (The panel reconfigures itself as you select different types of entities.)



### 5.4.9. Editing a Lifeform's DI-Guy Character, Appearance, and Animation

VR-Forces can specify the character, appearance, and animation values for DI-Guy models used in 3D visualization applications such as the VR-Forces 3D Front End and VR-Vantage. A model's character determines the kinds of movement it can make. The appearance determines what it looks like. The animation specifies the movements that the character makes. In the VR-Forces front-end, you can specify the animation with the DI-Guy Animation task. You can set the appearance with the DI-Guy Appearance set data request. This feature is valid only for lifeforms.

To specify the default DI-Guy characteristics for a lifeform:

1. In the Categories drop-down list, select the Lifeform category.
2. Select an entity in the entity list.
3. In the DI-Guy Character group box, select the default character from the Character Type drop-down list (Figure 5-11).

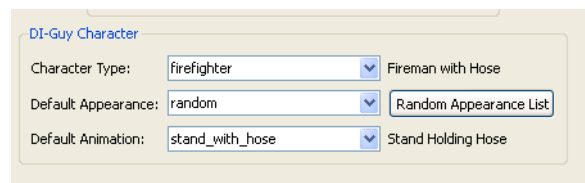


Figure 5-11. DI-Guy Character group box

4. Select the default appearance from the Default Appearance drop-down list. One of the default options is to use a random appearance.
5. If you selected random appearance, optionally, specify the list of appearances to choose from. For details, please see [“Configuring the Random Appearance List,”](#) on page 5-15.
6. Select the default animation from the Default Animation drop-down list. The default animation is the animation the entity uses if it is not tasked and if it has a posture and speed that is appropriate to the animation.

#### Configuring the Random Appearance List

If you specify that a lifeform use a random appearance for DI-Guy models, then VR-Forces chooses from a list of appearances when it specifies the appearance to use. By default, the list is all appearances available for the chosen character. However some appearances might not be appropriate for a scenario, so you can specify a subset of appearances to use.

To configure the random appearance list:

1. In the DI-Guy Character group box, click the Random Appearance List button. The Appearances dialog box opens (Figure 5-12). The appearances available for the character type are listed.

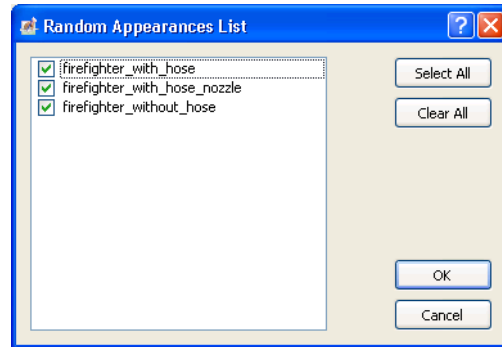


Figure 5-12. Random Appearances List dialog box

2. Select the appearances that you want to include in the random appearance list.
3. Clear any appearances that you do not want to include in the random appearance list.
4. Click OK.

#### **5.4.10. Undoing (Reverting) Changes to an Entity**

If you have made changes to an entity and have not yet saved the simulation model set, you can revert to the original entity configuration. You can revert an entity at any time prior to saving the SMS, regardless of how many other entities you may have changed since you edited the entity you want to revert.

To undo changes to an entity:

1. Select the entity in the entity list.
2. Choose **Model** → **Revert Model**.

## 5.5. Editing Aggregates

The procedures for creating aggregate entities and editing their “above-the-line” parameters are the same as for individual entities (described elsewhere in this chapter). This section describes how to edit aggregate composition and formations.

Each aggregate entity provided with VR-Forces has a pre-defined set of subordinates and the following defined formations:

- ♦ Column
- ♦ Line
- ♦ Vee
- ♦ Wedge.

You can easily edit subordinate lists and formations in the Entity Editor. You can add, remove, and replace subordinates and you can change their order in the subordinate list. (The order in the subordinate list determines which aggregate is considered the leader. It also affects aggregate reorganization.)

You can add, remove, and rename formations. You can also:

- ♦ Change the layout of subordinates in a formation
- ♦ Create new formations
- ♦ Specify the default formation.

The formation editor allows you to change the display of formations as follows:

- ♦ Expand and collapse the display of subordinate aggregates
- ♦ Zoom in and out on the display
- ♦ Display subordinate bounding boxes.

[Figure 5-13](#) illustrates the Entity Editor with an aggregate selected. Area 1 is “above the line”, as mentioned earlier in this section. Area 2 is the Composition Editor. Area 3 is the Formation Editor.

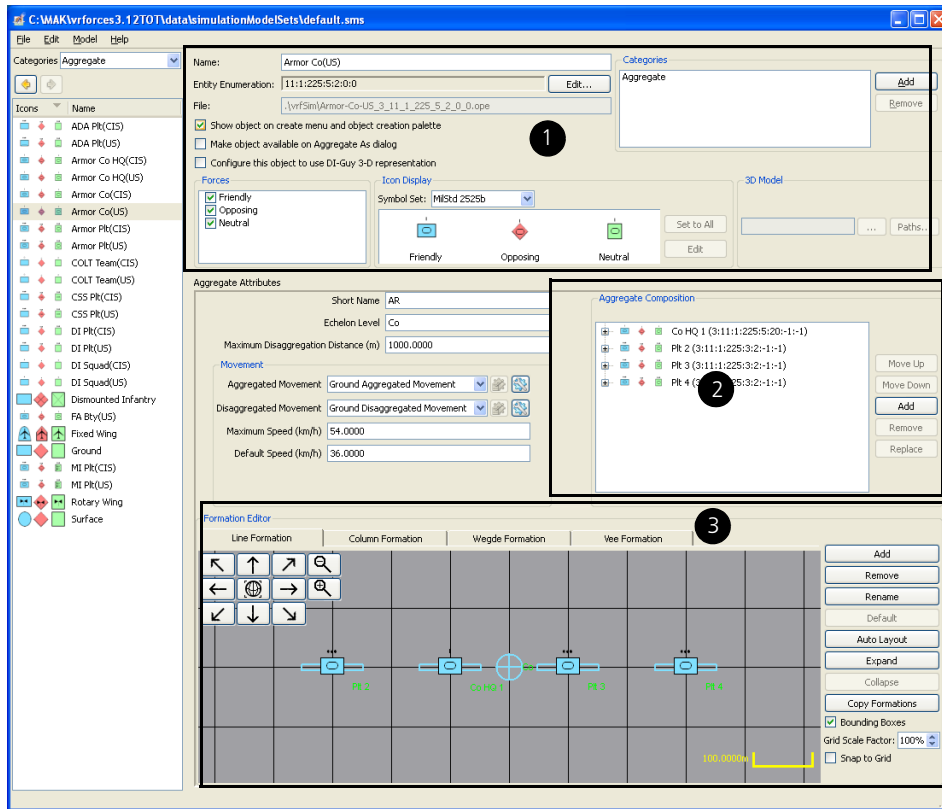


Figure 5-13. Aggregate page in Entity Editor

### 5.5.1. Adding Subordinates

When you create a new aggregate without copying an existing entity, you must specify its subordinates. You can also add subordinates to existing aggregates.

To add a subordinate to an aggregate:

1. In the Aggregate Composition group box (#2 in [Figure 5-13](#)), click the Add button. The Add to Hierarchy dialog box opens ([Figure 5-14](#)).

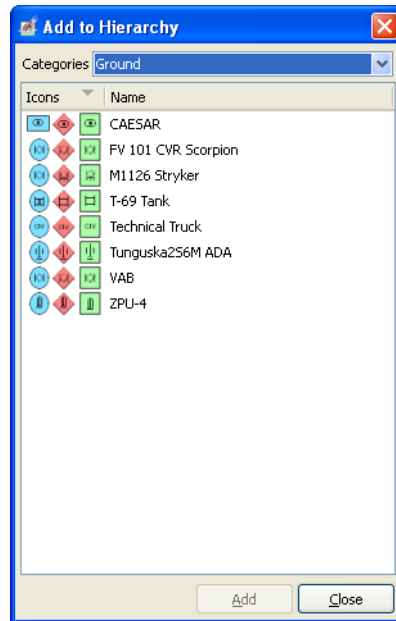


Figure 5-14. Add to Hierarchy dialog box

2. In the Categories drop-down list, select the category of the entity that you want to add as a subordinate.
3. In the list of entities, select the entity that you want to add.
4. Click Add. The entity is added to the list in the Aggregate Composition window. It is also added to each formation using a default layout algorithm.
5. Continue to add as many subordinates as desired.
6. Click Close.

### **5.5.2. Removing Subordinates**

To remove a subordinate from an aggregate:

1. In the Aggregate Composition window, select the subordinate that you want to remove.
2. In the Aggregate Composition group box, click the Remove button. The subordinate is removed from the aggregate.

### **5.5.3. Replacing Subordinates**

Replacing a subordinate is a convenient way to remove a subordinate and insert a different subordinate in the same location in one step.

To replace a subordinate:

1. In the Aggregate Composition window, select the subordinate that you want to replace.
2. In the Aggregate Composition group box, click the Replace button. The Add to Hierarchy dialog box opens (similar to [Figure 5-14](#)).
3. In the Categories drop-down list, select the category of the entity that you want to add as a replacement subordinate.
4. In the list of entities, select the entity that you want to add as a replacement.
5. Click OK. The entity replaces the selected entity in the Aggregate Composition window. It is also added to each formation in the location of the replaced entity.

### **5.5.4. Changing the Order of Subordinates**

The order of subordinates in the Aggregate Composition window determines which entity is the leader, the order in which aggregates are placed into formation, and the succession when aggregates are reorganized.

To change a subordinate's position in the aggregate composition:

1. In the Aggregate Composition window, select the subordinate that you want to move.
2. In the Aggregate Composition group box, click Move Up or Move Down.

### 5.5.5. Editing a Subordinate Aggregate

Aggregates can contain individual entities and other aggregates as subordinates. When you are editing an aggregate, you might decide that you want to edit one of its subordinates. You can jump to the subordinate quickly without selecting it in the main list of entities.



If you edit a subordinate aggregate, the changes affect all uses of that subordinate, not just the one in this particular aggregate.

To edit a subordinate:

1. In the Aggregate Composition window or in the Formation Editor, right-click the subordinate that you want to edit. A menu is displayed.
2. In the menu, choose **Edit**. The Entity Editor redisplay to show the selected entity.

### Navigating the Subordinate Hierarchy

If you jump to a subordinate in the Entity Editor, you can quickly back up to its superior aggregate by clicking the back arrow above the entity list ([Figure 5-19](#)).

### 5.5.6. Adding Formations

To add a formation:

1. In the Formation Editor group box, click Add. The Add Formation dialog box opens ([Figure 5-15](#)).

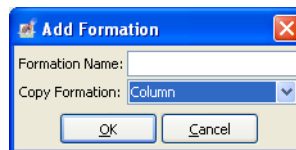


Figure 5-15. Add Formation dialog box

2. In the Formation Name box, type a name for the formation.
3. In the Copy Formation drop-down list, select an existing formation to copy to the new formation.
4. Click OK.
5. Edit the layout of the formation as described in [“Changing a Formation’s Layout by Hand,”](#) on page 5-24.

### **5.5.7. Removing Formations**

To remove a formation:

1. In the Formation Editor group box, select the tab for the formation that you want to remove.
2. Click the Remove button. The tab is removed.

### **5.5.8. Renaming Formations**

To rename a formation:

1. In the Formation Editor group box, select the tab for the formation that you want to rename.
2. Click the Rename button. The New Name dialog box opens.
3. Type a name for the formation.
4. Click OK.

### **5.5.9. Specifying the Default Formation**

The default formation is the formation assigned to an aggregate when you create it in VR-Forces. The default formation is the leftmost tab in the Formation Editor.

To specify the default formation:

1. In the Formation Editor group box, select the tab of the formation that you want to specify as the default formation.
2. Click the Default button. The selected tab gets moved to the leftmost position.

### **5.5.10. Expanding and Collapsing the Display of Formations**

You can expand and collapse the display of formations in the Formation Editor window. This lets you see how entities are dispersed over the terrain. It can help you quickly identify any unintended variations in subordinate formations.

- To expand or collapse the display of formations in the Formation Editor, click the Expand or Collapse buttons in the Formation Editor group box.

### **5.5.11. Displaying Bounding Boxes**

You can display bounding boxes for the subordinate aggregates in an aggregate. This may be useful when you collapse the views of individual aggregates, but you still want to have a sense for the area that they occupy.

- To display or hide bounding boxes, select or clear the Bounding Boxes check box.



### 5.5.12. Copying Formations

You can copy the formations from one aggregate to another. When you copy formations, the subordinates in the aggregate are laid out in each formation according to their order in the Aggregate Composition window.

To copy a formation:

1. In the entity list, select the aggregate that you want to copy formations to.
2. In the Formation Editor group box, click Copy Formations. The Select formations dialog box opens (Figure 5-16).

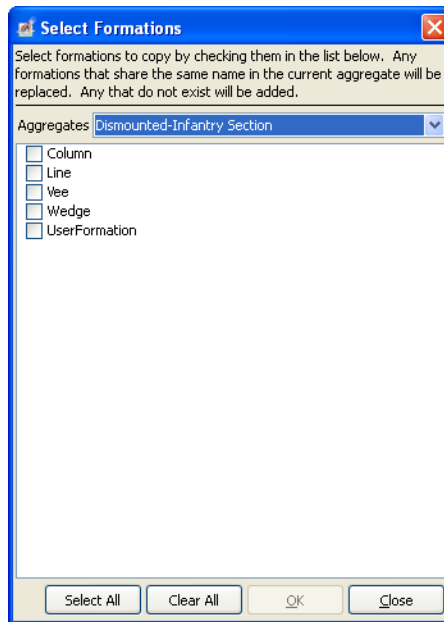


Figure 5-16. Select Formations dialog box

3. In the Aggregates drop-down list, select the aggregate whose formations you want to copy.
4. In the window, select the formations that you want to copy.
5. Click OK. The formations are copied from the selected aggregate to the currently displayed aggregate.

### 5.5.13. Laying Out a Formation Automatically

You can automatically distribute the entities in an aggregate into one of the built-in formations. You can specify the distance between each entity in the formation.

To lay out a formation automatically:

1. In the Formation Editor group box, select the tab for the formation that you want to lay out.
2. Click Auto Layout. The Auto Layout dialog box opens.
3. In the drop-down list, select a formation type.
4. Specify the distance, in meters, between each entity in the formation.
5. Click OK. The entities in the aggregate are reorganized to the new layout.

### 5.5.14. Changing a Formation's Layout by Hand

You can lay out a formation by hand. The crosshair in the formation editor indicates the center of the formation. This is where the aggregate icon is placed when the aggregate is collapsed or aggregated.

To change a formation's layout:

1. In the Formation Editor group box, select the tab for the formation that you want to lay out.
2. Drag each subordinate to the location where you want it to be.

#### Snapping Subordinates to the Grid

When you lay out a formation you can specify that entities snap to the grid. That is, they can only be placed at the intersection of vertical and horizontal grid lines. When snap-to-grid is enabled, if you want to place entities more closely together than the grid allows, you can change the scale factor.

- To enable snap-to-grid for entity placement, in the Formation Editor, select Snap To Grid.

#### Changing the Grid Scale Factor

The grid scale factor determines the distance, in meters, between grid lines. The default is 100 meters. When you change the scale factor, the distance between lines is changed by the specified percentage.

- To change the grid scale factor, in the Grid Scale Factor box, type a value or change the value using the spinner controls. The grid is resized to the new scale factor and the scale in the editing window shows the new distance between grid lines.

## 5.6. Editing an Entity's Systems

Entities have complex capabilities, such as the ability to move, to take damage, to sense other entities, and to fire munitions. These capabilities are organized as systems. You can quickly change how an entity functions by adding, removing, and changing its systems. The Entity Editor lists the systems available in each category and only lists systems that are appropriate for the entity type (Figure 5-17). For example, a lifeform can have an M-16 or an RPG, but not a cruise missile.

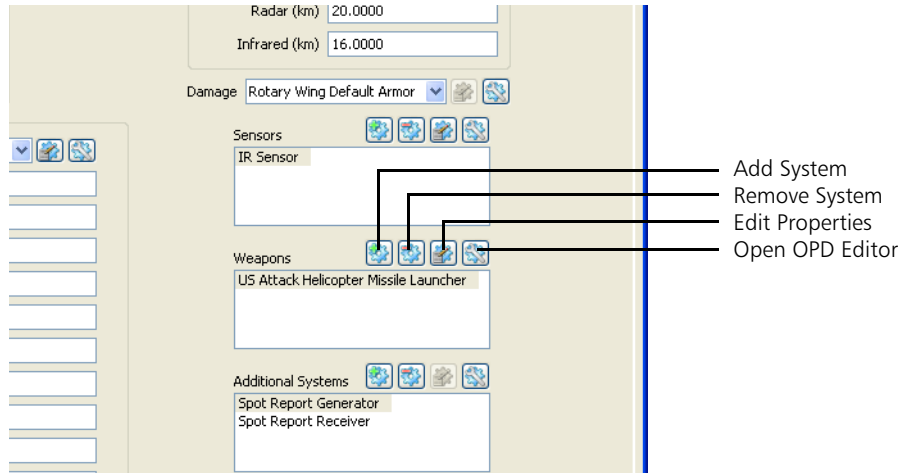


Figure 5-17. Entity systems

Some systems are interdependent. For example, if you add a weapon system to an entity, it cannot use it unless it also has a sensor system so that it can detect hostile entities. If you change or add systems, it is up to you to know if they have dependencies.

The components that make up a system have parameters and you can edit the parameters of system components by editing the properties of the system.

The Entity Editor lets you edit the following types of systems:

- ♦ **Damage.** You can specify the type of damage model that an entity uses. When you select a damage model, the Entity Editor configures the damage files that are appropriate for the entity type.
- ♦ **Sensors.** You can specify the sensors that an entity uses to locate other entities in the simulation. (By contrast, sensor signatures, which you can configure in the parameter section of the window, specify the distance at which other entity sensors can locate this entity.) When you add a weapon, the Entity Editor configures the components and connections necessary for the sensor to work.
- ♦ **Weapons.** The Entity Editor lists the weapons systems that are appropriate for the vehicle type. When you add a weapon, the Entity Editor configures the components, targets, ammunition, and connections necessary for the weapon to work.

- **Movement.** You can specify the type of movement system that an entity uses, for example, tracked, wheeled, pedestrian, and so on. When you select a movement system, the Entity Editor configures all of the actuators and controllers that the entity needs to implement the system.
- **Other systems.** This category includes spot reports and IFF transponders.

### 5.6.1. Adding A System

There is no specific limit to the number of sensors, weapons, or other systems that you can configure for an entity.

To add a system:

1. Click the Add System button for the system type you want to add (Figure 5-17). The Add System dialog box opens (Figure 5-18). It lists the systems of this type that are available for the entity you are editing. If you select a system in the window, the dialog box displays a brief description.

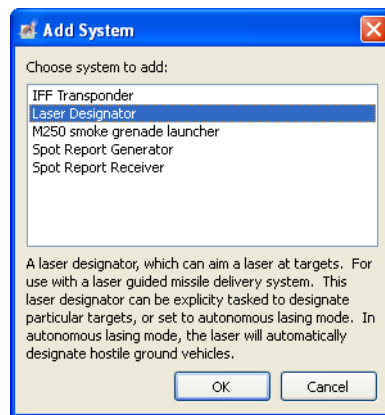


Figure 5-18. Add System dialog box

2. Select the system you want to add.
3. Click OK. The system is added to the list in the Entity Editor.

### 5.6.2. Selecting a Damage Model or Movement Model

You can only have one damage model and movement model specified for an entity.

- To specify a damage model, select an option from the Damage drop-down list.
- To specify a movement model, in the Movement group box, select an option from the Movement Dynamics drop-down list.

### 5.6.3. Editing a System's Properties

Some systems have properties that you can edit.

To edit a system's properties:

1. For sensors, weapons, or other systems, select the system whose properties you want to edit. (You do not have to select the damage or movement model, because there can only be one.) If the system has properties, the Properties button is enabled.
2. Click the Properties button ([Figure 5-17](#)). A Properties dialog box opens.
3. Edit the properties as desired.
4. Click OK.

### 5.6.4. Editing a System in the OPD Editor

Systems may have attributes that you cannot edit in the Entity Editor. You can jump directly to the OPD Editor from the Entity Editor to edit these attributes. When you open the OPD Editor from the Entity Editor, only the Systems page is displayed. You cannot edit other entity attributes.

Before you jump to the OPD Editor, you should save any changes that you have made to the simulation model set. If you make any changes in the OPD Editor, the Entity Editor is automatically updated after you save the changes in the OPD Editor.

To jump to a system in the OPD Editor:

1. Select the system that you want to edit.
2. Click the Open OPD Editor icon next to the system ([Figure 5-17](#)).

### 5.6.5. Removing a System

You can remove sensor, weapons, and additional systems.



If you remove a system that another system depends on, the other system will not work. It is your responsibility to know these dependencies.

---

To remove a system:

1. Select the system you want to remove.
2. Click the Remove System button ([Figure 5-17](#)).

## 5.7. Creating a New Entity

You can create a new entity by using an existing entity as a template or by building a completely new entity.

To create a completely new entity:

1. Choose **Model** → **New Model**. The Create New Entity dialog box opens.
2. Select an entity category from the Entity to Create drop-down list.
3. Type a name for the entity.
4. Click OK. The Entity Editor adds a new entry to the entity list. The editor shows default values for all the fields.
5. Specify an enumeration for the entity.
6. Specify symbol set and force values for the new entity.
7. Specify appropriate values for the entity parameters.
8. Type a name to be used in the entity's label in the Short Name text box.
9. Specify a movement system.
10. Specify a damage model.
11. Optionally, add weapons systems.
12. Optionally, add sensors.
13. Save the simulation model set.

### 5.7.1. Creating a New Entity from an Existing Entity

If you create a new entity from an existing entity, the new entity starts with all of the behavioral characteristics and systems of the original entity. It increments the entity enumeration. You just have to name it and customize it for the new entity.

To create a new entity from an existing entity:

1. In the entity list, select the source entity.
2. Choose **Model** → **New Model from Existing**. The Create New Entity dialog box opens.
3. Type a name for the entity.
4. Click OK. The new entity is added to the entity list. The enumeration is incremented and an OPE file name is created based on the entity name.
5. Edit the parameters to provide the new entity the characteristics you want for it.
6. Save the simulation model set.

## 5.8. Deleting an Entity Model from a Model Set

To delete an entity model from a simulation model set:

1. Select the entity in the entity list.
2. Choose **Model** → **Delete Model** or press **Delete**.

## 5.9. Navigating Among Edited Entities

If you have edited several entities and want to return to a previous entity without finding it in the entities list, you can use the Forward and Backward arrows to navigate among previously edited entities. The arrows act like the forward and back buttons in a web browser or online help viewer.

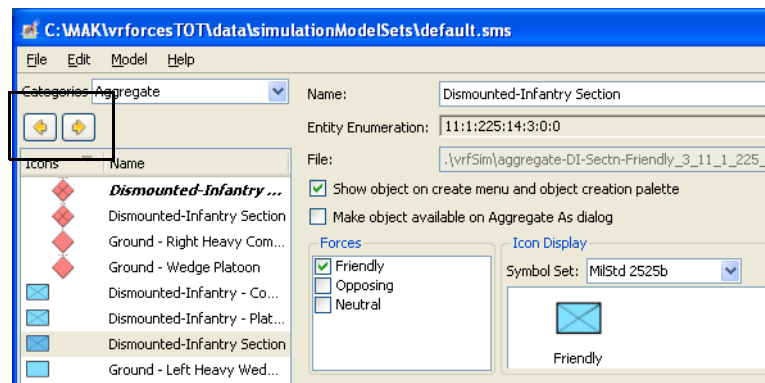


Figure 5-19. Forward and Back arrows

## 5.10. Editing the Create Menu

By default, VR-Forces automatically generates the Create menu based on the entity category and force assignments in the simulation model set. When you change the category or force assignments in the Entity Editor, these changes affect the automatically generated Create menu and the Entity Palette. Inclusion of an entity in these menus is controlled by the Show Object On Create Menu and Object Creation Palette check box. However, if you want to, you can configure a custom Create menu.



This procedure only affects the Create menu. It does not affect the organization of the Entity Palette.

### 5.10.1. Generating the Create Menu Automatically

To generate the Create menu automatically:

1. Choose **Edit** → **Edit Create Menu**. The Menu Editor dialog box opens ([Figure 5-20](#)).

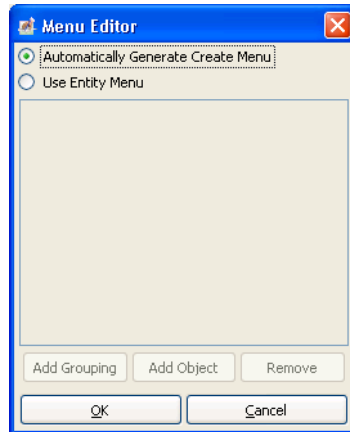


Figure 5-20. Menu Editor

2. Select Automatically Generate Create Menu.
3. Click OK.



### 5.10.2. Editing the Create Menu Manually

To edit the Create menu manually:

1. Choose **Edit** → **Edit Create Menu**. The Menu Editor dialog box opens.
2. Select Use Entity Menu. The window displays a tree list representing the current Create menu (Figure 5-21).

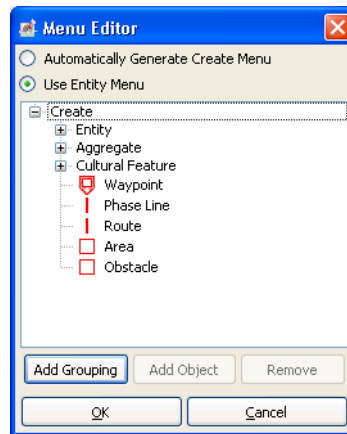


Figure 5-21. Menu Editor

3. To add a grouping:
  - a. Select a current grouping.
  - b. Click Add Grouping. The Add Group dialog box opens.
  - c. Select the location for adding the group: as a sibling, above or below the selection, or as a child.
  - d. Click OK. An entry called New Group is added (Figure 5-22).

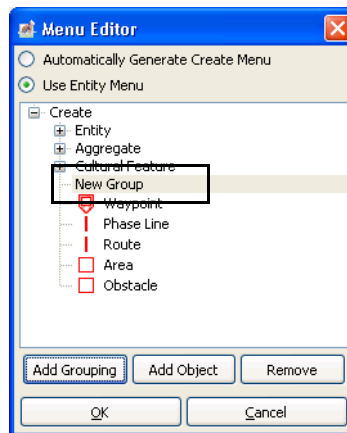


Figure 5-22. Menu Editor with new group added

- e. Double-click New Group to make it editable.
  - f. Type a name for the group.
4. To add an object to the Create menu:
  - a. Select a group title or an object in one of the groups.
  - b. Click Add Object. The Add Object dialog box opens (Figure 5-23).
  - c. Select the location where you want to add the object. If you selected a group title, the options are above or below it, or as a child. If you selected an object, the options are above and below it.
  - d. Select an object to add from the list. Filter the list as necessary to display the categories of objects that you want to add.

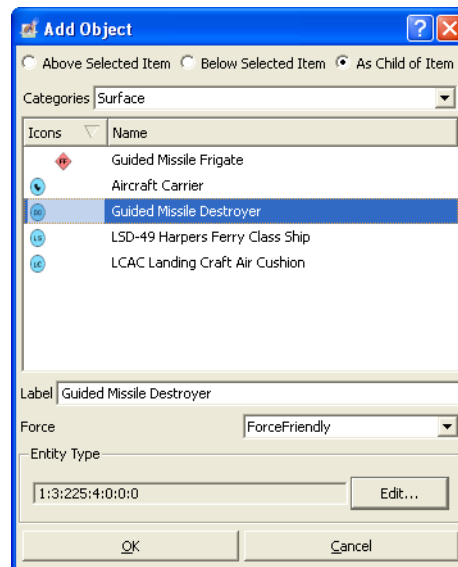


Figure 5-23. Add Object dialog box

- e. Optionally, edit the label and force for the object.
  - f. Click OK. The object is added to the menu (Figure 5-24).

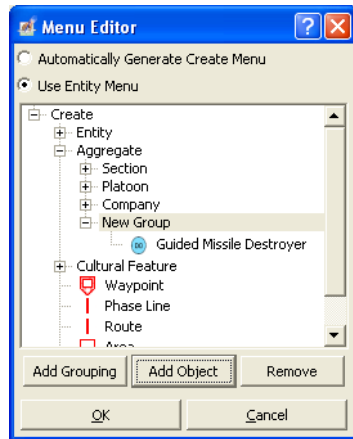


Figure 5-24. Menu Editor with new object added

5. You can edit any of the group or object labels.
6. You can drag an object to another group by dragging it onto the group title.

### 5.10.3. Undoing (Reverting) Changes to the Create Menu

If you have made changes to the Create menu, but have not saved the simulation model set, you can revert to the original menu.

- To undo changes to the Create menu, choose **Edit** → **Revert Create Menu**.

## 5.11. Editing the Category List

If you want to organize entities into categories that are not included in the default category list, you can add new categories. You can also remove and rename categories.

### 5.11.1. Adding a New Category

To add a new category:

1. Choose **Edit** → **Edit Categories**. The Categories Editor opens (Figure 5-25).

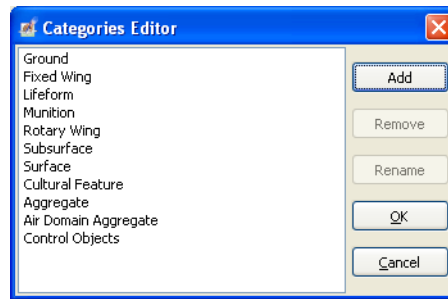


Figure 5-25. Categories Editor

2. Click Add. The New Category dialog box opens.
3. Type the name of the new category.
4. Click OK. The new category is added to the list in the Categories Editor.
5. Click OK.



If you add a category and do not assign an entity to it, it does not get saved when you save the simulation model set.

---

### 5.11.2. Removing a Category

To remove a category:

1. Choose **Edit** → **Edit Categories**. The Categories Editor opens.
2. Select the category that you want to remove.
3. Click Remove.
4. Click OK.

### 5.11.3. Renaming a Category

When you rename a category, all entities that are assigned to the old name are reassigned to the new name. The old name remains in the Categories drop-down list until you save the simulation model set, but it is not listed in the Categories Editor. If you assign an entity to the old category name before saving the SMS, the old name is retained and added to the list in the Categories Editor.

To rename a category:

1. Choose **Edit → Edit Categories**. The Categories Editor opens.
2. Select the category that you want to rename.
3. Click Rename. The Rename dialog box opens.
4. Type the new name.
5. Click OK. The categories list is updated.
6. Click OK.

### 5.11.4. Undoing Category Changes

You can undo changes made to the category list and entity category assignments.



---

When you undo category changes, all changes to the categories list and all changes to entity category assignments made since the last save of the simulation model set are reversed. Be sure that this is what you want before completing this procedure.

---

To undo category changes:

1. Choose **Edit → Revert Categories**. A confirmation prompt is displayed.
2. Click Yes.

## 5.12. Managing the Forces List

Each entity configuration page has a list of forces. You specify the forces for which an entity can be created. You can add new forces and removing existing forces. You can also change the color assigned to a force.

### 5.12.1. Adding a Force

Adding new forces may be useful if you are simulating participants from multiple countries, or if you want to easily model and change force relationships for non-governmental forces such as militias or insurgent groups.

To add a force:

1. Choose **Edit** → **Edit Forces**. The Forces Editor opens (Figure 5-26).

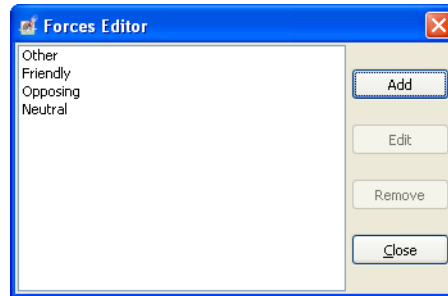


Figure 5-26. Forces Editor

2. Click Add. The Force Hostility Editor opens (Figure 5-27). The new force is assigned a force number. You cannot edit the force number.

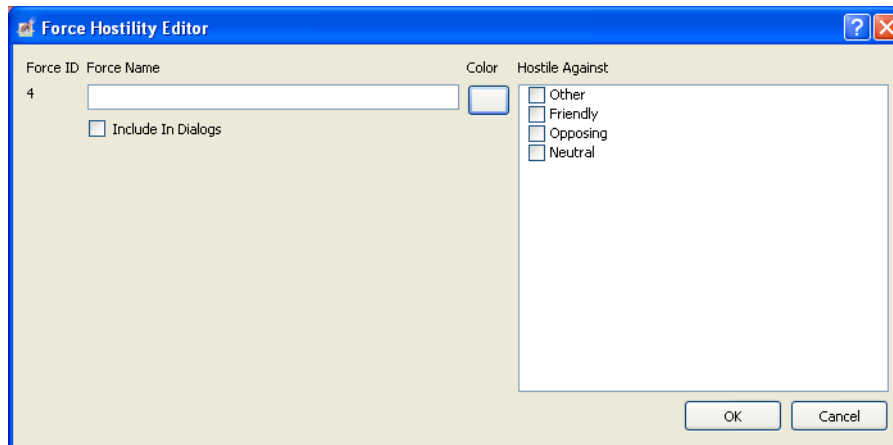


Figure 5-27. Force Hostility Editor

3. In the Force Name box, type a name for the force.
4. Click the Color button. A color picker opens.
5. Select a color for the force.
6. Optionally, in the Hostile Against window, select the forces that the new force will be hostile towards. (You can change hostility at run time in the Hostility Matrix. For details, please see Section 13.2, “[Managing Force Hostility Relationships](#)”, in *VR-Forces Users Guide*.)

7. If you want the force to be listed in dialog boxes that list forces (including the Forces list in the Entity Editor), select the Include in Dialogs check box.
8. Click OK.
9. Click Close.

### 5.12.2. Editing a Force

You can edit a force's name, color, hostility, and use in dialog boxes. Each force has a force number. When you edit a force, you are changing the attributes of that numbered force.

To edit a force:

1. Choose **Edit** → **Edit Forces**. The Forces Editor opens ([Figure 5-26](#)).
2. Select the force that you want to edit.
3. Click Edit. The Force Hostility Editor opens ([Figure 5-27](#)).
4. To edit the name, change the text in the Force Name box.
5. To edit the color associated with the force, click the Color button and choose a color from the color picker.
6. To edit hostility relationships, in the Hostile Against window, select the forces that the force should be hostile towards.
7. To edit whether or not the force is listed in dialog boxes that list forces (including the Forces list in the Entity Editor), select or clear the Include in Dialogs check box.
8. Click OK.
9. Click Close.

### 5.12.3. Removing a Force

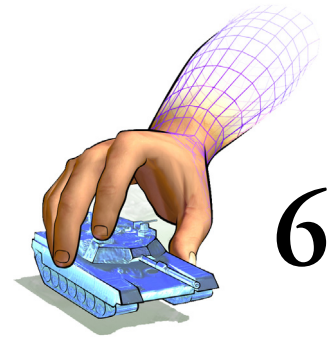
If you create a scenario that has entities with a particular force and you then remove the force, the next time you open the scenario, VR-Forces still maintains that force assignment for the entity. The force will be listed in the Hostility Matrix. However, entity icons will not be colored and you will not be able to create any entities with the removed force.

To remove a force:

1. Choose **Edit** → **Edit Forces**. The Forces Editor opens ([Figure 5-26](#)).
2. Select the force that you want to remove.
3. Click Remove. The force is removed from the list.
4. Click Close.







## ***Advanced Entity Model Editing***

---

This chapter describes how to use the OPD Editor to edit the object parameter database.

|                                                       |      |
|-------------------------------------------------------|------|
| The Object Parameter Database Editor .....            | 6-2  |
| Starting the OPD Editor .....                         | 6-2  |
| Loading an Object Parameter Database .....            | 6-3  |
| Changing the Object List View .....                   | 6-4  |
| Editing an Object's Parameters .....                  | 6-4  |
| Parameter Descriptions .....                          | 6-4  |
| Editing Individual Parameters .....                   | 6-5  |
| Editing Parameters for Multiple Objects .....         | 6-6  |
| Deleting Parameters and Components .....              | 6-6  |
| Restoring Parameters .....                            | 6-6  |
| Creating a New Object from an Existing Object .....   | 6-7  |
| Adding New Components and Parameters .....            | 6-7  |
| Adding New Components .....                           | 6-8  |
| Editing Components .....                              | 6-8  |
| Adding New Resource Parameters .....                  | 6-10 |
| Adding New Subordinate Object Parameters .....        | 6-10 |
| Adding New Soil Type Parameters .....                 | 6-11 |
| Editing Variables .....                               | 6-11 |
| Editing Systems .....                                 | 6-11 |
| Saving Changes to the Object Parameter Database ..... | 6-13 |

## 6.1. The Object Parameter Database Editor

The Object Parameter Database Editor (OPD Editor) is a graphical user interface for editing the object parameter database. It gives you access to all the components and entity parameters in the object parameter database. Using the OPD Editor can speed up editing of the object parameter database and prevent syntactical errors that can occur when you use a text editor to edit parameters.



If you want to add new entities or customize the entities supplied with VR-Forces, start with the Entity Editor, which is easier to use than the OPD Editor. Use the OPD Editor only if the Entity Editor does not let you make the changes you need to make.



- ♦ When you edit an entity's parameters in the OPD Editor, you are essentially editing the OPE file used by that entity. If more than one entity uses that OPE file, the change will apply to all of the entities.
- ♦ Do not try to edit an entity model in the Entity Editor and the OPD Editor at the same time, except as described in “Editing a System in the OPD Editor,” on page 5-27. Use one tool, save the changes, and close the tool before opening the other.

## 6.2. Starting the OPD Editor

- To start the OPD Editor, choose **Start** → **Programs** → **MÄK Technologies** → **VR-Forces x.x** → **Tools** → **OPD Editor**. The OPD Editor opens (Figure 6-1).

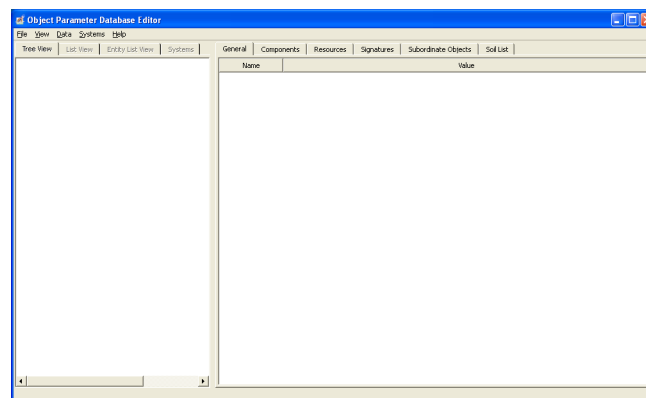


Figure 6-1. OPD Editor

To start the OPD Editor from the Windows or Linux command line:

1. Change directory to `./bin`.
2. Run `vrfOpdEditor`.

### 6.3. Loading an Object Parameter Database

To edit an object parameter database, you must load it into the editor.

To load an object parameter database:

1. Choose **File** → **Open SMS File**. An Open dialog box opens. By default it opens to `./data/simulationModelSets/default`.
2. Select the simulation model set file you want to load, for example, `default.sms`.
3. Click Open. The file is loaded and is listed as a tree hierarchy in the list window (Figure 6-2).

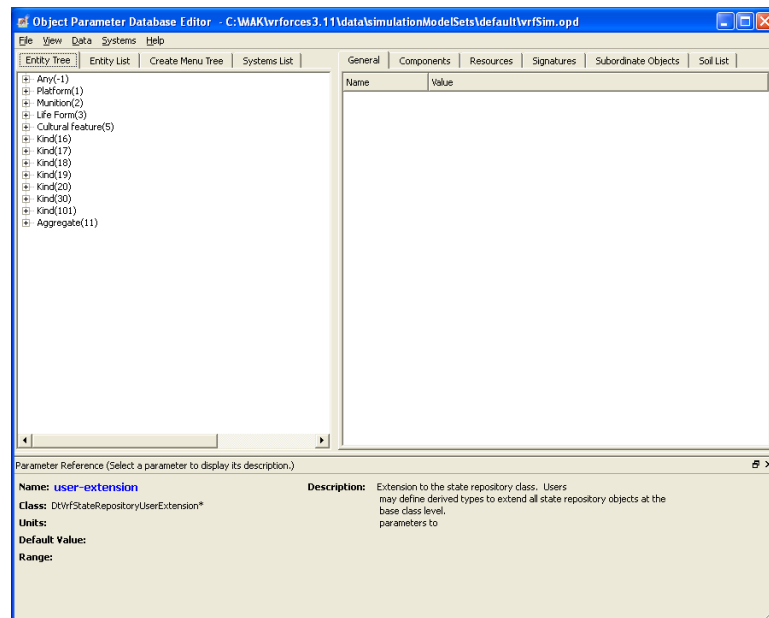


Figure 6-2. OPD Editor with SMS file loaded

### **6.3.1. Changing the Object List View**

The object list has the following views:

- ♦ Entity Tree: The hierarchical tree view allows you to configure any enumeration within the DIS/RPR FOM entity enumeration scheme
- ♦ Entity List: The list view lists all objects in the object parameter database
- ♦ Create Menu Tree: The entity list view lists entities and objects organized by categories as they are on the Create Menu
- ♦ Systems List: The systems view lists the systems configured in systems definition files.

➤ To change the list view, click the tab for the list you want to view.

## **6.4. Editing an Object's Parameters**

This section explains how to edit an object's parameters, how to delete parameters, and how to restore parameters that you have changed or deleted. The OPD Editor online help describes each component and component parameter.

### **6.4.1. Parameter Descriptions**

Parameter definitions are documented by component in the online help for the OPD Editor in the Object Parameter Descriptions folder. Individual parameter definitions are displayed in the Parameter Reference panel of the OPD Editor. The definitions are also in the header files and class documentation.

## 6.4.2. Editing Individual Parameters

The various parameters in the object parameter database are all edited essentially the same way. A parameter's value is a number or string that you type, a value that you select from a list, or a filename that you edit by hand or select from a file chooser. When you edit a parameter, the name of the parameter and all parents is displayed in bold (Figure 6-3).

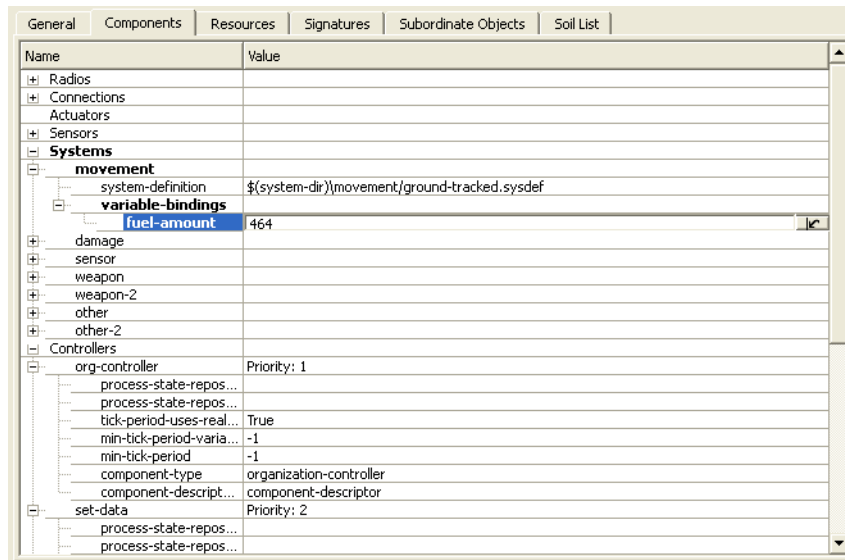


Figure 6-3. Parameter hierarchy

To edit a parameter:

1. Select the parameter by clicking in the Name or Value column. The Value field becomes editable.
2. Enter the new value. In most cases you enter a new value by typing it. If a parameter value has a File Chooser icon (Figure 6-4), click the icon to open a File Chooser dialog box. If the value has an edit icon, click the icon to open a window that allows you to edit the file.

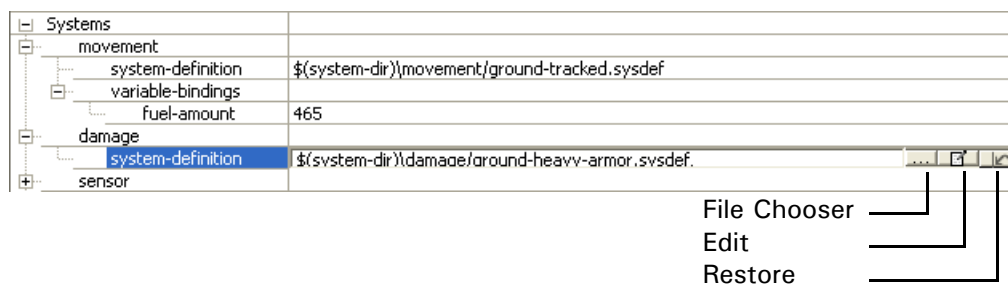


Figure 6-4. Parameter value icons

### 6.4.3. Editing Parameters for Multiple Objects

To edit parameters for multiple objects:

1. Select the objects that you want to edit by clicking them while holding down **Ctrl**. The OPD Editor enters multiple mode. It displays one tab, labeled **Multiple**. The selected objects do not have to be of the same platform type. If a particular node on the hierarchy has multiple leaf entries, selecting it puts the OPD Editor into multiple edit mode. If there is just one leaf entry, selecting a higher level node leaves the editor in individual edit mode.
2. Edit the parameters on the **Multiple** tab as you would edit an individual parameter. The changes are applied to all selected objects.

### 6.4.4. Deleting Parameters and Components

The parameters on the **General** tab cannot be deleted. Parameters on the other tabs can be deleted.

To delete a parameter or component:

1. Select the parameter or component in the Value list.
2. Choose **Data** → **Delete Parameter Entry** or click the Delete icon at the right end of the parameter entry (Figure 6-5).

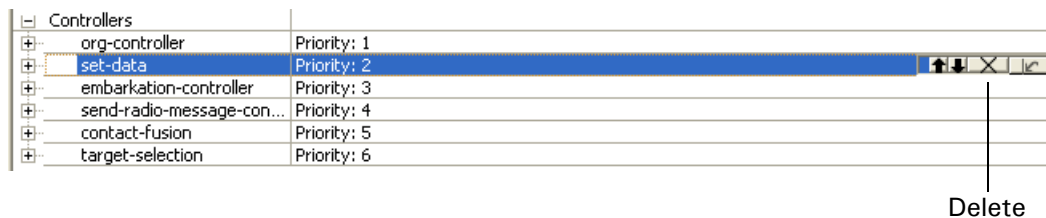


Figure 6-5. Delete parameter icon

### 6.4.5. Restoring Parameters

If you change a value or delete a parameter or component, you can restore the original value. You can restore at the individual parameter level and at any level in a given hierarchy of entries (Figure 6-3). For example, you can restore a change to the **at-distance** parameter for the **move-to** controller, all edits made to the **move-to** controller, or all edits made to all Controllers. However, since Actuators are not part of the Controllers hierarchy, restoring a change at the Actuators level would not affect changes to Controllers.

To restore a parameter or component:

1. Select the parameter to restore. If you are restoring a deleted parameter or component, select the parent component.
2. Click **Data** → **Restore Parameter Entry**, or click the Restore icon (Figure 6-4).

## 6.5. Creating a New Object from an Existing Object

You can add new entity types to the object parameter database using an existing object as a template.

To add a new object to the object parameter database:

1. Expand the Entity Tree to the leaf level and select the object on which you want to base the new object. (The other views do not support this procedure.)
2. Choose **Data** → **Create From Existing**. The Create Entry dialog box opens (Figure 6-6). It lists the object enumeration for the base object in the Base Entry box.

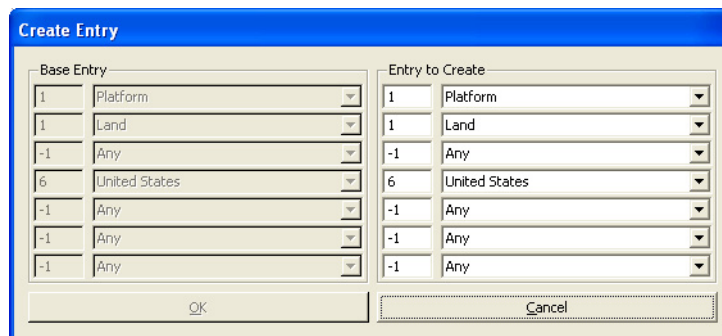


Figure 6-6. Create Entry dialog box

3. In the Entry to Create box, change the enumeration to the one for the object you want to create. You can type the enumeration directly, or build it by selecting options from the lists for each value.
4. Click OK. The new object is added to the Entity Tree in the appropriate location.
5. Edit the parameters for the new object to give it the proper behaviors.
6. Save the object parameter database.

## 6.6. Adding New Components and Parameters

You cannot add new parameters to the General tab. You can add new components, resources, subordinates, and soil types if an object supports them.



For most entities, the Components tab does not list all of the components that apply to the entity. This is because many components are referenced through the systems that are attached to an entity. If you do not see a component that you are looking for in an entity's component list, look at the component list for the relevant system in the Systems List.

### 6.6.1. Adding New Components



Before you add a component to an entity, consider whether or not the component should be added as part of a new or existing system, rather than being added directly to the entity.

To add a component to an object:

1. Select the Components tab.
2. Select the component type (Actuator, Controller, Sensor) you want to add.
3. Choose **Data** → **Create Parameter**, or click the Add icon (Figure 6-7). The Create New Component dialog box opens. The Component list lists all components that are available for this object.

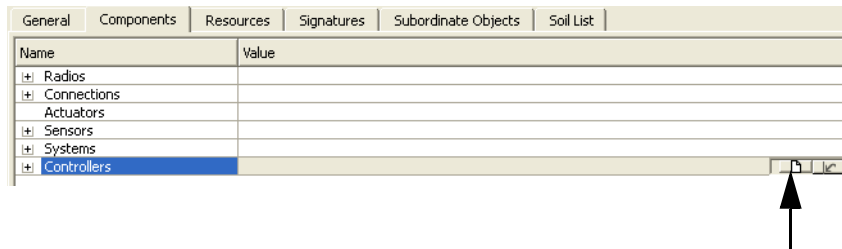


Figure 6-7. Add icon

4. From the Component list, select the component you want to add.
5. Click OK. The component is added to the object.

### 6.6.2. Editing Components

The Components tab lets you edit actuators, controllers, sensors, and connections. You can edit the individual parameters for components, as described in the other sections of this chapter. You can add and delete components and you can change their priorities.



If you do not see a component that you are looking for in an entity's component list, look at the component list for the relevant system in the Systems List.



## Changing A Component's Priority

When you change a component's priority, the other components are reprioritized as needed. All affected components are listed in bold.

To change a component's priority:

1. Select the component you want to edit.
2. Click the up or down icon on the value line (Figure 6-8).

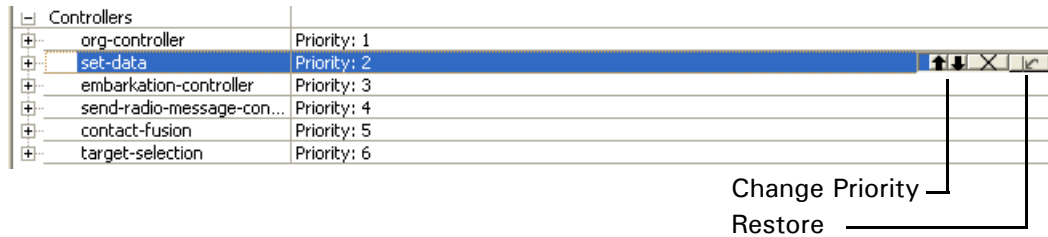


Figure 6-8. Change Priority and Delete Parameter icons

## Creating A Connection

When you add a component, you must also add connections between the component and other components it interacts with.

To add a connection:

1. On the Components tab, select Connections.
2. Choose **Data** → **Create Parameter**, or click the Add icon (Figure 6-7). The Create Connection dialog box opens (Figure 6-9).

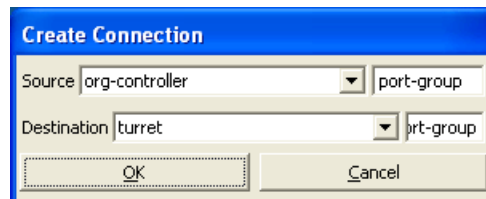


Figure 6-9. Create Connection dialog box

3. Choose a source component from the Source list.
4. Enter the port group for the source component. You must know the name of the port group you need to use. For a list of components and port groups, see the online help.
5. Choose the destination component from the Destination list.
6. Enter the port group for the destination component. You must know the name of the port group you need to use.

7. Click OK. The connection is added to the Connections list.

### 6.6.3. Adding New Resource Parameters

To add a resource parameter to an object:

1. Select the Resources tab.
2. Select Resources.
3. Choose **Data** → **Create Parameter**, or click the Add icon (Figure 6-7). The Create Resource dialog box opens.

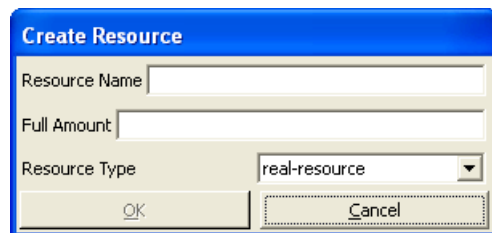


Figure 6-10. Create Resource dialog box

4. In the Resource Name box, enter the name of the resource you want to add. Resource names must start with a letter, not a number. You are responsible for knowing that this is a valid resource supported by your implementation of VR-Forces.
5. In the Full Amount box, enter the amount of the resource to assign to this object.
6. Select a resource type from the Resource Type list.
7. Click OK. The resource is added to the object.

### 6.6.4. Adding New Subordinate Object Parameters

You are responsible for knowing whether or not a subordinate object is appropriate for a given object.

To add a subordinate object parameter:

1. Select the Subordinate Objects tab.
2. Select Subordinate Objects.
3. Choose **Data** → **Create Parameter**, or click the Add icon (Figure 6-7). A subordinate parameter is added.
4. Expand the Subordinate Objects list and edit the parameters for the new subordinate object.

### 6.6.5. Adding New Soil Type Parameters

To add a soil type parameter to an object:

1. Select the Soil List tab.
2. Expand the Soil List to see which soil types are configured.
3. Choose **Data** → **Create Parameter**, or click the Add icon (Figure 6-7). The Add Soil Type dialog box opens (Figure 6-11).

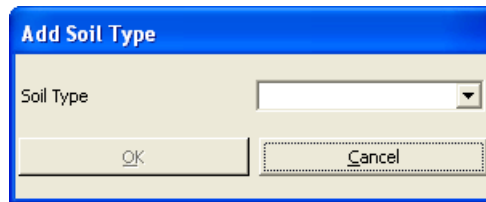


Figure 6-11. Add Soil Type dialog box

4. From the Soil Type list, select the soil type you want to add. The list of soil types is fixed. The Soil Type list contains only the supported soil types that are not already configured for the object.
5. Click OK. The soil type is added to the object.

## 6.7. Editing Variables

The object parameter database uses variables to specify the location of supporting files, such as ammo selection files, system definition files, and formation files. You can edit these variables.

To edit variables:

1. Choose **Data** → **Edit Variables**. The Edit Variables dialog box opens.
2. Change the variable values as desired.
3. Click OK.

## 6.8. Editing Systems

Systems are packages of components that get incorporated into entity models by reference. Internally, a system is just a list of components that could just as easily be part of the main OPE file for an entity. Consequently, within any selected system you edit components and parameters exactly the same way that you edit components that are listed for a specific entity. The only difference between a system and other components definitions is that systems contain metadata that enables them to be used in the Entity Editor. You edit a system's metadata in the OPD Editor just as you would any other parameters.

An object's systems are listed on its Components tab (Figure 6-12). Systems are also listed on the Systems List tab of the OPD Editor window.

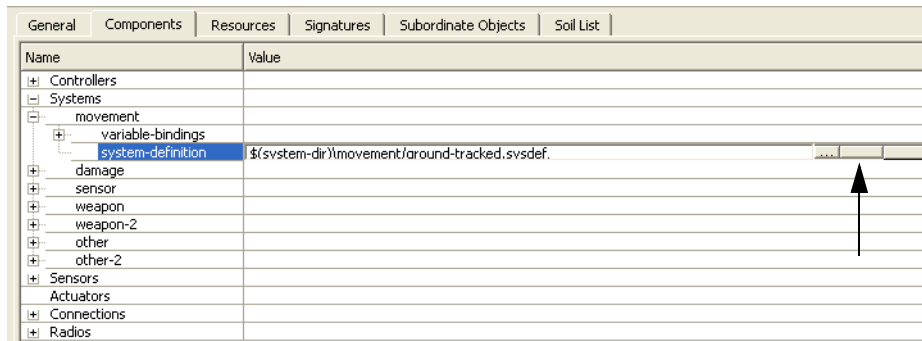


Figure 6-12. Systems list on a Components tab

If you want to edit a system and you have selected it on a Components tab, click the edit button. The OPD Editor changes the view to show the system in the Systems List.

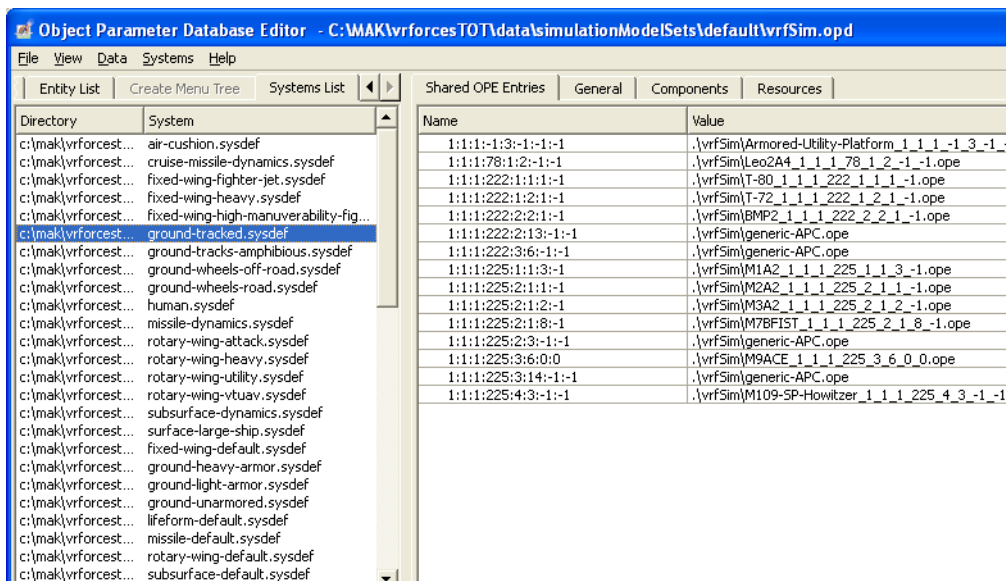


Figure 6-13. System entry in Systems List

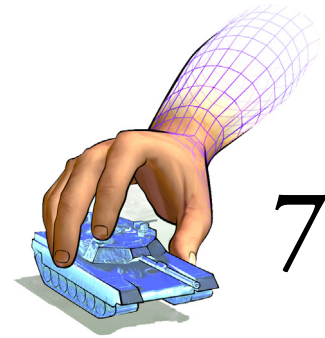
If you click on an entry in the Shared OPE Entries tab for a system, it takes you to the OPE entry for that entity.

You can create new systems in the OPD Editor, either by copying an existing system or starting a brand new system. These procedures are the same as those described in “Creating a New Object from an Existing Object,” on page 6-7 and “Adding New Components and Parameters,” on page 6-7.

## ***6.9. Saving Changes to the Object Parameter Database***

- To save an edited object parameter database, choose **File → Save OPD file**.





## ***The Object Parameter Database***

---

This chapter provides a detailed description of the object parameter database. It will be most useful to VR-Forces users who need to develop and configure new components.

|                                                                 |      |
|-----------------------------------------------------------------|------|
| The Object Parameter Database.....                              | 7-2  |
| VR-Forces Classes and Object Parameters .....                   | 7-6  |
| Object Types .....                                              | 7-6  |
| Parameter Types .....                                           | 7-9  |
| Local and Remote Subcomponents.....                             | 7-10 |
| Component Descriptors and Parameters .....                      | 7-13 |
| Common Elements of Component Descriptors.....                   | 7-14 |
| Configuring the Priority of Components.....                     | 7-16 |
| Adding a New Entity to the Object Parameter Database .....      | 7-17 |
| Systems Definition Files.....                                   | 7-18 |
| Creating a System Definition File.....                          | 7-19 |
| Configuring System Connections .....                            | 7-19 |
| Adding Variable Bindings .....                                  | 7-22 |
| Configuring Aggregate Entities .....                            | 7-23 |
| Adding an Aggregate to the Object Parameters Database .....     | 7-23 |
| Changing the Echelon ID Category and Echelon ID Level .....     | 7-26 |
| Configuring Graphical Objects .....                             | 7-27 |
| Pathname Interpretation in Simulation Model Set Files .....     | 7-27 |
| Updating Object Parameter Database Files for New Releases ..... | 7-28 |

## 7.1. The Object Parameter Database

VR-Forces entities and objects have many parameters. Parameter values are stored in a set of files that VR-Forces reads when you load a scenario or create new objects. We refer to these files collectively as the object parameter database (OPD). The object parameter database specifies both the characteristics of an entity and starting entity state values. You can modify the behavior of an entity by changing the parameters for that type of entity. You can add new types of entities to VR-Forces by adding new files to the object parameter database.



Before you make any changes to the object parameter database, make a backup copy of the file you are changing.

---

The object parameter database has an object parameter entry (OPE) file for each entity type configured for use by VR-Forces. When you load a scenario, the back-end reads the file *vrjSim.opd*. This file indexes the individual OPE files. When you create an object (either directly or by loading a scenario), VR-Forces reads the parameters for the object from its OPE file. OPE files may reference other files, in particular system definition files.

System definition files group related component and connection specifications into convenient packages that you can reference from OPE files. This makes OPE files easier to read. The system definition files also make it possible to add systems to entities in the Entity Editor.



By default, if successive scenarios use the same object parameter database, VR-Forces does not reload the object parameter database before running each scenario. With the default behavior, if you are making changes to the object parameter database, you must close VR-Forces and restart it to put the changes into effect. However, if you set **forceParameterDbReload** to true, VR-Forces reloads the object parameter database each time you load a scenario.

---

Example 7.1 is a highly edited example of an OPE file that includes the main features discussed in this manual. The relevant segments of the example (indicated by arrows) are cross referenced to the sections that discuss them.



### Example 7.1

```

1      ;; Entry for M1A2
2      (M1A2
→ 3      (parameter-type "ground-vehicle-param");; "Parameter Types," on
      page 7-9
4      (bounding-geometry-type Volume)
5      (bounding-volume
6      (local-bvol 3.994358 1.929996 0.896801)
7      (offset 0.000000 0.000000 -0.896801)
8      )
9      (formation "Other")
10     (force ForceOther)
→ 11     (category "M1A2");; "Changing the Echelon ID Category and Echelon
      ID Level," on page 7-26
12     (echelon-level "")
13     (is-organized True)
→ 14     (min-tick-period 0.000000);; 15 Hz ;; "Specifying the Tick Rate
      for Components," on page 1-12
15     (min-tick-period-variance 0.000000)
16     (dr-algorithm Rvw)
17     (radios
18     (main-radio
19     (radio-descriptor-type "radio-descriptor")
20     (radio-type "default-radio")
21     (comm-model-name "default-radio-model")
22     (publish-transmitter False)
23     (transmitter-params
24     (radio-type 7 1 225 2 1 20)
25     (initial-frequency 3000000.000000)
26     (power 30.000000)
27     )
28     )
29     )
30     (systems
31     (movement
32     (variable-bindings
33     (DtRwReal
34     (fuel-amount 465.000000
35     )
36     )
37     )
→ 38     (system-definition ;; "Systems Definition Files," on page 7-18
39     (filename "$(system-dir)\movement\ground-tracked.sysdef")
40     )
41     )
42     (sensor
43     (variable-bindings
44     (DtRwVector
45     (sensor-position -1.140000 0.570000 -2.900000
46     )
47     )
48     (DtRwReal
49     (max-range 4000.000000
50     )
51     )
52     )
53     (system-definition
54     (filename "$(system-dir)\sensors\visual-sensor.sysdef")
55     )

```

```

56         )
57     (weapon
58         (variable-bindings
59             (DtRwInt
60                 (num-ap-rounds 20
61                 )
62             )
63             (DtRwInt
64                 (num-heat-rounds 20
65                 )
66             )
67             ...
68         )
69         (system-definition
70             (filename "$(system-dir)\weapons/120mm-gun.sysdef")
71         )
72     )
73     (other
74         (system-definition
75             (filename "$(system-dir)\other/spot-report-
76 receiver.sysdef")
77         )
78     )
79 (sensors
80     ...
81     (under-fire-determination-sensor ;; "Configuring Sensors," on
82 page 8-3
83         (component-descriptor-type "under-fire-determination-sensor-
84 descriptor")
85         (component-type "under-fire-determination-sensor")
86         (min-tick-period -1.000000)
87         (min-tick-period-variance -1.000000)
88         (tick-period-uses-real-time True)
89         (process-state-repository-name "")
90         (process-state-repository-type "")
91         (under-fire-time 120.000000)
92         (under-fire-distance 1000.000000)
93         (under-fire-from-friendly True)
94     )
95     (controllers ;; "Component Descriptors and Parameters," on
96 page 7-13
97         ...
98         (set-data
99             (component-descriptor-type "component-descriptor")
100             (component-type "local-entity-set-controller")
101             (min-tick-period -1.000000)
102             (min-tick-period-variance -1.000000)
103             (tick-period-uses-real-time True)
104             (process-state-repository-name "")
105             (process-state-repository-type "")
106         )
107         (embarkation-controller
108             (component-descriptor-type "embarkation-descriptor")
109             (component-type "embarkation-controller")
110             (min-tick-period -1.000000)
111             (min-tick-period-variance -1.000000)
112             (tick-period-uses-real-time True)
113             (process-state-repository-name "")

```

```

112     (process-state-repository-type "")
113     (response-timeout 1.000000)
114     (response-retries 2)
115 )
116 ...
117 )
118 (actuators )
→ 119 (connections ;; "Common Elements of Component Descriptors," on
    page 7-14
120     (connect target-selection:weapon weapon:target-list)
121     (connect target-selection:object-types-to-detect sensor:object-
types-to-detect)
122     (connect sensor:detected-objects contact-fusion:detected-
objects)
123     (connect target-selection:weapon-2 weapon-2:target-list)
124     (connect other:detected-objects contact-fusion:detected-objects)
125 )
126 (state-repository-extension-type "default-vrf-state-repository-
user-extension")
127 ...
→ 128 (max-acceleration 1.270000) ;; Appendix B, Object Parameters
129 (max-deceleration 4.000000)
130 (fuel-efficiency 243.800000)
131 (object-geometry-file
132     (filename ".\objectModel\M1A2_1_1_1_255_1_1_3_-1.gdb")
133 )
134 (soil-list
135     (paved-road
136         (acceleration-factor 1.000000)
137         (stopping-factor 1.000000)
138         (max-speed-factor 1.000000)
139     )
140     (hard-packed
141         (acceleration-factor 0.720000)
142         (stopping-factor 0.720000)
143         (max-speed-factor 0.720000)
144     )
145 )
→ 146 (local-objects ;; "Local and Remote Subcomponents," on page 7-10
147     (state-repository "ground-entity-state-repository")
148     (state-repository-min-tick-period -1.000000)
149     (state-repository-min-tick-period-variance -1.000000)
150     (net-interface "individual-local-entity-net-interface")
→ 151     (net-interface-min-tick-period -1.000000) ;; "Tuning the Network
Interface," on page 1-13
152     (net-interface-min-tick-period-variance -1.000000)
153     (task-manager "local-task-manager")
154     (component-manager "component-manager")
155     (plan-manager "vrfoobject-plan-manager")
156 )
157 (remote-objects
158     ... )
159 (subordinate-objects )
160 (attached-objects )
161 (user-extension )
162 )

```

### 7.1.1. VR-Forces Classes and Object Parameters

When you create an object in VR-Forces, it creates an instance of an object defined by a VR-Forces class. A VR-Forces object class has relationships with other classes that are instantiated along with the object. For example, in the VR-Forces API, each *DtVrfObject* “has a” *DtVrfObjectBaseStateRepository*.

The entries for each object in the object parameter database correspond to VR-Forces classes and tell the simulation engine what type of object to create. For example, if you specify the **parameter-type** “ground-vehicle-param” for an entity, VR-Forces knows to use the *DtGroundVehicleParameters* class when it creates the entity.

### 7.1.2. Object Types

VR-Forces uses an eight-digit enumeration scheme for specifying objects. The enumerations are based on the seven-digit enumerations used by DIS and the RPR FOM. The additional field distinguishes between individual and aggregate objects.

This enumeration scheme describes a tree that goes from a more general specification to a more detailed one. [Figure 7-1](#) illustrates a partial entity type tree. To build an entity type enumeration, start at the Kind and work your way down the tree, adding the number for each element. So, for example, if you follow the tree down to M1A2 you build the enumeration: 1 1 225 1 1 3 0. Since M1A2 is a leaf in the tree, the last field is 0. For more information about the entity enumeration scheme, please see the DIS Enumerations Document (*Enumeration and Bit Encoded Values for Use with Protocols for DIS Applications*).

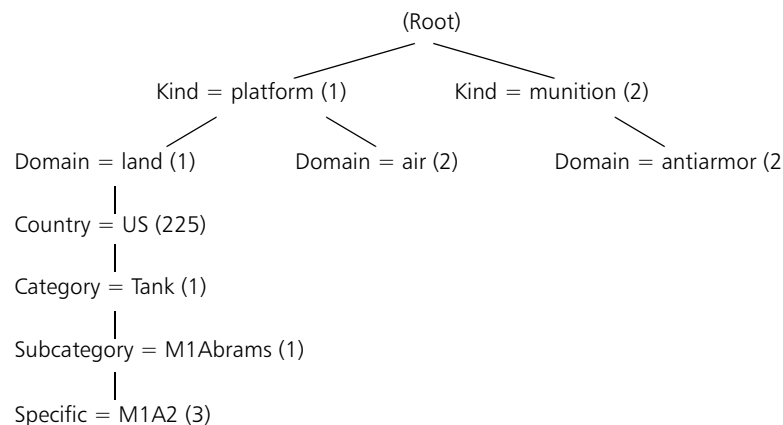


Figure 7-1. Entity type enumeration tree

## The Object Super-Type

VR-Forces adds a field to the standard 7-digit entity enumeration so that we can manage objects that are not part of the DIS/RPR FOM enumeration scheme, such as aggregate entities and control objects. We enclose the seven-digit enumeration in parentheses and preface it with the object super-type ([Figure 7-2](#)). For example, the entity enumeration for an M1A2, described in the previous paragraph, becomes the object type: 1 (1 1 225 1 1 3 0).

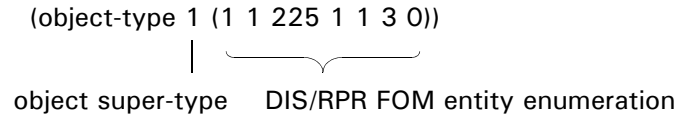


Figure 7-2. VR-Forces object-type parameter

[Table 7-1](#) lists the values VR-Forces can use for the object super-type. For more information, please see *objTypeEnum.h* or “[Object Type Enumerations](#),” on page B-9.

Table 7-1: Object super-type field values

| Object super-type | Meaning                                                    |
|-------------------|------------------------------------------------------------|
| 0                 | Other                                                      |
| 1                 | Individual (platform entities, individual control objects) |
| 2                 | Aggregate                                                  |
| 3                 | Pseudo-aggregate                                           |
| 4                 | True Aggregate                                             |

## The Object Type Hierarchy

Each simulated object on the Create menu is associated with an **object-type** (for details, please see [“Adding a New Entity or Control Object to vrf.ent,”](#) on page A-5.) When you create an entity, the front-end sends this information to the back-end. The back-end searches through the object parameter database and instantiates the entity with the proper set of values for the **object-type**.

The **object-type** sent by the front-end does not need to exactly match an **object-type** in the object parameter database. Objects in the object parameter database are specified in a hierarchy in which entries at the top of the hierarchy have wildcard values (-1s) in object type fields that do not need to be specified at that level of the hierarchy. When VR-Forces looks up an **object-type**, it begins at the root of the tree, working its way down until no better matches are found (the best-match method.)

An object parameter database might have the following entries:

```
(parameter-entry
  (object-type -1 (-1 -1 -1 -1 -1 -1 -1))
  (filename ".\vrfSim\base-sim-object_-1_-1_-1_-1_-1_-1_-1_-1.ope")
)
(parameter-entry
  (object-type 1 (3 1 -1 -1 -1 -1 -1))
  (filename
    ".\vrfSim\generic-lifeform-platform.ope")
)
(parameter-entry
  (object-type 1 (3 1 225 1 1 -1 -1))
  (filename
    ".\vrfSim\friendly-dismounted-infantry.ope")
)
```

As you replace each wildcard value (-1) with a specific enumeration, the scope of the object type narrows. With the illustrated hierarchy, an object with an object type of 1(3 1 225 -1 -1 -1 -1) would have a best match at the generic-lifeform-platform entry. The friendly-dismounted-infantry entry would be too specific for this example enumeration. The order in which parameters are specified in the object parameter database does not affect the hierarchy. The presence of the fixed-wing entity parameter block in this example does not interfere with the hierarchy of the ground-vehicle objects.

### 7.1.3. Parameter Types

In addition to specifying an **object-type** for each object, you must specify a **parameter-type** for each object. The **parameter-type** determines the parameters and components that can be specified for an object. For example, a rotary-wing or fixed-wing entity can have a **nose-support** parameter, but a ground vehicle cannot. [Figure 7-3](#) illustrates the object parameter type hierarchy.

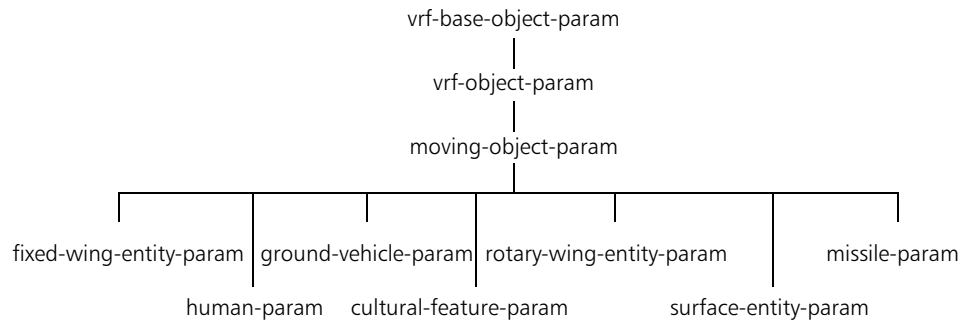


Figure 7-3. Object parameter hierarchy

The **parameter-type** specifies the set of parameters appropriate to an entity, and the entity state repository maintains state information for an entity, so the **parameter-type** and state repository must match. For more information, please see [“The State Repository Hierarchy,”](#) on page 7-12.

### 7.1.4. Local and Remote Subcomponents

As explained in Section 3.4.2, “[Local Objects and Remote Objects](#)” of *VR-Forces Users Guide*, each VR-Forces back-end simulates objects itself (local objects), and maintains state information about remote objects. All of these objects are instantiated as *DtVrfObjects*. The *DtVrfObject* class has a “has a” relationship with the following classes:

- ♦ *DtSimObjectStateRepository*
- ♦ *DtSimObjectNetInterface*
- ♦ *DtSimTaskManager*
- ♦ *DtSimComponentManager*
- ♦ *DtVrfObjectPlanManager*.

These classes are referred to as subcomponents of the object. [Table 7-2](#) maps the subcomponent classes to the parameters that specify them.

Table 7-2: Subcomponent classes and parameters

| Class                      | Parameter            |
|----------------------------|----------------------|
| DtSimObjectStateRepository | state-repository     |
| DtSimObjectNetInterface    | net-interface        |
| DtOrganizationManager      | organization-manager |
| DtSimTaskManager           | task-manager         |
| DtSimComponentManager      | component-manager    |
| DtVrfObjectPlanManager     | plan-manager         |



VR-Forces creates different subcomponents for an object depending on whether it is a local object or a remote object. The subcomponents are specified in the **local-objects** and **remote-objects** parameter blocks for that object. [Table 7-3](#) lists the permissible entries for each parameter listed in [Table 7-2](#).

Table 7-3: Permissible subcomponent parameters

| Parameter         | Permissible values                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| state-repository  | <ul style="list-style-type: none"> <li>♦ vrf-object-base-state-repository</li> <li>♦ vrf-overlay-object-state-repository</li> <li>♦ vrf-object-state-repository</li> <li>♦ vrf-moving-object-state-repository</li> <li>♦ vrf-aggregate-state-repository</li> <li>♦ fixed-wing-entity-state-repository</li> <li>♦ ground-entity-state-repository</li> <li>♦ human-state-repository</li> <li>♦ cultural-feature-state-repository</li> <li>♦ missile-entity-state-repository</li> <li>♦ rotary-wing-entity-state-repository</li> <li>♦ surface-entity-state-repository.</li> </ul> |
| net-interface     | <ul style="list-style-type: none"> <li>♦ aggregate-local-entity-net-interface</li> <li>♦ individual-local-entity-net-interface</li> <li>♦ local-environmental-net-interface</li> <li>♦ pseudo-aggregate-local-entity-net-interface</li> <li>♦ vrf-aggregate-remote-entity-net-interface</li> <li>♦ vrf-individual-remote-entity-net-interface</li> <li>♦ vrf-remote-environmental-net-interface</li> <li>♦ aggregate-remote-entity-net-interface</li> <li>♦ individual-remote-entity-net-interface</li> <li>♦ remote-environmental-net-interface.</li> </ul>                    |
| task-manager      | <ul style="list-style-type: none"> <li>♦ local-task-manage.r</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| component-manager | <ul style="list-style-type: none"> <li>♦ component-manager.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| plan-manager      | <ul style="list-style-type: none"> <li>♦ vrfobject-plan-manager.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

Every object must have a state repository specified. Most objects need a net interface. Other subcomponents are optional. However, for most entities, you will want to specify a task manager, plan manager, and component manager so that you can control the entity and give it tasks.

Use the existing entries in the object parameter database as guidance if you create a new entity or change an existing entry. A typical entry for local and remote objects is:

```
(local-objects
  (state-repository "ground-entity-state-repository")
  (net-interface "individual-local-entity-net-interface")
  (task-manager "local-task-manager")
  (component-manager "component-manager")
  (plan-manager "vrfoobject-plan-manager")
)
(remote-objects
  (state-repository "vrf-moving-object-state-repository")
  (net-interface "vrf-individual-remote-entity-net-interface")
  (task-manager "")
  (plan-manager "")
  (component-manager "")
)
```

## The State Repository Hierarchy

Each VR-Forces object must have a state repository to maintain essential state information about the object, such as its current location, orientation, appearance, and so on. Therefore, in the object parameter database, each object must specify a state repository within its parameter block and the state repository must match the parameter type for the object.

[Table 7-4](#) lists parameter types and the state repository type required.

Table 7-4: State repository and parameter types

| State Repository Type                | Parameter Type            |
|--------------------------------------|---------------------------|
| ground-entity-state-repository       | ground-vehicle-param      |
| fixed-wing-entity-state-repository   | fixed-wing-entity-param   |
| individual-lifeform-state-repository | individual-lifeform-param |
| rotary-wing-entity-state-repository  | rotary-wing-entity-param  |
| human-state-repository               | human-param               |
| cultural-feature-state-repository    | cultural-feature-param    |
| missile-entity-state-repository      | missile-param             |
| surface-entity-state-repository      | surface-entity-param      |
| vrf-moving-object-state-repository   | moving-object-param       |
| vrf-overlay-object-state-repository  | vrf-object-param          |
| vrf-object-state-repository          |                           |

### The Net Interface Hierarchy

VR-Forces manages network relationships of local and remote entities through the network interface classes. Choose the set of local and remote network interface types that most closely represent the type of object being simulated. For example, for an individual vehicle such as a tank, choose individual-local-entity-net-interface for the local network interface and vrf-individual-remote-entity for the remote network interface. Please see Appendix B, *Object Parameters* for tables that list the network interfaces used for the entities provided with VR-Forces.

## 7.2. Component Descriptors and Parameters

The parameters for an entity include a list of all the sensor, controller, and actuator components it uses (please see Section 3.5.4, “[VR-Forces Uses a Component Architecture](#)” in *VR-Forces Users Guide*.) These components might be in the OPE file or in a system definition file.

A component descriptor is a text block in the object parameter database that describes a component (see Example 7.2). A component descriptor must include the type of the descriptor and the type of the component that it is describing. A component descriptor can also contain parameters for the component. If there are no components of a certain type (for example, no sensors), you can omit the list. (In other words, you do not need a placeholder list for unused components.) “[Common Elements of Component Descriptors](#),” on page 7-14 describes the construction of a component descriptor.

Components can receive data from, and convey data to, other components through ports. In an entity’s entry in the object parameter database, you specify which ports a component sends output to and receives data from.

A component can create a port group, which groups a set of related ports together. For example, the automotive-control port group groups the throttle, steering, and brake ports. Use of port groups simplifies specification of connections because you just have to specify a port group instead of many individual ports. For component descriptions and a list of port groups, please see the online help for the OPD Editor. For a detailed discussion of ports and port groups, please see *VR-Forces Developers Guide*.

The port group also provides a mechanism for ensuring that only one data provider attempts to put values on those ports each simulation frame. For more information, please see “[Configuring the Priority of Components](#),” on page 7-16.

### 7.2.1. Common Elements of Component Descriptors

The common elements of component descriptors are:

- ♦ **Name** – The first element in a component descriptor specifies the component's name. This name can be whatever you want, but it must be unique within the entity. It gets used in the connections section of the entity's parameters. Other components may use this name to find the specified component. In Example 7.2, the name of the first component is `move-along` (line 2).
- ♦ **component-descriptor-type** – The component descriptor factory creates the type of component descriptor specified by this type. This lets the system know what types of additional data may be present. In Example 7.2, the component descriptor type for the `move-along` component is `ground-move-to-descriptor` (line 3).
- ♦ **component-type** – The component factory creates the type of component specified in this entry. In Example 7.2, the component type is `ground-auto-move-along-controller` (line 4).



The `component-descriptor-type` and `component-type` values are specified in code. You cannot make them up. Please see *compTypes.h* and *compDescTypes.h*.

---

- ♦ **connections (optional)** – The connections (Example 7.2, line 45) entry specifies how the components connect to one another (through ports, port groups, or both). Components attach to each other by connecting output and input ports to one another, or output port groups to input port groups. Since a component can have more than one port, port group, or both, the name of the port or port group to use is also specified. If an entity does not have any components that need to connect to one another, then you can omit this entry. Items in the connections entry can be in one of the following forms:

- Specify a port-to-port connection:

```
connect component_name:output_port component_name:input_port
```

- Specify a port group-to-port group connection:

```
connect component_name:output_port_group
      component_name:input_port_group
```

The connections in Example 7.2 use this format.

- Specify a port-to-port connection, in which one or both of the ports exists inside of a group, as follows:

The input port belongs to an input port group:

```
connect component_name:output_port
      component_name:input_port_group:input_port
```

The output port belongs to an output port group:

```
connect component_name:output_port_group:output_port
      component_name:input_port
```

Both input and output ports belong to groups:

```
connect component_name:output_port_group:output_port
component_name:input_port_group:input_port
```

## i

The port and port group names are defined in code for each component. Please see class documentation or the online help for the OPD Editor for the names of port and port groups used by a given component.

All of the entity components that you can use in the object parameter database are described in the online help for the OPD Editor.

Example 7.2 shows component descriptors for the Move Along and Patrol Between Points controllers and the Powertrain actuator for a ground vehicle (from the *ground-tracked.sysdef* file.) The Powertrain actuator simulates movement of a ground vehicle based on the throttle, brake, and steering inputs it receives from the automotive controllers. The example also shows how the output port groups for the controllers are connected to the input port group of the powertrain.

### Example 7.2

```
1  (controllers
2    (move-along
3      (component-descriptor-type "ground-move-to-descriptor")
4      (component-type "ground-auto-move-along-controller")
5      (min-tick-period -1.000000)
6      (min-tick-period-variance -1.000000)
7      (tick-period-uses-real-time True)
8      (process-state-repository-name "")
9      (process-state-repository-type "")
10     (near-distance 25.000000)
11     (at-distance 1.000000)
12     (approach-speed 2.000000)
13     (stay-in-lane False)
14     (road-center-offset 2.000000)
15   )
16   (patrol-between-points
17     (component-descriptor-type "ground-move-to-descriptor")
18     (component-type "ground-auto-patrol-between-controller")
19     (min-tick-period -1.000000)
20     (min-tick-period-variance -1.000000)
21     (tick-period-uses-real-time True)
22     (process-state-repository-name "")
23     (process-state-repository-type "")
24     (near-distance 25.000000)
25     (at-distance 1.000000)
26     (approach-speed 2.000000)
27     (stay-in-lane False)
28     (road-center-offset 2.000000)
29   )
30 ) ;; end controllers
31 (actuators
32   (powertrain
33     (component-descriptor-type "automotive-component-descriptor")
34     (component-type "automotive-actuator")
```

```
35      (min-tick-period -1.000000)
36      (min-tick-period-variance -1.000000)
37      (tick-period-uses-real-time True)
38      (process-state-repository-name "")
39      (process-state-repository-type "")
40      (art-part-list )
41      (check-terrain-collisions False)
42      (use-high-fidelity-ground-clamping True)
43    )
44  ) ;; end actuators
45  (connections
46    (connect move-along:automotive-control powertrain:automotive-
      control)
47    (connect patrol-between-points:automotive-control
      powertrain:automotive-control)
48  ) ;; end connections
```

The connection in line 46 would read: the move-along component sends output to its automotive-control output port group. The powertrain receives the move-along controllers output through its automotive-control input port group.

### 7.2.2. Configuring the Priority of Components

Each simulation frame, VR-Forces ticks each entity's components in the order in which they are listed in the object parameter database. If a component believes it should be providing data, it takes control of the receiving component's ports or port group. Once a component acquires the ports, no other component can send data to that component during that frame. Therefore the order in which components are listed provides a natural priority scheme.

For example, among the ground movement controllers, the collision avoidance controller is placed on the controller list before any of the other movement controllers. If this controller needs to provide input to the ground automotive actuator to avoid a collision, it takes control of the automotive-control port group on the next tick and overrides the controller that previously owned the port group (such as Move To or Move Along).

For more information about prioritizing components, please see Section 5.4.6, [“Setting the Priority of Components”](#) in *VR-Forces Developers Guide*.

## 7.3. Adding a New Entity to the Object Parameter Database

This section explains how to manually add new entries to the object parameter database. The easiest way is to copy the OPE file for a similar entity and change parameters to give it the characteristics you want for the new entity.



The recommended way to add new entities to the object parameters database is to use the Entity Editor ([“Creating a New Entity,”](#) on page 5-28) or the OPD Editor (see [“Creating a New Object from an Existing Object,”](#) on page 6-7.)



Before making changes to a configuration file, be sure to make a backup copy of the original file.

The object parameter database has an entry for an M1A2 tank, but not an M1A1. If you want to add an M1A1 you could:

1. Copy the *M1A2\_1\_1\_1\_225\_1\_1\_3\_-1.ope* file and rename it *m1a1.ope*.
2. Open *m1a1.ope*.
3. Change the vehicle name in the very first line of the block.
4. Change the 7-digit entity enumeration portion of the **object-type** to the entity-type for an M1A1. You can look this up in the DIS enumerations document.
5. Change the **category** string to M1A1. (For more information, please see [“Changing the Echelon ID Category and Echelon ID Level,”](#) on page 7-26.)
6. Change other parameters as desired and add or delete controllers and actuators to give the new vehicle the characteristics of a M1A1.
7. If you add controllers or actuators, add any necessary connections.
8. Save the *m1a1.ope* file.
9. In the OPD file, add a reference to the new OPE file.

The following code snippet shows the start and end of the parameter block for the M1A2 vehicle. The minimum changes that you have to make to turn it into an M1A1 (name, object-type, and category) are all in the first few lines of the parameter block.

```
(M1A1
  (parameter-type "ground-vehicle-param")
  (bounding-geometry-type Volume)
  (bounding-volume
    (local-bvol 3.994358 1.929996 0.896801)
    (offset 0.000000 0.000000 -0.896801)
  )
  (formation "Other")
  (force ForceOther)
```

```
(category "M1A1")
(echelon-level "")
...
```

The entry in the OPD file would be:

```
(parameter-entry
  (object-type 1 (1 1 225 1 1 2 -1))
  (filename ".\vrfSim\M1A1_1_1_1_225_1_1_2_-1.ope")
)
```

## **7.4. Systems Definition Files**

The Entity Editor allows VR-Forces users to easily add component systems, such as weapons, sensors, and movement, to entities without needing to understand the components, configuration files, and connections required to make these systems work. Editing of systems in the Entity Editor is facilitated by systems definition files (*.sysdef*) in the object parameter database. System definition files are referenced in an entity's OPE file. So, as a byproduct of the use of system definition files, OPE files can be shorter and easier to read. System definition files are read by the OPD Editor and can be edited in it.

Systems definition files have two major sections – data and metadata. The data is a list of the components and connections required to implement the system for an entity. The data could just as easily be written into the OPE file and have the same effect.

The metadata explains to the Entity Editor how to present the system to the user and how to integrate it into the entity model.

Given this information about systems definition files, if you are customizing the object parameter database by adding newly developed components or by using the components provided with VR-Forces, you have the following options:

- ♦ Add the new component configuration directly into OPE files. In this case, the new components will not be accessible from the Entity Editor, but could be edited in the OPD Editor.
- ♦ Package the new component information in a system definition file without metadata and reference the system definition file from the OPE file. Again, the system will not be accessible from the Entity Editor, but can be edited in the OPD Editor.
- ♦ Package the new component information in a system definition file and include metadata. The system will be accessible from the Entity Editor.



### 7.4.1. Creating a System Definition File

If you want to create a new system definition file, MÄK recommends that you copy an existing file and use it as a template for adding the components that you want to configure into a system. You can use the OPD Editor to do this or a text editor.

The format of a system definition file is as follows:

```
(system-name
  (controllers
  )
  (actuators
  )
  (connections
  )
  (resources
  )
  (meta-data
    (system-name "Spot Report Generator")
    (system-description "Allows the entity to send spot reports though
its radio when its sensors detect other entities in the scenario.")
    (allowed-state-repository-types "all")
    (system-categories "other")
    (meta-data-entry-list
    )
    (parameter-data-list
    )
  )
)
```

As you can see, the format is similar to that of the OPE file and other configuration files that use the MTL syntax. The system-name is an arbitrary name that uniquely identifies the system. The component sections list the various controllers, actuators, sensors, and resources that are needed to build the system. The connections list the connection among the components.

### 7.4.2. Configuring System Connections

Within each system definition file, you list the connections among the components in the file. (For general information about connections, please see [“Common Elements of Component Descriptors,”](#) on page 7-14.) For systems that are self-contained, for example movement systems, there is no need to connect to components in other systems. However, some systems need to connect to components outside their system. For example, weapons systems must have access to a target list. This section explains how to make these out-of-system connections. (The sample code is from the *M1A2\_1\_1\_1\_225\_1\_1\_3\_1.ope* file and the *120mm-gun.sysdef* file.)

To understand how system definition files and OPE files are integrated, we will look at:

1. A weapon definition in the OPE file.
2. The related system definition file.
3. The connection section in the OPE file.

The M1A2 is configured with a 120mm gun, using a system definition file.

```
1  (weapon
2    (variable-bindings
3      (DtRwInt
4        (num-ap-rounds 20
5          )
6        )
7      (DtRwInt
8        (gun-art-part-type 4416
9          )
10       )
11     ...
12   (system-definition
13     (filename "$(system-dir)\weapons/120mm-gun.sysdef")
14   )
15 )
```

This block is named `weapon`. It contains several variable bindings. The values attributed to the variables are used in the system definition file. The file itself is specified in the `system-definition` block. For more information, please see [“Adding Variable Bindings,”](#) on page 7-22.

The system definition file has the following relevant code:

```
1  (weapon-120mm-gun
2    (controllers
3      ...
4    )
5    (actuators
6      (main-gun-joy
7        (component-descriptor-type "ballistic-gun-descriptor")
8        ...
9        (process-state-repository-type "")
10       (art-part-list
11         (primary-gun
12           (art-part-type $gun-art-part-type)
13           (parent-part-type $turret-art-part-type)
14           (attach-point 2.331000 -0.125000 0.418000)
15           (art-part-param-list
16             (part-type 13)
17             (part-type 14)
18           )
19         )
20       )
21     ...
22   )
23   (connections
24     (connect system:target-list turret-acquire:target-list)
25     (connect main-gun-control-joy:gun-control main-gun-joy:gun-control)
26     ...
27   )
28   (meta-data
29     ...
30     (parameter-data-list
31       (art-part-parameter-data
32         (parameter-name "gun-art-part-type")
33         (variable-type "DtRwInt")
34       (art-part-category "gun")
```

```
35          (display-label "Gun Part Number")
36      )
37  )
38  )
```

The system component in the first connection (line 24) is a self reference. It links to an equivalent connection in the main OPE file.

The M1A2 OPE entry for weapon creates a variable called gun-art-part-type (line 8). The variable is used in line 12 of the system definition file. Other variables specified in the OPE file are used throughout the system definition file.

The system definition file defines connections for its components. The first connection in the list (line 24) maps system to target-list. If you look at the complete system definition file, you will see that there is no component called system. Nor is there one called target-list.

The use of system in a connection in a system definition file creates a port into the system so that components outside the system can connect to it. Line 24 indicates that a target-list component needs to connect to the weapon system through the system port.

The connections in the M1A2 OPE file are as follows:

```
(connections
  (connect target-selection:weapon weapon:target-list)
  (connect target-selection:object-types-to-detect
    sensor:object-types-to-detect)
  ...
)
```

In the first line in the list of connections, we see that the target-selection component sends output to its weapon output port. (The weapon output port is defined in the target-selection component's block, which is not shown in this section.) The weapon component gets input through its target-list input port. The weapon component is really a reference to the 120mm-gun system. The first connection in the 120mm-gun.sysdef file is:

```
(connect system:target-list turret-acquire:target-list)
```

In this line, "system" is a self-reference that is equivalent to the component name in the OPE file. Therefore, this line could be read as the weapon component sends output (which it received through its target-list input port in the OPE file connection) to its target-list output port. Then the turret-acquire component in the system definition file, takes input through its target-list port. (Figure 7-4) illustrates these connections.

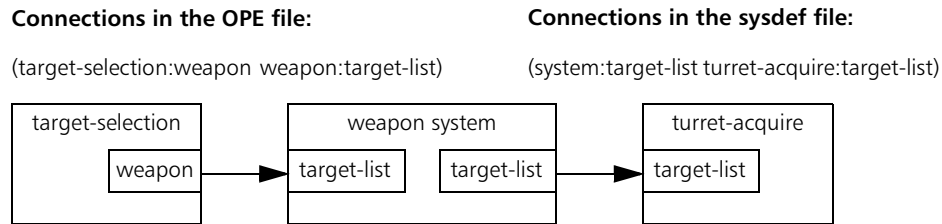


Figure 7-4. Flow of data in component connection

### 7.4.3. Adding Variable Bindings

Most system definition files are used by multiple entities. However, the values of some parameters in a system definition file (SDF) may need to vary for the different entities that use it. As previously noted, this is handled by using variables for the values in the system definition and specifying the values for variables in the OPE files. VR-Forces uses variable bindings for the parameters that you are most likely to want to change from entity to entity. If there are other parameters that you want to vary, you can add your own variable bindings.

The following example shows a variable binding for a movement system:

```

(movement
  (variable-bindings
    (DtRwReal
      (fuel-amount 462.000000)
    )
  )
  (system-definition
    (filename "$(system-dir)\movement/ground-tracked.sysdef")
  )
)

```

The variable is used in the SDF as follows:

```

(resources
  (fuel
    (resource-type "real-resource")
    (subresources )
    (amount $fuel-amount)
    (full-amount $fuel-amount)
  )
)

```

As you can see, the variable defined as fuel-amount in the OPE file is referenced as \$fuel-amount in the SDF file.

To add a variable binding to an SDF:

1. In the system definition file, substitute a variable name, in the format *\$variable-name* for the hard-coded value that you want to replace.
2. In each OPE file that uses this SDF, add a variable binding to the system block that references the SDF you edited. The variable binding must specify the type of the value, the name of the variable, and the value for the variable as shown in the example earlier in this section.



If you do not add a variable binding to an OPE that references the modified SDF, a default value will be used, which may result in unexpected behavior.

---

## 7.5. Configuring Aggregate Entities

All aggregate entities have the **parameter-type** “moving-object-param”. Aggregate entities have relatively few individual parameters. The principle function of the various parameter blocks is to specify controllers and formations, and to specify the **echelon-level** for the different levels of aggregates.

### 7.5.1. Adding an Aggregate to the Object Parameters Database

This section explains how to add a preconfigured aggregate to the object parameter database by directly editing the object parameter database. You can also add new entries with the OPD Editor (see Chapter 6, *Advanced Entity Model Editing*.) The procedure for adding a preconfigured aggregate to the object parameter database is similar to that of adding an entity, except that you must ensure that any subordinate units required by your new aggregate exist in the object parameter database. For example, VR-Forces provides preconfigured squads, platoons, and companies. If you wanted to add a preconfigured brigade, which had several battalions, you would have to add both brigades and battalions to the object parameter database, because the default object parameter database does not have any battalions.

The preconfigured aggregates in the object parameter database are based on the aggregate entries used when you aggregate entities using the **Aggregate As** command. Although they are not all available through the **Aggregate As** command, the object parameter database has entries for aggregates up through Corps.

The format for a preconfigured aggregate is:

```
(aggregate-local-entity
  (parameter-type "moving-object-param")
  ...
  (category "")
  (echelon-level "echelon_level")
  (subordinate-objects
    (sub1
      (object-type 3 (11 1 225 14 3 0 0))
      (position-offset 0.00000 0.000000 0.00000)
```

```
        (heading-offset 0)
      )
      ...
      (subn)
    )
  )
```

where:

- ♦ **parameter-type** must be moving-object-param.
- ♦ **category** should be left blank.
- ♦ **echelon-level** describes the hierarchical unit for this aggregate.
- ♦ The subordinate-objects block lists the entities that you want to be created as subordinates of this aggregate entity. In each subordinate block:
  - **object-type** specifies the object type of an entity defined elsewhere in the object parameter database
  - **position-offset** specifies the location of the entity in aggregate body coordinates
  - **heading-offset** specifies the heading of the entity relative to the aggregate.

### Editing a Preconfigured Aggregate

If you want to change the configuration of a preconfigured aggregate or add a variation of a provided type to the object parameter database, edit one of the current object parameter database entries as follows:

1. Copy the aggregate you want to edit to a new OPE file.
2. Change the subordinate list as desired.
3. Change the entity type enumeration to distinguish the entity from other aggregates of its type. The best way to do this is to change the last number in the enumeration.
4. Add a new entry to the *entity.mst* file so that the aggregate will be editable from the Entity Editor and be added to the Entity Palette. If you are using a customized Create menu (as opposed to having it generated automatically), add the new aggregate to *vrf.ent*.

## Adding a New Type of Preconfigured Aggregate

This section explains how to add a new type of preconfigured aggregate, such as a battalion, to the object parameter database.

To add a new preconfigured aggregate to the object parameter database:

1. Copy the existing object parameter database entry for the desired hierarchical unit, in this case, the battalion, which looks like:

```
(aggregate-local-entity
  (parameter-type "moving-object-param")
  ...
  (echelon-level "Bn")
)
```

2. Save the copied file to a new OPE file.

3. Add a category parameter.

```
(category "")
```

4. Add a subordinate-objects block.

5. Add entries for each subordinate entity, for example, four companies. Using the existing preconfigured company as a subordinate, an entry would look like:

```
(sub1
  (object-type 3 (11 1 225 5 -1 -1 1))
  (position-offset 0.00000 0.000000 0.00000)
  (heading-offset 0)
)
```

The resulting entry might look like:

```
(aggregate-local-entity
  (parameter-type "moving-object-param")
  (object-type 3 (11 -1 -1 6 -1 -1 -1))
  (category "")
  (echelon-level "Bn")
  (subordinate-objects
    (sub1
      (object-type 3 (11 1 225 5 -1 -1 1))
      (position-offset 0.00000 0.000000 0.00000)
      (heading-offset 0)
    )
    (sub2
      (object-type 3 (11 1 225 5 -1 -1 1))
      (position-offset 1000.00000 0.000000 0.00000)
      (heading-offset 0)
    )
    ...
  )
)
```

6. Add a new entry to the *vrSim.opd* file with an appropriate object type value.

7. Add a new entry to the *entity.mst* file so that the aggregate will be editable from the Entity Editor and be added to the Entity Palette. If you are using a customized Create menu (as opposed to having it generated automatically), add the new aggregate to *vrf.ent*.

## 7.6. Changing the Echelon ID Category and Echelon ID Level

You can change the category and echelon ID level used in echelon IDs by editing the object parameter database. Changes to the object parameter database take effect the next time you start VR-Forces.



Before making changes to a configuration file, be sure to make a backup copy of the original file.

---

To change the category or echelon name used for a platform, such as a tank:

1. Open the OPE file for the entity in a text editor.
2. Edit the value for the **category** or **echelon-level** parameter.
3. Save the file.



You can also add new entries with the OPD Editor. For details, please see Chapter 6, *Advanced Entity Model Editing*.

---



- ♦ Entity names have the following restrictions on their length:
    - Entities – 11 characters
    - Aggregates – 10 characters in HLA exercises, 32 characters in DIS.
  - ♦ Since the category is used to build the default entity name, make sure that it is short enough, in combination with the incremental entity number (for example, M1A2 23), that it will not exceed the character limits for entity names. If the default entity name is too long, it gets truncated and will not work properly.
  - ♦ If you change the **category** or **echelon-level** parameters (or both) for any entry, existing scenarios that reference this type of object will not work unless you edit the order of battle file, object map file, and plan file so that the object names use the new **category** and **echelon-level** strings.
-



## 7.7. Configuring Graphical Objects

The object parameter database includes parameters for graphical objects. Graphical objects provided with VR-Forces have the **parameter-type** `vrf-base-object-param`. You should not change the parameters for the objects provided with VR-Forces. However, if you are using the VR-Forces toolkit, you could create other types of graphical objects. You would have to include parameters for them in the object parameter database.

## 7.8. Pathname Interpretation in Simulation Model Set Files

The files that make up a simulation model set, such as the SMS file, the OPD file, and OPE files usually contain references to other files in the SMS. These supporting files may also have references to other configuration files. VR-Forces supports the file relationships by specifying files using relative paths and by defining directory variables. This allows you can maintain custom simulation model sets outside of the VR-Forces file hierarchy and not have to move them every time you install a new version of VR-Forces.

Examples of directory variables are in the SMS file and OPD file. The following lines from the *default.sms* file show a binding (`model-set-directory`) and its use:

```
(simulation-model-set
  (variable-bindings
    (DtRwString
      (model-set-directory "default")
    )
  )
  (opd-file "${model-set-directory}\vrfSim.opd")
  (physical-world "${model-set-directory}\physicalWorldParams.mtl")
  ...
)
```

Similarly, the *vrfSim.opd* file has bindings, which are then used in OPE files, as shown.

From *vrfSim.opd*:

```
(variable-bindings
  (DtRwString
    (system-dir "../data/simulationModelSets/default/vrfSim/systems")
  )
  (DtRwString
    (ammoselect-dir
      "../data/simulationModelSets/default/vrfSim/ammoselect")
    )
  (DtRwString
    (damage-dir "../data/simulationModelSets/default/vrfSim/damage")
  )
  ...
)
```

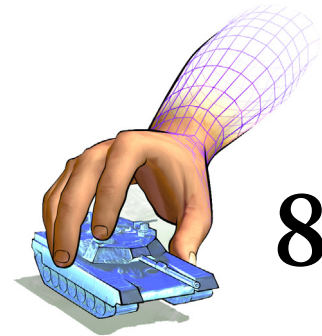
From *BMP2\_1\_1\_1\_222\_2\_2\_1\_1.ope* (system-dir variable):

```
(systems
  (tracked-movement
    (variable-bindings
      (DtRwReal
        (fuel-amount 462.000000)
      )
    )
  )
  (system-definition
    (filename "$(system-dir)\tracked-movement.sysdef")
  )
)
...
)
```

## **7.9. Updating Object Parameter Database Files for New Releases**

New releases of VR-Forces frequently include new components and new parameters for existing components. If you are using a customized object parameter database you will probably want to update it to take advantage of the new components and parameters. Unfortunately, there is no automatic way to migrate an object parameter database. We recommend using one or a combination of the following methods:

- ♦ Start with the object parameter database for the latest release and add your customizations to it.
- ♦ Use a text comparison tool (diff tool) to merge your customized files with the release version of the object parameter database.
- ♦ To merge in new parameters for existing components, in the OPD Editor, select the specific entities that you have customized, then save the object parameter database. The new parameters will be added with default values.
- ♦ The only way to add new components to an object parameter database or replace outmoded components (for example, to replace the target detection sensor with the new sensor mechanism in an older object parameter database) is to add them by hand in individual files or with the OPD Editor.
- ♦ Use multiple simulation model sets. VR-Forces can load multiple simulation model sets (for details, please see Section 5.1.1, “[Specifying Multiple Simulation Model Sets](#)”, in *VR-Forces Users Guide*.) If your customizations do not overlap the default SMS or are relatively easy to separate from the default SMS, you can create a separate SMS and load it along with the SMS supplied with VR-Forces.



# *Configuring Object Behaviors*

---

This chapter describes configuration files that affect entity behavior and the effects of munitions.

|                                                            |      |
|------------------------------------------------------------|------|
| Configuring Sensors.....                                   | 8-3  |
| Configuring a Sensor in an OPE File.....                   | 8-3  |
| Editing a Sensor System Definition File.....               | 8-4  |
| Specifying a Sensor Signature .....                        | 8-5  |
| Adding a New Sensor Type.....                              | 8-5  |
| Detection Probability Tables .....                         | 8-8  |
| Probability and Selection Tables for Munitions .....       | 8-9  |
| Ammo Select Tables .....                                   | 8-9  |
| Hit Probability Tables.....                                | 8-10 |
| Damage Probability Tables .....                            | 8-12 |
| Configuring an Entity to Take Damage from a Munition ..... | 8-13 |
| Configuring Formations.....                                | 8-14 |
| Formation Files .....                                      | 8-14 |
| Creating a User-Defined Formation .....                    | 8-15 |
| Configuring Emitters .....                                 | 8-16 |
| Configuring Radios.....                                    | 8-18 |
| Configuring Transmitters .....                             | 8-18 |
| Configuring Task Execution Rules .....                     | 8-19 |
| Configuring Joysticks .....                                | 8-20 |
| Configuring Obstruction Avoidance .....                    | 8-21 |
| Configuring Object Types .....                             | 8-21 |
| Configuring Vector Feature Types .....                     | 8-22 |
| Configuring Embarkation .....                              | 8-23 |

## **Configuring Object Behaviors**

---

|                                                          |      |
|----------------------------------------------------------|------|
| Editing Embarkation Parameters .....                     | 8-23 |
| Configuring VR-Forces Components on Remote Entities..... | 8-25 |

## 8.1. Configuring Sensors

VR-Forces includes several sensor systems and you can add them to entities using the Entity Editor or the OPD Editor. This section provides additional details about how sensors are configured in the OPE file and system definition file. A complete sensor configuration requires the following:

- ♦ One or more system definition entries in the sensor block in the sensing entity's OPE file
- ♦ An entry in the connections section of the entity's OPE file
- ♦ A sensor signature in the OPE file for each entity that can be sensed by a sensor type
- ♦ An signature propagator entry in the *physicalWorldParam.mtl* file (one per sensor type.)

VR-Forces includes radar, infrared, visual, and sonar sensors. The entities configured in the object parameter database are configured with appropriate sensors and sensor signatures. For details about sensor components and how they work, please see Section 5.2.6, “Sensors”, in *VR-Forces Developers Guide*.

### 8.1.1. Configuring a Sensor in an OPE File

All sensors provided with VR-Forces are signature-object-sensors. They are distinguished by the domain in which they operate (visual, radar, infrared, or sonar.)

To add a sensor to an entity:

1. Copy an existing sensor block in the OPE file and paste it into the sensors section.
2. Change the name of the system definition file to point to the system you want to add.
3. Change the values for the variables that must be specified for the system.

#### Connecting a Sensor

If you add a sensor, you must connect it to input through its input port and then send the data to its output port for the appropriate controller. The following is from the connections block for an M1A2. The sensor gets input through its object-types-to-detect input port. It sends that data out through its detected-objects output port:

```
(connect target-selection:object-types-to-detect sensor:object-types-to-detect)
(connect sensor:detected-objects contact-fusion:detected-objects)
```

If you were to add, for example, an infrared sensor to an entity and you called the sensor block infrared-sensor, you would add the following line:

```
(connect infrared-sensor:detected-objects target-selection:detected-objects)
```

### 8.1.2. Editing a Sensor System Definition File

The preferred method for editing a sensor system definition file, as with all system definition files, is to edit it in the OPD Editor. You can also edit the file by hand. The following example is from the visual sensor system definition file:

```
(visual-sensor
  (component-descriptor-type "signature-sensor-descriptor")
    (component-type "signature-sensor")
    (min-tick-period 2.000000)
    (min-tick-period-variance 0.100000)
    (tick-period-uses-real-time False)
    (process-state-repository-name "")
    (process-state-repository-type "")
    (detect-only-hostile-forces False)
    (detection-types )
    (detect-destroyed-objects False)
    (sensor-geometry
      (in-range
        (range $max-range)
      )
    )
    (sensor-domain "visual")
    (sensor-offset $sensor-position)
    (sensor-positional-error 0.000000)
    (detection-level-determinator
      (determinator-type "signature-detection-level-determinator")
      (detection-level-to-set-hostility 3)
      (combat-identification-level-table-file
        "$(detection-dir)\std-visual-detection-table.csv")
    )
  )
)
(controllers
)
(actuators
)
(connections
  (connect system:object-types-to-detect
    visual-sensor:object-types-to-detect)
  (connect visual-sensor:detected-objects system:detected-objects)
)
```

As you can see from viewing the file, most of the parameters that you would want to vary from entity to entity are variables that are set in the variable binding section of the OPE file.

### 8.1.3. Specifying a Sensor Signature

A sensor signature indicates that an entity can be detected by other entities that have a sensor for that signature. The value of the signature using the standard VR-Forces propagator represents the maximum distance, in kilometers, at which the entity can be detected. However, this value is affected by modifiers and the calculations made by the propagator.

The modifiers specify multipliers for that base value based on the current state of the object. For example, an entity might have a base value of 5.0 for its visual signature, and a modifier of 1.5 when moving. This means the signature of the entity will be 7.5 when moving and 5.0 when not moving.

The following is the sensor signature block from an M1A2:

```
(sensor-signatures
  (visual
    (base-signature 4.000000)
    (modifiers )
  )
  (radar
    (base-signature 27.000000)
    (modifiers )
  )
  (infrared
    (base-signature 4.000000)
    (modifiers )
  )
)
```

This sensor-signature block shows that an M1A2 can be detected by visual, infrared, and radar sensors. If you do not want an entity to be detectable by a particular type of sensor, remove it from the OPE file. If you want it to be detectable by an additional sensor type, add it to the OPE file. You can edit the values of sensor signatures in the Entity Editor. You cannot add sensor signatures in the Entity Editor. You can edit the values of sensor-signatures and add new sensor signatures in the OPD Editor.

### 8.1.4. Adding a New Sensor Type

You can easily add new sensor types to your entities. Suppose that you want to add an acoustic sensor. You would do the following:

1. Add a block to the signature-propagator-params block in *physicalWorldParams.mtl*.
2. Add a sensor block to each entity that can detect targets acoustically and connect the sensor to the detected-objects port.
3. Add a sensor signature to each entity that can be detected acoustically.

## Configuring a Signature Propagator

The *physicalWorldParams.mtl* file specifies the list of sensors available and the propagator to use for each. The following is a portion of this file:

```
(physical-world-params
  (signature-propagator-params
    (radar
      (propagator-descriptor-type "standard-propagator-descriptor")
      (propagator-type "standard-propagator")
      (check-line-of-sight True)
    )
    (visual
      (propagator-descriptor-type "standard-propagator-descriptor")
      (propagator-type "standard-propagator")
      (check-line-of-sight True)
    )
  )
)
```

If you edit the file to add an acoustic sensor, it might look like the following:

```
(physical-world-params
  (signature-propagator-params
    (visual
      (propagator-descriptor-type "standard-propagator-descriptor")
      (propagator-type "standard-propagator")
      (check-line-of-sight True)
    )
    (acoustic
      (propagator-descriptor-type "standard-propagator-descriptor")
      (propagator-type "standard-propagator")
      (check-line-of-sight False)
    )
    ...
  )
)
```

You don't have to be able to see an entity to hear it, so you do not need to check line of sight.



## Adding a Sensor Component

To add a new sensor component, follow the procedure in [“Configuring a Sensor in an OPE File,”](#) on page 8-3. A new acoustic sensor might look like the following:

```
(acoustic-sensor
  (component-descriptor-type "signature-sensor-descriptor")
  (component-type "signature-sensor")
  (min-tick-period 2.000000)
  (min-tick-period-variance 0.100000)
  (tick-period-uses-real-time False)
  (process-state-repository-name "")
  (process-state-repository-type "")
  (detect-only-hostile-forces False)
  (detection-types )
  (detect-destroyed-objects False)
  (sensor-geometry
    (in-range
      (range $max-range)
    )
  )
  (sensor-domain "acoustic")
  (sensor-offset $sensor-position)
  (sensor-positional-error 0.000000)
  (detection-level-determinator
    (determinator-type "signature-detection-level-determinator")
    (detection-level-to-set-hostility 3)
    (combat-identification-level-table-file
      "$(detection-dir)\std-acoustic-detection-table.csv")
  )
)
)
(controllers
)
(actuators
)
(connections
  (connect system:object-types-to-detect
    acoustic-sensor:object-types-to-detect)
  (connect acoustic-sensor:detected-objects system:detected-objects)
)
```

The connection for the acoustic sensor would look like:

```
(connect target-selection:object-types-to-detect sensor:object-types-to-detect)
(connect sensor:detected-objects contact-fusion:detected-objects)
```

### Adding a Sensor Signature

If you add a new sensor, you need to add a signature for it to each entity that can be detected by the sensor. If you add an acoustic sensor, a sensor signature might look like the following:

```
(sensor-signatures
  (visual
    (base-signature 4.000000)
    (modifiers )
  )
  (acoustic
    (base-signature 2.000000)
    (modifiers )
  )
  ...
)
```

You can probably see entities at a greater distance than you can hear them, so we have a lower base-signature value.

## 8.2. Detection Probability Tables

The detection probability tables determine the likelihood that an entity will be detected by another entity.

Detection probability tables are comma-delimited files that are most easily read in a spreadsheet. As illustrated in [Figure 8-1](#), each column represents a detection interval, in seconds, from 0 through 160. The rows represent the apparent signature from one through 4. The detection probability is determined by matching the apparent signature to the detection interval used. The detection probability is used in the Detect condition in plans. The apparent signature maps to the certainty levels as follows:

- ♦ 1 – Detected
- ♦ 2 – Classified
- ♦ 3 – Identified
- ♦ 4 – Full Knowledge.

|    | A                           | B                                | C | D | E | F  | G  | H  | I  | J   | K |
|----|-----------------------------|----------------------------------|---|---|---|----|----|----|----|-----|---|
| 1  | CID Table-radar Version 1.0 | Detection interval in seconds >> |   |   |   |    |    |    |    |     |   |
| 2  |                             |                                  | 0 | 1 | 5 | 10 | 20 | 40 | 80 | 160 |   |
| 3  | Apparent signature          | 1                                | 1 | 1 | 2 | 2  | 2  | 3  | 3  | 3   |   |
| 4  |                             | 1.5                              | 1 | 1 | 2 | 2  | 3  | 3  | 3  | 3   |   |
| 5  |                             | 2                                | 1 | 1 | 2 | 2  | 3  | 3  | 3  | 4   |   |
| 6  |                             | 2.5                              | 1 | 2 | 2 | 3  | 3  | 3  | 4  | 4   |   |
| 7  |                             | 3                                | 1 | 2 | 3 | 3  | 3  | 4  | 4  | 4   |   |
| 8  |                             | 4                                | 2 | 3 | 3 | 3  | 4  | 4  | 4  | 4   |   |
| 9  |                             |                                  |   |   |   |    |    |    |    |     |   |
| 10 |                             |                                  |   |   |   |    |    |    |    |     |   |

Figure 8-1. Detection table displayed in a spreadsheet

For details about CID detection levels, please see Section 13.3, “[Detecting Targets](#)”, in *VR-Forces Users Guide*.

## 8.3. Probability and Selection Tables for Munitions

When the entities in a scenario shoot at each other, VR-Forces uses an ammo select table to select the ammunition to be used. It uses a set of tables to calculate the probability of:

- ♦ Hitting a target (hit probability table)
- ♦ Being damaged when hit by hostile munitions (damage probability table).

### 8.3.1. Ammo Select Tables

The ballistic gun controller descriptor refers to an ammo select table file, which contains the ammo select table for that controller. Each entry in this file specifies a target type (using the best-match method) and a prioritized list of ammunition that should be used for that type of target (highest priority listed first). The name of the entry is not used, and so may be something descriptive for readability. Each entry has the following fields:

- ♦ **target-type** – a 7-digit entity type enumeration, which can use -1s to denote wild-cards.
- ♦ **ammo** – list of ammunition entries. The names of the ammunition entries are assumed to be the resource names. If the resource name is found in the Object Resource Manager, the amount of the ammunition is decremented appropriately, and is not used if the resource amount is 0. In this case, the next ammunition entry (if any) in the ammunition list is used. Each ammunition entry has the following fields:
  - **munition-type** – 7 digit entity type enumeration. This is the entity type that is used in the fire and detonation interactions created, and should NOT contain -1s.
  - **warhead** – VR-Link warhead enumeration value to use in fire and detonation interactions for this munition.

The following is an example of an Ammo Select Table:

```
(ammo-select-table
  (default ;; matches any ground vehicle
    (target-type 1 1 -1 -1 -1 -1 -1)
    (ammo
      (M830-HEAT-120mm ;; these names need to match resource names
        (munition-type 2 2 225 2 13 3 0)
        (warhead 0)
      )
      (M829A1-AP-120mm ;; these names need to match resource names
        (munition-type 2 2 225 2 13 2 0)
        (warhead 0)
      )
    )
  )
  (battle-tank ;; matches any tank
    (target-type 1 1 -1 1 -1 -1 -1)
    (ammo
      (M829A1-AP-120mm ;; these names need to match resource names
        (munition-type 2 2 225 2 13 2 0)
      )
    )
  )
)
```

```

        (warhead 0)
      )
    (M830-HEAT-120mm ;; these names need to match resource names
      (munition-type 2 2 225 2 13 3 0)
      (warhead 0)
    )
  )
)

```

### 8.3.2. Hit Probability Tables

A weapon system definition file specifies a hit probability file. This file contains a hit probability table that is used by the system’s actuator to determine whether or not it hits a target that it fires upon. A hit probability table has one or more entity-range entries. The name of the entity-range entry is not used, and can be something descriptive. An entity range entry has the form:

- ♦ **entity-type** – 7 digit entity type enumeration, which may use -1s as wildcards.
- ♦ **range-determinant** – a range determinant can be either a list of probability-range entries with the name “range-list”, or a probability-coefficient list with the name “coefficients”.
- ♦ **range-list** – a list of probability-range entries. A probability-range entry has a name, “range”, a range value (in meters) and a list of probability entries. The entry applies to targets in the range from the last entry to the current one (or from 0.0 in the case of the first entry). Each probability entry has a name that the table user uses to find the probability value it is interested in and the value itself (a real number between 0.0 and 1.0.) The ballistic gun looks for the probability entry with the name “probability”.

#### i

- ♦ For ranges that are greater than the last specified range, the probability is zero.
- ♦ The range for missiles is the range from the missile to the target, not from the entity that fired the missile.

- ♦ **coefficients** – a coefficient probability list is a list of coefficient probability entries. Each entry has a probability name and the coefficients of a standard polynomial that can be used with a given range to determine the specified probability. The order of the polynomial is determined by the number of coefficients provided. For example, an entry of:

```
(probability -5.00000E-008 -300000E-012 1.0000)
```

indicates that the formula to use for a given range  $r$  would be:

$$(- 5.00000E-008 * r^2) + (-300000E-012 * r) + 1.0$$

Values calculated will be constrained to be between 0.0 and 1.0. Zero values are factored out, for example:

```
(probability -5.00000E-008 0.0000 1.0000)
```

indicates that the formula to use for a given range  $r$  would be:

$$(- 5.00000E-008 * r^2) + (0.0 * r) + 1.0$$

The following is an example of the *M1A2MainGun.hit* hit probability table:

```
(hit-probability-table
  (entity-range
    (entity-type 1 1 -1 -1 -1 -1 -1)
    (range-determinant
      (range-list
        (range 1000.000000
          (probability 0.90000)
        )
        (range 2000.000000
          (probability 0.8000)
        )
        (range 3000.000000
          (probability 0.75000)
        )
        (range 4000.000000
          (probability 0.70000)
        )
      )
    )
  )
)
```

### 8.3.3. Damage Probability Tables

The damage actuator uses damage probability tables specified by its component descriptor to determine direct and indirect fire damage based on several factors. The damage probability table has a list of surface entries, which provide a sub-table for each surface upon which a round might detonate. The name of the surface entry is the surface name used in the damage model.

VR-Forces has damage probability tables for direct fire and collateral damage. A munition type does not need to have both types of probability table if one of the types is not appropriate. For example, an M16 does not have a collateral damage file. Conversely, an artillery shell does not have a direct fire probability table.

The damage actuator delivered with VR-Forces uses surfaces of front, rear, left-side, right-side, bottom, and top (other components could model the entity's geometry differently, and therefore use different surface strings). A surface entry has a list of angle-of-incidence entries. An **angle-of-incidence** entry has an angle (in radians), and a **range-determinant**. The entry is intended for use with angles of incidence that go from the previous entry's specified angle to the current one (or in the case of the first entry, from 0.) The range determinant is exactly the same as the range determinant described in [“Hit Probability Tables,”](#) on page 8-10. The damage actuator provided by MÄK looks for a probability entry with the name **catastrophic-kill**.

---

**i**

The range for missiles is the range from the missile to the target, not from the entity that fired the missile.

---

The following damage probability table fragment has two surface entries, front and rear. The first uses a coefficients range determinant, the second surface entry uses a **range-list** determinant.

```
(damage-table
  (front
    (angle-of-incidence
      (angle 0.5236) ;; 30 degrees
      (range-determinant
        (coefficients
          (catastrophic-kill -5.0000E-008 0.0000 1.0000)
        )
      )
    )
    (angle-of-incidence
      (angle 1.0472) ;; 60 degrees
      (range-determinant
        (coefficients
          (catastrophic-kill -5.0000E-008 0.0000 1.0000)
        )
      )
    )
  )
  (rear
    (angle-of-incidence
```

The structure of a collateral damage probability file is similar to the direct damage file. However the file is processed differently. VR-Forces calculates a range around the munition impact within which other entities can be damaged.

## 8.5. Configuring Formations



---

Although you can create and edit formation files by hand as described in this section, the recommended way to create and edit formations is to use the Entity Editor.

---

The object parameter database lists the formations available to an aggregate entity in the **aggregate-formation-controller** block in its OPE file. Each formation is identified by its name and the relative path to the formation file, for example:

```
(aggregate-formation-controller
 (component-descriptor-type "formation-controller-descriptor")
 (component-type "aggregate-formation-controller")
 ...
 (formation-list
  (column-formation
   (formation Column)
   (formation-file
    (filename ".\formation\platoonColumn.frm")
   )
  )
  (line-formation
   (formation Line)
   (formation-file
    (filename ".\formation\platoonLine.frm")
   )
  )
 )
 ...
)
```

### 8.5.1. Formation Files

A formation file specifies the name of the formation and has an entry for each entity in the default formation definition. Each entry specifies a **promotion-id**, a **leader-promotion-id**, and a **position-offset**, where:

- ♦ The **promotion-id** specifies the order in which entities are promoted to aggregate leader if the leader is incapacitated.
- ♦ The **leader-promotion-id** specifies the aggregate member from whom the entry's position will be offset.
- ♦ The **position-offset** specifies how many meters behind, to the right, and above the entry will follow its leader. The **position-offset** is specified in terms of the body coordinate system of its leader.

The following is a portion of the formation file for a column formation:

```
(column-formation
 (entry-0
  (promotion-id 0)
  (leader-promotion-id -1)
  (position-offset 0.000000 0.000000 0.000000)
 )
 (entry-1
```



```
(promotion-id 1)
(leader-promotion-id 0)
(position-offset -25.000000 0.000000 0.000000)
)
(entry-2
  (promotion-id 2)
  (leader-promotion-id 1)
  (position-offset -25.000000 0.000000 0.000000)
)
(entry-3
  (promotion-id 3)
  (leader-promotion-id 2)
  (position-offset -25.000000 0.000000 0.000000)
)
```

In a column, each entity follows the entity in front of it by 25 meters. Therefore, the **leader-promotion-id** for each entry is the **promotion-id** of the entity in front of it, and each entry has a **position-offset** of -25 meters.

The number of entries in a formation file is finite. However, it is possible to place more entities into an aggregate than there are entries in a formation file. If this happens, VR-Forces increments the **promotion-id** of the last entry in the formation file and assigns the new entity the same **position-offset** as the last entry in the formation file. For a simple formation like a column, this works well. If you are using a more complex formation, this mechanism might create unintended relationships among entities.

If you want to change the layout of the formations provided with VR-Forces, you can edit the formation files.

### 8.5.2. Creating a User-Defined Formation

You can create formations other than those provided by VR-Forces. It is recommended that you create formations in the Entity Editor. If you use the Entity Editor, you do not have to edit any formation files or OPE files by hand.

To create a formation:

1. Using one of the formation files supplied with VR-Forces as your template, place as many entries as you want in the formation file and specify their offsets.
2. Save the new formation under a new name.
3. Add an entry to the formation list block in the OPE file for each aggregate that you want to be able to use the formation. Be sure to use the same structure and naming conventions as the default entries. For example:

```
(my-formation-1
  (formation myFormation)
  (formation-file
    (filename ".\formation\myformation.frm")
  )
)
```

We recommend that you configure your formations so that entities follow the leader, rather than the next closest member of the formation (as is done in column formation.) This ensures that formations have the best chance of maintaining their formation as they maneuver.

## 8.6. Configuring Emitters

All fixed-wing aircraft have emitters as part of their radar sensor. You can configure entities with emitters that have more than one beam. You can also configure multiple emitters. The basic radar sensor with an emitter is as follows:

```
(active-radar-sensor-system
  (systems )
  (sensors
    (radar-sensor
      (component-descriptor-type "radar-signature-sensor-descriptor")
      (component-type "radar-signature-sensor")
      . . .
      (emitter-system
        (system-id 0)
        (system-name-enum 5570)
        (system-function-enum 5)
        (beam-list
          (beam-0
            (index 0)
            (type "track")
            (frequency 4999999488.000000)
            (frequency-range 5000.000000)
            (power 70.000000)
            (pulse-repetition-frequency 0.000000)
            (pulse-width 0.000000)
            (azimuth-center 0.000000)
            (azimuth-sweep 3.142857)
            (elevation-center 0.000000)
            (elevation-sweep 1.000000)
            (sweep-sync 0.000000)
            (beam-function-enum 5)
            (beam-param-index 0)
          )
        )
      )
    )
  )
)
```

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

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

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

```
(radar-sensor
  (component-descriptor-type "radar-signature-sensor-descriptor")
  (component-type "radar-signature-sensor")
  . . .
  (emitter-system
    (system-id 0)
    (system-name-enum 5570)
    (system-function-enum 5)
    (beam-list
      (beam-0
        ...
      )
      (beam-1
        ...
      )
    )
  )
)
```

## 8.7. Configuring Radios

All radios are *DtVrfRadio* classes or subclasses. The radio is responsible for managing incoming and outgoing radio messages on a single entity. Each radio is associated with a specific communication model, and uses this communication model to send and receive messages.

Radios are configured in an entity's OPE file. The radios block in the OPE file lists all the radios on the entity and the parameters for each radio. Parameters include the type of radio descriptor and type of radio class, which are strings used to create the C++ classes from the factories. The **comm-model-name** parameter identifies which communication model a radio is associated with. Every radio must be associated with a communication model. A radio can also specify transmitter parameters (please see the next section for details.)

A typical OPE radio entry is as follows:

```
(main-radio
  (radio-descriptor-type "radio-descriptor")
  (radio-type "default-radio")
  (comm-model-name "default-radio-model")
  (publish-transmitter False)
  (transmitter-params
    (radio-type 7 1 225 2 1 20)
    (initial-frequency 3000000.000000)
    (power 30.000000)
  )
)
```

### 8.7.1. Configuring Transmitters

You can configure entities to publish a transmitter object corresponding to a radio. A transmitter object is a DIS or HLA object that represents an object that is capable of transmitting radio messages. By default, radio transmitters are not published. You can enable publication of transmitters in the OPE file or by enabling the external communication model. For details, please see [“Enabling Radio Transmitter Publishing,”](#) on page 9-3.

To enable publication of transmitters, in an entity's radio entry, set the **publish-transmitter** parameter to True for each radio that is to publish its transmitter state.

You can specify some of the fields in the transmitter state that will be published to the network. These parameters have no effect on the operation of the radio. They only affect which values are published that relate to this transmitter. Some external communication effects servers require that the transmitter state be published for the server to operate, and may use some of the published parameters.

## 8.8. Configuring Task Execution Rules

Each entity OPE file contains an entry for a task execution rule configuration file, named **task-execution-rules**. (The default rules file is *./data/simulationModelSets/default/vrfSim/taskRules/default-task-rules.tsk*.) The task execution rule file specifies whether or not tasks can be executed concurrently, and if so, which tasks are mutually exclusive.

The task execution rules file has the following parameters:

- ♦ **all-tasks-exclusive** enables (False) or disables (True) use of concurrent execution of tasks. When True, the Task Manager executes only one task at a time, and the **mutually-exclusive-task-groups** parameter is ignored. This causes the Task Manager to act as it did in VR-Forces 3.11 and earlier.
- ♦ **mutually-exclusive-task-groups** is a list of groups. Each group contains a list of tasks that cannot be executed at the same time. Any time a task from a particular group is started by the Task Manager, any currently running task that is part of the same group is interrupted. Tasks that are part of other groups, or not part of any group, are allowed to continue running.

The task names that are listed in the **mutually-exclusive-task-groups** parameter are the names that are registered with the task factory when a task is added to the system. Groups can contain any number of tasks, and there can be any number of groups.

Tasks are always considered to be mutually exclusive with themselves. For that reason, it is impossible to execute two tasks of the same type at the same time on a single entity.

## 8.9. Configuring Joysticks

The `./config/joyParam.dat` file specifies the default values for joystick parameters. The parameters are:

- ♦ **joy-device-type** – Possible values are:
  - `direct-input-joydevice` – used for direct-input joysticks on Windows 2000 or XP
  - `pre-direct-input-joydevice` – used on Windows NT, for joysticks that do not use DirectInput.
  - `linux-joydevice` – used for joysticks on Linux.

Default: `direct-input-joydevice`.

- ♦ **joy-trans-sensitivity** and **joy-rot-sensitivity** – affect the translation and rotation displacement required to make VR-Forces respond. The value should be between 0 and 1.0, with 0 being not sensitive at all, and 1.0 being the most sensitive it can be.
- ♦ **joy-trans-gain** and **joy-rot-gain** – affect the velocity of motion for a given displacement. The value is a multiplier, so the default value of 1.0 has no effect on the gain.
- ♦ **joy-trans-invert-x** (y, z) and **joy-rot-invert-x** (y, z) – allow you to invert axes on an individual basis.

---

### i

- ♦ The joystick must be attached to the computer on which the entity is being simulated.
  - ♦ Joystick configuration parameters are documented in the online help for the OPD Editor and in the context sensitive parameter descriptions for joystick controllers in the OPD Editor.
- 

You can override the default values for joystick parameters for specific entity types in the object parameter database. Change the values in the joystick parameter block for the system definition file.

```
(joystick
  (component-descriptor-type "base-joy-controller-descriptor")
  (component-type "ground-move-joystick-controller")
  (joy-trans-sensitivity 0.3)
  (joy-rot-sensitivity 0.3)
  (joy-trans-gain 0.1)
  (joy-rot-gain 0.2)
)
(turret-joystick
  (component-descriptor-type "base-joy-controller-descriptor")
  (component-type "ground-turret-joystick-controller")
  (joy-trans-sensitivity 0.8)
  (joy-rot-sensitivity 0.8)
  (joy-trans-gain 0.1)
  (joy-rot-gain 0.2)
)
)
```

The rotary wing controller has some extra values that you can set, for example:

```
(joystick
  (component-descriptor-type "rotary-wing-joy-controller-descriptor")
  (component-type "rotary-wing-joy-flight-controller")

  ;; controller rise time. Default: 1.5
  ;; rotary-wing-joy-controller-descriptor only
  (controller-rise-time 1.6)

  ;; pitch feedback constant. Default: 1.0
  ;; rotary-wing-joy-controller-descriptor only
  (pitch-control-feedback-constant 1.0)

  ;; roll feedback constant. Default: 2.2
  ;; rotary-wing-joy-controller-descriptor only
  (roll-control-feedback-constant 2.265)

  ;; yaw feedback constant. Default: 2
  ;; rotary-wing-joy-controller-descriptor only
  (yaw-control-feedback-constant 2.1)
)
```

## 8.10. Configuring Obstruction Avoidance

You can configure ground entities to avoid specific entity types, vector features, or both.



The more vector feature types and entity types specified for avoidance, the more memory and processing power required.

Obstruction avoidance is influenced by the **stopping-distance-factor** and **lateral-clearance** parameters. If entities display erratic behavior, try raising the values, lowering the speed of the entities, or both.

### 8.10.1. Configuring Object Types

To specify the types of objects that an entity type should avoid, add or remove the appropriate object type from the obstruction sensor entry in the movement system definition file. For example, the following object types are configured for ground entities:

```
(object-type
  (object-type 1 (18 -1 -1 2 -1 -1 -1)) ;; vrfObstacles object type
  (object-type 1 (1 -1 -1 -1 -1 -1 -1)) ;; DtPlatform entity kind
  (object-type 1 (3 -1 -1 -1 -1 -1 -1)) ;; DtLifeForm entity kind
)
```

## 8.10.2. Configuring Vector Feature Types

Configuring avoidance of vector features relies on feature specifications in GDB databases. To configure a vector feature type for obstruction avoidance, modify the appropriate **vector-type** parameter in the **vector-obstruction-type** parameter block for the obstruction sensor in an entity's object parameter database entry.

The vector obstruction types are used to create *DtSpecifications*, which are then used by the vector network intersection metrics to determine if obstructions are the correct types or not. Any attribute specified for avoidance must be in the vector feature for it to be considered an obstruction. Because the vector feature may contain additional properties, simple vector obstruction types can map to a large number of features.

For example, to specify all types of buildings, add a **vector-type** parameter with the single attribute:

```
(vector-obstruction-type
  (vector-type
    (string-attributes
      (DtFeatureGroup "MAK000")
    )
  )
)
```



For details about the MÄK specification model, please see Chapter A, *The MÄK Specification Model* in *MÄK Terrain Database Tool Users Guide*.

---

If you only want to avoid a specific feature, create a more detailed vector-type. For example, the following will cause only standard lights to be considered obstructions:

```
(vector-obstruction-type
;; Light Standard (street light) detailed spec
;; illustrates how to create a detailed obstruction type
(vector-type
  (integer-attributes
    (DtFidCode 537)
    (DtFeatureType 0)
    (DtSmc 1)
    (DtSedrisCodes 0)
    (DtOrientation 0)
    (DtHeight 3)
  )
  (string-attributes
    (DtFeatureName "537 Light Standard")
    (DtFeatureGroupName "Power")
    (DtFeatureGroup "MAK050")
    (DtDescription "US-Light Standard/Light Support UK-Light
      Standard/Light Support/Lamp Post")
    (ECC "AL110")
  )
  (real-attributes
    (GDB_Width 0.0)
    (GDB_Length 2.0)
  )
))
```



## 8.11. Configuring Embarkation

Entities that are configured to allow other entities to embark on them have an occupancy-director-controller component. Its parameters configure embarkation, as follows:

- ♦ The locations at which entities can embark on an entity are configured as **ingress-points**
- ♦ The locations from which entities can disembark are configured as **egress-points**
- ♦ The number of entities that can embark on an entity is defined by the **embarkation-slots** parameter.

The **embarkation-slots** parameter has one or more **embarkation-slot-entry** parameters that identifies a location on the entity where an embarked entity can be placed. The embarkation-slot-entries also specify the orientation of the embarked entity and the types of entities that can occupy the slot.

The **ingress-points** and **egress-points** parameters each have one or more **load-entry** parameters. The **load-entry** parameters specify one or more **load-entry-point** parameters. If there is one **load-entry-point**, an embarked entity enters the parent entity at that point and is immediately placed in an available slot. If a **load-entry** has multiple load entry points, they specify the points in a route that the embarking entity follows to reach its slot. This configuration is for large parent entities, such as aircraft carriers, on which entities land and then taxi to a slot.

### 8.11.1. Editing Embarkation Parameters

You can edit embarkation parameters in the OPD Editor and the Entity Editor. Ingress and egress points are specified as X, Y, Z values. This section provides a very general introduction to editing embarkation. Please contact MÄK technical support for more information.

The Entity Editor lets you edit single ingress and egress points; it does not let you edit embarkation routes. When you edit ingress and egress points in the Entity Editor, it does not show you the existing points. In fact, it clears out all existing ingress and egress values before it allows you to add new values. Therefore, it is recommended that you use the Entity Editor to edit ingress and egress points only if you know precisely what you want to do.

The OPD Editor lets you edit all aspects of embarkation. To edit embarkation parameters:

1. Select the entity whose embarkation values you want to edit.
2. Select the Components tab.
3. Expand the Controllers list.
4. Expand the occupancy-director-controller component.
5. Edit values as desired.
6. Save the changes.

## Adding a New Ingress or Egress Point

To add a new ingress or egress point:

1. Expand the occupancy-director-controller component as explained in the previous procedure.
2. Select the ingress-points parameter or egress-points parameter, as desired (Figure 8-2).

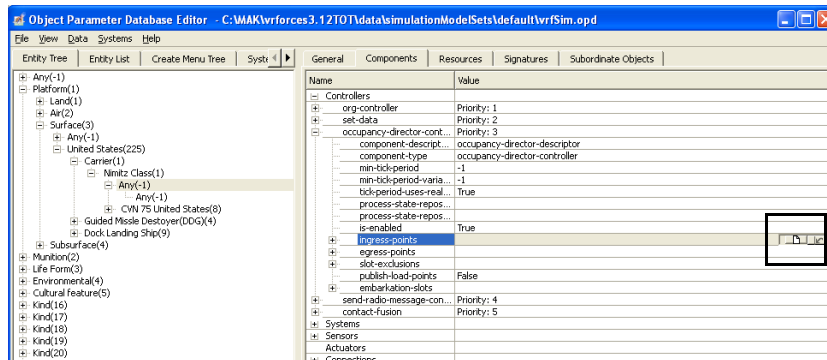


Figure 8-2. occupancy-director-controller component

3. Click the New button. The Set Load Points dialog box opens (Figure 8-3).

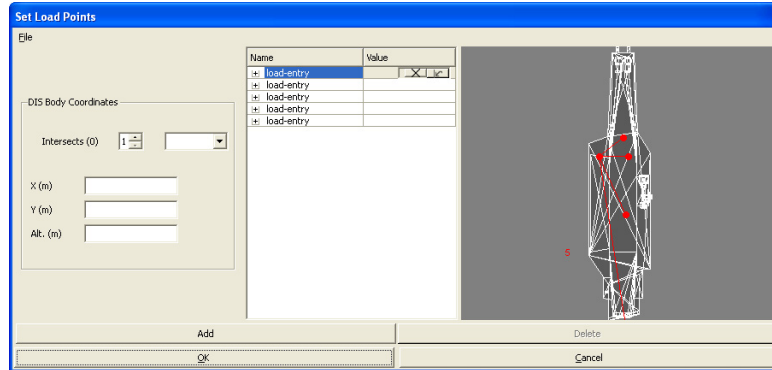


Figure 8-3. Set Load Points dialog box

4. You can edit existing load points and add new ones. Please contact MÄK technical support for assistance.

## 8.12. Configuring VR-Forces Components on Remote Entities

You can attach VR-Forces sensors, actuators, and controllers to remote (non-VR-Forces) entities. This feature allows you to add to the capabilities of the remote entity and better integrate it with the VR-Forces simulation. For example, you could add a radio to an entity or you could enable spot reporting. When you attach a component to a remote entity, you can write a plan for that entity that uses the attached components. For details, please see Section 15.9, “Writing Plans for Remote Entities”, in *VR-Forces Users Guide*.

VR-Forces includes an example scenario, `remoteAttachment`, that uses remote attachment. Its use requires that you run an external application, the `f18` example that comes with VR-Link. For details about how to run the scenario, please see the readme file in `./data/scenarios/remoteAttachment`.

When you add a component to a remote entity, the application that is simulating the entity is responsible for updating its position and health status. VR-Forces simulates the attached components. You can specify that VR-Forces simulate the components on one back-end or across all back-ends.

Simulation of components on remote entities has the following limitations:

- Only individual platforms and lifeforms can attach components. Aggregates and environmental process objects are not supported.
- VR-Forces components cannot control the remote entities. That is, they cannot alter its position or state. Therefore, you should not attach components such as movement systems to remote entities, because they will have no effect on the entities. Similarly, components that expect to be attached to a locally simulated entity such as an articulated part, may not work properly when attached to remote entities.

Components are attached to remote entities on the basis of entries in a component attachment table. The component attachment table provided by VR-Forces is `./data/simulationModelSets/default/extra/componentAttachmentTable.mtl`. The entries are essentially extensions to the configuration data in OPE files.

The component attachment table associates entities with attachment key-names. The following is a sample component attachment table:

```
(remote-configurations
  (attachment-entry
    ;; CIS MIG 29
    (attach-by object-type (obj-type 1 (1 2 222 1 2 1 -1)))
    (key-name "mig29-config")
  )
  (attachment-entry
    ;; US F-18
    (attach-by object-type (obj-type 1 (1 2 225 1 9 -1 -1)))
    (key-name "f18-config")
  )
)
```

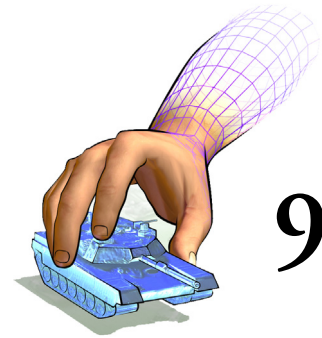
The entries in this table associate the key-names `mig29-config` and `f18-config` with appropriate entity type enumerations. You can also associate a key-name with marking text, as follows:

```
(attachment-entry
  (attach-by marking-text "HMMWV 1)
  (key-name "hmmwv-attachments")
)
```

The key-names in the component attachment table have corresponding entries in *vrfSim.opd* in the `remote-configurations` block, as follows:

```
(remote-configurations
  (remote-configuration
    (key-name "f18-config")
    (configuration-path "$(remote-config-dir)/f18.mtl")
  )
  (remote-configuration
    (key-name "mig29-config")
    (configuration-path "$(remote-config-dir)/mig29.mtl")
  )
)
```

The key-names are associated with specific configuration files, which by default are located in `./data/simulationModelSets/default/vrfSim/remoteConfigurations`. The configuration files contain the components that are to be attached to the remote entity. For examples, please see the files in the *remoteConfigurations* directory.



## *Using VR-Forces with an External Comms Effects Server*

---

This chapter explains how to configure VR-Forces to interoperate with an external communications effects server.

|                                                    |     |
|----------------------------------------------------|-----|
| Introduction .....                                 | 9-2 |
| Configuring VR-Forces.....                         | 9-2 |
| Enabling Radio Transmitter Publishing .....        | 9-3 |
| Enabling the External Communication Model .....    | 9-4 |
| Configuring the External Communications Model..... | 9-5 |
| Using VR-Forces with Qualnet .....                 | 9-5 |
| Running VR-Forces with Qualnet .....               | 9-6 |
| Example Scenario.....                              | 9-6 |
| Running the Scenario .....                         | 9-7 |

## 9.1. Introduction

You can configure VR-Forces to use an external communications effects server to control the transmission of radio messages between VR-Forces entities. The communications effects server determines which entities are able to communicate with each other, and how long it takes a message to be transmitted from the originator of the message to the receiver of the message.

VR-Forces supports the interface specified in the file *./doc/QualNet\_HLA\_interface\_RPR-FOM 1.0\_extensions\_ICD3.pdf*. This Interface Control Document (ICD) was created by Scalable Network Technologies and is distributed with their permission. Any communications effects server that conforms to this interface can be used with VR-Forces.

## 9.2. Configuring VR-Forces

By default, VR-Forces is configured to use a simple internal communications effects model that allows all messages to be passed to their recipients. To use an external communications effects server, you must:

- ♦ Enable the publishing of radio transmitters for any entity types that will be sending messages through the communications effects server
- ♦ Set the external communications model as the communications model to be used by VR-Forces
- ♦ Configure the external communications model.

### 9.2.1. Enabling Radio Transmitter Publishing

The external communications model requires that any entities sending or receiving messages through the communications effects server publish a DIS or HLA radio transmitter object. By default, VR-Forces entities do not publish this object. You can enable publishing of transmitters for individual entities or for all entities. Enabling publishing for all entities is done through the GUI. For details, please see [“Enabling the External Communication Model,”](#) on page 9-4.

To enable publishing of the radio transmitter object for an entity:

1. In the OPE file for each entity type for which you want to enable radio transmitter publishing, locate the **radios** block.
2. Change the **publish-transmitter** parameter to True for each radio, for example:

```
(radios
  (main-radio
    (radio-descriptor-type "radio-descriptor")
    (radio-type "default-radio")
    (comm-model-name "default-radio-model")
    (publish-transmitter True)
    (transmitter-params
      (radio-type 7 1 225 2 1 20)
      (initial-frequency 3000000.000000)
      (power 30.000000)
    )
  )
)
```

To verify that transmitters are being published, create a scenario that has the entity types for which you enabled radio transmitter publishing. A radio icon should be displayed next to the entity symbol for each radio transmitter on the entity.

### 9.2.2. Enabling the External Communication Model

You can enable external communication in the front-end or by editing *commModelParams.mtl*. When you enable the external communication model in the front-end, VR-Forces also enables publishing of transmitters on all entities that have radios. When you change the communication model setting, it gets saved. The next time you run VR-Forces, the saved setting is used.



When you change the communication model setting, existing transmitters are destroyed and new transmitters are created. Therefore, changing this option while running a scenario with QualNet will cause QualNet to no longer recognize the transmitters being published.

To enable the external communication model:

1. Choose **Options** → **Communication Model Settings**. The Communication Model Settings dialog box opens (Figure 9-1).

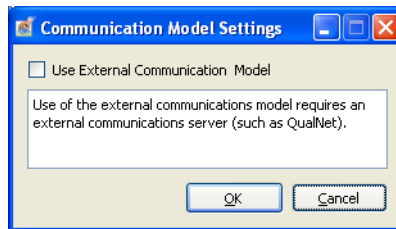


Figure 9-1. Communication Model Settings dialog box

2. Select the Use External Communication Model check box.
3. Click OK.

#### Enabling the External Communication Model in *commModelParams.mtl*

To enable the External Communications Model, edit the file *commModelParams.mtl*, which is in the top directory of the simulation model set that you are using. This file defines the communication models available to the scenarios. By default, VR-Forces is configured to use the simple-radio-comm-model, which allows all communication between entities to be received.

- To use an external server for all radio communication, comment out the **simple-radio-comm-model** block, and uncomment the **external-comm-model** block.

This causes the external server to be used for all radio communication.



### 9.2.3. Configuring the External Communications Model

If you enable the external communications model, you can set the following configuration options in *commModelParams.mtl*:

- ♦ **use-actual-message-size** specifies how the size of the messages will be modeled. When a message is sent using the external communications model, a message size is sent to the external communications effects server, and that number may affect how long it takes for the message to reach its recipient. If this parameter is set to True (the default), the actual number of bytes of the VR-Forces message is used for this size value. This includes the VR-Forces message header of the sender or recipient and the contents of the message. If this parameter is set to False, the size specified by **substitute-message-size** is used for all messages.
- ♦ **substitute-message-size** specifies the message size, in bytes, to send to the external communications effects server. It is used only if **use-actual-message-size** is False.
- ♦ **message-timeout** specifies the timeout value, in seconds, for messages sent to the communications effects server. Messages that are not delivered to the server by the end of the timeout period are dropped.

## 9.3. Using VR-Forces with Qualnet

To use VR-Forces with Qualnet, it must be configured to use the external communications effects server, as described in “[Configuring VR-Forces](#),” on page 9-2. Please note the following requirements:

- ♦ Qualnet only supports particular network configurations. Please refer to the Qualnet documentation for details.
- ♦ If you are using the MÄK RTI, you must run the rtiexec.
- ♦ You must create a Qualnetscenario that matches the VR-Forces scenario to be run using Qualnet as the communications effects server. You can do this by hand, or using the Qualnet HLA Extractor tool. Please see the documentation provided with the HLA Extractor for instructions on its use.



A version of QualNet is available that is specially configured to work well with VR-Forces. Please contact your Qualnet or MÄK salesperson for details about this version of QualNet. Other versions of QualNet are also compatible with VR-Forces, but require additional configuration to work with it. Please see QualNet documentation and or contact Qualnet support to configure it to work with VR-Forces.

---

### 9.3.1. Running VR-Forces with Qualnet

To use VR-Forces with Qualnet, the two programs must be loaded and scenarios started in the following sequence:

1. If you are using the MÄK RTI, start the rtiexec. If you are using a different RTI, run its equivalent server application.
2. Start Qualnet and load the desired Qualnet scenario. (Right click the *.config* file, and choose Run.)
3. Start VR-Forces, and load the VR-Forces scenario.
4. After all entities have been displayed in the VR-Forces GUI, click the Play button on the Qualnet interface.
5. Run the VR-Forces scenario.



- ♦ If you want to load a new VR-Forces scenario, you must stop the QualNet scenario and load a new QualNet scenario using the procedure described in this section.
  - ♦ The VR-Forces rewind functionality does not work correctly with QualNet. To replay a scenario you must restart QualNet and VR-Forces as described in the procedure.
- 

## 9.4. Example Scenario

VR-Forces includes an example VR-Forces scenario and a matching QualNet scenario to demonstrate the interoperability of VR-Forces and QualNet. The QualNet scenario was generated using rprsych.

In the scenario, the M1A2 1 tank commands the UAV to fly over the area where the opposing forces are and send back spot reports. When the first tank receives spot reports, it commands the second one to move to the waypoint, and also moves there itself. All commands and spot reports are sent through the radio, which is passed through QualNet. View the plans for M1A2 1 and UAV 1 to see the details of how they are communicating.

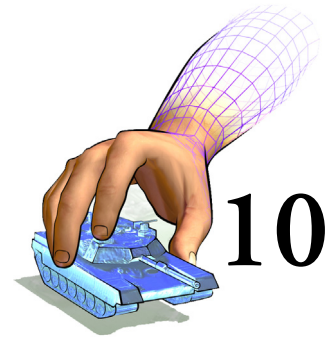
If none of the entities start moving after 10 seconds of running, there is probably an error communicating with QualNet.

### 9.4.1. Running the Scenario

To run the scenario, use the following steps:

1. Start your RTI. If you are using the MÄK RTI, run the rtiexec.
2. Run QualNet.
3. In the QualNet window, select the Scenario Designer tab.
4. Expand the file system tree for the *scenarios* folder.
5. Expand the tree for the *vrforces-CommDemo* scenario.
6. Right-click *vrforces-CommDemo.config*. A popup menu is displayed.
7. Choose **Run Scenario**. QualNet loads the scenario and creates a federation.
8. Start VR-Forces. Be sure that you use a simulation connection that matches the protocol that QualNet is using. If you are using HLA, be sure that you connect to the same rtiexec as you did for QualNet.
9. Choose **Options** → **Communication Model Settings**. The Communication Model Settings dialog box opens ([Figure 9-1](#)).
10. Select the Use External Communication Model check box.
11. Click OK.
12. Open the *NetworkCentricForces/vrforces-CommDemo* scenario. When the scenario is fully loaded, the Run button on the QualNet toolbar should become enabled. If it does not, there may be a communication problem between the two applications.
13. If you want to see communications lines in VR-Forces, choose **Options** → **Transmitters**. In the Transmitter Options dialog box, select the Transmitting check box and any of the other check boxes in the Transmitting group box.
14. Click the Play button on the QualNet toolbar.
15. Run the scenario in VR-Forces.





## *Configuring 3D Models*

---

This chapter explains how to map 3D models to entities and how to create and edit model definitions.

|                                                           |       |
|-----------------------------------------------------------|-------|
| Entity, Object, and Effects Models .....                  | 10-3  |
| Mapping Models to Entities and Objects .....              | 10-4  |
| Adding a Model Mapping .....                              | 10-5  |
| Editing a Model Mapping .....                             | 10-6  |
| Deleting a Model Mapping .....                            | 10-7  |
| Creating and Editing Model Definitions .....              | 10-8  |
| Creating a Model Definition .....                         | 10-9  |
| Deleting a Model Definition .....                         | 10-9  |
| Copying a Model Definition .....                          | 10-10 |
| Filtering the List of Model Definitions .....             | 10-10 |
| Editing a Model Definition .....                          | 10-11 |
| Saving Model Definitions .....                            | 10-13 |
| Configuring Model Instancing .....                        | 10-14 |
| Clearing the Model Instancing Cache .....                 | 10-14 |
| Creating and Editing Model Definition Schemata .....      | 10-15 |
| Creating Model Definition Schemas .....                   | 10-15 |
| Deleting Model Definition Schemas .....                   | 10-16 |
| Copying Model Definition Schemas .....                    | 10-17 |
| Adding a Parameter to a Model Definition Schema .....     | 10-17 |
| Deleting a Parameter from a Model Definition Schema ..... | 10-18 |
| Editing a Model Definition Schema Parameter .....         | 10-18 |
| Configuring SpeedTree Trees .....                         | 10-19 |
| Specifying the SpeedTree Performance Profile .....        | 10-20 |
| Adding a Model to VR-Forces - A Tutorial .....            | 10-20 |

Create a Model Definition ..... 10-21

Add a Model Mapping..... 10-24

Test the Model Mapping ..... 10-26

## **10.1. Introduction to VR-Forces 3D Object Modeling**

In VR-Forces, visualization of 3D objects (entities, props, and effects) is managed by the integration of schemas, model definitions, and model mapping.

A schema is a set of parameters for a scene element, such as an entity. VR-Forces uses schema definitions to validate model definitions. A model definition specifies values for the required and optional parameters for an instance of a schema type. In particular, a model definition specifies the 3D model to be used to represent an object in the scene. Model definitions are associated with particular object types by mapping the model definition to an entity type enumeration. These concepts and the procedures associated with them are described in detail throughout this chapter.

### **10.1.1. Entity, Object, and Effects Models**

VR-Forces includes models for vehicles, lifeforms, environmental objects, and effects. The models are in a compressed, encrypted format (extension *.medf*).

You can provide your own models if they are in one of the following supported formats:

- ♦ OpenFlight
- ♦ 3ds
- ♦ OBJ.

## 10.2. Mapping Models to Entities and Objects

Model mapping is the process of associating an entity or object with a 3D model definition.

DIS and HLA objects are specified by an entity type enumeration. An entity type enumeration is a list of seven components as specified by the DIS protocol and HLA RPR FOM. If none of the seven components contains a wildcard (-1) value, then the entity type refers to a specific, defined entity. If some components contain a wildcard, then the entity type refers to a class of entities. A larger number of wildcards indicates a broader, more general class. For complete details about entity type enumerations, please see *Enumeration and Bit Encoded Values for Use with Protocols for DIS Applications* (DIS enumerations document). It is available at the IEEE World Wide Web site: <http://shop.ieee.org/store/> and at the SISO web site: <http://www.sisostds.org/>.

You map models in the Shared Simulation Settings dialog box. It has pages for mapping:

- ♦ Entity models
- ♦ Firing effects
- ♦ Detonation effects
- ♦ Flaming effects
- ♦ Trailing effects
- ♦ Decal effects
- ♦ Head-up displays.

The mapping process is the same for all model types. You can add new model mappings, edit existing model mappings, and delete mappings.



You can map more than one model definition to an entity type. If you do this, when entities of this type appear in a simulation, VR-Forces randomly assigns one of the model definitions to the entity.

---



### 10.2.1. Adding a Model Mapping

This procedure describes how to map an entity model to an enumeration. The procedure is the same for any page on the Shared Simulation Settings dialog box.

To add a model mapping:

1. Choose **Visual Mappings** → **Entity Models**. The Shared Simulation Settings dialog box opens to the Entity Mapping Settings page (Figure 10-1). The page shows a list of entity type enumerations in the Entity Type column. The model definition mapped to the entity type is listed in the Model Definition column. By default, the list of entity types is filtered to just show entities.

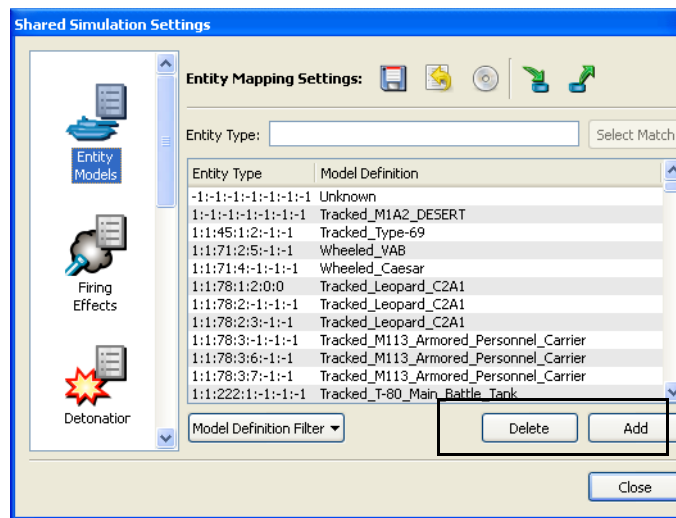


Figure 10-1. Entity Mapping Settings page

- Click the Add button. The Create New Entity Type Model Mapping dialog box opens (Figure 10-2). It lists a default entity type enumeration.

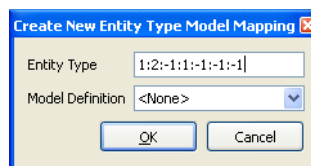


Figure 10-2. Create New Entity Type Model Mapping dialog box

3. In the Entity Type text box, type the enumeration you want to map to a model.
4. In the Model Definition drop-down list, select the model definition that you want to map to this entity type.
5. Click OK.
6. The new model mapping is added to the list.

### 10.2.2. Editing a Model Mapping

This procedure describes how to edit a model mapping for an entity. The procedure is the same for any page on the Shared Simulation Settings dialog box.

To edit a model mapping:

1. Choose **Visual Mappings** → **Entity Models**. The Shared Simulation Settings dialog box opens to the Entity Mapping Settings page (Figure 10-1).
2. Select the model mapping that you want to edit. If you know the entity enumeration that you want to edit, you can type the enumeration in the Entity Type text box and click Select Match. VR-Forces highlights the entry for that enumeration if it is present.
3. To change the enumeration associated with a model definition:
  - a. In the Entity Type column, click the enumeration to make it editable.
  - b. Change the values of the enumeration.
  - c. Click someplace else in the window.
4. To change the model definition associated with an enumeration:
  - a. In the Model Definition column, click the model definition to make it editable (as a drop-down list (Figure 10-3)).

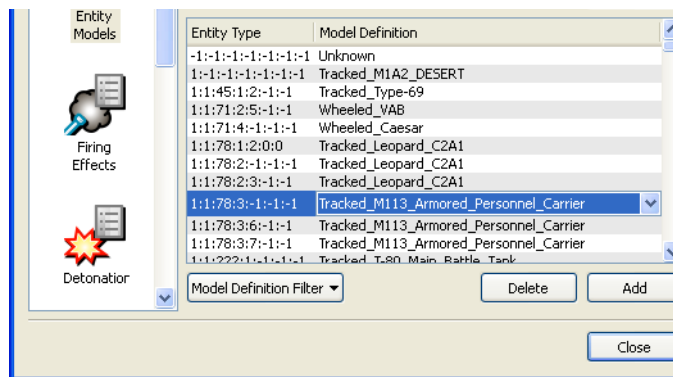


Figure 10-3. Editable model definition

- b. Select the new model definition from the drop-down list.
- c. Click someplace else in the window.

### **10.2.3. Deleting a Model Mapping**

The procedure for deleting a model mapping is the same for any page on the Shared Simulation Settings dialog box.

To delete a model mapping:

1. Choose **Visual Mappings** → **Entity Models**. The Shared Simulation Settings dialog box opens to the Entity Mapping Settings page ([Figure 10-1](#)).
2. Select the model mapping that you want to delete. If you know the entity enumeration that you want to edit, you can type the enumeration in the Entity Type text box and click Select Match. VR-Forces highlights the entry for that enumeration if it is present.
3. Click Delete.

### 10.3. Creating and Editing Model Definitions

Each model displayed in VR-Forces is defined by a set of parameters that are appropriate to the model type. (The parameters are specified in the model's schema. For information about model schema, please see [“Creating and Editing Model Definition Schemata,”](#) on page 10-15.) VR-Forces comes with an extensive set of model definitions that are mapped to the included models. You can edit or delete these model definitions and you can create new ones.

All model configuration is done in the Model Definitions page of the Applications Settings dialog box ([Figure 10-4](#)).

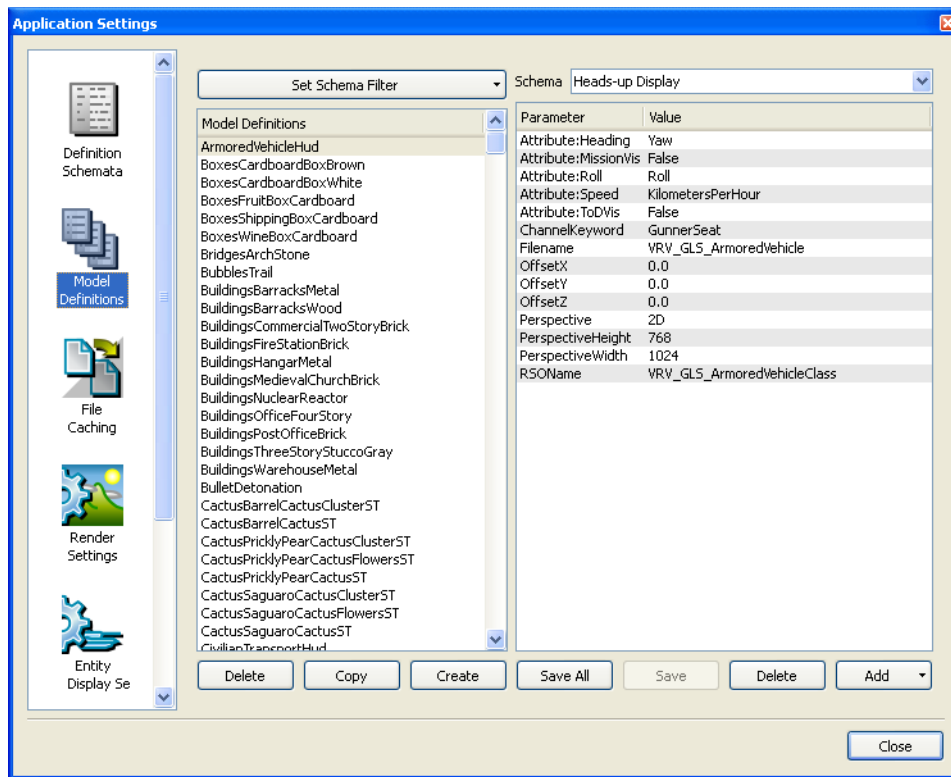


Figure 10-4. Model Definitions page

### 10.3.1. Creating a Model Definition

To create a model definition:

1. Choose **Settings** → **Model Definitions**. The Application Settings dialog box opens to the Model Definitions page ([Figure 10-4](#)).
2. Optionally, filter the model definitions in the list. (For details, please see [“Filtering the List of Model Definitions,”](#) on page 10-10.)
3. Click the Create button. A dialog box opens ([Figure 10-5](#)).

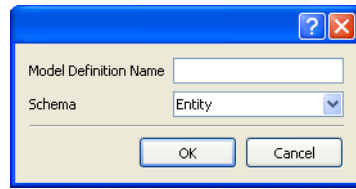


Figure 10-5. Create model definition

4. Type a name for the new model definition.
5. In the Schema drop-down list, select a schema for the model.
6. Add parameters to the model, as described in [“Adding Parameters to a Model Definition,”](#) on page 10-11.

### 10.3.2. Deleting a Model Definition

To delete a model definition:

1. Choose **Settings** → **Model Definitions**. The Application Settings dialog box opens to the Model Definitions page ([Figure 10-4](#)).
2. Optionally, filter the model definitions in the list. (For details, please see [“Filtering the List of Model Definitions,”](#) on page 10-10.)
3. Select the model that you want to delete.
4. Click the Delete button. You are prompted to confirm the deletion.
5. Click Yes. The model definition is removed from the list.

### 10.3.3. Copying a Model Definition

If you want a new model that is a variant of an existing one, it can be easier to copy the existing model and edit it, than to create a new model and add all of the parameters.

To copy a model definition:

1. Choose **Settings** → **Model Definitions**. The Application Settings dialog box opens to the Model Definitions page (Figure 10-4).
2. Optionally, filter the model definitions in the list. (For details, please see “[Filtering the List of Model Definitions](#),” on page 10-10.)
3. Select the model that you want to copy.
4. Click the Copy button. A dialog box opens (Figure 10-6).

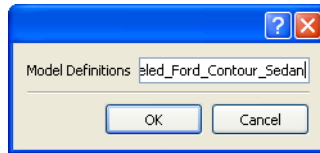


Figure 10-6. Copy model definition

5. Type a name for the copied version of the model.
6. Click OK.
7. The new model definition is added to the list.
8. Edit the parameters as described in “[Editing a Model Definition](#),” on page 10-11.

### 10.3.4. Filtering the List of Model Definitions

By default, the list of models in the Model Definitions page lists all models that VR-Forces knows about. You can filter the list to more easily work with a specific type of model.

- To filter the list of model definitions, select a schema type from the Set Schema Filter drop-down list.

### 10.3.5. Editing a Model Definition

You can add and delete the parameters in a model definition and can change their values. You can only add parameters that are appropriate to the schema type for a model.

#### Adding Parameters to a Model Definition

When you create a new model definition, you must specify the parameters required by its schema. You can only add parameters that are permitted by the schema. For existing model definitions, you cannot add parameters unless a required parameter is missing.

To add a parameter to a model:

1. Choose **Settings** → **Model Definitions**. The Application Settings dialog box opens to the Model Definitions page ([Figure 10-4](#)).
2. Optionally, filter the model definitions in the list. (For details, please see “[Filtering the List of Model Definitions](#),” on page 10-10.)
3. Select the model for which you want to add a parameter.
4. Click the Add button. A list of parameters is displayed ([Figure 10-7](#)).

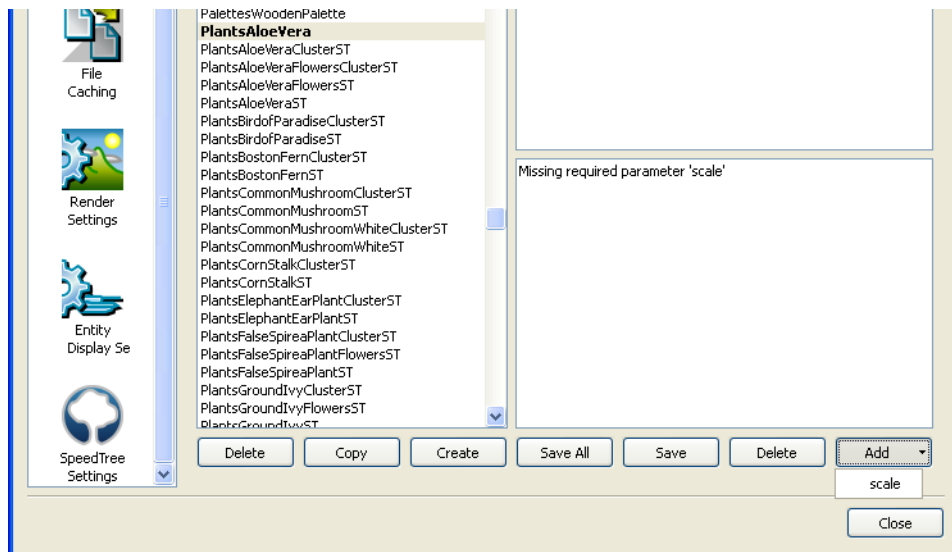


Figure 10-7. Add parameter list

5. Select the parameter you want to add. It is added to the parameter list.
6. Click the value for the parameter. The field becomes editable ([Figure 10-8](#)).

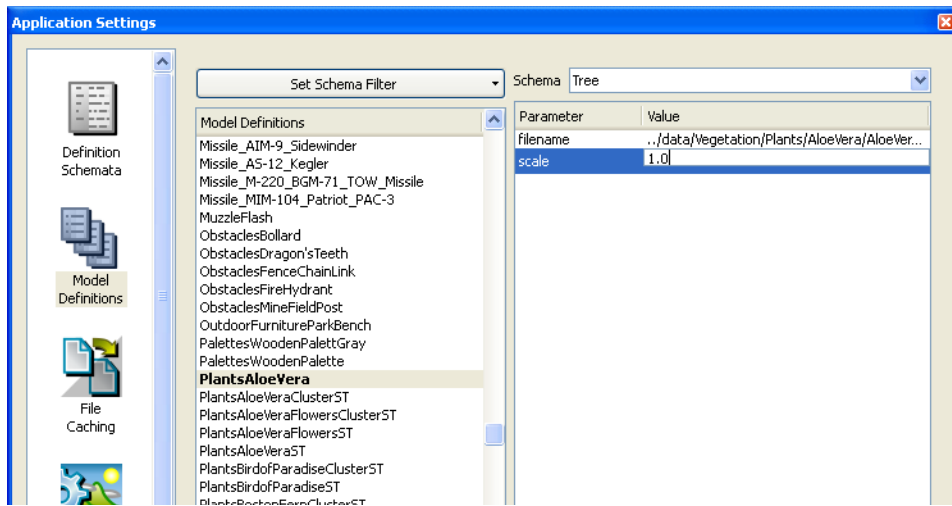


Figure 10-8. Editable parameter

7. Type a value for the parameter.
8. Click away from the field.

### Editing a Model Definition Parameter

To edit a parameter:

1. Choose **Settings** → **Model Definitions**. The Application Settings dialog box opens to the Model Definitions page (Figure 10-4).
2. Optionally, filter the model definitions in the list. (For details, please see “[Filtering the List of Model Definitions](#),” on page 10-10.)
3. Select the model for which you want to edit a parameter.
4. Click the value for the parameter you want to edit. The field becomes editable.

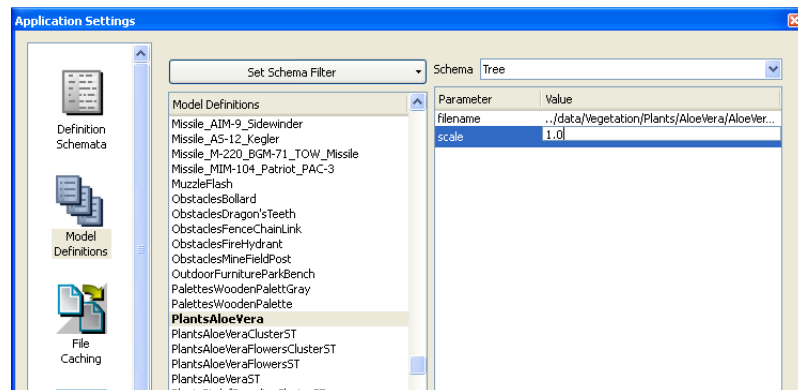


Figure 10-9. Editable parameter



5. Type a value for the parameter.
6. Click away from the field.

### Deleting Parameters from a Model Definition



---

If you delete a parameter from a model definition, the definition may become invalid. Invalid model definitions are shown in red.

---

To delete a parameter from a model:

1. Choose **Settings** → **Model Definitions**. The Application Settings dialog box opens to the Model Definitions page ([Figure 10-4](#)).
2. Optionally, filter the model definitions in the list. (For details, please see [“Filtering the List of Model Definitions,”](#) on page 10-10.)
3. Select the model from which you want to delete a parameter.
4. Select the parameter that you want to delete.
5. Click the Delete button under the parameter list. The parameter is removed.

### 10.3.6. Saving Model Definitions

When you change a model definition, the changes persists until you close VR-Forces. If you want the changes to be saved for future use, you can save changes for individual models and for all changed models.

- To save changes to the currently selected model definition, click the Save button.
- To save all changed model definitions, click the Save All button.

## 10.4. Configuring Model Instancing

When model instancing is enabled, models that use the same resources can share them. Usually, models can share geometry and textures. Use of model instance caching decreases memory use and improves performance.

To configure model instancing:

1. Choose **Settings** → **File Caching**. The Application Settings dialog box opens to the File Cache Settings page (Figure 10-10).

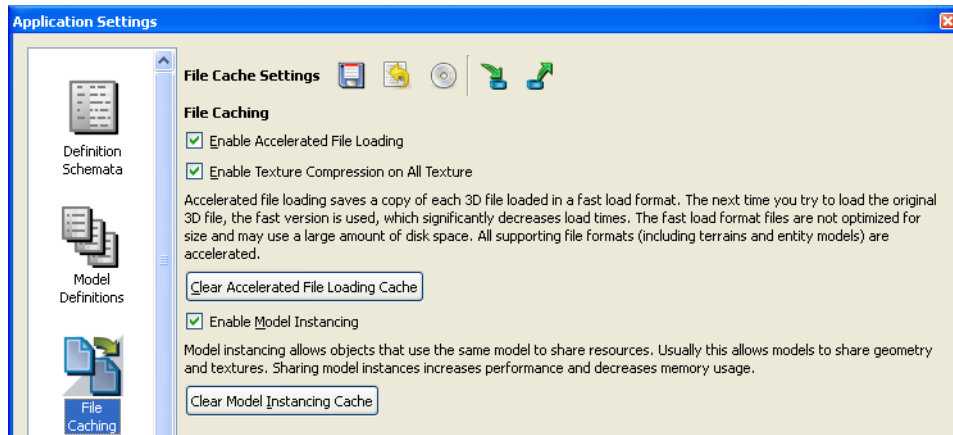


Figure 10-10. File Caching page

2. To enable model instancing, select the Enable Model Instancing check box. To disable model instancing, clear the check box.

### 10.4.1. Clearing the Model Instancing Cache

You can clear the model instancing cache if you need the disk space.

To clear the model instancing cache:

1. Choose **Settings** → **File Caching**. The Application Settings dialog box opens to the File Cache Settings page (Figure 10-10).
2. Click the Clear Model Instancing Cache button.

## 10.5. Creating and Editing Model Definition Schemata

The models and terrain patches that make up a scene have parameters that help define them, for example a filename or a scale. The set of parameters for a particular scene element is a schema. VR-Forces uses schema definitions to validate model definitions. You can create, copy, and delete model definition schemas. For any schema, you can add and delete parameters and edit parameter values.

### 10.5.1. Creating Model Definition Schemas

To create a model definition schema:

1. Choose **Settings** → **Definition Schemata**. The Application Settings dialog box opens to the Definition Schemata page (Figure 10-11).

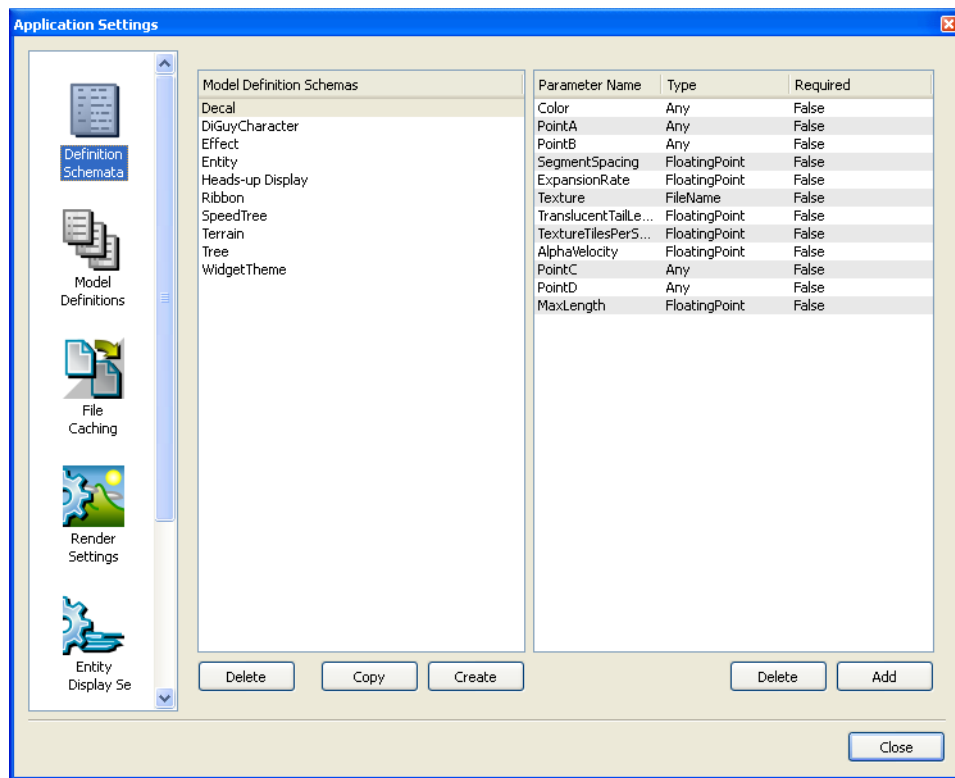


Figure 10-11. Definition Schemata page

2. Click the Create button. A dialog box opens (Figure 10-12).

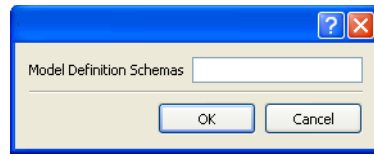


Figure 10-12. Model definition schemas

3. Type a name for the new schema.
4. Click OK.
5. Add parameters as described in [“Adding a Parameter to a Model Definition Schema,”](#) on page 10-17.

### 10.5.2. Deleting Model Definition Schemas



---

If you delete a model definition schema, any model definitions that use the schema become invalid. Invalid model definitions are shown in red in the list of model definitions.

---

To delete a model definition schema:

1. Choose **Settings** → **Definition Schemata**. The Application Settings dialog box opens to the Definition Schemata page ([Figure 10-11](#)).
2. In the Model Definition Schemas list, select the schema definition you want to delete.
3. Click the Delete button under the list of schemas. You are prompted to confirm the deletion.
4. Click Yes. The schema is removed from the list.

### 10.5.3. Copying Model Definition Schemas

If you want a new schema that is a variant of an existing one, it can be easier to copy the existing schema and edit it, than it is to create a new schema and add all of the parameters.

To copy a model definition schema:

1. Choose **Settings** → **Definition Schemata**. The Application Settings dialog box opens to the Definition Schemata page ([Figure 10-11](#)).
2. Select the schema that you want to copy.
3. Click the Copy button. A dialog box opens.
4. Type a name for the copied version of the schema.
5. Click OK.
6. The copied schema is added to the list.
7. Edit the parameters as described in “[Editing a Model Definition Schema Parameter](#),” on page 10-18.

### 10.5.4. Adding a Parameter to a Model Definition Schema

To add a parameter to a model definition schema:

1. Choose **Settings** → **Definition Schemata**. The Application Settings dialog box opens to the Definition Schemata page ([Figure 10-11](#)).
2. In the Model Definition Schemas list, select the schema to which you want to add a parameter.
3. Click the Add button. A parameter called “key” is added to the parameter list.
4. Double-click “key” to enter edit mode.
5. Type a name for the parameter.
6. If the parameter has a specific type, edit the value in the Type column.
7. If the parameter is required, edit the value in the Required column.

### 10.5.5. Deleting a Parameter from a Model Definition Schema



---

If you delete a parameter from a model definition schema, any model definitions that use the schema may become invalid. Invalid model definitions are shown in red in the list of model definitions.

---

To delete a parameter from a model definition schema:

1. Choose **Settings** → **Definition Schemata**. The Application Settings dialog box opens to the Definition Schemata page ([Figure 10-11](#)).
2. In the Model Definition Schemas list, select the schema from which you want to delete a parameter.
3. Select the parameter that you want to delete.
4. Click the Delete button under the parameter list.

### 10.5.6. Editing a Model Definition Schema Parameter

To edit a parameter for a model definition schema:

1. Choose **Settings** → **Definition Schemata**. The Application Settings dialog box opens to the Definition Schemata page ([Figure 10-11](#)).
2. In the Model Definition Schemas list, select the schema for which you want to edit a parameter.
3. Double-click the parameter attribute that you want to edit. The attribute becomes editable.
4. Type the new attribute value.

## 10.6. Configuring SpeedTree Trees

SpeedTree trees are high resolution models of trees. To improve performance, you can set clipping plane and levels-of-detail (LOD) values to vary the resolution of the trees placed in the scene based on their distance from the observer. For details about clipping planes, please see [“Setting the Clipping Planes,”](#) on page 11-11.

To configure SpeedTree trees:

1. Choose **Settings** → **SpeedTree Settings**. The Application Settings dialog box opens to the SpeedTree Settings page ([Figure 10-13](#)).

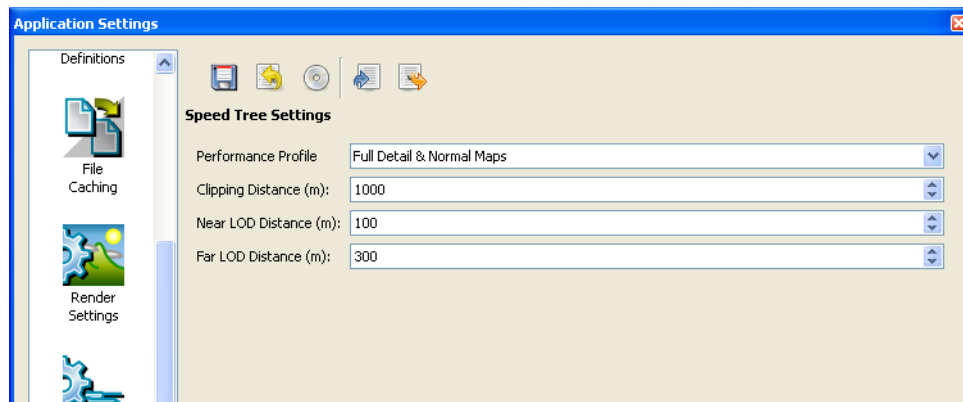


Figure 10-13. SpeedTree Settings page

2. Select a performance profile. (For details, please see [“Specifying the SpeedTree Performance Profile,”](#) on page 10-20.)
3. Specify the clipping distance.
4. Specify the near and far LOD distance.
5. Click Close.

### 10.6.1. Specifying the SpeedTree Performance Profile

The number of SpeedTree objects in a scene can affect performance. Therefore, VR-Forces provides several profiles for display of SpeedTrees so that you can balance performance against visual quality. You specify the performance profile on the SpeedTree Settings page of the Application Settings dialog box. [Table 10-1](#) describes the profiles.

Table 10-1: SpeedTree Performance Profiles

| Profile                            | Branch Texture                          | Fronde Texture                | Vertical Billboards | Horizontal Billboards |
|------------------------------------|-----------------------------------------|-------------------------------|---------------------|-----------------------|
| Disabled                           |                                         |                               |                     |                       |
| Single Color Map                   | Color                                   | Color                         | Low LOD             |                       |
| Partial Detail & Normal Maps       | Color<br>Detail<br>Normal map           | Color                         | Low LOD             |                       |
| Full Detail & Normal Maps          | Color<br>Detail<br>Normal map           | Color<br>Normal map           | Low LOD             |                       |
| Full Detail & Normal & Shadow Maps | Color<br>Detail<br>Normal map<br>Shadow | Color<br>Detail<br>Normal map | Low LOD             | Low LOD               |

---

## 10.7. Adding a Model to VR-Forces - A Tutorial

This section is a brief tutorial that shows how to add a new model to the 3D Front End. To help you with the tutorial, VR-Forces includes a model of an F/A-18 Hornet that is not mapped to any entities. You can substitute your own model in the tutorial if you want to.

The basic steps are:

1. Create a model definition.
2. Map the new model to an entity or object type enumeration.
3. Test the new model.



### 10.7.1. Create a Model Definition

To create the model definition:

1. Choose **Settings** → **Model Definitions**. The Application Settings dialog box opens to the Model Definitions page (Figure 10-14).

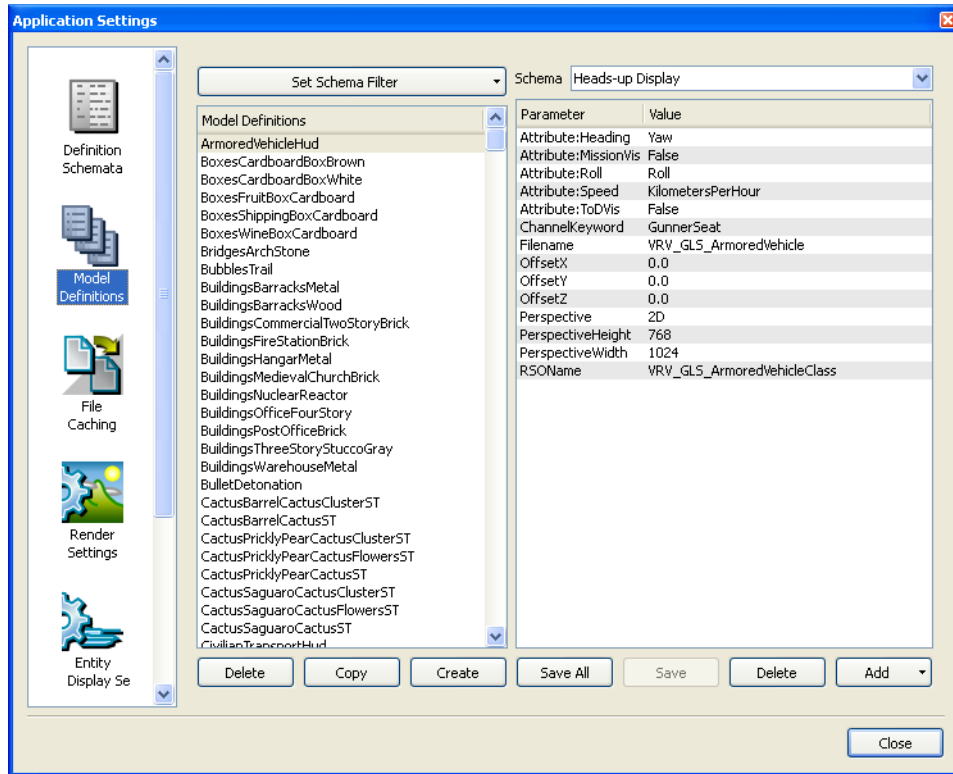


Figure 10-14. Model Definitions page

2. Optionally, filter the model definitions in the list. (For details, please see [“Filtering the List of Model Definitions,”](#) on page 10-10.)
3. Click the Create button. A dialog box opens (Figure 10-15).

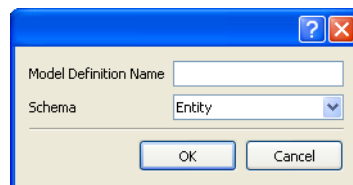


Figure 10-15. Create Model Definition dialog box

4. Type a name for the new model definition, for example, myF18.

5. In the Schema drop-down list, select Entity. (If you are adding a model for an object that is not an entity, choose the proper schema from the list.) The dialog box now looks like (Figure 10-16).

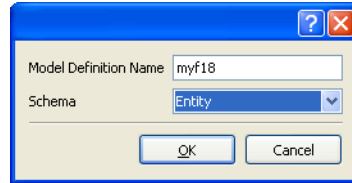


Figure 10-16. New model definition

6. Click OK. The new definition is added to the list (Figure 10-17). Now you must specify the model file for this model definition.

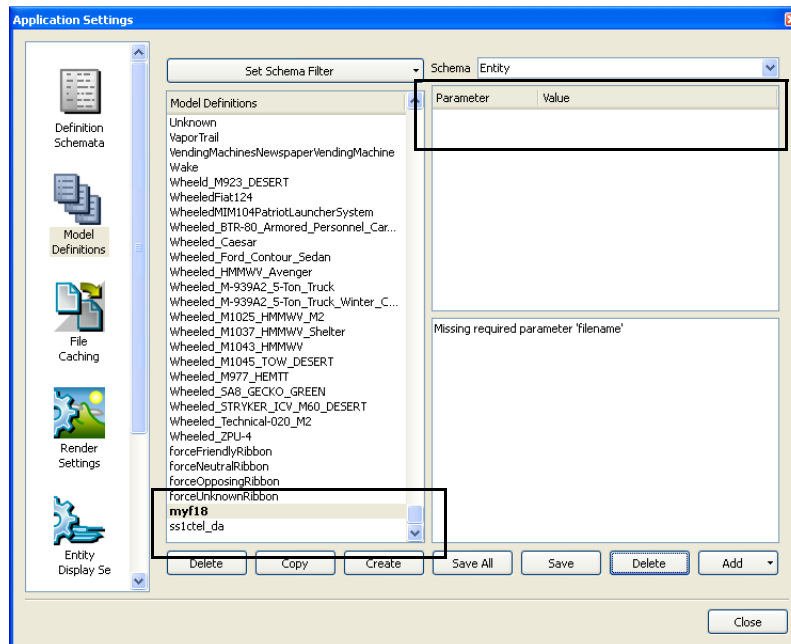


Figure 10-17. New model definition in list

7. Click the Add button. A list of parameters is displayed (Figure 10-18).

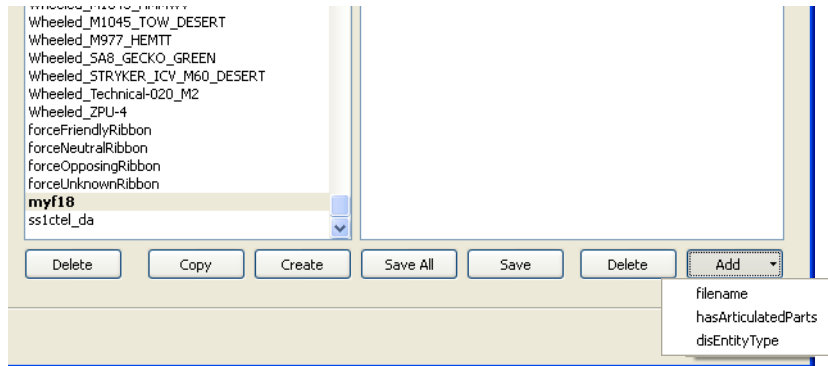


Figure 10-18. List of parameters

8. Select filename. It is added to the parameter list.
9. Click the value for the parameter. A file selection dialog box opens.
10. Navigate to *./data/Vehicles/FixedWing/FA-18\_Hornet* and select *FA-18\_Hornet.flt*. (Or enter the filename for the model that you are using. It must be in one of the supported formats (FLT, OBJ, 3ds).)
11. Click OK. The value is updated (Figure 10-19).

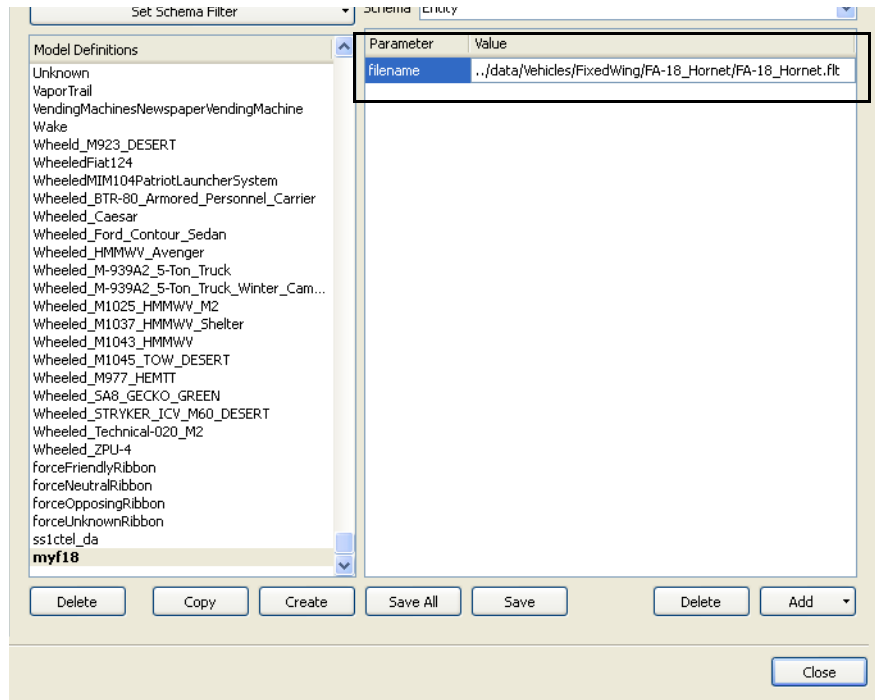


Figure 10-19. Model definition parameter

12. Click the Save button. You have finished creating the model definition.

## 10.7.2. Add a Model Mapping

Now, you have to map the new model definition to an entity type enumeration.

To add a model mapping:

1. Choose **Visual Mappings** → **Entity Models**. The Shared Simulation Settings dialog box opens to the Entity Mapping Settings page (Figure 10-20).

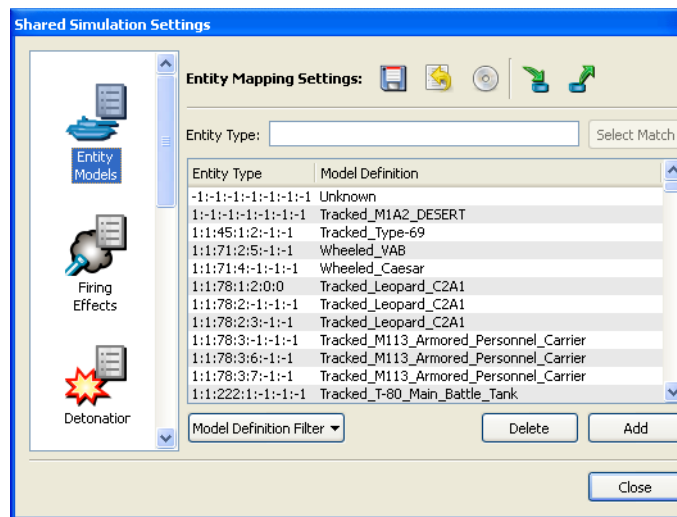


Figure 10-20. Entity Mapping Settings page

2. Click the Add button. The Create New Entity Type Model Mapping dialog box opens (Figure 10-21). It lists a default entity type enumeration.

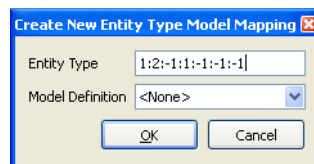


Figure 10-21. Create New Entity Type Model Mapping dialog box

3. In the Entity Type text box, type the enumeration you want to map to the new model mapping. (The enumeration for an F/A18 is 1:2:225:1:9:1:-1.)



VR-Forces already has an F/A18 mapped to this enumeration. However you can map multiple model definitions to an enumeration. When an entity type that has multiple model definitions mapped to it is discovered in a simulation, VR-Forces randomly assigns a model definition from among those mapped to the entity type.

4. In the Model Definition drop-down list, select the model definition that you added in the previous procedure (myF18) (Figure 10-22).

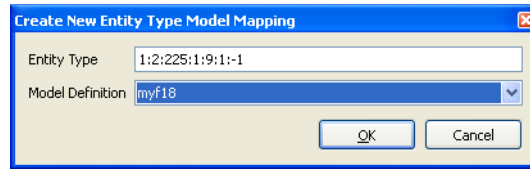


Figure 10-22. Create new model mapping

5. Click OK.
6. The new model mapping is added to the list.

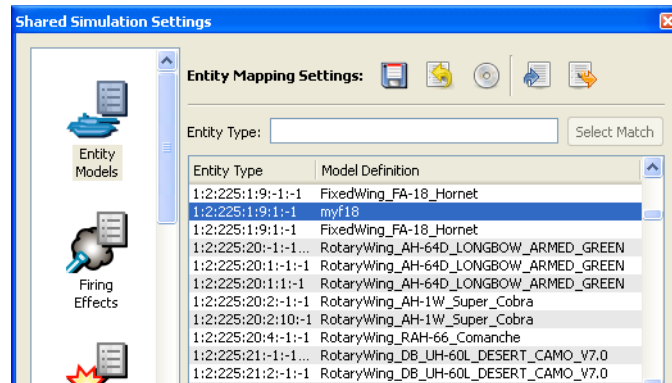


Figure 10-23. New model mapping

### 10.7.3. Test the Model Mapping

It is a good idea to test the new model mapping to see if the correct model is being displayed. You can test a model by running a simulation that has the entity or object type, or you can perform the simple test described in this section.

To test a model:

1. Open a terrain database that has easily identifiable props. For the purposes of this tutorial, open the Makland terrain.
2. In the VR-Forces window, move the observer so that you are viewing the palace on top of the hill near the town in Makland.
3. Select the palace ([Figure 10-24](#)).

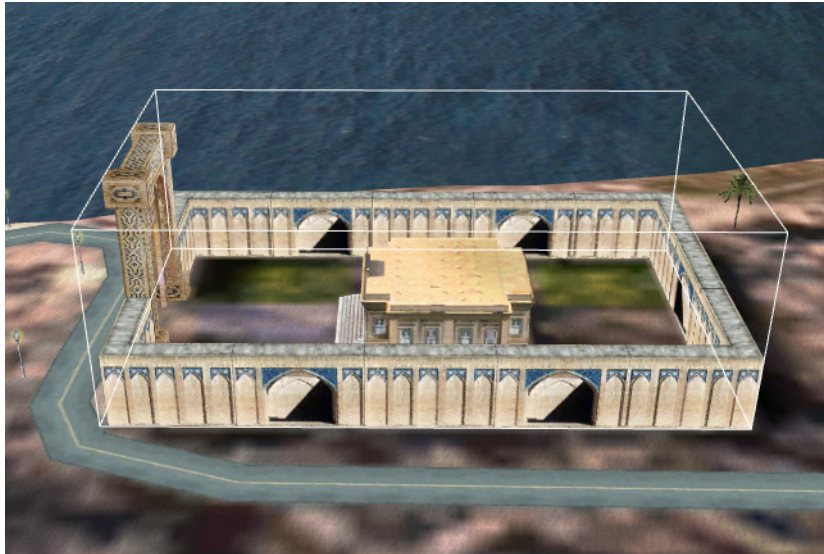


Figure 10-24. Palace

4. Choose **Terrain** → **Props**. The Terrain Settings dialog box opens. The list of props shows that Building26, the palace, is selected ([Figure 10-25](#)).

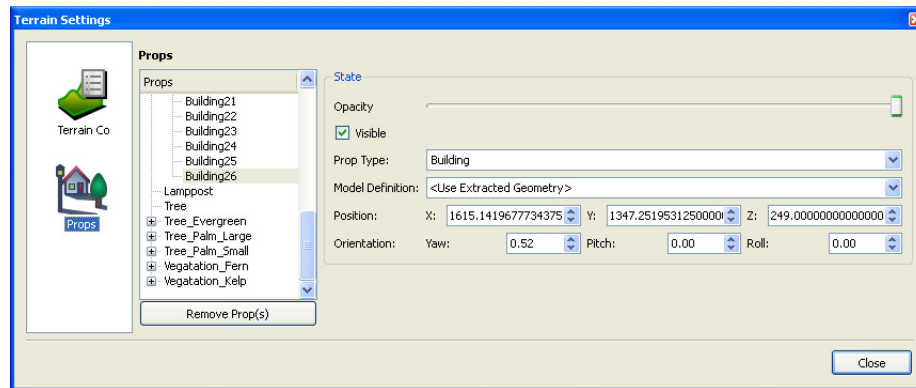


Figure 10-25. Terrain Settings dialog box

5. In the Model Definition drop-down list, select the model definition that you added earlier in this tutorial (Figure 10-26). The model of the palace is replaced with the model for the F/A-18 (Figure 10-27). If this is the model that you wanted to add, then you know that you completed the procedure correctly. If this is not the model, you expected, check the filename you specified for the model definition.

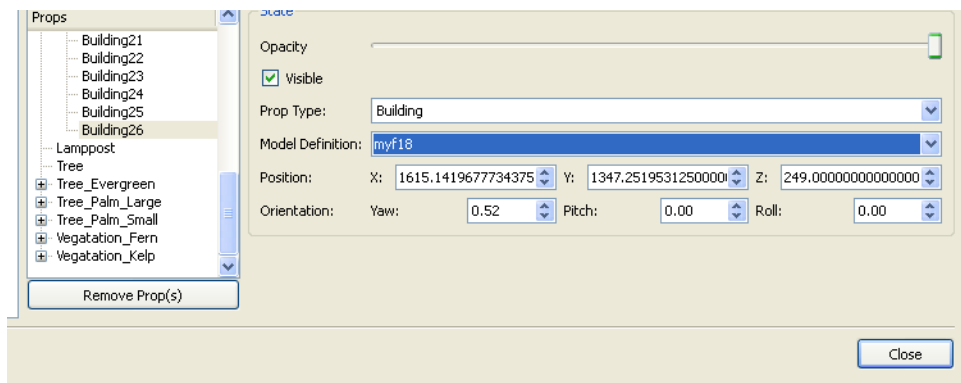


Figure 10-26. Changed model definition

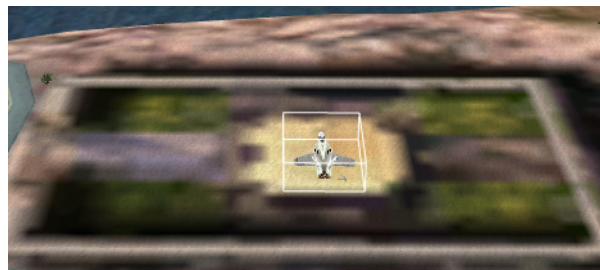
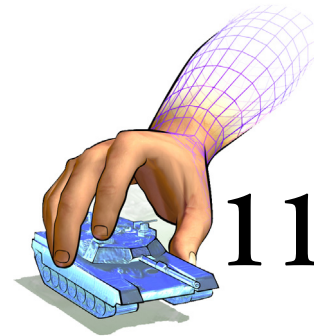


Figure 10-27. F/A-18 replaces the palace







# *Configuring 3D Windows and Channels*

---

This chapter explains how to add and configure windows and channels in the VR-Forces 3D Front End and how to connect to remote display engines.

|                                                            |       |
|------------------------------------------------------------|-------|
| The VR-Vantage Display Engine .....                        | 11-2  |
| How VR-Forces Responds to Display Engine Connections ..... | 11-2  |
| The Display Engine's Connection States.....                | 11-3  |
| Windows, Channels, and the Observer .....                  | 11-3  |
| Managing Display Engine Configurations.....                | 11-3  |
| Adding a Window .....                                      | 11-4  |
| Adding a Channel to a Window .....                         | 11-6  |
| Removing a Window.....                                     | 11-6  |
| Removing a Channel.....                                    | 11-6  |
| Saving a Display Engine Configuration .....                | 11-7  |
| Loading a Display Engine Configuration.....                | 11-7  |
| Changing a Window's Properties.....                        | 11-7  |
| Changing a Channel's Properties.....                       | 11-9  |
| Setting the Clipping Planes .....                          | 11-11 |
| Specifying the Projection Resize Policy Property .....     | 11-12 |
| Changing a Channel's Frustum (Field of View) .....         | 11-14 |
| Changing the Viewport .....                                | 11-15 |
| Starting a VR-Vantage Display Engine.....                  | 11-16 |
| Connecting to a Display Engine.....                        | 11-17 |
| Disconnecting from a Display Engine .....                  | 11-18 |
| Configuring Multi-Channel Displays.....                    | 11-19 |
| Changing the Camera's Position and Orientation Offset..... | 11-21 |

## 11.1. The VR-Vantage Display Engine

VR-Forces includes an application called the VR-Vantage Display Engine. A VR-Vantage Display Engine has a very limited GUI. Its primary purpose is to provide 3D visualization windows that can be controlled by the VR-Forces (master) display engine.

The master display engine can connect to multiple VR-Vantage Display Engines (typically running on other computers) and control what is displayed in their windows.

A display engine can display multiple windows, each of which can have zero or more channels. The windows are all configured in the 3D Front End as part of a display engine configuration. You can save a display engine configuration and load it at a later time.

The 3D Front End drives all of the display engines it is connected to, as illustrated in [Figure 11-1](#).

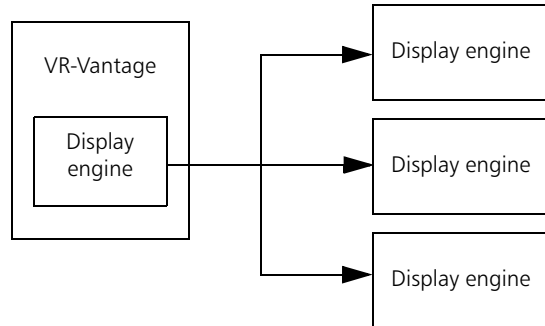


Figure 11-1. VR-Forces and display engines



Display engines must have the same terrains, models, and other visualization data available as the 3D Front End.

---

### 11.1.1. How VR-Forces Responds to Display Engine Connections

The 3D Front End reloads the current scene each time a display engine is connected. (A scene is a term for the terrain and environmental settings in a scenario.) This ensures that the display engine has an exact duplicate of the current scene. The 3D Front End does not perform any special actions when a display engine disconnects.

### **11.1.2. The Display Engine's Connection States**

An external display engine is initially in an unconnected state. While it is unconnected, it cannot display a scene. When the 3D Front End connects to a display engine, the display engine creates an identical copy of the scene regardless of whether or not it can render it. The display engine is automatically sent a default display configuration that has a single channel on a single window. However, you can change this at runtime or programmatically. You could also load a saved configuration. If a display engine loses its connection to the 3D Front End, it clears its local scene and returns to an unconnected state.

## ***11.2. Windows, Channels, and the Observer***

To view a scene and simulation data, a display engine must have a window. The 3D Front End supports three types of windows:

- ♦ Full screen. A full screen window uses the entire area of a monitor. It does not have any menus, panels, or other GUI controls.
- ♦ Resizable. A resizable window is similar to a typical window in a graphical windowing environment. It can be moved, resized, maximized, and minimized. It does not have any GUI controls.
- ♦ Embedded. An embedded window is part of an application. The 3D display in VR-Forces is an embedded window. The application provides a variety of controls that affect the view in the window. You could, if you wanted to, change the 3D Front End's window to a different type, in which case it would no longer be embedded in the the 3D Front End window.

A channel exists within a window. It is what the observer sees. The observer is a place in the 3D world. It has an orientation, an offset, and a camera, through which it sees the world. When you look at the rendered display in a the 3D Front End window, you are seeing the world as viewed by the observer.

## ***11.3. Managing Display Engine Configurations***

The 3D Front End has one display engine. You can add windows to the display engine and you can add channels to the windows. The windows and channels make up a display engine configuration. You can save a display engine configuration and load a saved configuration.

### 11.3.1. Adding a Window

You can add as many windows as you want. New windows are named Window 1, Window 2, and so on. Each window is created with one channel.

To add a window:

1. Choose **View** → **Display Engine Configuration Editor**. The Display Engine Configuration Editor panel is added to the the 3D Front End window (Figure 11-2).

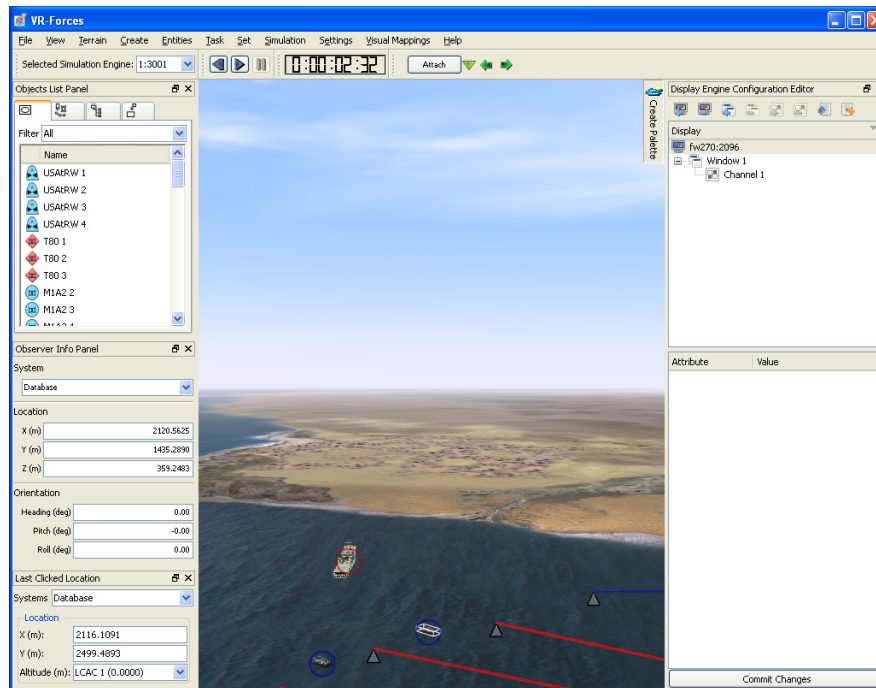


Figure 11-2. Display Engine Configuration Editor



The Display Engine Configuration Editor is available only in VR-Forces, not on display engines.

2. Select the display engine at the top of the Display tree.
3. Click the Add a Window icon (Figure 11-4), or right-click the display engine name and choose **Add a Window** on the popup menu. A new window is displayed and listed in the Display Engine Configuration Editor (Figure 11-3).

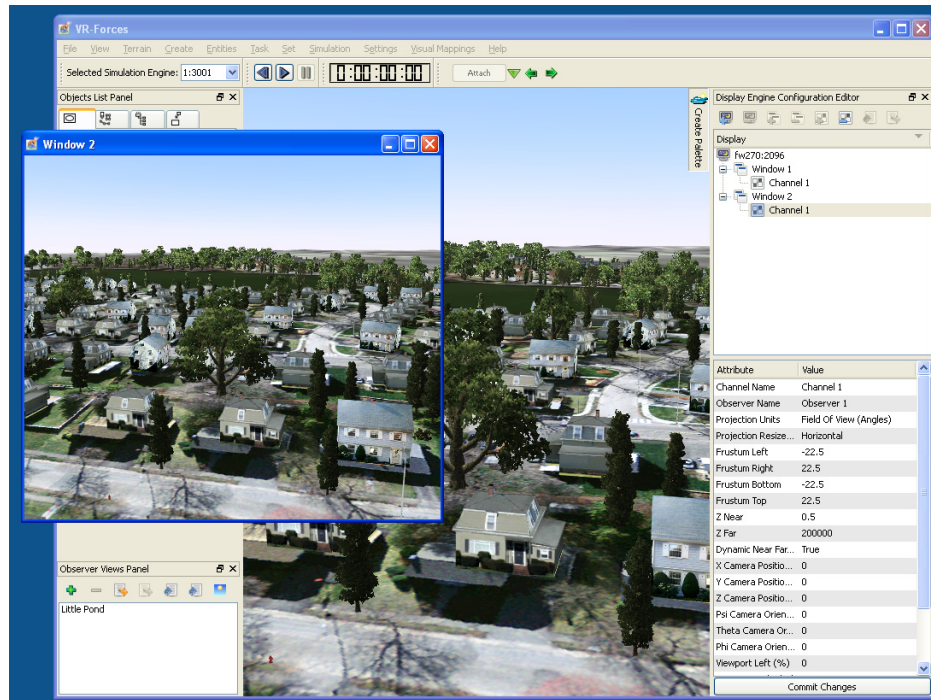


Figure 11-3. Display Engine Configuration Editor with new window

### The Display Engine Configuration Editor Panel

The Display Engine Configuration Editor panel (Figure 11-4) has icons for the following actions:

- ♦ Connect (1) and disconnect (2) a display engine
- ♦ Add (3) and remove (4) a window
- ♦ Add (5) and remove (6) a channel
- ♦ Save (7) and load (8) a display engine configuration.

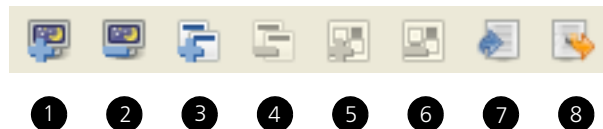


Figure 11-4. Display Engine Configuration Editor icons

### 11.3.2. Adding a Channel to a Window

Adding a channel to a window lets you configure multiple views within that window.

To add a channel to a window:

1. In the Display Engine Configuration Editor, select the window to which you want to add a channel.
2. Click the Add a Channel icon ([Figure 11-4](#)), or right-click the window name and choose **Add a Channel** from the popup menu.



To see a difference between the channels, change the properties of the new channel. For details about editing channel properties, please see “[Changing a Channel’s Properties](#),” on page 11-9.

---

### 11.3.3. Removing a Window

To remove a window:

1. In the Display Engine Configuration Editor, select the window that you want to remove.
2. Click the Remove Window icon ([Figure 11-4](#)), or right-click the window name and choose **Remove Window** from the popup menu.

### 11.3.4. Removing a Channel

To remove a channel:

1. In the Display Engine Configuration Editor, select the channel that you want to remove.
2. Click the Remove Channel icon ([Figure 11-4](#)), or right-click the window name and choose **Remove Channel** from the popup menu.

### 11.3.5. Saving a Display Engine Configuration

You can save a display engine configuration so that you can easily replicate the configuration at a later time.

To save a display engine configuration:

1. In the Display Engine Configuration Editor, select the master display engine.
2. Click the Save Display Engine Configuration icon ([Figure 11-4](#)), or right-click the name of the display engine and choose **Save Display Configuration** on the popup menu. The Save Display Configuration dialog box opens.
3. Type a name for the display configuration.
4. Click Save.

### 11.3.6. Loading a Display Engine Configuration

If you have any open windows in the current display, you must remove them before you can load a display engine configuration.

To load a display engine configuration:

1. In the Display Engine Configuration Editor, select the master display engine.
2. Click the Load Display Engine Configuration icon ([Figure 11-4](#)), or right-click the name of the display engine and choose **Load Display Configuration** on the popup menu. The Load Display Configuration dialog box opens.
3. Select the display configuration you want to load.
4. Click Open.

## 11.4. Changing a Window's Properties

You can change the following properties of a window:

- ♦ Window Name
- ♦ Window Type (for details about window types, please see [“Windows, Channels, and the Observer,”](#) on page 11-3)
- ♦ Screen Number
- ♦ Position (X, Y coordinates of upper left corner, in screen pixels)
- ♦ Width and height, in pixels
- ♦ Fixed size.

To change a window's properties:

1. Select the window in the Display Engine Configuration Editor. A list of properties is displayed at the bottom of the editor (Figure 11-5).

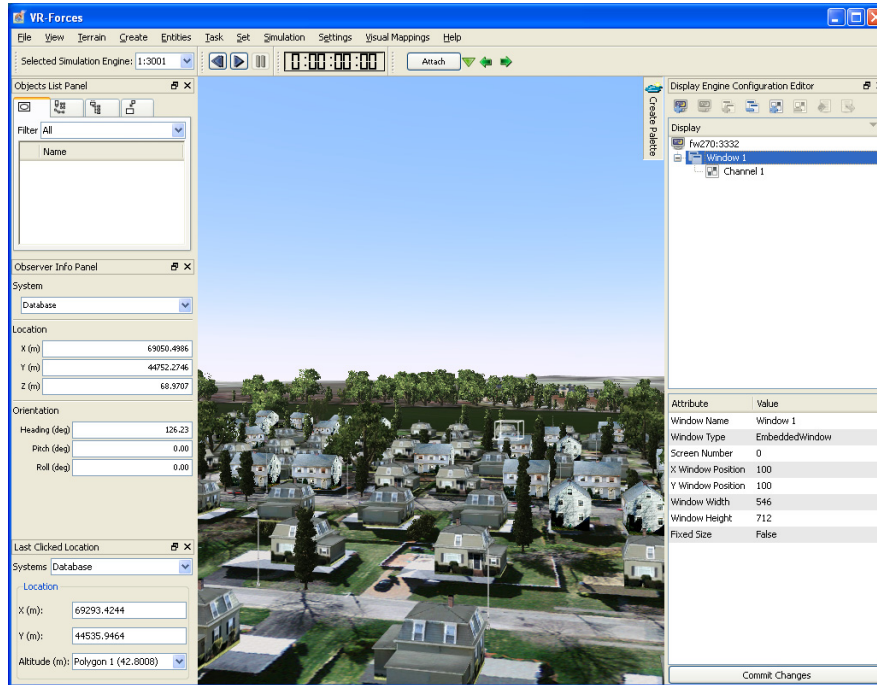


Figure 11-5. Window properties

2. Click the value of the property that you want to change.
3. To change a numeric value, type a new value or use the spin control to change the value. To change the Window Type or Fixed Size, select a value from the drop-down list. To change the name, type a new name.
4. Click Commit Changes. The window is updated.



You can also change a window's position by dragging it to a new location. You can change its size by resizing the window directly. The new values are shown in the properties list the next time you select the window in the Display Engine Configuration Editor.

---



## **11.5. Changing a Channel's Properties**

You can change the following properties of a channel:

- ♦ Channel Name
- ♦ Observer Name
- ♦ Projection Units – field of view, in angles, or database, in meters
- ♦ Projection Resize Policy – fixed, vertical, or horizontal
- ♦ Frustum (Left, Right, Top, and Bottom) (field-of-view)
- ♦ Z Near and Z Far (clipping planes)
- ♦ Dynamic Near Far Planes
- ♦ Dynamic Near/Far Ratio
- ♦ Parent Channel
- ♦ Camera Offset Position (X, Y, Z)
- ♦ Camera Orientation Offset (Psi (heading), Theta (pitch), Phi (roll))
- ♦ Viewport (Left, Right, Bottom, Top)
- ♦ Keywords – user-defined keywords to associate with the channel, for example, “left”.

To change a channel's properties:

1. Select the channel in the Display Engine Configuration Editor. A list of properties is displayed at the bottom of the editor (Figure 11-6).

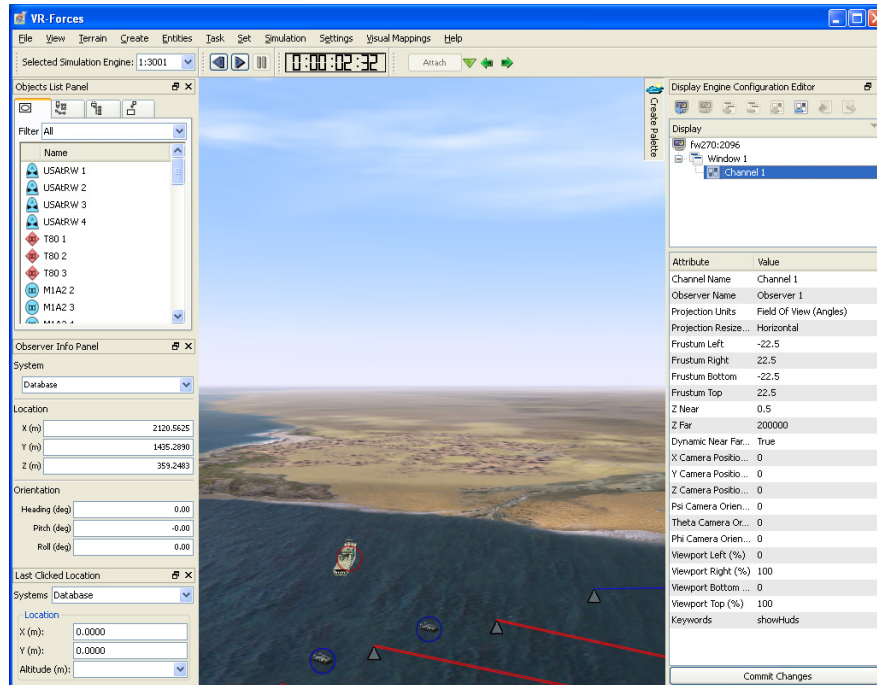


Figure 11-6. Channel properties

2. Click the value of the property that you want to change.
3. To change a numeric value, type a new value or use the spin control to change the value. To change the Observer Name, Project Units, Projections Resize Policy, or Parent Channel, select a value from the drop-down list. To change the Channel Name, type a new name. (For details about some of these attributes, please see the sections that follow this one.)
4. Click Commit Changes. The window is updated.

### 11.5.1. Setting the Clipping Planes

To improve performance, the graphics engine does not render anything that falls outside of a specified distance range. This range is determined by the near and far clipping planes.

By default, the 3D Front End uses dynamic clipping planes. The near and far clipping values are determined based on the altitude of the observer, using the Dynamic Near/Far Ratio attribute. You can disable dynamic clipping planes and set the clipping planes manually with the Z Near and Z Far properties of a channel.

Figure 11-7 illustrates the effect of changing the Z Near value. Figure 11-8 illustrates the effect of changing the Z Far value.

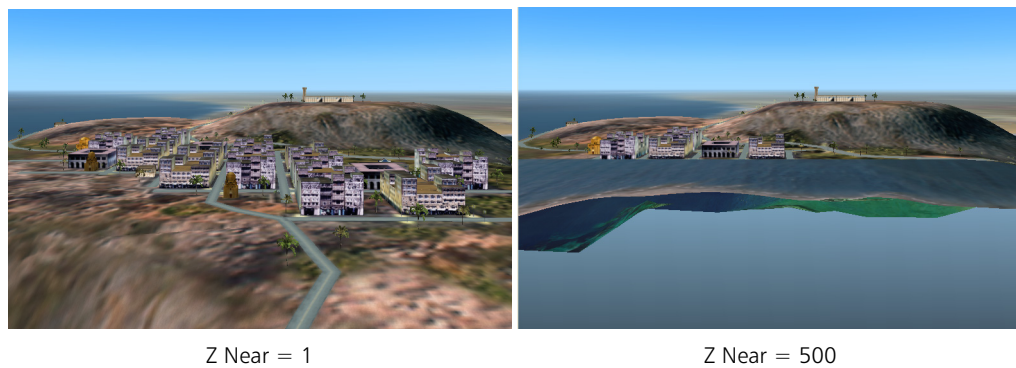


Figure 11-7. Near clipping

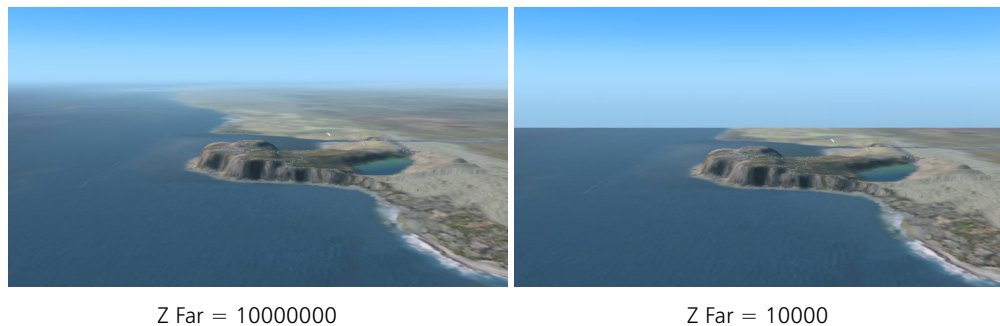


Figure 11-8. Far clipping

- To enable or disable use of dynamic clipping planes, set the Dynamic Near Far Planes attribute to True or False.
- To change the near clipping plane, edit a channel's Z Near property. To change the far clipping plane, edit the Z Far property.

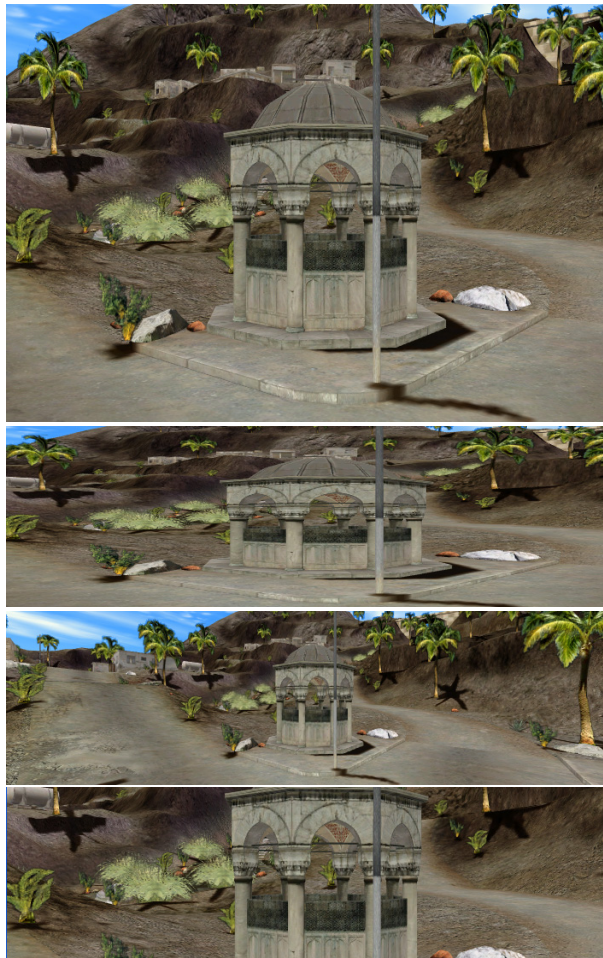
### **11.5.2. Specifying the Projection Resize Policy Property**

The Projection Resize Policy determines how a channel behaves when you resize it. The options are:

- ♦ Fixed: Keep the aspect ratio the same regardless of how the window is resized
- ♦ Horizontal: Maintain the horizontal aspect ratio when the window is resized
- ♦ Vertical: Maintain the vertical aspect ratio when the window is resized.

Figure 11-9 illustrates the effect of different resize policies. In the window with the fixed resize policy, all of the visual data in the initial view is still in the resized view, although the vertical dimensions are distorted. In the window with the horizontal resize policy, the field-of-view is widened to maintain the correct aspect ratio of the visual data. In the window with the vertical resize policy, the field-of-view is shortened.

You can change the projection resize policy for individual channels by editing the Projection Resize Policy attribute. You can also change the projection resize policy for all channels.



Initial view

Resized with Fixed resize policy

Resized with Horizontal resize policy

Resized with Vertical resize policy

Figure 11-9. Effect of different projection resize policies

To change the projection resize policy for all channels:

1. Choose **Settings** → **Render Settings**. The Application Settings dialog box opens to the Render Settings page.
2. Select an option on the Channel Projection Resize Policy drop-down list.

### 11.5.3. Changing a Channel's Frustum (Field of View)

The field-of-view setting affects perspective in a manner similar to a camera lens.

A wide field-of-view creates an effect like that of a wide-angle lens. Objects appear smaller and farther away from the observer, since the observer coverage spans a wider area. Depth distances between objects become exaggerated.

A narrow field-of-view creates an effect like that of a telephoto lens. Objects appear larger and closer to the observer, and the overall scene depth appears flattened. The distances between objects appears compressed.

Figure 11-10 illustrates different field-of-view settings. The view on the left has default frustum values (-30, 30, -30, 30). The view on the right has the frustum values -10, 10, -10, 10.

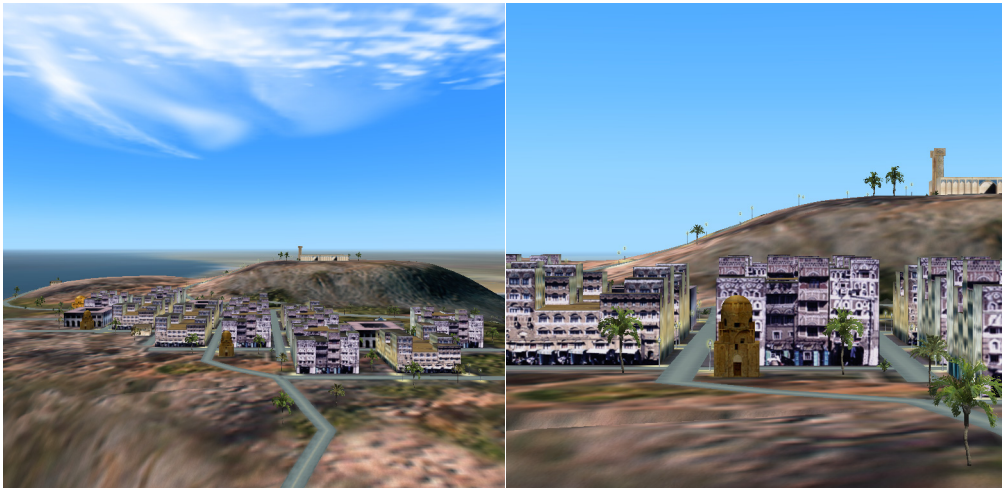


Figure 11-10. Field-of-view

- To change a channel's field of view, edit the frustum values in its properties list.

### 11.5.4. Changing the Viewport

When you change the viewport of a channel, you change the portion of the window in which the channel is displayed. This does not change the extents of the scene. A narrow viewport compresses the scene width. A short viewport compresses the height. [Figure 11-11](#) shows a window that has three channels, each with a different viewport. (The arrows show which portion of the window corresponds to each channel.) Notice that each channel has the same field-of-view.

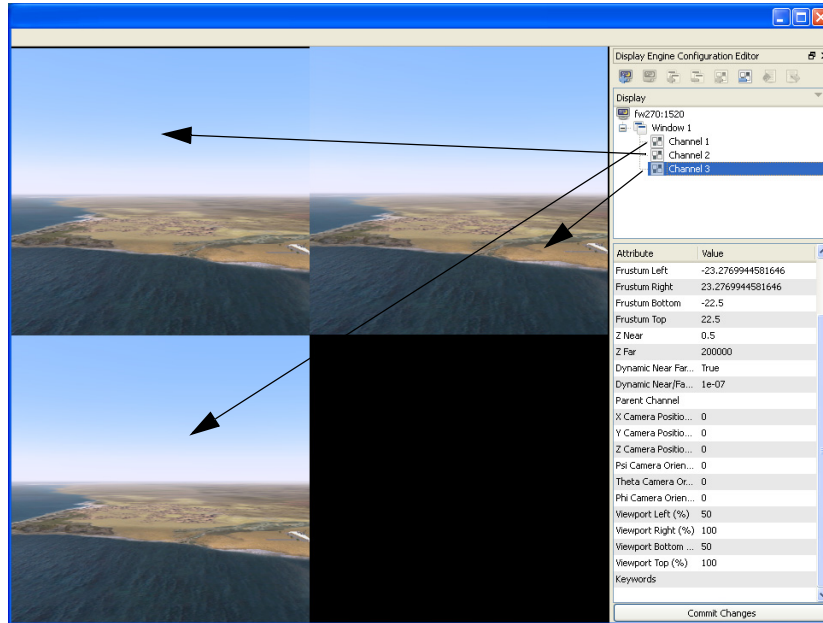


Figure 11-11. Window with three channels and three viewports

Viewports are specified as a percentage of the window from left to right and top to bottom. [Figure 11-12](#) shows the percentage allocation for the viewports in ([Figure 11-11](#)).



|                                                |                                                  |      |
|------------------------------------------------|--------------------------------------------------|------|
| 0                                              | 50%                                              | 100% |
| Left: 0<br>Right: 50<br>Top: 100<br>Bottom: 50 | Left: 50<br>Right: 100<br>Top: 100<br>Bottom: 50 |      |
| Left: 0<br>Right: 50<br>Top: 50<br>Bottom: 0   | Left: 50<br>Right: 100<br>Top: 50<br>Bottom: 0   | 50%  |
|                                                |                                                  | 0    |

Figure 11-12. Viewport configuration



When you change viewports, the window might not completely refresh. To see the changes, resize the window.

---

## 11.6. Starting a VR-Vantage Display Engine

On Windows, you can start a display engine from the Start menu or by running a batch file. On Linux, you start a display engine from a script.

- To start a display engine on the Windows Start menu, choose **Programs → MÄK Technologies → VR-Forces X.X → Tools → VR-Vantage Display Engine**.
- To start a display engine from the Windows console, or on Linux, run:
 

```
./bin/displayEngine.extension
```

where *extension* is the appropriate script extension for the platform.



## 11.7. Connecting to a Display Engine

If you want to control a display engine, you must first connect to it.

Connections are made using TCP networking. All computers running display engines together must be on the same network. The display engines listen for the 3D Front End on a TCP socket. A display engine can only be connected to a single master application.



---

It is possible to run multiple display engines on the same machine with different ports. However, this is not a recommended configuration.

---

To connect to a display engine:

1. On one computer (the master), run the 3D Front End.
2. On another computer, run the VR-Vantage Display Engine application, as described in “[Starting a VR-Vantage Display Engine](#),” on page 11-16.
3. On the master computer, choose **View** → **Display Engine Configuration Editor**. The Display Engine Configuration Editor panel is displayed ([Figure 11-2](#)).
4. In the Display Engine Configuration Editor, click the Connect a Display Engine button. The Connect To Display Engine dialog box opens ([Figure 11-13](#)).

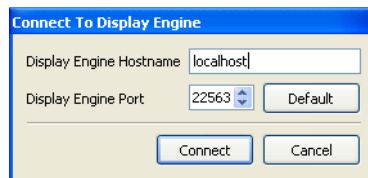


Figure 11-13. Connect to Display Engine dialog box

5. In the Display Engine Hostname box, type the host name or IP address of the computer on which the display engine is running.
6. Optionally, in the Display Engine Port box, specify a port for the display engine. The default is 22563. In most cases the default value will work well.
7. Click Connect. The display engine is added to the Display Engine list. A window is added to the display engine. It shows the same view as Window 1 on the 3D Front End.

### 11.7.1. Disconnecting from a Display Engine

To disconnect from a display engine:

1. In the Display Engine Configuration Editor, select the display engine that you want to disconnect.
2. Click the Disconnect a Display Engine icon ([Figure 11-4](#)), or right-click the display engine name and choose **Disconnect a Display Engine** on the popup menu.



You cannot disconnect the display engine running on the 3D Front End.

---

## 11.8. Configuring Multi-Channel Displays

People running simulations often want to set up a multi-channel configuration using multiple monitors (on one or more computers) to display exercises. Typical configurations are a horizontal alignment, which widens the view, and the angled alignment, which surrounds the observer ([Figure 11-14](#)) to simulate an out-the-window view. Since the 3D Front End just sends out its view, if you do not configure the displays for multi-channel, the subordinate monitors display the exact same view as the master ([Figure 11-16](#), default window views). Therefore, for multiple views, you must configure the channels to calculate an offset from the master view.

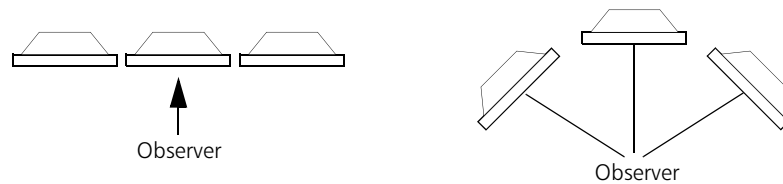


Figure 11-14. Multi-channel displays (top-down view)

The distribution of the view across multiple channels is controlled by the frustum. [Figure 11-15](#) illustrates the default frustum for Channel 1. It has a 60° field of view, from -30° to +30°.

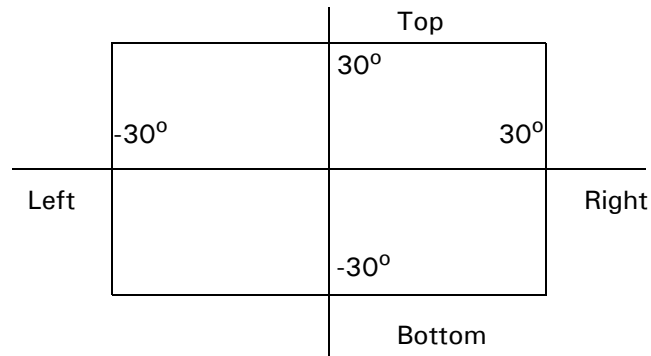


Figure 11-15. Default frustum for Channel 1

If you wanted to split the view in [Figure 11-15](#) into, for example, three horizontal channels, you would edit the left and right frustum values for each channel, as shown in ([Figure 11-16](#)). The field of view is the same: -30° to +30°. However, now each channel shows a 20° segment of the total field of view.

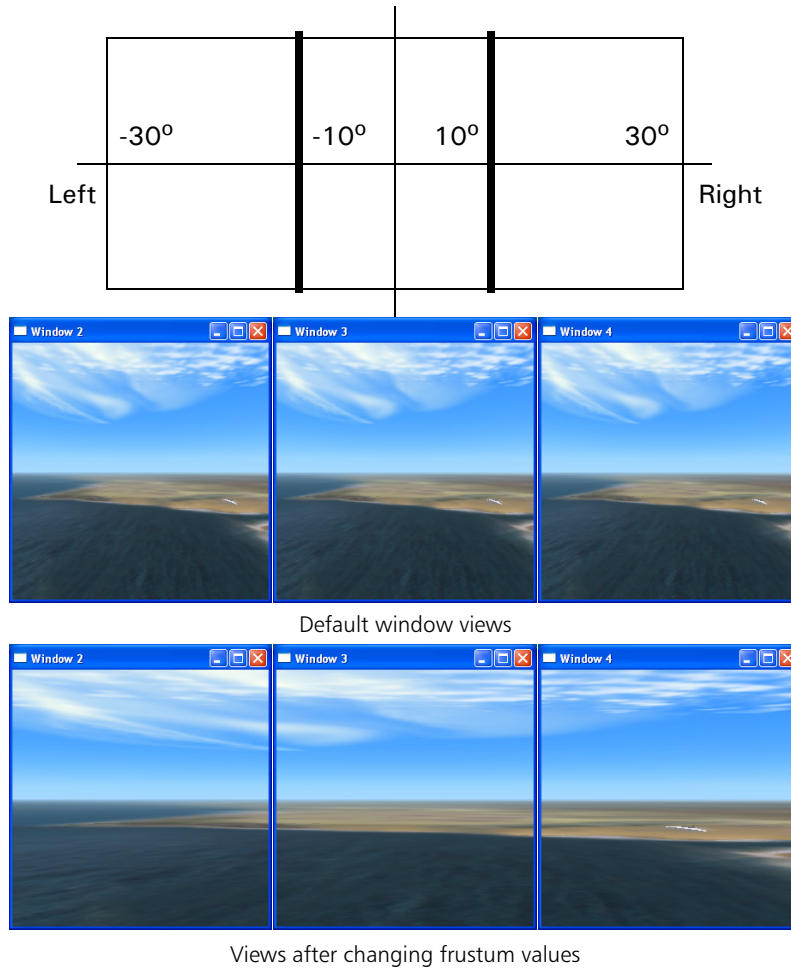


Figure 11-16. Frustum values for three channels

### 11.8.1. Changing the Camera's Position and Orientation Offset

Changing the frustum affects how much of the database is in the view. Changing the camera's position and orientation affects the point from which you look at the view. For example, imagine you are simulating the view from a vehicle, as in vehicle 1 in [Figure 11-17](#). The default camera location is the black circle. If you wanted to show the view as experienced by a gunner riding in the back of the vehicle, or in a turret, you would change the X, Y, and Z Camera Position Offsets in the channel's property list (vehicle 2).  
2).

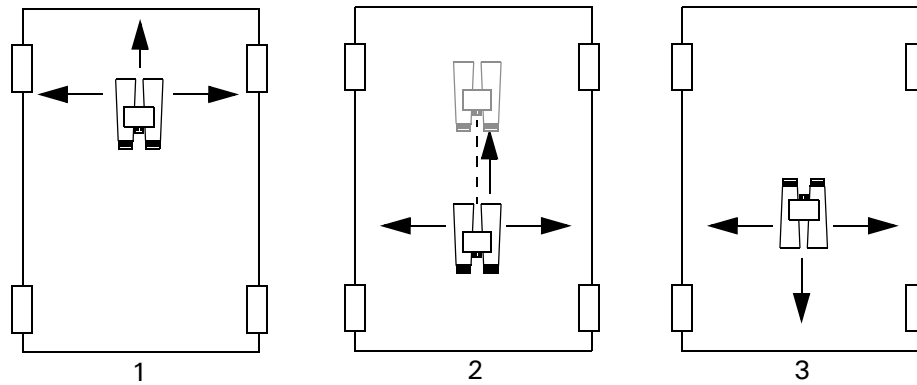
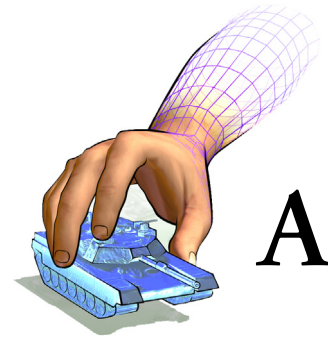


Figure 11-17. Position and orientation offsets

If you wanted the gunner to look out the rear of the vehicle, you would change its orientation, with the Psi (heading), Theta (pitch), and Phi (roll) Camera Orientation Offsets.





## *Editing the Create Menu*

---

This chapter describes how to edit the Create menu manually.

|                                                        |     |
|--------------------------------------------------------|-----|
| The Create Menu.....                                   | A-2 |
| The vrf.ent Configuration File .....                   | A-2 |
| Object Blocks in vrf.ent.....                          | A-4 |
| Reorganizing the Create Menu .....                     | A-4 |
| Editing Menu Items .....                               | A-4 |
| Adding New Entity Types to VR-Forces.....              | A-5 |
| Adding a New Entity or Control Object to vrf.ent ..... | A-5 |
| Editing the Overlay Object Menus.....                  | A-5 |

## A.1. The Create Menu

By default, the Create menu is generated automatically from the *entity.mst* file. However, you can use a customized Create menu if you want to. The recommended way to create a customized Create menu is to use the Entity Editor, as described in “Editing the Create Menu,” on page 5-29. This appendix explains how to edit the Create menu in a text editor.

You can change the Create menu by:

- ♦ Reorganizing entries
- ♦ Editing existing entries
- ♦ Adding new entries.

You modify the Create menu and its sub-menus by editing the following files:

- ♦ *vrf.ent* – configures entities, aggregates, and control objects
- ♦ *overlayMenu.mtl* – configures the overlay object sub-menus.

If you make changes to *vrf.ent*, you may also need to make changes to the object parameter database.

## A.2. The *vrf.ent* Configuration File

The *vrf.ent* file contains the information VR-Forces uses to build the Entity and Aggregates menu options on the Create menu, and the list of control objects. The menu structure is completely configurable. It consists of a set of nested menu blocks. The highest block specifies the name of the menu. The second level specifies entries on the main menu. Nested menu blocks specify entries for submenus. The `create-entity-menu-item` entries specify the objects you can create. [Figure A-1](#) is a highly edited version of the *vrf.ent* file that shows nested menus and an illustration of the menu structure that would be created by the sample *vrf.ent* file.



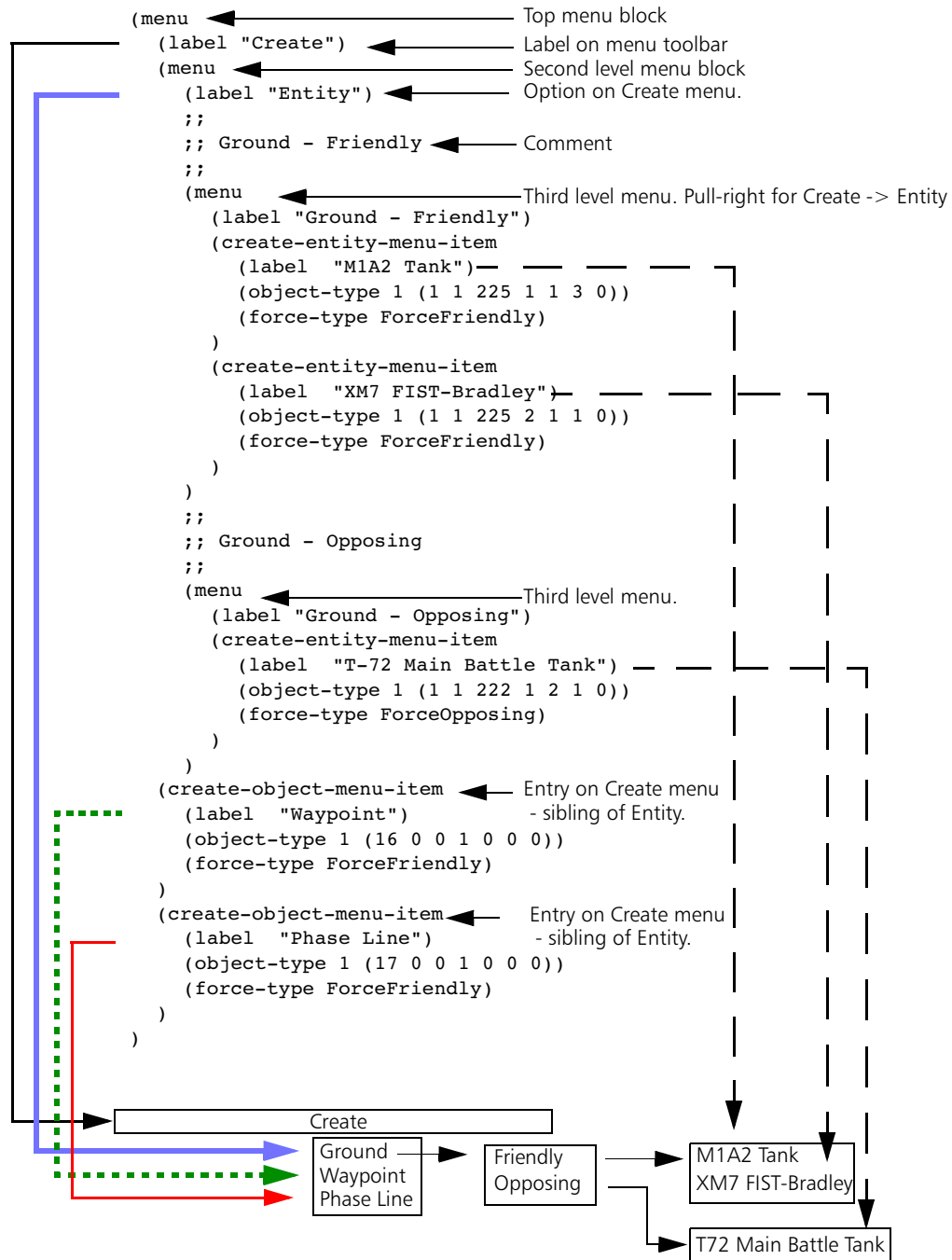


Figure A-1. Abbreviated *vrf.ent* file and menus created

### A.2.1. Object Blocks in *vrf.ent*

The entry for each type of entity that you can create defines a label, object type, and force type, similar to the following:

```
(create-entity-menu-item
  (label "M1A2 Tank")
  (object-type 1 (1 1 225 1 1 3 0))
  (force-type ForceFriendly)
)
```

where:

- ♦ **label** specifies the text that is displayed on the menu.
- ♦ **object-type** is the eight-digit object type for the object. It must match an object-type in the object parameter database using exact match or best match criteria.
- ♦ **force-type** determines the force assigned to a newly created object. The options are:
  - ForceOther
  - ForceNeutral
  - ForceFriendly
  - ForceOpposing
  - ForceUserDefined5 up to ForceUserDefined255.

### A.2.2. Reorganizing the Create Menu

The default *vrf.ent* file organizes entities by force and domain. It organizes aggregates by hierarchical level. If you would prefer to organize entities differently, all you have to do is create menu blocks with your preferred organization and move the appropriate `create-entity-menu-item` blocks to the new menu blocks. For example, you could move the control object entries to a Control Objects sub-menu.

### A.2.3. Editing Menu Items

You can change the labels and force types for entries on the Create Menu by editing the text in *vrf.ent*. You can also change the objects associated with the entries by editing the **object-type**. However, if you change an object type, you must be sure that it matches an entry in the object parameter database. Please see [“Adding New Entity Types to VR-Forces,”](#) on page A-5 for details.

### A.3. Adding New Entity Types to VR-Forces

You can add new entity types to VR-Forces without using the VR-Forces API toolkit.

To add a new entity type, you must:

1. Add the entity to the Create menu.
2. Ensure that the object parameter database has an entry whose object-type matches the object type in *vrf.ent* or meets best-match criteria.

#### A.3.1. Adding a New Entity or Control Object to *vrf.ent*



---

Before making changes to a configuration file, be sure to make a backup copy of the original file.

---

To add a new object to *vrf.ent*:

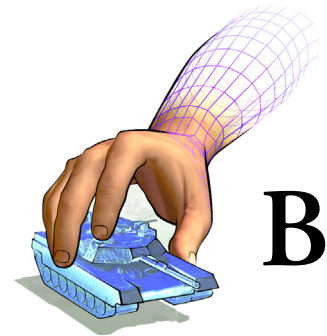
1. Open *vrf.ent* in a text editor.
2. To add an entity (including preconfigured aggregates), add a new **create-entity-menu-item** block to the file. To add a control object, add a new **create-object-menu-item**. Be sure that you match the syntax and parentheses of an existing block. (You could copy a block, paste it at the end of the existing block, and change the entries for the various fields as described in the following steps.)
3. In the **label** field, enter a string for the new entity. This is the text that will be displayed on the menu.
4. In the **object-type** field, enter an 8-digit object-type for the entity. Do not use wildcard values. The object type must match (using the best match method) an object type that is specified in *vrfSim.opd* (or an alternative OPD file.)
5. In the **force-type** field, enter the force type for the entity.
6. Save the file.

The next time you start VR-Forces, the new entity will be listed on the appropriate submenu of the Create menu.

### A.4. Editing the Overlay Object Menus

The layout of the Overlay Objects submenus is configured in the *overlayMenu.mtl* file. For details, please see [“Configuring Overlay Objects,”](#) on page 2-21.





## *Object Parameters*

---

This appendix describes the high-level parameters in the object parameter database. The OPD Editor has context-sensitive help for most parameters and also describes them in the online help system. It also has reference material for local and remote subcomponents and object type enumerations.

|                                                      |      |
|------------------------------------------------------|------|
| <a href="#">The Structure of an Entity Parameter</a> | B-2  |
| <a href="#">The Parameter Type Hierarchy</a>         | B-3  |
| <a href="#">Vrf Base Object Parameters</a>           | B-3  |
| <a href="#">Vrf Object Parameters</a>                | B-7  |
| <a href="#">Moving Object Parameters</a>             | B-8  |
| <a href="#">Parameter Types for Entity Platforms</a> | B-8  |
| <a href="#">Entity Parameters</a>                    | B-8  |
| <a href="#">Object Type Enumerations</a>             | B-9  |
| <a href="#">Object Super-type</a>                    | B-9  |
| <a href="#">AggregateKind</a>                        | B-9  |
| <a href="#">ObjectKind</a>                           | B-10 |
| <a href="#">Aggregate Category</a>                   | B-10 |
| <a href="#">Aggregate Specific</a>                   | B-11 |
| <a href="#">Point Object Category</a>                | B-12 |
| <a href="#">LinearObjectKind</a>                     | B-12 |
| <a href="#">LinearObjectCategory</a>                 | B-12 |
| <a href="#">Interaction</a>                          | B-12 |

## ***B.1. The Structure of an Entity Parameter***

The object parameter database uses MÄK Technologies LISP (MTL). (For details about MTL, please see “[MÄK Technologies Lisp \(MTL\) Files](#),” on page 2-2.) A parameter block has the form:

```
(block_name
  ( parameter_name value)
  ...
  (parameter_name
    (sub-parameter value)
  )
  and so on
)
```

Entries have the following requirements:

- ♦ The block\_name can be anything that you like.
- ♦ The parameter\_name must be exactly as documented (except that in some cases, a parameter may contain an arbitrary length list of sub-parameters, for which the parameter names are not used – these will be noted). Parameter names must not contain spaces, and must not start with a digit (or a dash followed by a digit, for example, -1).
- ♦ The first parameter in any parameter block must be **parameter-type**.
- ♦ You can include comments in the file. A comment begins with a semicolon(;) and continues to the end of the line.
- ♦ All numerical parameters use standard MKS values: meters, kilograms, seconds, radians, liters, and so on.
- ♦ Real constants must begin with a digit and contain a decimal point. For example, 0.5 and 0. are acceptable, but .5 and 0 are not.

## B.2. The Parameter Type Hierarchy

Figure 7-3, in “Parameter Types,” on page 7-9, illustrates the hierarchy of parameter types. In this section, we describe the parameters associated with each **parameter-type**.

### B.2.1. Vrf Base Object Parameters

All object parameters inherit from the parameter type **vrf-base-object-param**. The parameters associated with **vrf-base-object-param** are potentially applicable to any type of object (ground, rotary-wing, control object, and so on). Please see the General Parameters topic in the OPD Editor online help for parameter descriptions. In addition to parameters, each object has an object type and parameter type, as described in [Table B-1](#).

Table B-1: Object and parameter type

| Parameter      | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| object-type    | 8-digit object type. Not actually a parameter. Specifies the key used to look up the entry. (Required) For more information, please see “Object Types,” on page 7-6.                                                                                                                                                                                                                                                                                                      |
| parameter-type | <p>One of the following (string):</p> <ul style="list-style-type: none"> <li>♦ "vrf-base-object-param"</li> <li>♦ "vrf-object-param"</li> <li>♦ "moving-object-param"</li> <li>♦ "fixed-wing-param"</li> <li>♦ "ground-vehicle-param"</li> <li>♦ "rotary-wing-entity-param"</li> <li>♦ "missile-param"</li> <li>♦ "surface-entity-param"</li> <li>♦ "human-param"</li> <li>♦ "cultural-feature-param".</li> </ul> <p>(Required) The value must be in quotation marks.</p> |

### Local Objects Subcomponent Parameters

The `local-objects` parameter specifies the types of subcomponents to create for locally simulated objects. You can omit optional items, or set them with an empty string `""`. [Table B-2](#) lists the subcomponent parameters and their possible values.

Table B-2: Local-objects subcomponent parameters (Sheet 1 of 2)

| Subcomponent parameter        | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|-------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>state-repository</code> | <p>(Required) Maintains the state information for the object. An object that has components (sensors, controllers, and actuators) for simulation sets and retrieves information in the repository. Some simulation components expect a specific kind of state repository. In general, you will want to specify the state repository type that most closely matches the kind of entity you expect to simulate. May be one of the following types:</p> <ul style="list-style-type: none"><li>♦ <code>"vrf-object-base-state-repository"</code>: Used for the base object in the object parameter database.</li><li>♦ <code>"vrf-object-state-repository"</code>: Derived from <code>vrf-object-base-state-repository</code>, it adds the concepts of resources, sector of responsibility, engagement rules, and targeting. Because there are more entity-specific state repository types that are derived from this type, this kind of repository is not typically specified.</li><li>♦ <code>"vrf-overlay-object-state-repository"</code>: For graphical objects.</li><li>♦ <code>"vrf-moving-object-state-repository"</code>: For any object that you expect to simulate movement. Pseudo-aggregate entities have this kind of state repository. The more entity-specific repository types that follow inherit this kind of state repository, so choose one of those when simulating a specific type of individual entity.</li><li>♦ <code>"ground-entity-state-repository"</code>: For ground vehicles, such as tanks, trucks, dismounted infantry.</li><li>♦ <code>"rotary-wing-entity-state-repository"</code>: For rotary-wing aircraft.</li><li>♦ <code>"fixed-wing-entity-state-repository"</code>: For fixed-wing aircraft.</li><li>♦ <code>"missile-entity-state-repository"</code>: For missile and rocket entities.</li><li>♦ <code>"surface-entity-state-repository"</code>: For surface entities.</li><li>♦ <code>"human-state-repository"</code>: For lifeform entities.</li><li>♦ <code>"cultural-feature-state-repository"</code>: For cultural features that are created as entities.</li></ul> |



Table B-2: Local-objects subcomponent parameters (Sheet 2 of 2)

| Subcomponent parameter                 | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| network-interface                      | <p>(Optional) The network interface handles the publication of the object over the network. It acts as an interface between an entity's state repository and its network (VR-Link) representation. Choose one that most closely matches the kind of entity you are simulating. May be one of the following types:</p> <ul style="list-style-type: none"> <li>♦ "aggregate-local-entity-net-interface": For aggregate entities.</li> <li>♦ "pseudo-aggregate-local-entity-net-interface": For aggregate entities that have subordinate entities that are separately simulated.</li> <li>♦ "individual-local-entity-net-interface": For individual platform entities, such as tanks, aircraft, dismounted infantry.</li> <li>♦ "local-environmental-net-interface": For entities that are best represented as an environmental process. Non-entity objects such as control objects (waypoints, routes, areas) use this type of network interface.</li> </ul> |
| net-interface-min-tick-period          | (Optional) Specifies the shortest period of time, in seconds, that can elapse between ticks of an entity's network interface.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| net-interface-min-tick-period-variance | (Optional) Specifies an amount within which the net-interface-min-tick-period can vary. Defined as:<br>plus or minus $\text{rand}(\text{net-interface-min-tick-period}/2)$                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| component-manager                      | <p>(Optional) A component manager contains the lists of sensor, controller, and actuator components used to simulate the behavior of a given object (typically entities).</p> <ul style="list-style-type: none"> <li>♦ "component-manager".1</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| plan-manager                           | (Optional) The entity plan manager. If you want an entity to be able to execute a plan, it must have a plan manager.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| task-manager                           | <p>(Optional) If a component manager is specified for an object, then you must specify a task manager. The task manager works in conjunction with the component manager to manage the processing and message handling associated with execution of a task.</p> <ul style="list-style-type: none"> <li>♦ "local-task-manager": For individual entities or true aggregates.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

## Remote Object Subcomponent Parameters

The `remote-objects` parameter specifies the types of sub-components to create for remote objects. Remote objects are those objects simulated by other applications on the network (including other VR-Forces applications). You can omit optional items, or set them with an empty string `""`. [Table B-3](#) lists the subcomponent parameters and their possible values.

Table B-3: Remote object subcomponent parameters (Sheet 1 of 2)

| Subcomponent parameter        | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>state-repository</code> | <p>(Required) Maintains the state information for the object. Unlike the local-object entry, it is not as important to specify the closest match based on type. For entities, the <code>"vrf-moving-object-state-repository"</code> is sufficient. A more specific state repository may be specified. However, some of the entity-specific data may not be set. May be one of the following types:</p> <ul style="list-style-type: none"><li>♦ <code>"vrf-object-state-repository"</code>: Derived from <code>vrf-object-base-state-repository</code>, it adds the concepts of resources, sector of responsibility, engagement rules, and targeting. Because there are more entity-specific state repository types that are derived from this type, this kind of repository is not typically specified.</li><li>♦ <code>"vrf-overlay-object-state-repository"</code>: For graphical objects.</li><li>♦ <code>"vrf-moving-object-state-repository"</code>: For any object that you expect to simulate movement, including individual and aggregate entities.</li><li>♦ <code>"ground-entity-state-repository"</code>: For ground vehicles, such as tanks, trucks, dismounted infantry.</li><li>♦ <code>"rotary-wing-entity-state-repository"</code>: For rotary-wing aircraft.</li><li>♦ <code>"fixed-wing-entity-state-repository"</code>: For fixed-wing aircraft.</li><li>♦ <code>"missile-entity-state-repository"</code>: For missile and rocket entities.</li><li>♦ <code>"human-state-repository"</code>: For lifeform entities.</li><li>♦ <code>"cultural-feature-state-repository"</code>: For cultural features that are created as entities.</li><li>♦ <code>"surface-entity-state-repository"</code>: For surface entities.</li></ul> |

Table B-3: Remote object subcomponent parameters (Sheet 2 of 2)

| Subcomponent parameter                 | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| network-interface                      | <p>(Optional) The network interface is the interface between the remote network (VR-Link) representation of the object and its state repository. It acts as an interface between an entities state repository and its network (VR-Link) representation. Choose one that most closely matches the kind of entity you are simulating. May be one of the following types:</p> <ul style="list-style-type: none"> <li>• "vrf- aggregate -remote-entity-net-interface": For remote aggregate entities.</li> <li>• "vrf-individual-remote-entity-net-interface": For individual platform entities, such as tanks, aircraft, dismounted infantry.</li> <li>• "vrf-remote-environmental-net-interface": For entities that are best represented as an environmental process. Non-entity objects such as control objects (waypoints, routes, areas) use this type of network interface.</li> </ul> |
| net-interface-min-tick-period          | (Optional) Specifies the shortest period of time, in seconds, that can elapse between ticks of an entity's network interface.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| net-interface-min-tick-period-variance | (Optional) Specifies an amount within which the net-interface-min-tick-period can vary. Defined as:<br>plus or minus rand(net-interface-min-tick-period/2)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| plan-manager                           | Do not specify a plan manager for remote objects. (Give it the value "" (empty string). If you specify a plan manager, it is ignored.)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| component-manager                      | Do not specify a component manager for remote objects. (Give it the value "" (empty string). If you specify a component manager, it is ignored.)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| task-manager                           | (Optional) Do not specify a task manager for remote objects. (Give it the value "" (empty string). If you specify a task manager, it is ignored.)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

## B.2.2. Vrf Object Parameters

The parameter type **vrf-object-param** inherits from **vrf-base-object-param** and is the parent of the moving object parameter type. Vrf object parameters include resources, engagement rules, and target acquisition time. The **vrf-object-param** parameter type represents an intermediate class in the parameter hierarchy. It is not usually used to represent objects in VR-Forces. You will probably want to instantiate a derived type, (for aggregates and entities) or you will want to specify vrf base object parameters (for case of control objects.)

Control objects have the parameter type **vrf-base-object-param**.

### B.2.3. Moving Object Parameters

Moving objects have the parameter type `moving-object-param`. Moving objects inherit parameters from vrf objects and vrf base objects. The `moving-object-param` parameter type is used for objects that are capable of movement, such as individual entities and aggregates. Parameters include support points, maximum speed, maximum reverse speed, turning radius, maximum slope, ordered speed, maximum acceleration, maximum deceleration, and fuel efficiency. Aggregates are moving objects and have this parameter type. All of the platform-specific parameter types derive from this one.

### B.2.4. Parameter Types for Entity Platforms

[Table B-4](#) lists the parameter types used by each entity type.

Table B-4: Parameter types for entities

| Platform         | Parameter Type                        | Inherits from                    |
|------------------|---------------------------------------|----------------------------------|
| Ground vehicle   | <code>ground-vehicle-param</code>     | <code>moving-object-param</code> |
| Fixed-wing       | <code>fixed-wing-entity-param</code>  | <code>moving-object-param</code> |
| Human            | <code>human-param</code>              | <code>moving-object-param</code> |
| Rotary-wing      | <code>rotary-wing-entity-param</code> | <code>moving-object-param</code> |
| Surface          | <code>surface-entity-param</code>     | <code>moving-object-param</code> |
| Cultural feature | <code>cultural-feature-param</code>   | <code>moving-object-param</code> |
| Missile          | <code>missile-param</code>            | <code>moving-object-param</code> |
| Pseudo-aggregate | <code>moving-object-param</code>      | <code>vrf-object-param</code>    |

## B.3. Entity Parameters

For descriptions of entity parameters, please see the descriptions of individual components in the OPD Editor online help.

## B.4. Object Type Enumerations

This section lists the object type enumerations as listed in the header file *objTypeEnums.h*. It is presented here primarily for customers who have purchased VR-Forces, but not the toolkit. Therefore we have departed from the typical header file format for readability.

### B.4.1. Object Super-type

The header file *objTypeEnums.h* contains the VR-Forces extensions to the DIS/RPR FOM enumerations. The enumerations specify individual, aggregate, and non-entity object types. The sim object super-type is an enumeration used in the *DtObjectType* class (an extension of the *DtEntityType* class). It extends the DIS/RPR FOM entity type enumeration by specifying the sort of object associated with the *DtEntityType*. Because entity types are not guaranteed to be unique across all of the possible individual and aggregate types, the extra enumerated field provides a way to uniquely identify all objects in a VR-Forces simulation.

VR-Forces uses the value 3 in the object super-type for all of the aggregates that it simulates.

|                 |   |
|-----------------|---|
| Other           | 0 |
| Individual      | 1 |
| Aggregate       | 2 |
| PseudoAggregate | 3 |
| TrueAggregate   | 4 |
| Interaction     | 5 |

### B.4.2. AggregateKind

The enumeration for *DtAggregateKind* should start with 0, but VR-Forces starts with 10, to ensure that entity types will be unique.

|                     |    |
|---------------------|----|
| Other               | 10 |
| MilitaryHierarchy   | 11 |
| CommonType          | 12 |
| CommonMission       | 13 |
| SimilarCapabilities | 14 |
| CommonLocation      | 15 |
| DtSimArtPartKind    | 20 |

### B.4.3. ObjectKind

Extension of entity kind to represent non-entity objects, such as control objects (control points, routes, phase lines, obstacles, etc.)

|                    |    |
|--------------------|----|
| DtPointObjectKind  | 16 |
| DtLinearObjectKind | 17 |
| DtArealObjectKind  | 18 |
| DtCircleObjectKind | 19 |
| DtTextObjectKind   | 20 |

### B.4.4. Aggregate Category

VR-Forces extends the enumeration to include *DtAggregateCategoryForce*, which represents the force level. All of the entries from Force through the end of the list, are extensions to the standard.

|                   |    |
|-------------------|----|
| Other             | 0  |
| IndividualVehicle | 1  |
| Element           | 2  |
| Platoon           | 3  |
| Battery           | 4  |
| Company           | 5  |
| Battalion         | 6  |
| Regiment          | 7  |
| Brigade           | 8  |
| Division          | 9  |
| Corps             | 10 |
| Force             | 11 |
| Team              | 12 |
| Squad             | 13 |
| Section           | 14 |

### Aggregate Subcategory

The Subcategory field enumerations for Kind field = Military Hierarchy specify the Echelon Type of the aggregate. Not all of the echelon types are applicable to each of the categories.

In the following list, ForceFriendly, ForceOpposing, ForceNeutral, and ForceOther are extensions to the DIS/RPR FOM enumerations.

|                        |    |                          |    |
|------------------------|----|--------------------------|----|
| Other                  | 0  | ArmyAviationFixedWing    | 13 |
| CavalryTroop           | 1  | ArmyAviationRotaryWing   | 14 |
| Armor                  | 2  | ArmyAttackHelicopter     | 15 |
| Infantry               | 3  | AirCavalry               | 16 |
| MechanizedInfantry     | 4  | ArmorHeavyTaskForce      | 17 |
| Cavalry                | 5  | MotorizedRifle           | 18 |
| ArmoredCavalry         | 6  | MechanizedHeavyTaskForce | 19 |
| Artillery              | 7  | CommandPost              | 20 |
| SelfPropelledArtillery | 8  | CEWI                     | 21 |
| CloseAirSupport        | 9  | TankOnly                 | 22 |
| Engineer               | 10 | ForceFriendly            | 23 |
| AirDefenseArtillery    | 11 | ForceOpposing            | 24 |
| AntiTank               | 12 |                          |    |

### B.4.5. Aggregate Specific

The Specific field enumerations for Kind field = Military Hierarchy specify whether the aggregate unit contains a headquarters, and is enumerated as follows.

|                      |   |
|----------------------|---|
| Noheadquarters       | 0 |
| ContainsHeadquarters | 1 |

#### B.4.6. Point Object Category

Subcategories for use with the Kind field = *DtPointObjectKind*.

|              |   |
|--------------|---|
| Other        | 0 |
| ControlPoint | 1 |
| TargetPoint  | 2 |
| Marker       | 3 |

#### B.4.7. LinearObjectKind

Subcategories for use with the Kind field = *DtLinearObjectKind*.

|           |   |
|-----------|---|
| Other     | 0 |
| PhaseLine | 1 |
| Route     | 2 |

#### B.4.8. LinearObjectCategory

Subcategories for use with the Kind field = *DtArealObjectKind*.

|             |   |
|-------------|---|
| Other       | 0 |
| ControlArea | 1 |
| Obstacle    | 2 |

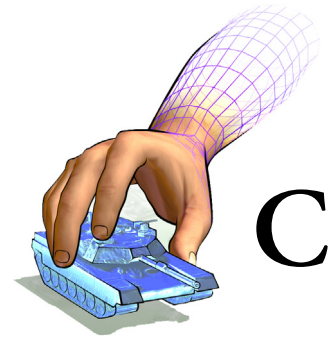
ArealObjectCategory

#### B.4.9. Interaction

Kinds for Supertype interaction.

|          |   |
|----------|---|
| Other    | 0 |
| Fire     | 1 |
| Detonate | 2 |





## ***Creating and Editing Key Maps***

---

The 3D Front End comes with key maps that we think will meet the needs of most the 3D Front End users. However, you can edit the default keyboard maps and create new mappings of your own with the Key Map Editor.

|                                                        |                      |
|--------------------------------------------------------|----------------------|
| <a href="#">The Key Map Editor.....</a>                | <a href="#">C-2</a>  |
| <a href="#">Binary Key Mappings .....</a>              | <a href="#">C-2</a>  |
| <a href="#">Editing a Key Map .....</a>                | <a href="#">C-3</a>  |
| <a href="#">Adding a Key Mapping .....</a>             | <a href="#">C-4</a>  |
| <a href="#">Changing a Key Mapping.....</a>            | <a href="#">C-5</a>  |
| <a href="#">Deleting a Key Mapping .....</a>           | <a href="#">C-5</a>  |
| <a href="#">Using Combined Key Mappings .....</a>      | <a href="#">C-6</a>  |
| <a href="#">Filtering the Function List .....</a>      | <a href="#">C-6</a>  |
| <a href="#">Creating a Key Map .....</a>               | <a href="#">C-7</a>  |
| <a href="#">Deleting a Key Map .....</a>               | <a href="#">C-7</a>  |
| <a href="#">Loading a Key Map.....</a>                 | <a href="#">C-7</a>  |
| <a href="#">Keyboard Mappings.....</a>                 | <a href="#">C-7</a>  |
| <a href="#">Mouse Navigation Button Mappings .....</a> | <a href="#">C-11</a> |

## C.1. The Key Map Editor

The Key Map Editor (Figure C-1) lists all of the functions that you can control from the keyboard and the keys to which they are mapped. Some functions are mapped to more than one key or key combination. A key can be mapped to more than one function (combined mappings). You can:

- Edit a function's key mappings
- Add new mappings
- Delete mappings
- Save the changes to the current key map
- Save the changes to a different key map.

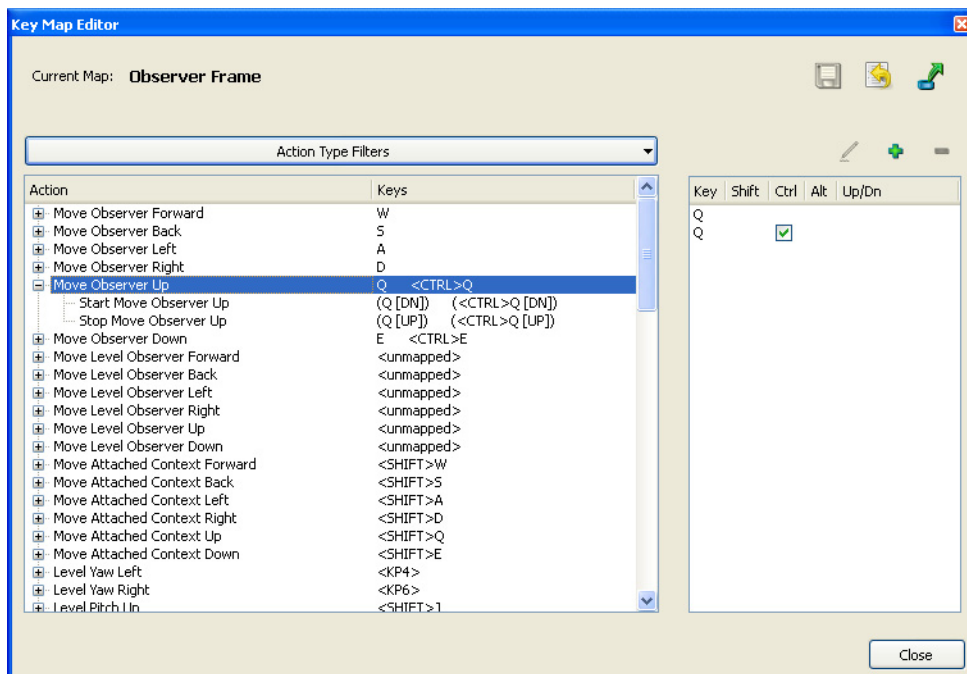


Figure C-1. Key Map Editor

### C.1.1. Binary Key Mappings

Many navigation functions require you to press and hold a key. For example, you press and hold **W** to move the observer forward. (By contrast some keyboard actions are just toggles, such as pressing period to attach to the next entity.) For these actions, the 3D Front End needs to keep track of when the key is pressed down (start the action) and when it is released (stop the action). To a human user, a function like Move Observer Forward is experienced as one action. But the computer experiences it as two actions. Therefore, key mappings must take this into account. The Key Map Editor shows the key mappings for these functions as pairs (Figure C-1).

## C.2. Editing a Key Map

The Key Map Editor lists all of the the 3D Front End functions that you can control from the keyboard.

To edit a key map:

1. In the Observer Control Panel, in the Keyboard Mapping drop-down list (Figure C-2), select the key map you want to edit.

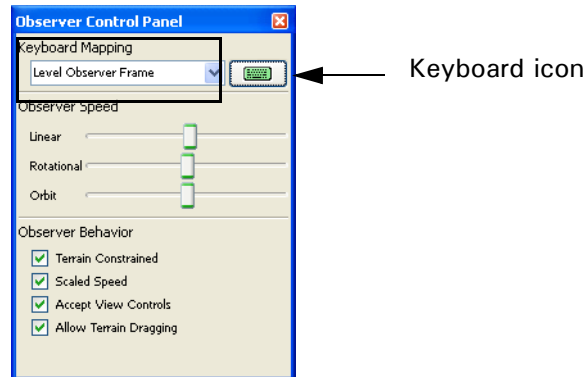


Figure C-2. Keyboard Mapping drop-down list and keyboard icon

2. Click the keyboard icon and choose **Edit Key Map**. The Key Map Editor opens (Figure C-1).
3. Select the function whose key mappings you want to edit.
4. Edit the key mappings as described in:
  - “Adding a Key Mapping,” on page C-4
  - “Changing a Key Mapping,” on page C-5
  - “Deleting a Key Mapping,” on page C-5.
5. Save the changes to the current file or to a different key map file.
6. Click OK.

### C.2.1. Adding a Key Mapping

If a function is not mapped to a key, it is marked <unmapped>. You can add a mapping to unmapped functions and you can add additional key mappings to functions that are already mapped. If you add a key mapping to a binary function, the Key Map Editor automatically applies it to the child (UP and DN) functions. You can add key mappings to the child functions of binary mappings. When you add a key mapping to a child function, it is not applied to its paired function or the parent function.

To add a key mapping:

1. Select the function to which you want to add a key mapping.
2. Click the Add Key Mapping icon (plus sign) (Figure C-3). The Key Entry dialog box opens (Figure C-4).

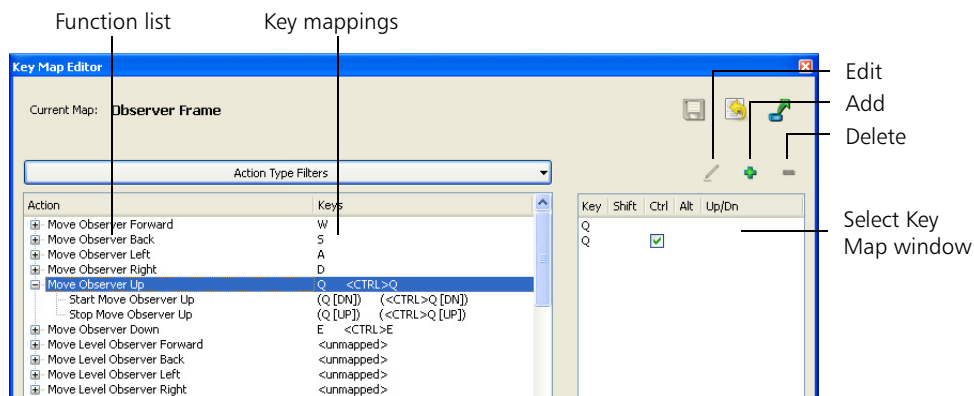


Figure C-3. Key Map Editor

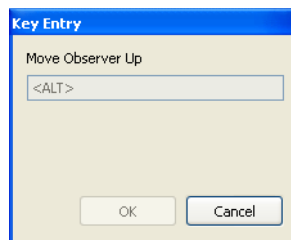


Figure C-4. Key Entry dialog box

3. On the keyboard, press the key or key combination (such as **Ctrl**+key or **Shift**+key) that you want to map to this function.
4. Click OK. (If you choose a key combination that is already in use, the 3D Front End notifies you and asks how to proceed. For details about duplicate key mappings, please see “Using Combined Key Mappings,” on page C-6.) The mapping is added to the function and is listed in the Keys column of the function list and in the Select Key Map window.

### C.2.2. Changing a Key Mapping

If a key mapping is assigned to a binary function, the key is automatically assigned to the child functions. You cannot edit the key mappings for the child functions. You can only change the parent's key mapping. However, if you add a key mapping to a child function, you can edit it.

To change a key mapping:

1. Select the function whose key mapping you want to change. Its key mappings get listed in the Select Key Map window (Figure C-3).
2. In the Select Key Map window, select the key mapping that you want to change.
3. Click the Edit Key Mapping icon (pencil). The Key Entry dialog box opens (Figure C-5). It shows the current key mapping.

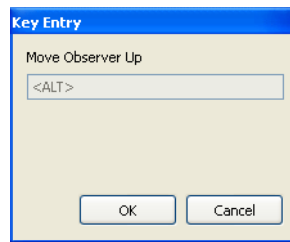


Figure C-5. Key Entry dialog box

4. On the keyboard, press the key or key combination (such as **Ctrl**+key or **Shift**+key) that you want to map to this function.
5. Click OK. The mapping is changed.

### C.2.3. Deleting a Key Mapping

To delete a key mapping:

1. Select the function whose key mapping you want to delete. Its key mappings get listed in the Select Key Map window (Figure C-3).
2. In the Select Key Map window, select the key mapping that you want to delete.
3. Click the Delete Mapping icon (minus sign) (Figure C-3). The mapping is deleted. The function list is updated.

### C.3. Using Combined Key Mappings

You can assign a key combination to more than one function. This allows you to combine the behaviors of two or more functions into one key combination. For example, the **Z** key detaches the observer from the attached entity. The **Space Bar** resets the observer view to the default view. Suppose that whenever you detach from an entity, you want the observer to return to the default view. You could map the Reset function to **Z**. **Z** is now mapped to two functions. When you press **Z**, the 3D Front End detaches from the attached entity and moves the observer to the default view.

To manage combined key mappings:

1. Add the appropriate key mapping to a function.
2. Click OK. The dialog box opens (Figure C-6). It lists the functions that already use this key mapping.

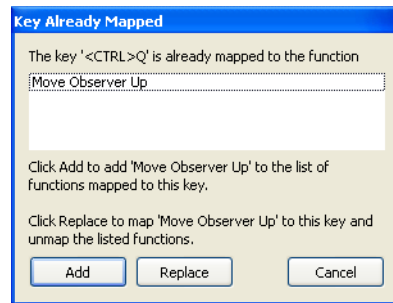


Figure C-6. Key Already Mapped dialog box

3. If the functions listed are the ones with which you want to share this key mapping, click Add. The key mapping is added to the function.
4. If you do not want to share this key mapping with other functions and want to use it for the function that you are editing, click Replace. The key mapping is applied to the function you are editing and removed from the functions listed in the dialog box.
5. If you do not want to remove the key mapping from existing functions and do not want to share it with the other functions, click Cancel. Presumably you wanted a unique key mapping for the function you are editing and did not realize you chose one that was already in use. Choose a different key mapping for this function.

### C.4. Filtering the Function List

You can filter the function list in the Key Map Editor. Functions are grouped thematically, similar to Table C-1 through Table C-7.

- To filter the function list, select an option on the Action Type Filters drop-down list.

## C.5. Creating a Key Map

To create a key map:

1. In the Observer Control Panel ([Figure C-2](#)), click the keyboard icon and choose **New Key Map**. The Key Map Editor opens. It lists all of the movement options. None of them are mapped to keys.
2. Add key mappings for the movement actions as described in “[Adding a Key Mapping](#),” on page C-4.
3. Click the Save icon. The Save Key Map dialog box opens.
4. Type a name for the key map.
5. Click Save.

## C.6. Deleting a Key Map

To delete a key map

1. In the Observer Control Panel, in the Keyboard Mapping drop-down list ([Figure C-2](#)), select the key map you want to delete.
2. Click the keyboard icon and choose **Delete Key Map**.
3. Confirm that you want to delete the selected key map.

## C.7. Loading a Key Map

To load a key map:

1. In the Observer Control Panel, click the keyboard icon and choose **Load Key Map**. The Load Key Map dialog box opens.
2. Select the key map file you want to load.
3. Click Open.

## C.8. Keyboard Mappings

The tables in this section list the default key mappings for navigation and miscellaneous keyboard control of the 3D Front End. You can change these mappings using the Key Map Editor.



To use the keypad for navigation, NumLock must be on.

---

Table C-1: Observer Frame movement key mappings

| Observer Movement              | Key                |
|--------------------------------|--------------------|
| Move forward.                  | <b>w</b>           |
| Move back.                     | <b>s</b>           |
| Move right.                    | <b>d</b>           |
| Move left.                     | <b>a</b>           |
| Move up.                       | <b>q, Ctrl + q</b> |
| Move down.                     | <b>e, Ctrl + e</b> |
| Move attached context forward. | <b>Shift + w</b>   |
| Move attached context back.    | <b>Shift + s</b>   |
| Move attached context right.   | <b>Shift + d</b>   |
| Move attached context left.    | <b>Shift + a</b>   |
| Move attached context up.      | <b>Shift + q</b>   |
| Move attached context down.    | <b>Shift + e</b>   |

Table C-2: Observer Frame rotation key mappings

| Observer Movement | Key              |
|-------------------|------------------|
| Level yaw right.  | Keypad <b>6</b>  |
| Level yaw left.   | Keypad <b>4</b>  |
| Pitch up.         | Keypad <b>8</b>  |
| Pitch down.       | Keypad <b>2</b>  |
| Roll right.       | Keypad <b>9</b>  |
| Roll left.        | Keypad <b>3</b>  |
| Yaw right.        | <b>Shift + l</b> |
| Yaw left.         | <b>Shift + h</b> |
| Level pitch up.   | <b>Shift + j</b> |
| Level pitch down. | <b>Shift + k</b> |
| Level roll right. | <b>Shift + i</b> |
| Level roll left.  | <b>Shift + m</b> |



Table C-3: Observer Frame orbit key mappings

| Observer Movement | Key                          |
|-------------------|------------------------------|
| Orbit right.      | <b>Ctrl + d, Right arrow</b> |
| Orbit left.       | <b>Ctrl + a, Left arrow</b>  |
| Orbit up.         | <b>Ctrl + w, Up arrow</b>    |
| Orbit down.       | <b>Ctrl + s, Down arrow</b>  |

Table C-4: Observer Frame attachment key mappings

| Observer Movement                                               | Key                   |
|-----------------------------------------------------------------|-----------------------|
| Detach.                                                         | <b>z</b>              |
| Toggle attach mode.                                             | <b>Shift + z</b>      |
| Attach to the next entity allowed by the attachment filter.     | Period (.)            |
| Attach to next entity or prop.                                  | Greater than sign (>) |
| Attach to the previous entity allowed by the attachment filter. | Comma (,)             |
| Attach to previous entity or prop.                              | Less than sign (<)    |

Table C-5: Observer Frame gain control key mappings

| Observer Movement      | Key                   |
|------------------------|-----------------------|
| Increase linear speed. | Keypad 7              |
| Decrease linear speed. | Keypad 1              |
| Increase orbit speed.  | Keypad Plus sign (+)  |
| Decrease orbit speed.  | Keypad Minus sign (–) |

Table C-6: Observer Frame camera control key mappings

| Observer Movement | Key                    |
|-------------------|------------------------|
| Look right.       | <b>Ctrl + Keypad 6</b> |
| Look left.        | <b>Ctrl + Keypad 4</b> |
| Look up.          | <b>Ctrl + Keypad 8</b> |

Table C-6: Observer Frame camera control key mappings

| Observer Movement | Key                           |
|-------------------|-------------------------------|
| Look down.        | <b>Ctrl</b> + Keypad <b>2</b> |
| Look tilt right.  | <b>Ctrl</b> + Keypad <b>9</b> |
| Look tilt left.   | <b>Ctrl</b> + Keypad <b>3</b> |

Table C-7: Observer Frame miscellaneous key mappings

| Observer Movement                                                                                                                                               | Key                     |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------|
| Level the observer.                                                                                                                                             | Keypad <b>5</b>         |
| Reset the camera.                                                                                                                                               | Keypad <b>0</b>         |
| Reset the observer:<br>In absolute mode, go to the center of the scene.<br>If attached, reset the attach position and orientation.                              | <b>Space</b>            |
| Suppress constrain to terrain.                                                                                                                                  | <b>x</b>                |
| Toggle speed scaling on or off.                                                                                                                                 | Keypad asterisk (*)     |
| Toggle constrain to terrain on or off.                                                                                                                          | <b>/</b>                |
| Clear the current movement mode. This key is an escape mechanism if keys are not behaving as you expect. This is typically due to a loss of the window context. | <b>c</b>                |
| Toggle view controls on or off.                                                                                                                                 | <b>Shift</b> + <b>c</b> |
| Toggle entity labels on or off.                                                                                                                                 | <b>1</b>                |

Table C-8: Level Observer Frame movement key mappings

| Observer Movement            | Key                               |
|------------------------------|-----------------------------------|
| Move forward level observer. | <b>w</b>                          |
| Move back level observer.    | <b>s</b>                          |
| Move right level observer.   | <b>d</b>                          |
| Move left level observer.    | <b>a</b>                          |
| Move up level observer.      | <b>q</b> , <b>Ctrl</b> + <b>q</b> |
| Move down level observer.    | <b>e</b> , <b>Ctrl</b> + <b>e</b> |

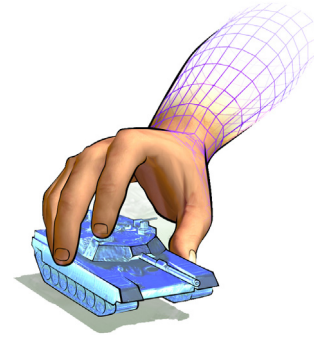
## C.9. Mouse Navigation Button Mappings

Table C-9 lists mouse button mappings for navigation. These are the default mappings. You can change them using the Key Map Editor.

Table C-9: Mouse button mappings

| Navigation Action                                  | Mouse Action                            | Context                   |
|----------------------------------------------------|-----------------------------------------|---------------------------|
| Attach to the selected entity.                     | <b>Shift</b> + Left-click               | Entity                    |
| Attach to the selected prop.                       | <b>Ctrl</b> + <b>Shift</b> + Left-click | Prop                      |
| Deselect all.                                      | Left-click                              | Terrain or sky            |
| Drag heading and pitch.                            | Right Drag                              |                           |
| Drag terrain.                                      | Left-click + drag                       | Terrain or prop           |
| Object remains selected, deselect all others.      | Left-click                              | Selected entity or prop   |
| Orbit drag terrain.                                | <b>Ctrl</b> + Left-click + drag         | Terrain or prop           |
| Select entity or prop.                             | <b>Ctrl</b> + Left-click                | Unselected entity or prop |
| Select object and unselect all others.             | Left-click                              | Unselected entity or prop |
| Teleport 90% of distance to the selected location. | <b>Shift</b> + Left-click               | Terrain                   |
| Teleport 90% of distance to the selected prop.     | <b>Shift</b> + Left-click               | Prop                      |
| Teleport 90% of distance to the selected entity.   | <b>Ctrl</b> + <b>Shift</b> + Left-click | Entity                    |
| Unselect.                                          | <b>Ctrl</b> + Left-click                | Selected entity or prop   |
| Drag the view.                                     | Middle mouse button drag                |                           |
| <b>Mouse wheel while observer is not moving</b>    |                                         |                           |
| Scroll forward.                                    | Mouse wheel forward                     |                           |
| Scroll back.                                       | Mouse wheel back                        |                           |
| <b>Mouse wheel while moving observer</b>           |                                         |                           |
| Increment linear speed.                            | Mouse wheel forward                     |                           |
| Decrement linear speed.                            | Mouse wheel back                        |                           |
| Increment linear speed.                            | <b>Ctrl</b> + mouse wheel backward      |                           |
| Decrement linear speed.                            | <b>Ctrl</b> + mouse wheel forward       |                           |
| Scroll level forward.                              | <b>Shift</b> + mouse wheel forward      |                           |
| Scroll level backward.                             | <b>Shift</b> + mouse wheel backward     |                           |





# Index

## Numerics

-1 (wildcard) 7-8  
 3D Front End, performance 1-15  
 3D model 5-14

## A

--targetFrameRate command-line option 1-16  
 absolute, pathname 3-6  
 Add Soil Type dialog box 6-11  
 adding  
   category 5-34  
   channel 11-6  
   component to object in OPD Editor 6-8  
   entities to Create menu A-5  
   entity to object parameter database 7-17  
   formation 5-21  
   ingress or egress points 8-23  
   key mapping C-4  
   model mapping 10-5  
   new preconfigured aggregate 7-25  
   object in OPD Editor 6-7  
   parameter  
     model definition 10-11  
     model definition schema 10-17  
   scenarios to Scenario Merge project 4-4  
   sensor signature 8-5  
   sensor type 8-5  
   soil type parameter 6-11  
   subordinate object 6-10  
   subordinate to aggregate 5-19  
   symbol set 5-10  
   system to entity 5-26  
   window 11-4

additional-ops parameter 3-6  
 addMulticastAddr parameter 2-13, 2-17  
 addSymbolMapFile parameter 2-5, 2-18  
 advisory message, enabling and disabling 2-13, 2-15, 2-17  
 aggregate B-8  
   adding new preconfigured type 7-25  
   adding preconfigured to object parameter database 7-23  
   bounding box, Entity Editor 5-22  
   connection mode 2-23  
   creating 5-17  
   editing 5-17  
   editing preconfigured 7-24  
   formation  
     adding 5-21  
     automatic layout 5-24  
     copying 5-23  
     default 5-22  
     editing 5-17  
     manual layout 5-24  
     removing 5-22  
     renaming 5-22  
     snap to grid 5-24  
   formations 8-14  
   parameter-type 7-23  
   subordinate  
     adding 5-19  
     changing order 5-20  
     editing 5-21  
     removing 5-20  
     replacing 5-20  
 aggregate-formation-controller parameter 8-14  
 aggregatesFilename parameter 2-5  
 all, connection mode 2-23

**all-tasks-exclusive** parameter 8-19  
**ammo** parameter 8-9  
**ammo select** table 8-9  
**angle, damage** 8-12  
**angle-of-incidence** parameter 8-12  
**animated**  
     icons 2-4  
     image, configuring symbols for 2-19  
**animation, configuring symbol map** 2-19  
**API, scenario merge** 4-2  
**appearance**  
     DI-Guy, random 5-15  
     specifying for DI-Guy model 5-15  
**appearance attribute, specifying configuration file** 2-5  
**appearanceFile** parameter 2-5  
*appearance.mtl* file 2-4  
**appldRange** parameter 2-13, 2-17  
**application number** 2-5, 3-11  
     back-end 2-6  
     back-end for remapping objects 2-11  
     configuring 2-14  
     remapping for batch mode 2-12  
**Application Settings dialog box** 10-14, 10-15  
**appNumber** parameter 2-5, 2-14  
**argument**  
     language 2-7  
     usingGlobalMiniMap 2-10  
**arrow, forward and back** 5-21, 5-29  
**askSaveScenarioIfChanged** parameter 2-5  
**asynchronous I/O** 2-14, 2-18  
     using for performance 1-11  
**attached, keyboard mappings** C-7  
**autoRemapInCombinedMode** parameter 2-10  
**autoRemapInSeparateMode** parameter 2-11  
**auto-reorganize** parameter 3-5  
**availableTaskSetRetrievalScheme** parameter 2-6  
**avoiding, objects** 8-21

## B

**back arrow** 5-21, 5-29  
**back-end**  
     application number 2-6, 2-11  
     console 2-15  
     discovery by front-end 2-7  
     heartbeat 2-16  
     multiple 3-11  
     reloading OPD 2-15  
     remapping for batch mode 2-11

**back-end (continued)**  
     site ID 2-6  
         for remapping 2-11  
         specifying name of 2-6  
         timeout interval 2-6  
**Back-end Settings tab** 1-7  
**backendApplicationName** parameter 2-6  
**backendAppNum** parameter 2-6  
**backendSiteId** parameter 2-6  
**backendTimeoutInterval** parameter 2-6  
**ballistic gun, controller descriptor** 8-9  
**batch file, configuring** 2-14  
**batch mode**  
     remapping application number 2-12  
     remapping back-end 2-11  
**batchFile** parameter 2-14  
**batchModeModeRemapAppNum** parameter 2-12  
**batchModeModeRemapSiteId** parameter 2-11  
**batchScenarioFilename** parameter 2-6  
**binary key mapping** C-2  
**bounding box, aggregate** 5-22  
*boundingVolume.mtl* file 2-4, 2-21  
     specifying path 2-6  
**boundingVolumesFile** parameter 2-6

## C

**cache**  
     clearing model instancing 10-14  
     model instance 10-14  
**camera** 11-3  
     position and orientation 11-21  
**Camera Offset Position** 11-9  
**Camera Orientation Offset** 11-9  
**Camera Position Offsets** 11-21  
**capabilities, configuration file** 2-6  
**capabilitiesFile** parameter 2-6  
*capabilities.mtl* file 2-4  
**catastrophic-kill** parameter 8-12  
**category**  
     adding 5-34  
     changing 5-10  
     changing echelon 7-26  
     removing 5-34  
     renaming 5-34, 5-35  
     undoing changes 5-35  
**category** parameter 7-17, 7-26  
**cgf, tuning tick() sleep interval** 2-15  
**cgfDispatcherReceivePort** parameter 2-14

- changing
  - channel properties 11-9
  - component priority 6-9
  - echelon ID category and echelon ID level 7-26
  - entity
    - category 5-10
    - enumeration 5-12
    - force 5-11
    - icon 5-8
    - image 5-8
    - name 5-12
    - parameter 5-14
    - symbols 2-20
    - view in OPD Editor 6-4
  - field-of-view 11-14
  - frame rate mode 3-5
  - grid scale factor 5-24
  - key mapping C-5
  - site ID 3-11
  - subordinate order 5-20
  - symbol set 5-8
  - terrain database for scenario 3-7
  - viewport 11-15
  - window properties 11-7
- channel 11-2, 11-3
  - adding 11-6
  - changing properties 11-9
  - projection resize policy 11-12
  - removing 11-6
- Channel Name 11-9
- character, specifying for DI-Guy model 5-15
- character\_animations\_table.mtl* 2-22
- character\_animations\_table.mtl configuration file 2-4
- character\_digest.xml* 2-22
- character\_digest.xml configuration file 2-4
- class
  - DtSimComponentManager* 7-10
  - DtSimObjectNetInterface* 7-10
  - DtSimObjectStateRepository* 7-10
  - DtSimTaskManager* 7-10
  - DtVrfObject* 7-6, 7-10
  - DtVrfObjectBaseStateRepository* 7-6
- clearing, model instancing cache 10-14
- clipping 11-11
- clipping plane, dynamic 11-11
- clock mode 3-5
- closing, scenario file in Scenario Merge 4-8
- coefficient probability 8-10
- collateral damage, probability table 8-12
- collision avoidance, configuring 8-21
- combined mode, remapping objects 2-10
- combining, key mapping C-6
- command line, running Scenario Merge from 4-9
- command-line option, --targetFrameRate 1-16
- commandLineDiscoveryTimeout** parameter 2-7
- comment, MTL parameter 2-2
- comm-model-name** parameter 8-18
- commModelParams.mtl* file 2-23, 9-4, 9-5
- commonObjects.mtl* file 2-4, 2-20
- communication model 2-23, 8-18
  - custom 2-24
  - setting 9-4
  - simple-radio-comm-model 2-23
- Communication Model Settings dialog box 9-4, 9-7
- component 7-9
  - adding to object in OPD Editor 6-8
  - attaching to remote entity 8-25
  - creating connection in OPD Editor 6-9
  - deleting in OPD Editor 6-6
  - descriptor, elements of 7-14
  - editing in OPD Editor 6-8
  - entity 7-13
  - occupancy-director-controller 8-23
  - priority 6-9
  - tuning 1-12
    - performance 1-12
- component attachment table 8-25
- Component Manager B-5
- Component-Attachment** parameter 3-4
- Component-Attachment-Table** parameter 3-5
- componentAttachmentTable.mtl 8-25
- component-descriptor-type 7-14
- component-manager** parameter B-5, B-7
- component-type 7-14
- conditional expression 3-9
  - entity type pattern 2-8
- configuration
  - display engine
    - loading 11-7
    - saving 11-7
- configuration file 2-4
  - appearance attribute 2-5
  - boundingVolume.mtl* 2-4, 2-21
  - capabilities 2-6
  - character\_animations\_table.mtl 2-4
  - character\_animations\_table.mtl* 2-22
  - character\_digest.xml 2-4
  - character\_digest.xml* 2-22

- configuration file 2-4 (continued)*
  - commModelParams.mtl* 2-23, 9-4, 9-5
  - configurationsSettings.xml 2-4
  - entityEditor.mtl 2-4
  - for object parameter database 5-4
  - opdEditor.mtl 2-4
  - overlayMenu.mtl* 2-21, A-2
  - overlayPalette.mtl* 2-21
  - plugin.mtl* 2-20
  - rangeRings.mtl* 2-21
  - settings 2-7
  - symbolmap-images.mtl* 2-4
  - vr3DGui.mtl* 2-5
  - vr.ent* 2-4, A-2, A-5
  - vrSim.mtl* 2-14
  - vrSimSettings.xml* 1-14
  - vrSimSettings.xml* 2-5
- configurationsSettings.xml configuration file 2-4
- configuring
  - back-end
    - application number 2-6
    - console 2-15
    - heartbeat 2-16
    - name 2-6
    - site ID 2-6
    - timeout interval 2-6
  - batch file 2-14
  - communications model 2-23
  - context sensitivity for tasks and sets 2-6
  - Create menu 2-21
  - default scenario path 2-7
  - destination address 2-13, 2-18
  - DI-Guy character and appearance 2-22, 5-15
  - DIS 2-13, 2-14, 2-17, 2-18
  - display engines 11-2
  - drainInput 2-7, 2-16
    - DIS 2-13
    - HLA 2-13
  - drainInput for DIS 2-18
  - drainInput for HLA 2-17
  - emitter 8-16
  - environmental data updates 2-15
  - ethernet address for file save 2-7
  - exercise ID 2-13, 2-18
  - External Communications Model 9-5
  - feature avoidance 8-22
  - FED file name 2-12, 2-17
  - federation execution 2-12, 2-17
    - name 2-12, 2-17
  - FOM Mapper 2-12, 2-17
  - configuring (continued)
    - formations 8-14
    - frame rate buffer 2-12
    - frequency of data sends 2-15
    - heartbeat threshold 2-13, 2-17
    - HLA 2-12, 2-17
    - joysticks 8-20
    - language 2-7
    - location of help file 2-8
    - map datum 2-10, 2-17
    - model definition 10-8
    - model definition schema 10-15
    - multicast address 2-13, 2-17
    - multi-channel display 11-19
    - mutually exclusive tasks 8-19
    - network frame length 2-12
    - object avoidance 8-21
    - obstruction avoidance 8-21
    - overlay menu 2-7
    - Overlay Palette 2-21
    - path to OPD file 2-15
    - performance 1-10, 2-11
    - port number 2-13, 2-18
    - projection resize policy 11-12
    - radio message model 2-23
    - radios 8-18
    - range rings 2-9
    - receive port 2-14
    - relative scenario paths 2-7
    - save scenario prompt 2-5
    - scenario file to use 2-16
    - scenario save port 2-9
    - sensors 8-3
    - session ID 2-16
    - settings 2-7
    - simulation management message 2-16
    - site ID 2-16
    - sleep interval 2-16
    - start mode 2-16
    - symbols 2-12
    - terrain database
      - path 2-15
      - to load 2-16
    - tick management 2-12
    - time stamp order 2-13, 2-17
    - track history 2-11
    - transmitter 2-8
    - transmitters 8-18
    - visual quality slider 1-15



- configuring (continued)
    - VR-Forces for external communications effects server 9-2
  - configuring slider 1-15
  - Connect To Display Engine dialog box 11-17
  - connecting, display engine to 11-17
  - connection
    - component 6-9
    - sensor 8-3
  - connection mode 2-23
    - aggregate 2-23
    - all 2-23
    - force 2-23
  - connection model, external 2-23
  - consecutiveOverFrameRateThreshold** parameter 2-11
  - consecutiveTimesOverThreshold** parameter 1-9
  - console
    - configuring for back-end 2-15
    - enabling or disabling, in Windows 2-10
  - context sensitivity, configuring 2-6
  - control object 3-11, B-7
    - parameter-type 7-27
  - copying
    - formation 5-23
    - model definition 10-10
    - model definition schema 10-17
  - CPU time, giving up 1-16
  - Create Connection dialog box 6-9
  - Create Entry dialog box 6-7
  - Create menu A-2
    - adding entity to A-5
    - assigning or removing entity 5-13
    - configuring 2-21, A-2
    - editing 5-29, A-4
      - manually 5-31
    - reorganizing A-4
    - reverting 5-33
  - Create Menu Tree, OPD Editor 6-4
  - Create New Component dialog box 6-8
  - Create Resource dialog box 6-10
  - create-entity-menu-item A-5
  - creating
    - aggregate 5-17
    - component connection 6-9
    - entity A-5
      - from existing 5-28
      - new 5-28
    - key map C-7
    - model definition 10-9, 10-21
    - model definition schema 10-15
  - creating (continued)
    - Scenario Merge project 4-3
    - simulation model set 5-5
    - user-defined formation 8-15
  - cultural-feature-param** parameter 7-12, B-8
  - cultural-feature-state-repository 7-12
  - custom, communication model 2-24
  - customizing, object parameter database 7-28
- ## D
- daemon, running vrfSim as 2-14
  - daemon** parameter 2-14
  - damage
    - configuring entity to take 8-13
    - probability table 8-12
  - damage actuator 8-12
  - damage file 5-4, 7-27
  - damage model 5-25
    - specifying 5-26
  - data
    - configuring frequency of sends 2-15
    - environmental, updates 2-15
    - input 7-13
  - database, object parameter 7-2
  - decal effects 10-4
  - decayTrackHistory** parameter 2-11
  - default
    - formation 5-22
    - scenario 2-7
    - value, parameter 3-12
  - defaultFreehandSamplingRate** parameter 2-7
  - defaultHeartbeatThreshold** parameter 2-13, 2-17
  - defaultMainLoopSleepInterval** parameter 2-15
  - defaultObjectTimeout** parameter 2-13, 2-18
  - defaultOverlayMenu** parameter 2-7
  - defaultParameterDatabasePath** parameter 2-15, 3-6
  - defaultScenario** parameter 2-7
  - defaultSettingsFile** parameter 2-7
  - default.sms* file 5-4, 5-6
  - defaultSymbolMapFile** parameter 2-7
  - defaultTerrainDatabasePath** parameter 2-15, 3-6
  - definition, parameter 6-4
  - Definition Schemata page 10-15, 10-16, 10-17, 10-18
  - deleting
    - category 5-34
    - channel 11-6
    - entity from simulation model set 5-29
    - key map C-7

- deleting (continued)
    - key mapping C-5
    - model definition 10-9
    - model definition schema 10-16
    - model mapping 10-7
    - parameter
      - model definition 10-13
      - model definition schema 10-18
    - parameters and components in OPD Editor 6-6
    - window 11-6
  - description, parameter 6-4
  - destAddrString** parameter 2-13, 2-18
  - destination address 2-13, 2-18
  - destroyFedExec** parameter 2-12, 2-17
  - destroying, federation execution 2-12, 2-17
  - detonation effects 10-4
  - dialog box
    - Add Soil Type 6-11
    - Application Settings 10-14
    - Communication Model Settings 9-4, 9-7
    - Connect To Display Engine 11-17
    - Create Connection 6-9
    - Create Entry 6-7
    - Create New Component 6-8
    - Create Resource 6-10
    - Display Options 1-4
    - Frame Rate Monitor 1-7
    - New Scenario 3-3
    - Open Model Set 5-6
    - Performance Options, configuring slider 1-15
    - Performance options 1-14
    - Shared Simulation Settings 10-4
    - Terrain View Options 1-10
  - DI-Guy 2-4
    - configuring character and appearance 5-15
  - DIS
    - configuring 2-13, 2-14, 2-17, 2-18
    - enumerations document 10-4
    - heartbeat, threshold 2-13, 2-17
  - DIS Enumeration Document 7-6
  - disabling
    - advisory messages 2-13, 2-15, 2-17
    - console window 2-10
    - creation of an entity type 5-13
    - frame rate statistics 2-15
    - global Minimap 2-10
    - track histories 1-10
    - view control messages 2-10
  - disconnecting, display engine 11-18
  - display
    - field-of-view 11-14
    - frame rate 1-4
      - setting 1-4
      - viewing 1-5
    - perspective 11-14
  - display engine 11-2
    - configuration
      - loading 11-7
      - saving 11-7
    - connecting to 11-17
    - connection states 11-3
    - disconnecting from 11-18
  - Display Engine Configuration Editor 11-4, 11-5, 11-6, 11-7, 11-8, 11-10, 11-17, 11-18
  - Display Options dialog box 1-4
  - displaying
    - performance statistics 1-17
    - polygons 1-10
    - scenario file in Scenario Merge 4-7
  - disPort** parameter 2-13, 2-18
  - doNotUseConsole** parameter 2-15
  - doNotUseSettingsFile** parameter 2-7
  - drainInput
    - configuring 2-7, 2-16
    - configuring for DIS 2-13, 2-18
    - configuring for HLA 2-13, 2-17
  - dr-allow-gui-overrides** parameter 1-15
  - drawing mode 1-10
  - DtNotifyLevel** parameter 2-7, 2-15
  - DtSimComponentManager* class 7-10
  - DtSimObjectNetInterface* class 7-10
  - DtSimObjectStateRepository* class 7-10
  - DtSimTaskManager* class 7-10
  - DtVrfObject* class 7-6, 7-10
  - DtVrfObjectBaseStateRepository* class 7-6
  - duplicate, key mapping C-6
  - dynamic clipping plain 11-11
  - Dynamic Near Far Planes 11-9
  - Dynamic Near Far Planes attribute 11-11
  - Dynamic Near/Far Ratio 11-9
  - Dynamic Near/Far Ratio attribute 11-11
- E**
- echelon, changing level 7-26
  - echelon ID, category and echelon ID level 7-26
  - echelon-level** parameter 7-23, 7-26
  - editing
    - aggregate 5-17

- 
- editing (continued)
    - aggregate subordinate 5-21
    - channel properties 11-9
    - components in OPD Editor 6-8
    - Create menu 5-29, 5-31, A-4
    - embarkation ingress and egress points 8-23
    - embarkation parameters 8-23
    - entity model 5-7
    - formation 5-17
    - frustum 11-14
    - key map C-3
    - key maps C-2
    - model definition 10-11
    - model mapping 10-6
    - object map file 3-11
    - parameter
      - model definition 10-12
      - model definition schema 10-18
    - parameters 5-14, 6-4
    - parameters for multiple objects 6-6
    - plan file 3-8
    - preconfigured aggregate 7-24
    - scenario file 3-3
      - in Scenario Merge 4-8
    - sensor 8-4
    - sensor signature 8-5
    - system 5-27
    - system properties 5-27
    - systems 6-11
    - window properties 11-7
  - effects
    - mapping 10-4
    - model 10-3
  - egress point 8-23
  - egress-points** parameter 8-23
  - embarkation
    - editing ingress and egress points 8-23
    - occupancy-director-controller 8-23
    - slots 8-23
  - embarkation-slot-entry** parameter 8-23
  - embarkation-slots** parameter 8-23
  - embedded window 11-3
  - emitter, configuring 8-16
  - enableEmitterHandling** parameter 2-7
  - enableHistoryTracking** parameter 1-10, 2-11
  - enableLogoOverlay** parameter 2-7
  - enabling
    - advisory messages 2-13, 2-15, 2-17
    - console window 2-10
    - creation of an entity type 5-13
  - enabling (continued)
    - External Communications Model 9-4
    - frame rate statistics 2-15
    - global Minimap 2-10
    - radio transmitter 9-3
    - view control messages 2-10
  - entity
    - 3D model 5-14
    - adding new to parameter database 7-17
    - category, changing 5-10
    - changing
      - force 5-11
      - name 5-12
      - symbol set 5-8
      - undoing changes 5-16
    - component, attaching to remote 8-25
    - components 7-13
    - configuring
      - damage by munitions 8-13
      - emitter 8-16
      - obstruction avoidance 8-21
    - creating from existing 5-28
    - creating in Entity Editor 5-28
    - creation, enabling and disabling 5-13
    - deleting from simulation model set 5-29
    - editing 5-3
    - icon
      - changing 5-8
      - mapping 2-20
    - image, changing 5-8
    - model 10-3
      - editing 5-7
    - models 5-4
    - systems 5-25
      - adding 5-26
    - tick rate 1-12
    - type 10-4
      - enumeration 10-5
    - vrf.ent* file 2-4, A-2, A-5
  - Entity Editor 5-3
    - configuring DI-Guy character and appearance 5-15
  - force
    - adding 5-36
    - editing 5-37
    - removing 5-37
  - formation, bounding box 5-22
  - managing forces 5-35
  - navigating edited entities 5-29
  - opening OPD Editor from 5-27

Entity Editor 5-3 (continued)  
     starting 5-4  
 entity enumeration, changing 5-12  
 Entity List, OPD Editor 6-4  
 Entity Palette, assigning or removing entity 5-13  
 Entity Resource Manager 8-9  
 Entity Tree, OPD Editor 6-4  
 entity type  
     changing 5-12  
     enumeration, for conditional expressions 2-8  
 entityEditor.mtl configuration file 2-4  
 entityEditorUi.mtl file 2-4  
 entity-type parameter 8-10, A-5  
 entVrfDataTimeThreshold parameter 2-15  
 enumeration 7-6, 10-5  
     document 10-4  
     entity, changing 5-12  
     object type B-9  
 Enumeration and Bit Encoded Values for Use with  
     Protocols for DIS Applications 7-6  
 environmental data, frequency of updates 2-15  
 envVrfDataTimeThreshold parameter 2-15  
 ethernet address, for file save 2-7  
 execName parameter 2-12, 2-17  
 exercise ID 2-13, 2-18  
 exerciseld parameter 2-13, 2-18  
 exercise-start-time parameter 3-6  
 external communication model 2-23  
     enabling 9-4  
 external communications effects server  
     configuring VR-Forces for 9-2  
     QualNet 9-5  
 External Communications Model  
     configuring 9-5  
     enabling 9-4  
 external communications model 9-3

## F

far clipping plane 11-11  
 feature  
     avoiding 8-21  
     configuring avoidance of 8-22  
 FED file, name 2-12, 2-17  
 federate, name 2-12, 2-17  
 federateName parameter 2-12, 2-17  
 federation execution  
     destroying 2-12, 2-17  
     name 2-12, 2-17  
 fedFileName parameter 2-12, 2-17

field of view 11-14  
 field-of-view, changing 11-14  
 file  
     appearance.mtl 2-4  
     capabilities.mtl 2-4  
     commonObjects.mtl 2-4, 2-20  
     default.sms 5-4  
     entityEditorUi.mtl 2-4  
     formation 8-14  
     joyParam.dat 8-20  
     MTL 2-2  
     object map 3-11  
     order of battle 3-11  
     overlayMenu.mtl 2-4  
     plan 3-8  
     plugin.mtl 2-4  
     pvd.mtl 2-4  
     rangeRings.mtl 2-4  
     scenario 3-3, 3-6  
     simulation model set 5-4  
     symbolmap-bmp.mtl 2-4  
     symbolmap-ttf.mtl 2-4  
     vrf.ent 2-4, A-2, A-5  
     vrfGui.mtl 2-5  
     vrfGuiSettings.mtl 2-5  
     vrfSim.mtl 2-5, 2-14  
     vrfSim.opd 7-2  
     vrfSimSettings.xml 1-14  
 File Caching page 10-14  
 filtering  
     function list C-6  
     model definition list 10-10  
 firing effects 10-4  
 fixed-frame 3-5  
 Fixed-Frame Best-Effort 3-5  
 Fixed-Frame Run-To-Complete 3-5  
 fixed-frame-run-to-complete 3-5  
 fixed-wing-entity-param 7-12  
 fixed-wing-entity-param parameter B-8  
 fixed-wing-entity-state-repository 7-12  
 flaming effects 10-4  
 FLT file, mapping to models 10-4  
 FOM Mapper  
     configuring 2-12, 2-17  
     initialization string 2-12, 2-17  
 fomMapperInitData parameter 2-12, 2-17  
 fomMapperLib parameter 2-12, 2-17  
 font, mapping 2-20  
 force  
     adding 5-36

force (continued)

- color, changing 5-37
- connection mode 2-23
- editing 5-37
- entity, changing 5-11
- hostility, changing 5-37
- image 5-9
- name, changing 5-37
- removing 5-37

**Force-Hostility** parameter 3-5

**forceParameterDbReload** parameter 2-15

**force-type** parameter A-4, A-5

formation

- adding 5-21
- bounding box 5-22
- configuring 8-14
- copying 5-23
- default 5-22
- editing 5-17
- expanding and collapsing in Formation Editor 5-22
- file 5-4, 8-14
- layout
  - automatic 5-24
  - manual 5-24
- removing 5-22
- renaming 5-22
- snap to grid 5-24
- user-defined 8-15

Formation Editor 5-21, 5-22, 5-23, 5-24

forward arrow 5-21, 5-29

frame, length of 3-6

frame rate 1-16, 2-11

- configuring 2-12
- display, setting 1-4
- mode, specifying 3-5
- setting network 1-5
- statistics, enabling and disabling 2-15
- viewing network and display 1-5

Frame Rate Monitor dialog box 1-7

**frame-mode** parameter 3-5

**frame-time** parameter 3-5, 3-6

freehand line, number of points 2-7

frequency, back-end statistics 1-7

front-end, waiting for back-end 2-7

frustum 11-9

- editing 11-14

full screen window 11-3

function list, filtering C-6

## G

gain

- speed, keyboard mappings C-7

geometry, sharing 10-14

global Minimap, enabling and disabling 2-10

**Global-Plan** parameter 3-5

graphics, performance impact 1-11

grid scale factor, changing 5-24

ground vehicle, configuring obstruction avoidance 8-21

ground-entity-state-repository 7-12

ground-vehicle-param 7-12

**ground-vehicle-param** parameter 7-6, B-8

**Gui-Terrain-Database** parameter 3-5, 3-7

## H

hazard cloud, configuring performance 1-11

header file, *objTypeEnum.h* B-9

head-up display 10-4

heartbeat

- back-end, configuring 2-16
- DIS 2-13, 2-17
- relationship to thresholds 1-14

heavy network traffic 1-9, 2-11

- configuring 2-12

help file, configuring location for 2-8

**hideOffScreenSymbols** parameter 2-12

hit probability file 7-27

hit probability table 8-10

HLA, parameters 2-12, 2-17

**hostAddrString** parameter 2-7

**human-param** parameter 7-12, B-8

human-state-repository 7-12

## I

icon

- animated 2-4
- changing entity 5-8
- changing symbol map file 2-20
- for all forces 5-9

ID, site 2-16

ignore list 4-5

**ignoreAdvisories** parameter 2-15

image

- entity, changing 5-8
- for all forces 5-9

improving, performance 1-10

- improving performance 1-10
- individual-lifeform-param 7-12
- individual-lifeform-state-repository 7-12
- information message 2-15
- ingress point 8-23
- ingress-points** parameter 8-23
- initialization string, FOM Mapper 2-12, 2-17
- input
  - of data 7-13
  - port and port group 7-14
- intersection, intervisibility 1-11
- intervisibility, intersection 1-11, 2-21

## J

- joy-device-type** parameter 8-20
- joyParam.dat* file 8-20
- joy-rot-gain** parameter 8-20
- joy-rot-invert** parameter 8-20
- joy-rot-sensitivity** parameter 8-20
- joystick, configuring 8-20
- joy-trans-gain** parameter 8-20
- joy-trans-invert** parameter 8-20
- joy-trans-sensitivity** parameter 8-20

## K

- key map C-1
  - creating C-7
  - deleting C-7
  - editing C-3
  - loading C-7
- Key Map Editor C-1, C-2
  - adding mapping C-4
  - binary mapping C-2
  - changing mapping C-5
  - deleting mapping C-5
  - filtering function list C-6
- key mapping
  - adding C-4
  - binary C-2
  - changing C-5
  - combining C-6
  - deleting C-5
- keyboard C-1
  - mapping C-2
  - mappings C-7
- key-names, location of configuration files 8-26

## L

- label** parameter A-5
- language, configuring 2-7
- language** argument 2-7
- lateral-clearance** parameter 8-21
- layout
  - formation
    - automatic 5-24
    - manual 5-24
- leader-promotion-id** parameter 8-14
- level observer frame C-7
- lifeform
  - appearance, random 5-15
  - configuring DI-Guy appearance and character 2-22
- line
  - freehand, points in 2-7
  - keeping settings 2-10
- lisp B-2
- listing, components in parameters database 7-16
- load-entry** parameter 8-23
- load-entry-point** parameter 8-23
- loading
  - display engine configuration 11-7
  - key map C-7
  - scenario, timeout 2-9
  - Scenario Merge project 4-6
  - symbol map file 2-20
- loadPluginIfVersionMismatch** parameter 2-15
- local
  - object 7-10
  - network interface 7-13
- local-objects** parameter 7-10, 7-11, B-4
- logFrameRateStatistics** parameter 1-7, 2-15
- lookahead** parameter 2-13

## M

- MÄK Technologies LISP 2-2
- makePathsRelative** parameter 2-7
- map
  - datum, configuring 2-10
- map datum, configuring 2-17
- mapping
  - keys C-2
  - model to entity type 10-5
  - mouse button C-11
  - object, back-end 2-11
  - objects to back-end 2-11

- mapping (continued)
    - ttf and utf fonts 2-20
  - maxDrainInputReads** parameter 2-13, 2-18
  - maxDrainInputTime** parameter 2-7, 2-16
  - maxDRRotationThreshold** parameter 1-15
  - maxDRTranslationThreshold** parameter 1-15
  - maximumSignalTimeDifferent** parameter 2-8
  - mcastTtl** parameter 2-14, 2-18
  - memory 1-16
  - memory use, improving 10-14
  - menu, adding to Create A-5
  - merging, scenarios 4-2, 4-7
  - message
    - frequency of 2-15
    - information and warning 2-15
    - notification for Scenario Merge 4-8
    - sending and receiving 8-18
  - message-timeout** parameter 9-5
  - migrating, object parameter database to new release 7-28
  - minDrainInputTime** parameter 2-13, 2-17
  - minimum tick period 1-12
  - minimum tick period variance 1-12
  - min-tick-period** parameter 1-12
  - min-tick-period-variance** parameter 1-12
  - minTrackHistoryMovement** parameter 2-11
  - missile-entity-state-repository 7-12
  - missile-param 7-12
  - missile-param** parameter B-8
  - model 10-3
    - communication 2-23
    - entity 5-4
      - editing 5-3, 5-7
    - mapping
      - adding 10-5
      - deleting 10-7
      - editing 10-6
      - to FLT files 10-4
    - object 5-4
    - sharing resources 10-14
    - simulation 5-4
    - specifying 3D 5-14
  - model definition 10-4
    - configuring 10-8
    - copying 10-10
    - creating 10-9, 10-21
    - deleting 10-9
    - editing 10-11
    - filtering list of 10-10
  - model definition 10-4 (continued)
    - parameter
      - adding 10-11
      - deleting 10-13
      - editing 10-12
      - saving 10-13
  - model definition schema
    - adding parameter to 10-17
    - copying 10-17
    - creating 10-15
    - deleting 10-16
    - deleting parameter from 10-18
    - editing parameter 10-18
  - Model Definitions page 10-8, 10-9, 10-10, 10-11, 10-12, 10-13, 10-21
  - model instance, caching 10-14
  - model instancing 10-14
  - model instancing cache, clearing 10-14
  - model schema 10-8
  - monitoring, network and display frame rates 1-5
  - mouse, button mappings C-11
  - movement
    - keyboard mappings C-7
    - mouse buttons C-11
  - movement model 5-25
    - specifying 5-26
  - movement system 5-26
  - moving object parameter B-8
  - moving-object-param 7-12
  - moving-object-param** parameter 7-23, B-8
  - moving-object-param parameter type B-8
  - MTL file 2-2
    - vrfSim.mtl* 2-14
  - multicast address, configuring 2-13, 2-17
  - multicast time-to-live 2-14, 2-18
  - multi-channel, configuring 11-19
  - multiple, back-ends 3-11
  - munition, configuring entity for damage by 8-13
  - munition-type parameter 8-9
  - mutually exclusive task, configuring 8-19
  - mutually-exclusive-task-groups** parameter 8-19
- ## N
- name
    - changing, entity 5-12
    - component 7-14
    - FED file 2-12, 2-17
    - federate 2-12, 2-17
  - navigating, aggregate hierarchy 5-21

- navigation
  - function, editing key map C-3
  - key map C-1
  - keyboard mappings C-7
- near clipping plane 11-11
- net-interface-min-tick-period** parameter 1-13, B-5, B-7
- net-interface-min-tick-period-variance** parameter 1-13, B-5, B-7
- network
  - frame, configuring time to execute 2-12
  - frame rate 1-4
  - setting 1-5
  - viewing 1-5
  - performance 1-15
  - traffic, heavy 1-9
- network card, multiple 2-7
- network interface B-5
  - hierarchy 7-13
- networkFrameRateBuffer** parameter 1-9, 2-12
- network-interface** parameter B-5, B-7
- networkTimeThreshold** parameter 1-9, 2-12
- New Scenario dialog box 3-3
- notification level 2-15
  - Scenario Merge 4-8

## O

- object
  - adding, resources in OPD Editor 6-10
  - configuring, avoidance of 8-21
  - count 1-16
  - editing, multiple 6-6
  - mapping to back-end 2-10, 2-11
  - model 10-3
  - models 5-4
  - parameter B-3
    - default value 3-12
  - subcomponents 7-10
  - super-type 7-7
  - timeout threshold 2-13, 2-18
  - type 7-6
    - hierarchy 7-8
- object map file 3-11
- object parameter database 5-4, 7-2
  - adding
    - component in OPD Editor 6-8
    - new entity 7-17
    - new object in OPD Editor 6-7
    - preconfigured aggregate 7-23, 7-25
    - soil type parameter 6-11

- object parameter database 5-4, 7-2 (continued)
  - adding
    - subordinate object 6-10
  - configuring path to 2-15
  - configuring sensors 8-3
  - loading in OPD Editor 6-3
  - migrating to new release 7-28
  - pathname interpretation 7-27
  - reloading 2-15
  - saving 6-13
- Object Parameter Database Editor. *See* OPD Editor
- object type enumeration B-9
- objectConsoleHistoryLimit** parameter 2-8
- objectInformationToolbar** parameter 2-8
- Object-Map** parameter 3-5
- object-type** parameter 7-8, 7-17, B-3
- objTypeEnum.h* header file B-9
- observer 11-3
- observer frame C-7
- Observer Name 11-9
- obstacle, configuring avoidance of 8-21
- obstruction, configuring avoidance of 8-21
- occupancy-director-controller, role in embarkation 8-23
- occupancy-director-controller component 8-23
- OPD, reloading 2-15
- OPD Editor 6-2
  - adding
    - new object 6-7
    - resources 6-10
  - adding subordinate object 6-10
  - changing
    - component priority 6-9
    - list views 6-4
  - compared to Entity Editor 5-3
  - connecting components 6-9
  - deleting parameters and components 6-6
  - editing
    - components 6-8
    - multiple objects 6-6
    - preconfigured aggregate 7-24
  - editing parameters 6-4
  - loading object parameter database 6-3
  - opening from Entity Editor 5-27
  - restoring values 6-6
  - saving object parameter database 6-13
  - starting 6-2
- OPD editor, ingress and egress points 8-23
- opdEditor.mtl* configuration file 2-4



OPE file 7-27  
 Open Model Set, dialog box 5-6  
 opening, simulation model set 5-6  
 optimal performance level 1-8  
 optimizing, performance 1-10, 1-16  
 orbit, keyboard mappings C-7  
 order of battle  
   file 3-11  
   default values in 3-12  
**Order-Of-Battle** parameter 3-5  
 orientation  
   camera 11-21  
   keyboard mappings C-7  
 output, port group 7-14  
 overlay  
   menu, default configuration file 2-7  
 Overlay file 3-12  
 Overlay Palette, configuring 2-21  
**Overlay** parameter 3-5  
**overlayLogoFile** parameter 2-8  
**overlayLogoPosition** parameter 2-8  
**overlayLogoPositionX** parameter 2-8  
**overlayLogoPositionY** parameter 2-8  
**overlayLogoScaleFactor** parameter 2-8  
*overlayMenu.mtl* file 2-4, 2-21, A-2  
   specifying default 2-7  
*overlayPalette.mtl* file 2-4, 2-21  
*ovl* file 3-12

## P

panel, Display Engine Configuration Editor 11-5  
 parameter  
   adding to model definition 10-11  
   adding to model definition schema 10-17  
   additional-opds 3-6  
   addMulticastAddr 2-13, 2-17  
   addSymbolMapFile 2-5, 2-18  
   aggregate-formation-controller 8-14  
   aggregatesFilename 2-5  
   all-tasks-exclusive 8-19  
   ammo 8-9  
   angle-of-incidence 8-12  
   appearanceFile 2-5  
   appldRange 2-13, 2-17  
   appNumber 2-5, 2-14  
   askSaveScenarioIfChanged 2-5  
   autoRemapInCombinedMode 2-10  
   autoRemapInSeparateMode 2-11  
   auto-reorganize 3-5

parameter (continued)  
   availableTaskSetRetrievalScheme 2-6  
   backendApplicationName 2-6  
   backendAppNum 2-6  
   backendSitelD 2-6  
   backendTimeoutInterval 2-6  
   batchFile 2-14  
   batchModeModeRemapAppNum 2-12  
   batchModeModeRemapSitelD 2-11  
   batchScenarioFilename 2-6  
   boundingVolumesFile 2-6  
   capabilitiesFile 2-6  
   catastrophic-kill 8-12  
   category 7-17, 7-26  
   cgfDispatcherReceivePort 2-14  
   commandLineDiscoveryTimeout 2-7  
   commenting out 2-2  
   comm-model-name 8-18  
   component list order 7-16  
   Component-Attachment 3-4  
   Component-Attachment-Table 3-5  
   component-manager B-5, B-7  
   consecutiveOverFrameRateThreshold 2-11  
   consecutiveTimesOverThreshold 1-9  
   cultural-feature-param 7-12, B-8  
   daemon 2-14  
   decayTrackHistory 2-11  
   defaultFreehandSamplingRate 2-7  
   defaultHeartbeatThreshold 2-13, 2-17  
   defaultMainLoopSleepInterval 2-15  
   defaultObjectTimeout 2-13, 2-18  
   defaultOverlayMenu 2-7  
   defaultParameterDatabasePath 2-15, 3-6  
   defaultScenario 2-7  
   defaultSettingsFile 2-7  
   defaultSymbolMapFile 2-7  
   defaultTerrainDatabasePath 2-15, 3-6  
   definitions 6-4  
   deleting, model definition 10-13  
   deleting from model definition schema 10-18  
   deleting in OPD Editor 6-6  
   destAddrString 2-13, 2-18  
   destroyFedExec 2-12, 2-17  
   DIS 2-13, 2-17  
   disPort 2-13, 2-18  
   doNotUseConsole 2-15  
   doNotUseSettingsFile 2-7  
   dr-allow-gui-overrides 1-15  
   DtNotifyLevel 2-7, 2-15  
   echelon-level 7-23, 7-26

parameter (continued)

- editing in Entity Editor 5-14
- editing in OPD Editor 6-4
- editing model definition schema 10-18
- egress-points 8-23
- embarkation, editing 8-23
- embarkation-slot-entry 8-23
- embarkation-slots 8-23
- enableEmitterHandling 2-7
- enableHistoryTracking 1-10, 2-11
- enableLogoOverlay 2-7
- entity-type 8-10, A-5
- entry B-2
- entVrfDataTimeThreshold 2-15
- envVrfDataTimeThreshold 2-15
- execName 2-12, 2-17
- exerciseld 2-13, 2-18
- exercise-start-time 3-6
- federateName 2-12, 2-17
- fedFileName 2-12, 2-17
- fixed-wing-entity-param B-8
- fomMapperInitData 2-12, 2-17
- fomMapperLib 2-12, 2-17
- Force-Hostility 3-5
- forceParameterDbReload 2-15
- force-type A-4, A-5
- frame-mode 3-5
- frame-time 3-5, 3-6
- Global-Plan 3-5
- ground-vehicle-param 7-6, B-8
- Gui-Terrain-Database 3-5, 3-7
- hideOffScreenSymbols 2-12
- HLA 2-12, 2-17
- hostAddrString 2-7
- human-param 7-12, B-8
- ignoreAdvisories 2-15
- ingress-points 8-23
- joy-device-type 8-20
- joy-rot-gain 8-20
- joy-rot-invert 8-20
- joy-rot-sensitivity 8-20
- joy-trans-gain 8-20
- joy-trans-invert 8-20
- joy-trans-sensitivity 8-20
- label A-5
- lateral-clearance 8-21
- leader-promotion-id 8-14
- load-entry 8-23
- load-entry-point 8-23
- loadPluginIfVersionMismatch 2-15

parameter (continued)

- local-objects 7-10, 7-11, B-4
- logFrameRateStatistics 1-7, 2-15
- lookahead 2-13
- makePathsRelative 2-7
- maxDrainInputReads 2-13, 2-18
- maxDrainInputTime 2-7, 2-16
- maxDRRotationThreshold 1-15
- maxDRTranslationThreshold 1-15
- maximumSignalTimeDifferent 2-8
- mcastTtl 2-14, 2-18
- message-timeout 9-5
- minDrainInputTime 2-13, 2-17
- min-tick-period 1-12
- min-tick-period-variance 1-12
- minTrackHistoryMovement 2-11
- missile-param B-8
- model definition, editing 10-12
- moving-object-param 7-23, B-8
- mutually-exclusive-task-groups 8-19
- net-interface-min-tick-period 1-13, B-5, B-7
- net-interface-min-tick-period-variance 1-13, B-5, B-7
- networkFrameRateBuffer 1-9, 2-12
- network-interface B-5, B-7
- networkTimeThreshold 1-9, 2-12
- object B-3
- objectConsoleHistoryLimit 2-8
- objectInformationToolbar 2-8
- Object-Map 3-5
- object-type 7-8, 7-17, B-3
- Order-Of-Battle 3-5
- Overlay 3-5
- overlayLogoFile 2-8
- overlayLogoPosition 2-8
- overlayLogoPositionX 2-8
- overlayLogoPositionY 2-8
- overlayLogoScaleFactor 2-8
- parameter-type 7-6, 7-9, 7-27, B-2, B-3
- patternsFile 2-8
- performance 1-9
- Plan 3-5
- plan-manager B-5, B-7
- pluginsDirectory 2-8, 2-16
- position-offset 8-14
- preserveEmptyOverlayLayers 2-8
- processSignalInteractions 2-8
- promotion-id 8-14
- publishSimulationTimeForStealth 2-16
- publish-transmitter 8-18, 9-3
- qtAssistantLocation 2-8

## parameter (continued)

- qtHelpFile 2-8
- random-number-seed 3-5
- range-determinant 8-10, 8-12
- range-list 8-10
- rangeRingsFile 2-9
- recvBufferSize 2-14, 2-18
- remoteControllerReceivePort 2-9
- remote-objects 7-11, B-6
- rotary-wing-entity-param B-8
- rprFomVersion 2-13, 2-17
- runInTimeConstrainedMode 2-13
- runInTimeManagementMode 2-17
- runInTimeRegulatingMode 2-13
- saveTimeoutInterval 2-9
- scenario, file 3-4
- scenarioFileName 2-9, 2-16
- scenarioTimeoutInterval 2-9
- SelectionGroups 3-5
- sendBufferSize 2-14, 2-18
- sendFedTime 2-13, 2-17
- separateModeRemapAppNum 2-11
- separateModeRemapSiteld 2-11
- separateObjectLists 2-9
- sessionId 2-9, 2-16
- set-data-request-type 3-10
- ShowGroundTruthForForce 2-9
- showSplashScreen 2-9
- ShowSpotReportsForForce 2-9
- simMgmtPduProcessLevel 2-16
- simTimeScale 2-13
- simulationModelSet 2-9, 2-16, 5-6
- Simulation-Model-Set-Files 3-6
- siteld 2-9, 2-16
- sleepInterval 2-16
- slowNetworkFrameOverride 2-12
- SpotReportSystemEnabled 2-9
- startInRunMode 2-16
- state-repository B-4, B-6
- state-repository-min-tick-period 1-13
- state-repository-min-tick-period-variance 1-13
- statusUpdateHeartbeat 2-16
- stopping-distance-factor 8-21
- storeLineValuesMode 2-10
- structure of entries B-2
- substitute-message-size 9-5
- surface-entity-param B-8
- system-id 8-17
- task-manager B-5, B-7
- task-type 3-10

## parameter (continued)

- Terrain-Database 3-5
- terrainDatabase 2-10
- terrainFileName 2-16
- tickManagingEnabled 1-9, 2-12
- time-multiplier 3-5
- timeMultiplierLowerBound 2-10
- timeMultiplierUpperBound 2-10
- trackHistoryTimeout 2-11
- useAbsoluteTimeStamps 2-13, 2-14, 2-17, 2-18
- use-actual-message-size 9-5
- useAdvisories 2-13, 2-17
- useAsyncIO 1-11, 2-14, 2-18
- useCustomMapDatum 2-10, 2-17
- usingConsole 2-10
- value, restoring in OPD Editor 6-6
- vector-obstruction-type 8-22
- vector-type 8-22
- viewControlMessages 2-10
- vrf-base-object-param 7-27, B-3
- warhead 8-9
- parameters database, component list order 7-16
- parameter-type
  - aggregate 7-23
  - control object 7-27
- parameter-type parameter 7-6, 7-9, 7-27, B-2, B-3
- Parent Channel 11-9
- path, relative and absolute 2-7
- pathname
  - interpreting by object parameter database 7-27
  - scenario files 3-6
- pattern, for conditional expressions 2-8
- patternsFile parameter 2-8
- pause mode 2-16
- PDU
  - siman, configuring 2-16
- performance 1-10
  - asynchronous I/O 1-11
  - configuring 2-11
  - factors affecting 1-3
  - frame rate 1-16
  - hazard cloud images 1-11
  - heavy network traffic level 1-9
  - improving 10-14
  - intervisibility intersections 1-11
  - network, tuning 1-14
  - optimal 1-8
  - optimizing 1-3, 1-10, 1-16
  - parameters 1-9
  - SpeedTree profile 10-20

- performance 1-10 (continued)
  - statistics, displaying 1-17
  - tuning
    - cgf 2-15
    - network interface 1-13
    - state repository 1-13
  - tuning visual for 3D application 1-14
  - visual quality versus network performance 1-15
- Performance Options dialog box 1-14, 1-15
- Performance Options page 1-15
- perspective, in display 11-14
- plan
  - file, editing 3-8
- plan manager B-5
- Plan parameter 3-5
- plan-manager parameter B-5, B-7
- plugin.mtl* file 2-4, 2-20
- pluginsDirectory parameter 2-8, 2-16
- point, number in freehand line 2-7
- point-to-point communications, configuring 2-13, 2-18
- polygon, displaying 1-10
- port 7-14
  - for save scenario request 2-9
  - for scenario files 2-14
  - group 7-14
  - number 2-13
- port number 2-18
- position
  - camera 11-21
  - keyboard mappings C-7
  - window 11-7
- position-offset parameter 8-14
- precedence, components in parameters database 7-16
- preserveEmptyOverlayLayers parameter 2-8
- priority
  - component, changing 6-9
- probability table, direct and collateral damage 8-12
- probability-coefficient list 8-10
- processSignalInteractions parameter 2-8
- profile, SpeedTree performance 10-20
- project
  - adding scenarios to 4-4
  - creating Scenario Merge 4-3
  - file 4-6
    - Scenario Merge 4-3
  - loading Scenario Merge 4-6
  - removing scenarios from 4-4
  - saving Scenario Merge 4-6

- project (continued)
  - specifying ignore list for 4-5
- Projection Resize Policy 11-9
  - configuring 11-12
- Projection Units 11-9
- promotion-id parameter 8-14
- prompt, saving scenario 2-5
- properties, editing system 5-27
- publishing, simulation time 2-16
- publishSimulationTimeForStealth parameter 2-16
- publish-transmitter parameter 8-18, 9-3
- pvd.mtl* file 2-4

## Q

- qtAssistantLocation parameter 2-8
- qtHelpFile parameter 2-8
- QualNet
  - example scenario 9-6
  - using with VR-Forces 9-5

## R

- radio
  - configuring 8-18
  - message, configuring 2-23
  - transmitter 8-18
- radio transmitter, enabling 9-3
- random, DI-Guy appearance 5-15
- random-number-seed parameter 3-5
- range, Time Multiplier toolbar 2-10
- range ring, configuring 2-9
- range-determinant parameter 8-10, 8-12
- range-list parameter 8-10
- rangeRingsFile parameter 2-9
- rangeRings.mtl* configuration file 2-4, 2-21
- receive buffer size 2-14, 2-18
- receiving, radio messages 8-18
- recvBufferSize parameter 2-14, 2-18
- reference ellipsoid 2-10
  - configuring 2-17
- refresh rate 1-4
- relative path 3-6
- remapping object
  - combined mode 2-10
  - separate mode 2-11
- remote
  - object 7-10, B-6
  - network interface 7-13
- remote attachment, key-names 8-26

- remote entity, attaching components 8-25
  - remoteAttachment example scenario 8-25
  - remoteConfigurations directory 8-26
  - remote-configurations entry 8-26
  - remoteControllerReceivePort** parameter 2-9
  - remote-objects** parameter 7-11, B-6
  - removing
    - category 5-34
    - channel 11-6
    - force 5-37
    - formation 5-22
    - scenarios from Scenario Merge project 4-4
    - subordinate 5-20
    - system 5-27
    - window 11-6
  - renaming
    - category 5-34, 5-35
    - formation 5-22
  - Render Settings page 1-17, 11-13
  - reorganizing, Create menu A-4
  - replacing, subordinate 5-20
  - repository B-4
  - resizeable window 11-3
  - resizing, windows 11-12
  - resource, adding to object 6-10
  - restoring, parameter values in OPD Editor 6-6
  - reverting
    - category changes 5-35
    - Create menu 5-33
  - rotary-wing-entity-param 7-12
  - rotary-wing-entity-param** parameter B-8
  - rotary-wing-entity-state-repository 7-12
  - RPR FOM 10-4
    - specifying version 2-13, 2-17
  - rprFomVersion** parameter 2-13, 2-17
  - run mode 2-16
  - runInTimeConstrainedMode** parameter 2-13
  - runInTimeManagementMode** parameter 2-17
  - runInTimeRegulatingMode** parameter 2-13
- S**
- sample size, back-end statistics 1-7
  - save scenario prompt 2-5
  - saveTimeoutInterval** parameter 2-9
  - saving
    - display engine configuration 11-7
    - model definition 10-13
    - object parameter database 6-13
    - Scenario Merge project 4-6
    - saving scenario, timeout 2-9
  - scenario
    - adding to Scenario Merge project 4-4
    - closing in Scenario Merge 4-8
    - default 2-7
    - displaying scenario file in Scenario Merge 4-7
    - editing 4-8
    - file
      - editing 3-3
      - parameters 3-4
      - pathnames 3-6
      - relative path 2-7
      - specifying which to use 2-16
    - loading
      - from command line 2-7
      - timeout 2-9
    - merging 4-2, 4-7
    - parameter, **version** 3-6
    - Qualnet 9-6
    - remoteAttachment 8-25
    - removing from Scenario Merge project 4-4
    - saving 2-9
      - timeout 2-9
    - temporary directory 3-12
    - terrain database 3-7
  - Scenario Merge 4-2
    - closing scenario file 4-8
    - command line syntax 4-9
    - console output 4-8
    - creating project 4-3
    - displaying scenario file 4-7
    - editing scenario file 4-8
    - ignore list 4-5
    - loading project 4-6
    - project file 4-3, 4-6
    - saving project 4-6
  - scenario merge, API 4-2
  - scenarioFileName** parameter 2-9, 2-16
  - scenarioTimeoutInterval** parameter 2-9
  - schema, model definition 10-15
  - Screen Number 11-7
  - Selection Groups file 3-12
  - SelectionGroups** parameter 3-5
  - send buffer size 2-14, 2-18
  - sendBufferSize** parameter 2-14, 2-18
  - sendFedTime** parameter 2-13, 2-17
  - sending
    - data, configuring 2-15
    - radio messages 8-18

- sensor 8-3
  - adding new 8-5
  - attaching to remote entity 8-25
  - configuring 8-3
  - connecting 8-3
  - signature 8-5
  - signature-object-sensor 8-3
  - system definition file, editing 8-4
- sensor signature 5-25
- sensor system 5-25
- separate mode, remapping objects 2-11
- separateModeRemapAppNum** parameter 2-11
- separateModeRemapSiteId** parameter 2-11
- separateObjectLists** parameter 2-9
- session ID 2-9
  - configuring 2-16
- sessionId** parameter 2-9, 2-16
- set data request, configuring context sensitivity 2-6
- set statement 3-9
- Set to All 5-9
- set-data-request-type** parameter 3-10
- setting
  - back-end statistics frequency and sample size 1-7
  - communication model 9-4
  - display frame rate 1-4
  - network frame rate 1-5
  - range of Time Multiplier toolbar 2-10
- settings
  - configuring 2-7
  - default 2-7
  - line creation 2-10
- sgr* file 3-12
- Shared Simulation Settings dialog box 10-4
- sharing, resources 10-14
- ShowGroundTruthForForce** parameter 2-9
- showSplashScreen** parameter 2-9
- ShowSpotReportsForForce** parameter 2-9
- signature, sensor 8-5
- signature-object-sensor 8-3
- simMgmtPduProcessLevel** parameter 2-16
- simple-radio-comm-model 2-23
- simTimeScale** parameter 2-13
- simulation
  - objects 1-16
  - start mode 2-16
- simulation management message, configuring 2-16
- simulation model set 5-4
  - adding symbol map file 5-10
- simulation model set 5-4 (continued)
  - creating 5-5
  - deleting entity 5-29
  - file 5-4
  - opening 5-6
- simulation time, publishing 2-16
- simulationModelSet** parameter 2-9, 2-16, 5-6
- Simulation-Model-Set-Files** parameter 3-6
- site ID 3-11
  - back-end 2-6
  - configuring 2-16
  - remapping back-end 2-11
  - remapping batch mode 2-11
  - specifying 2-9
- siteId** parameter 2-9, 2-16
- sleep interval 2-15
  - configuring 2-16
- sleepInterval** parameter 2-16
- slider
  - visual quality, configuring 1-15
- slot, embarkation 8-23
- slowNetworkFrameOverride** parameter 2-12
- snap to grid, formation 5-24
- soil type, adding to object 6-11
- specifying 5-14
  - 3D model 5-14
  - appearance attribute configuration file 2-5
  - boundingVolume.mtl* path 2-6
  - default *default.sms* 5-6
  - frame length 3-6
  - frame rate mode 3-5
  - movement model 5-26
  - site ID 2-9
- specifying default 5-6
- speed
  - gain, keyboard mappings C-7
- SpeedTree
  - configuring 10-19
  - performance profile 10-20
- SpeedTree Settings page 10-19
- spot report, affect on performance 1-10
- SpotReportSystemEnabled** parameter 2-9
- starting
  - Entity Editor 5-4
  - OPD Editor 6-2
  - Scenario Merge tool 4-3
- startInRunMode** parameter 2-16
- state repository B-4
  - hierarchy 7-11, 7-12

- statement
    - conditional 3-9
    - set 3-9
  - state-repository** parameter B-4, B-6
  - state-repository-min-tick-period** parameter 1-13
  - state-repository-min-tick-period-variance** parameter 1-13
  - statistics, displaying 1-17
  - statusUpdateHeartbeat** parameter 2-16
  - Stealth, simulation clock 2-16
  - stopping-distance-factor** parameter 8-21
  - storeLineValuesMode** parameter 2-10
  - subcomponent B-4
    - classes 7-10
  - subordinate
    - adding to aggregate 5-19
    - editing 5-21
    - order, changing 5-20
    - removing from aggregate 5-20
    - replacing 5-20
  - subordinate object, adding 6-10
  - substitute-message-size** parameter 9-5
  - super-type 7-7
  - surface, damage specification 8-12
  - surface-entity-param 7-12
  - surface-entity-param** parameter B-8
  - surface-entity-state-repository 7-12
  - symbol
    - animated 2-19
    - configuring 2-12
    - for all forces 5-9
  - symbol map file 5-4
    - adding to simulation model set 5-10
    - animations 2-19
    - default 2-7
    - for fonts 2-20
    - loading 2-20
  - symbol set
    - adding 5-10
    - changing 5-8
  - symbolmap-bmp.mtl* file 2-4
  - symbolmap-images.mtl* file 2-4
  - symbolmap-utf.mtl* file 2-4
  - synchronizing, VR-Forces and Stealth simulation time 2-16
  - system
    - adding to entity 5-26
    - editing 5-27, 6-11
    - editing properties 5-27
    - entity 5-25
  - system (continued)
    - removing 5-27
    - sensor 8-3
  - system definition file, sensor 8-4
  - system-id** parameter 8-17
  - Systems List, OPD Editor 6-4
- ## T
- target-type parameter 8-9
  - task
    - configuring context sensitivity 2-6
    - configuring mutually exclusive 8-19
  - task-manager** parameter B-5, B-7
  - task-type** parameter 3-10
  - temporary directory, scenario 3-12
  - terrain database
    - changing for a scenario 3-7
    - configuring 2-16
    - configuring path to 2-15
  - Terrain View Options dialog box 1-10
  - Terrain-Database** parameter 3-5
  - terrainDatabase** parameter 2-10
  - terrainFileName** parameter 2-16
  - texture, sharing 10-14
  - threshold
    - data updates 2-15
    - DIS heartbeat 2-13, 2-17
    - object timeout 2-13, 2-18
  - tick
    - configuring 2-12
    - rate, tuning 1-12
    - sleeping between 2-16
  - tickManagingEnabled** parameter 1-9, 2-12
  - time
    - multiplier, range 2-10
  - Time Multiplier toolbar 2-10, 3-5
  - time stamp order, configuring 2-13, 2-17
  - time-multiplier** parameter 3-5
  - timeMultiplierLowerBound** parameter 2-10
  - timeMultiplierUpperBound** parameter 2-10
  - timeout
    - for sending save message 2-9
    - interval, back-end 2-6
    - object 2-13, 2-18
    - scenario load 2-9
  - toolbar, Time Multiplier 2-10
  - track history
    - affect on performance 1-10
    - configuring 2-11

- trackHistoryTimeout** parameter 2-11
- trailing effects 10-4
- transmitter
  - configuring 2-8, 8-18
  - radio 8-18
- transparency, affect on performance 1-10
- tree, SpeedTree 10-19
- TrueType font, mapping 2-20
- tuning
  - components 1-12
  - network performance 1-14
  - performance
    - component 1-12
    - network interface 1-13
    - state repository 1-13
  - visual performance 1-14

## U

- undoing
  - category changes 5-35
  - changes to Create menu 5-33
  - entity edits in Entity Editor 5-16
- unicode font, mapping 2-20
- update, frequency of 2-15
- useAbsoluteTimeStamps** parameter 2-13, 2-14, 2-17, 2-18
- use-actual-message-size** parameter 9-5
- useAdvisories** parameter 2-13, 2-17
- useAsyncIO** parameter 1-11, 2-14, 2-18
- useCustomMapDatum** parameter 2-10, 2-17
- USE-DEFAULT keyword 3-12
- user-defined, formation 8-15
- usingConsole** parameter 2-10
- usingGlobalMiniMap** argument 2-10

## V

- variable, tick rate 1-12
- variable frame mode 1-16
- variable-frame 3-5
- Variable-Frame Run-To-Complete 3-5
- vector
  - feature, configuring avoidance of 8-22
- vector-obstruction-type** parameter 8-22
- vector-type** parameter 8-22
- version, RPR FOM 2-13, 2-17
- version** parameter, scenario file 3-6
- video card, processor speed 1-16
- view control message, enabling and disabling 2-10

- viewControlMessages** parameter 2-10
- viewing, network and display frame rates 1-5
- Viewport 11-9
- viewport, changing 11-15
- visual quality slider, configuring 1-15
- vrf A-2
- vrf3DGui.mtl** configuration file 2-5
- vrf-base-object-param** parameter 7-27, B-3
- vrf-base-object-param parameter type B-7
- vrf.ent* file 2-4, A-2, A-5
  - editing A-4
  - reorganizing A-4
- vrfGui.mtl** 1-15
- vrfGui.mtl* file 2-5
- vrfGuiSettings.mtl* file 2-5
- vrf-moving-object-state-repository 7-12
- vrf-object-param 7-12
- vrf-object-param parameter type B-7
- vrf-object-state-repository 7-12
- vrf-overlay-object-state-repository 7-12
- vrfSim, running as daemon 2-14
- vrfSim.mtl* file 2-5, 2-14, 5-6
- vrfSim.opd, remote-configurations block 8-26
- vrfSim.opd* file 7-2
- vrfSimSettings.xml**, configuration file 2-5
- vrfSimSettings.xml* file 1-14

## W

- warhead** parameter 8-9
- warning message 2-15
- weapon system 5-25
- wildcard 7-8
- window 11-2
  - adding 11-4
  - adding channel 11-6
  - properties, editing 11-7
  - removing 11-6
  - types 11-3
- Window Name 11-7
- Window Type 11-7

## X-Y-Z

- Z Far 11-9, 11-11
- Z Near 11-9, 11-11







Link - Simulate - Visualize

VRF-3.12-7-090707