Copy No._____

Defence Research and Development Canada
Recherche et développement pour la défense Canada

DEFENCE R&D DÉFENSE

SIMDIS VMSA Federate

User Guide and Technical Description

Allan Gillis

Defence R&D Canada – Atlantic

Technical Memorandum
DRDC Atlantic TM 2005-026
August 2005

Canada

This page intentionally left blank.

SIMDIS VMSA Federate

User Guide and Technical Description

Allan Gillis

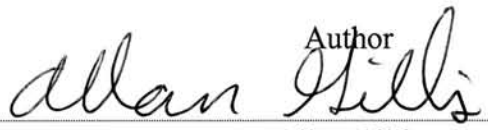
Defence R&D Canada – Atlantic

Technical Memorandum

DRDC Atlantic TM 2005-026

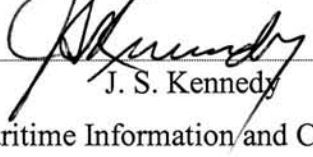
August 2005

Author



Allan Gillis

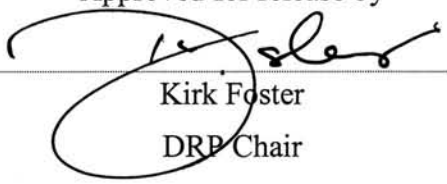
Approved by



J. S. Kennedy

Head, Maritime Information and Combat Systems

Approved for release by



Kirk Foster

DRP Chair

The information contained herein has been derived and determined through best practices and adherence to the highest levels of ethical, scientific, and engineering investigative principles. The reported results, their interpretation, and any opinions expressed therein, remain those of the authors and do not represent, or otherwise reflect, any official opinion of position of DND or the Government of Canada.

Abstract

This federate provides a software gateway from Virtual Maritime Systems Architecture (VMSA) simulations to SIMDIS, the Naval Research Laboratory's 3D visualization software. The federate can both log VMSA simulations for later viewing with SIMDIS and communicate with SIMDIS for real-time viewing.

Since visual representation is not part of VMSA the federate uses an XML dictionary file to relate VMSA Composite Entities (ships, submarines, aircraft, etc) to SIMDIS platforms. The dictionary relates specific entities to particular 3D model files and SIMDIS types.

This document describes the federate software and how to integrate the federate into DRDC Atlantic's VMSA execution system. As well, the software design and technical details are explained.

This document covers version 1.0 of the SIMDIS Federate.

Résumé

Ce logiciel fédérateur réalise une passerelle entre les simulations VMSA (Virtual Maritime Systems Architecture) et SIMDIS, le logiciel de visualisation 3D du Naval Research Laboratory. Il est en mesure d'enregistrer des simulations VMSA dans le but de les visualiser ultérieurement au moyen de SIMDIS, et de communiquer avec ce dernier afin de réaliser une visualisation en temps réel.

Comme VMSA ne comporte pas de composante de représentation visuelle, le logiciel fédérateur se sert d'un fichier dictionnaire XML pour établir un lien entre les entités composites de VMSA (navires, sous-marins, aéronefs, etc.) et les plates-formes SIMDIS. Le dictionnaire établit une relation entre des entités spécifiques d'une part, et des types SIMDIS et des fichiers modèles 3D d'autre part.

Ce document décrit le logiciel fédérateur et montre comment l'intégrer dans le système d'exécution VMSA de RDDC Atlantique. Il explique en outre la conception du logiciel et présente des détails techniques.

Ce document décrit la version 1.0 du logiciel fédérateur SIMDIS.

Executive Summary

Report title

Gillis, A. D.; 2005; SIMDIS VMSA Federate: User guide and technical description; DRDC Atlantic TM 2005-026;Defence R&D Canada - Atlantic; Unclassified.

Background

SIMDIS is a 3D visualization tool created by the US Navy's Naval Research Laboratory (NRL) and available free for defence purposes. Sponsorship by an individual or agency within the US military is required to obtain the account needed to download the software. For more information visit the SIMDIS site at the URL below:

https://simdis.nrl.navy.mil/

The Virtual Maritime Systems Architecture (VMSA) is a framework for distributed simulations in the maritime environment, based on the High Level Architecture (HLA). VMSA was originally developed by the Australian Defence Science and Technology Organisation (DSTO) and is now in use by all of The Technical Co-operation Program (TTCP) countries including Canada.

The SIMDIS application was originally designed to visualize telemetry data in real time for missile test ranges. To allow the interface between telemetry equipment and SIMDIS a communication application programming interface (API) called Display Client Server protocol (DCS) was developed as part of SIMDIS. DCS is a client-server architecture in which the SIMDIS application acts as a client able to receive data from any DCS server.

Principal results

The SIMDIS VMSA Federate described in this document is a gateway allowing the SIMDIS application to provide visualization for VMSA simulations. The federate joins VMSA federations and acts as a DCS server that the SIMDIS application, acting as a DCS client, may connect to. The federate always sends "simulation truth" via DCS so all platforms involved in the simulation can be seen at all times. Any number of DCS clients may receive the information from the server.

While the federate is most often used for visualization of a simulation in progress it can also be used in a stand-alone mode with no DCS clients. In this mode it stores all of the required information in a SIMDIS compatible ASCII Input (ASI) file. This file can be read by SIMDIS and used to review the simulation output.

Significance of results

As SIMDIS is the main visualization package in use at the Canadian Forces Maritime Warfare Centre (CFMWC) this federate provides both a 3D stealth viewer for VMSA federations and interoperability with CFMWC simulations.

Future work

It is intended that the next version of the SIMDIS federate will include more functionality and be compatible with the new versions of DCS and SIMDIS. In particular:

- Pass beam data, especially from the Sonar v3.0.0 VMSA federate,
- Pass gate data when available,
- Upgrade the DCSGateway to make it DCS version 2 compliant,
- And allow for non-multicast DCS connections.

Sommaire

Titre du rapport

Gillis, A. D.; 2005; SIMDIS VMSA Federate: User guide and technical description; RDDC Atlantique TM 2005-026; R & D pour la défense Canada – Atlantique; non classifié.

Contexte

SIMDIS est un outil de visualisation 3D créé par le Naval Research Laboratory (NRL) de la US Navy et il est disponible gratuitement aux fins de la défense. Pour obtenir un compte, nécessaire pour télécharger le logiciel, le parrainage d'une personne ou d'une agence militaire américaine est requis. Pour de plus amples renseignements à ce sujet, visitez le site de SIMDIS à l'adresse suivante :

https://simdis.nrl.navy.mil/

VMSA (Virtual Maritime Systems Architecture) est un cadre de réalisation de simulations distribuées dans l'environnement maritime, basé sur l'architecture HLA (High Level Architecture). À l'origine, VMSA a été développé par la DSTO (Defence Science and Technology Organisation) australienne, et il est maintenant utilisé par tous les pays qui participent au Programme de coopération technique (PCT), notamment le Canada.

À l'origine, l'application SIMDIS avait été conçue pour visualiser en temps réel des données de télémétrie dans des polygones de tir de missiles. Afin de permettre d'interfacer l'équipement de télémétrie et SIMDIS, une interface de programmation d'applications (API) de communication appelée DCS (Display Client Server protocol) a été développée dans le cadre de SIMDIS. DCS possède une architecture client-serveur dans laquelle l'application SIMDIS fait office de client pour la réception des données émanant de n'importe quel serveur DCS.

Principaux résultats

Le logiciel fédérateur SIMDIS VMSA décrit dans ce document est une passerelle qui permet de visualiser des simulations VMSA au moyen de l'application SIMDIS. Il permet de réunir entre elles des fédérations VMSA et fait office de serveur DCS auquel l'application SIMDIS, qui joue le rôle de client DCS, peut être reliée. Le logiciel fédérateur transmet toujours les données de simulation via DCS, de sorte que toutes les plates-formes concernées par la simulation peuvent être visualisées en tout temps. Le serveur peut transmettre des informations à un nombre arbitraire de clients DCS.

Bien que ce logiciel fédérateur soit la plupart du temps utilisé pour visualiser une simulation en cours, on peut également s'en servir en mode autonome, sans client DCS. Dans ce mode, il stocke toutes les informations requises dans un fichier d'entrée ASCII (ASI) compatible SIMDIS. Ce fichier peut être lu par SIMDIS et utilisé pour examiner les résultats de la simulation.

Importance des résultats

Comme SIMDIS est le principal logiciel de visualisation utilisé par le Centre de guerre navale des Forces canadiennes (CGNFC), ce logiciel fédérateur constitue un visualisateur furtif 3D pour les fédérations VMSA, et il assure l'interopérabilité avec les simulations du CGNFC.

Travaux ultérieurs

On prévoit que la prochaine version du logiciel fédérateur SIMDIS comprendra des fonctionnalités enrichies et qu'elle sera compatible avec les nouvelles versions de DCS et SIMDIS. En particulier :

- Transfert de données de faisceau, émanant en particulier du logiciel fédérateur Sonar v3.0.0 VMSA;
- Transfert de données de passerelle le cas échéant;
- Mise à niveau de la passerelle DCS pour la rendre compatible avec la version 2 de DCS;
- Prise en charge de connexions DCS non-multidestinataire.

Table of contents

List of figures

List of tables

Document revision history

Table 1. *Document revision history.*		
DATE	**VERSION**	**SUMMARY OF CHANGES**
30 August 2005	1.0	First release, covering version 1.0 of the SIMDIS federate

This page intentionally left blank.

1. Introduction

SIMDIS is a 3D visualization tool created by the US Navy's Naval Research Laboratory (NRL) [1] and available free for defence purposes. Sponsorship by an individual or agency within the US military is required to obtain the account needed to download the software. For more information visit the SIMDIS site:

https://simdis.nrl.navy.mil/

Note: the "s" after HTTP in the above uniform resource locator (URL) is required to access the NRL site.

The Virtual Maritime Systems Architecture (VMSA) [2] is a framework for distributed simulations in the maritime environment, based on the High Level Architecture (HLA). VMSA was originally developed by the Australian Defence Science and Technology Organisation (DSTO) and is now in use by all of The Technical Co-operation Program (TTCP) countries including Canada.

The SIMDIS application was originally designed to visualize telemetry data in real time for missile test ranges. To allow the interface between telemetry equipment and SIMDIS a communication application programming interface (API) called Display Client Server protocol (DCS) was developed as part of SIMDIS. DCS is a client-server architecture in which the SIMDIS application acts as a client able to receive data from any DCS server.

The SIMDIS VMSA Federate described in this document is a gateway allowing the SIMDIS application to provide visualization for VMSA simulations. The federate joins VMSA federations and acts as a DCS server that the SIMDIS application, acting as a DCS client, may connect to. The federate always sends "simulation truth" via DCS so all platforms involved in the simulation can be seen at all times. Any number of DCS clients may receive the information from the server.

While the federate is most often used for visualization of a simulation in progress it can also be used in a stand-alone mode with no DCS clients. In this mode it stores all required information in a SIMDIS compatible ASCII Input (ASI) file. This file can be read by SIMDIS and used to review the simulation.

This document is comprised of two main sections; the first (section 2) is a user manual for this federate and the second (section 3) provides the technical description.

2. User Manual

2.1 Introduction

The SIMDIS federate allows you to use the US Navy's SIMDIS application to record a VMSA federation's simulation truth and/or watch it in 3D as it runs. Only the composite entity data is recorded and viewed. It is intended that later versions will allow beam patterns and gates to be displayed as well.

The SIMDIS application is a complicated piece of software in its own right and is not explained here. Please see the SIMIDS documentation and web site for more information about using SIMDIS effectively:

https://simdis.nrl.navy.mil/

The SIMDIS VMSA Federate described in this document is a gateway allowing the SIMDIS application to provide visualization for VMSA simulations. The federate joins a VMSA federation and acts as a data logger for later display with the SIMDIS application, and can also simultaneously act as a DCS server for any number of DCS clients (the SIMDIS application is a DCS client). Figure 1 shows how information is passed from the VMSA federation to DCS clients.

The federate subscribes to all CompositeEntity objects, and records their position/orientation, information. Whenever a new entity is detected, or changes occur in position/orientation the federate broadcasts this information to all connected DCS clients as platform data. The federate always sends "simulation truth" via DCS so all platforms involved in the simulation can be seen at all times.

The SIMDIS application also requires general information such as descriptions of the platforms, where the event took place, the wind speed, etc. The headers which communicate this information are created from data discovered in the VMSA federation, items passed by the command line arguments at start-up, a platform dictionary file, and a configuration file. These parameters and files are described in detail later in this document. The header information is sent to all DCS clients when they first connect to the federate. For a full description of the data required by the SIMDIS application please see the SIMDIS on-line manual [1].

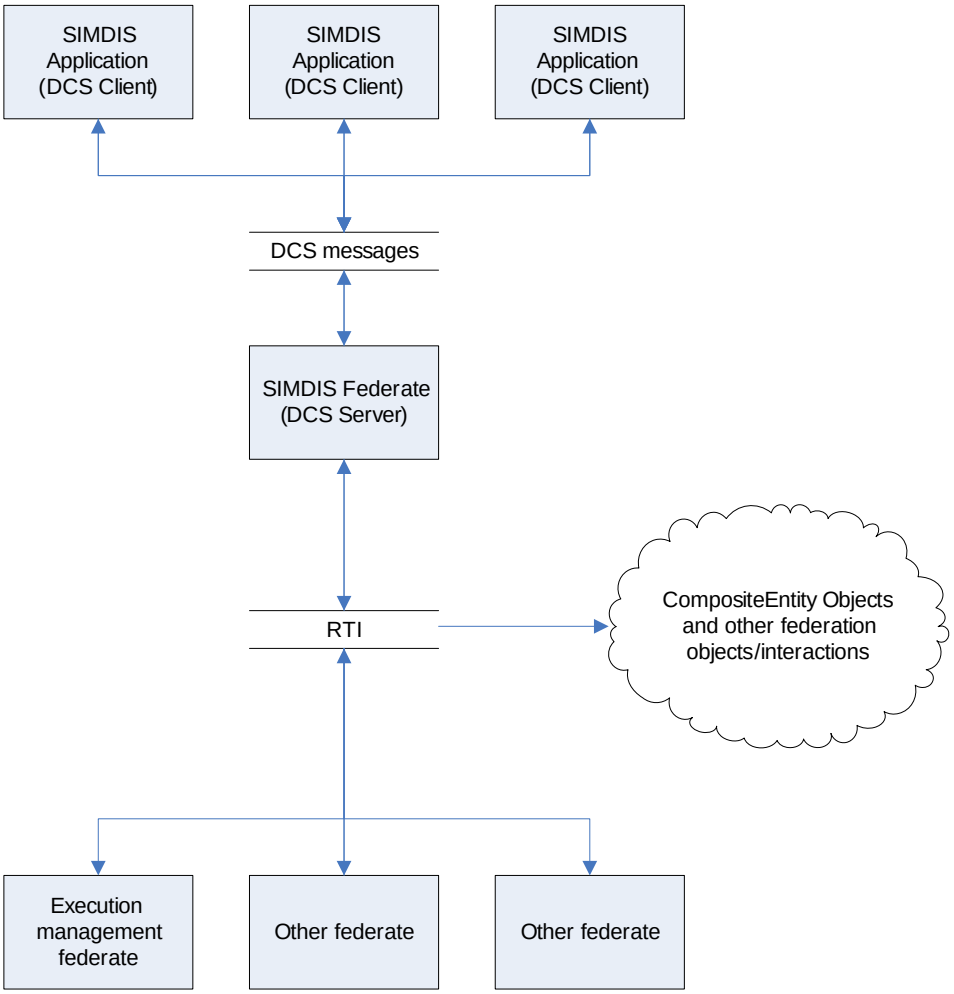


Figure 1. Diagram showing how the SIMDIS federate communicates with a VMSA federation and DCS clients.

2.1.1 Version History

Version 1.0 is a fully functional federation logger and real-time viewer. It creates a SIMDIS readable ASI file by watching a federation and also allows you to connect DCS clients to view the data in real-time.

2.1.2 Federate Compliance

Table 2. Federate Compliance

	VERSION	COMMENTS
SIMDIS Federate	1.0.0	Supports both DCS and stand-alone ASI file creation.
Programming Language	Java	1.4.2 SE SDK
IDE	Eclipse 3.0 M5	
VMS-FOM	3.0.0	
RTI	DMSO 1.3NG4	
Platform	Windows 2000	
DCS	1.0	
SIMDIS	8.5.1, 8.6	Tested with both versions

2.2 Installation

2.2.1 System Requirements

SIMDIS is a 100% Pure Java Federate and should run on any platform for which VMSA base classes and Java bindings for the RTI are available.

We have not attempted to determine the minimum system requirements, but the lowest-end machine it has been run on to date is:

Table 3. *System Requirements.*

ITEM	DETAILS
Make and model	Dell Latitude D800
CPU	Intel Centrino 1.7GHz
Memory	1.0 GB DDR RAM
Network connection	10/100 Switched Ethernet
Operating System	Windows 2000 Professional

This configuration was more than capable of handling the demands of the federate and of the SIMDIS application. This federate does not make large demands on either the processor or memory and should run on lower-end machines.

2.2.2 Installation Instructions

The installation zip file includes all the executables, Java Archive (JAR) files, and documentation specific to this federate in a hierarchy of sub-directories. Simply unzip the file to your federate directory. For example, in the Virtual Combat Systems (VCS) group at Defence R&D Canada – Atlantic (DRDC Atlantic) unzip the installation file to:

V:\DRDC-Simulations\apps\vmsa\federates

This will create the following directory structure:

V:\DRDC-Simulations\apps\vmsa\federates\simdis-1.0.0-vmsfom-3.0.0

|-> bin

|-> doc

|-> lib

|-> src

"bin" holds the configuration file, dictionary file, and start batch file. The complete list of files is given in table 4.

Table 4. *Files included in the "bin" directory.*

FILE	NOTES
simdis.bat	A batch file to start the federate.
simdis-som.xml	The federate's simulation object model (SOM) file.
config.xml	A sample configuration file.
dictionary.xml	A sample dictionary file.
simdisconfig.xsd	The extensible mark-up language (XML) schema for the configuration files.
PlatformDictionary.xml	The XML schema for dictionary files.

"doc" holds the Javadoc Hypertext ,mark-up language (HTML) files that describe the source and this document (as PDF). The most important files in this directory are given in table 5.

Table 5. *Files included in the "doc" directory.*

FILE	NOTES
TECH NOTE VERSION OF MANUALS.PDF	This document
Index.html	The starting point for the Javadocs

"lib" holds all required JAR files, including the one for this federate. The files in this directory are given in table 6.

Table 6. *Files included in the "lib" directory.*

FILE	NOTES
Log4j-1.2.8.jar	Required for some debug data statements.
Resolver.jar	Required to parse XML.
Simdis-1.0.0-vmsfom-3.0.0.jar	The Jar file containing the SIMDIS federate code.
Simdis-jdcs-1.0.0.jar	The jar file from NRL containing the classes required for DCS.
xercesImpl.jar	Required to parse XML.
xml-apis.jar	Required to parse XML.
xmlParserAPIs.jar	Required to parse XML.

"src" holds the Java source code. The classes are well described in the Javadocs and elsewhere in this document (see sections 3.3 and 3.6).

2.3 Start-up

Starting a SIMDIS federate is normally done with a set of batch files. There are several systems for doing this but this document will only refer the method currently in use at DRDC Atlantic.

2.3.1 The command line

The command line needed to run the SIMDIS federate is:

```
java -DVMSBC_ROOT=%VMSBC_ROOT%
-
DVMSBC_JAVA_VERSION=%VMSBC_JAVA_VERSIO
N%
-classpath %SIMDIS_CLASSPATH%
ca.gc.drdc_rddc.atlantic.vcs.simdis.Simdis
]%CMD_LINE_ARGS%
```

All the options must appear on a single line when using a batch file to start the federate. The environment variables %VMSBC_ROOT%, %VMSBC_JAVA_VERSION%, %SIMDIS_CLASSPATH%, and %CMD_LINE_ARGS% are set using the system of batch files that runs the command line. The arguments are explained in table 7.

Table 7. *Explanation of command line items for starting the SIMDIS federate.*

ITEM	NOTES
DVMSBC_ROOT	This argument specifies the root directory for the VMSBC base-classes.
DVMSBC_JAVA_VERSION	This argument specifies which version of the VMSBC base classes are required. For this version of the SIMDIS federate, version 1.0.2 is required.
Classpath	The Java class path needed to include all the required libraries.
ca.gc.drdc_rddc.atlantic.vcs.simdis.Simdis	The package and class for the entry point to the SIMDIS federate. The package is called “ca.gc.drdc_rddc.atlantic.vcs.simdis”, and the main class is “Simdis”.
%CMD_LINE_ARGS%	This environment variable is the set of command line arguments that are passed to the SIMDIS federate. Command line arguments to the federate do not need to be passed as an environment variable, but this is how it is done with the system currently in use at DRDC – Atlantic (see section 2.3.2).

2.3.2 Command line arguments

The SIMDIS federate takes a number of command-line arguments, and with the exception of –d, they are all required. Each command line argument is made up of a switch and data. For example:

-d 4

sets the debug level for the base classes to 4. The full list of command line arguments is given in table 8.

Table 8. *Command line arguments accepted by the SIMDIS federate.*

ARGUMENT	EXPLANATION
-fx	The name of the federation; required.
-fn	The name of the federate; required.
-fed	The name of the FED file to use; required.
-som	The name of the SOM file to use; required.
-d	The debug level. This is optional and the argument is any positive integer or 0. SIMDIS does not make much use of this flag, but the base-classes do report debug information based on its value. In general, the higher the number, the more debug information you will see.

An example command line argument string is given below:

-fx SIMDIS_DEBUG
-fed C:\federation\vmsfom-3.0.0.fed
-fn SIMDIS -c demo.xml –d 2

Here the federation name is “SIMDIS_DEBUG”, the FED file is “C:\federation\vmsfom-3.0.0.fed”, the federate name is “SIMDIS”, the configuration file is “demo.xml” and the debug level for the base classes is set to 2.

2.3.3 Configuration Files

The SIMDIS federate requires two configuration files, one to define options for the federate (Configuration File), the other to define mappings between the CompositeEntity objects in VMSA and their visual representation in SIMDIS (Platform dictionary).

Both files are XML and schemas are provided for validation (see table 3 for file locations). Using these schemas with a good XML editor to reduce errors in the configuration is highly recommended. While the schemas cannot catch every possible error they can catch missing elements and out-of-range values. They are included in the "bin" directory of the installation.

2.3.3.1 SIMDIS configuration file

The schema file (SIMDISConfig.xsd) is available in the "bin" directory of the SIMDIS federate installation, and should be used to validate your own configuration files. The valid XML tags for these files are explained below.

DictionaryFileName: this parameter defines the file name for the Platform Dictionary data. The file name must have the extension "XML".

TemporaryFileName: the name of the file to use for writing platform data during the simulation.

ASIFileName: the name of the ASI file that is generated at the end of the simulation. The file name must have the extension "ASI".

Classification: the security classification. Choices are restricted to those allowed in the SIMDIS documentation, namely Unclassified, For Official Use Only, Confidential, Secret, Secret – WN, Secret – NF, Secret – NC, Secret – OC, Secret – PR, Top Secret. All the SIMDIS application does with the classification is display it at the top and bottom of the screen. It is marked as optional in the ASI v4.0 manual [1], so it may be possible to enter any text for this tag (if the schema is not used), but this has not been tested.

UseDCS: whether or not the DCS server should be started and used; true or false. If this is set to false only the ASI file will be created. If it is set to true then the ASI file is created and DCS clients can connect to watch the simulation in real time.

DigitalElevationMap (optional): each of these tags allows you to specify a Digital Elevation Data (DED) file to use. You may specify 0 or more DED files by adding these tags. These files are large and will slow down SIMDIS' loading and rendering, so only use the ones required.

WorldVectorShoreLineMap (optional): each tag allows you to add a World Vector Shoreline (WVS) file for SIMDIS to load. You may specify 0 or more WVS files by adding these tags. These are also (usually) large files so only specify the ones required.

GOGFile (optional): these are graphic over-lay files that SIMDIS can use to show more information. You may specify 0 or more by adding these tags. The same caution about size applies to General Overlay Graphics (GOG) files as for WVS and DED files.

SoundFile (optional): this tag holds the name of a sound file you'd like to play, synched to SIMDIS time. SIMDIS can only handle one file so this tag may appear once, or not at all. There are also two attributes that must be set:

StartTime: the simulation time (in seconds) from the beginning of the data to start playing the file.

EndTime: the simulation time (in seconds) from the beginning of the data to stop playing the sound file.

ViewFile (optional): the name of a SIMDIS eye position file. These files must be created within SIMDIS and exported. Only one file is allowed per ASI.

DataKeyLock (optional): a SIMDIS license key. If this is present then only the copy of SIMDIS with that license key will be able to export binary multimedia (FCT) files of the data as ASI. Only one of these tags is allowed.

WindSpeed (optional): the wind speed, in metres/second. This is not part of the VMSA FOM, so unless there is a federate using wind speed as part of its internal calculations it shouldn't be used.

WindAngle (optional): the wind direction in degrees. The direction is the true bearing the wind is blowing from). This is not part of the VMSA FOM, so unless there is a federate using wind speed as part of its internal calculations it shouldn't be used.

DCSServerName (optional): the name of the DCS discovery server. This is what appears in the server browse dialog within the SIMDIS application. If this tag is not present the server name will be "SIMDIS Federate".

An example configuration file is given in table 9.

Table 9. *Example SIMDIS configuration file.*

```
<?xml version="1.0" encoding="UTF-8"?>
<SIMDISConfiguration xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="SIMDISConfig.xsd">
    <DictionaryFileName>v:\drdc-
simulations\federations\Apollo\ini\Simdis\apollo_dictionary.xml</DictionaryFileName>
    <TemporaryFileName>v:\drdc-
simulations\federations\Apollo\output\SIMDIS\platform_data_temp.txt</TemporaryFileName>
    <ASIFileName>v:\drdc-simulations\federations\Apollo\output\SIMDIS\apollo.asi</ASIFileName>
    <Classification>UNCLASSIFIED</Classification>
    <UseDCS>true</UseDCS>
    <WorldVectorShoreLineMap>world40.wvs</WorldVectorShoreLineMap>
</SIMDISConfiguration>
```

2.3.3.2 PlatformDictionary file

The platform dictionary file provides mappings between the CompositeEntity objects in the federation and the particular 3D models that will represent them in the SIMDIS application.

The 3D files referenced in the dictionary file must be present and properly located on every computer that will be running SIMDIS. SIMDIS uses a default white cube to represent platforms when it doesn't find the 3D file specified. How and where to place these models is explained in the SIMDIS documentation [1].

SIMDIS uses a proprietary 3D file format. While a converter tool is included the only files it con covert from are Designer's Workbench (DWB). SIMDIS proprietary files must end with the extension "opt" and this is enforced by the dictionary schema file.

The schema file (PlatformDictionary.xsd) is available in the "bin" directory of the SIMDIS federate installation, and should be used to validate dictionary files.

Default: every dictionary must have a single "default" tag. This tag defines the information that SIMDIS will use if there are no matching "Platform" tags for a particular CompositeEntity. The content of this tag is the file name of one of the SIMDIS 3D models.

Platform: there can be any number of "Platform" tags, or none. The Platform tag has a single attribute, "Name" which is the VMSA Object Instance Name (not the CompositeEntityName attribute) that maps to the enclosed type and file. Each platform tag has the following attributes:

Type: the SIMDIS "type"

FileName: the name of the 3D file to display.

An example dictionary file is given in table 10.

Table 10. *Example SIMDIS dictionary file.*

```
<?xml version="1.0" encoding="UTF-8"?>
<Platform3DFiles xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="PlatformDictionary.xsd">

        <Default Type="unknown" FileName="NTDSUnknownUnknown2D.opt" />

        <!-- Friendly Warships -->
       <Platform Name="HMCS_Halifax" Type="ship" FileName="cpf.opt" />
       <Platform Name="HMCS_Vancouver" Type="ship" FileName="cpf.opt" />
       <Platform Name="PNS_M_Saad" Type="submarine" FileName="ussr_kilo.opt" />

        <!-- Friendly aircraft -->
        <Platform Name="Hunter" Type="aircraft" FileName="p-3c_orion.opt" />

        <!-- Hostile forces -->
        <Platform Name="IIS_Bayandor" Type="ship" FileName="ussr_tarantul.opt" />
        <Platform Name="Smuggler" Type="ship" FileName="boghammer.opt"/>

        <!-- Commercial Aircraft -->
        <Platform Name="AF132" Type="aircraft" FileName="boeing_747-400_no_windows.opt" />
        <Platform Name="AQ467" Type="aircraft" FileName="boeing_757-200.opt" />
        <Platform Name="BA453" Type="aircraft" FileName="boeing_707-320.opt" />

        <!-- Commercial shipping -->
        <Platform Name="Maersk_Estelle" Type="ship" FileName="esso_sabba_tanker.opt" />
        <Platform Name="Mars_Glory" Type="ship" FileName="bulkcarrier.opt" />
        <Platform Name="Maersk_Dubai" Type="ship" FileName="containership.opt" />

</Platform3DFiles>
```

2.4 Operation

2.4.1 Graphical User Interface (GUI)

There is no GUI for the SIMDIS federate.

2.4.2 Debugging Output

There are two sources of debugging output; the SIMDIS federate and the base classes. There are no switches for SIMDIS debugging, but messages are sent to STDOUT and STDERR (normally the console window) when interesting events and errors happen.

Messages are generated by the SIMDIS federate when (in no particular order):

- there is a problem with the command line arguments,
- there is an error opening or reading the configuration file,
- the ASI header information is written,
- there is a problem creating the temporary file used to record simulation data,
- the ASI file is created,
- there is a problem writing the ASI file,
- there is a problem shutting down,
- the federate is getting ready to shut down,
- there is a problem talking to the Run-Time Infrastructure (RTI)
- important (to the SIMDIS federate) federation events occur,
- new CompositeEntity objects are discovered.

The base class debugging is controlled with the command line flag “-d x” where “x” is a positive integer (see section 2.3.2, Command line arguments). More information about base class debugging information can be found in their Javadocs [6].

2.5 Federate Behaviour

The SIMDIS federate joins a federation and watches for creation, removal, and movement of the CompositeEntity objects in the simulation. It logs all physical domains (Air, Sea Surface, Sub Surface, and Land) and records the position, orientation, and speed of all CompositeEntity objects in a temporary file.

A valid SIMDIS ASI file (version 4) is always created on completion of the simulation and can be read by the SIMDIS application to visualize the simulation off-line. The header and platform information is determined from the configuration and dictionary files which are described in section 2.3 of this document.

If the DCS gateway is in use, the federate will allow clients to join and leave at will. All simulation data is sent to both the clients and the ASI file mentioned above. When the simulation ends the client connections are terminated before the federate exits.

2.5.1 Time Management Policies

This version is both time regulating and time constrained. The look-ahead is hard coded to 1.0.

2.5.2 Models

This is a logging federate. It does not publish any objects, and the only model it has is an internal representation of CompositeEntity objects.

2.5.2.1 Simulation Object Model (SOM)

Table 11. *Simulation Object Model.*

CLASSES	PUBLISH	SUBSCRIBE
CompositeEntity	No	Yes
SeaSurface	No	Yes
SubSurface	No	Yes
Air	No	Yes
Land	No	Yes
Interactions		
ExecutionManagment	Yes	Yes
InitialiseFederate	Yes	Yes
CreateEntity	Yes	Yes
ExecutionManagementError	Yes	Yes
TerminateIteration	Yes	Yes
TerminateFederation	Yes	Yes
RemoveEntity	Yes	Yes

2.5.3 Execution Management

SIMDIS must be included in the “Participating Federates” section of the Virtual Maritime System Execution Manager (VMSEM) federate script file, just like any other federate.

It also responds properly to the “End Federation”. It uses the “End Federation” interaction as a trigger to create the ASI file from the scenario information and the saved position data. It has not been tested with the “End Iteration” interaction.

2.5.4 Known Faults

This version is time regulating and does not need to be.

The ASI file creation process depends on receiving a correct “End Federation” signal from the execution manager.

The federate hasn’t been tested with multiple iterations (loops) and may not work correctly in that mode (this has not been tested). At the least, the ASI file is likely to be overwritten during each iteration.

The SIMDIS application and other DCS clients may not be able to connect to the federate’s DCS server if federation time has not begun to advance. A simple work-around is to connect only after the federation time is 1 or greater.

2.5.5 Known Limitations

This version does not handle beam or gate data, or beam events.

The 3D models included with the SIMDIS application are limited and the proprietary file format makes it difficult to supplement them. This is not a limitation of the federate per se, but of the SIMDIS application.

2.6 Bug Reports

Please report any bugs found with this federate to:

Allan Gillis
Virtual Combat Systems group,
Maritime Information and Combat Systems section,
DRDC Atlantic
(902) 426 - 3100 ext 203
allan.gillis@drdc-rddc.gc.ca

3. Federate Technical Description

3.1 Introduction

This is a VMSA utility federate that logs position and orientation data for CompositeEntity objects and does two things with it. First it creates a SIMDIS compatible text file (ASI) for later viewing. Second, it communicates with the US Navy's SIMDIS application to display the CompositeEntity's position and orientation immediately as the federation executes.

The SIMDIS federate was developed using Java (SDK 1.4.2) and the Eclipse 3.0M5 IDE. The federate code is adapted from that of DRDC Atlantics Sonar federate (version 2) which was in turn based on DRDC Atlantic's TMA federate, and the original DSTO Sonar federate.

3.1.1 Federate Compliance

Table 12 summarizes the compliance for this federate.

Table 12. *Federate Compliance.*

	VERSION	COMMENTS
SIMDIS Federate	1.0.0	This federate creates an ASI file and communicates via DCS with the SIMDIS application.
Programming Language	Java (SDK 1.4.2 and 1.5.0)	It was originally written with v1.4.2, but final modifications were done using v1.5.0)
VMS-FOM	3.0.0	
RTI	DMSO RTI1.3NG-V4	
Platform	Windows 2000/XP, Intel processors	Since this is a pure Java app and makes no use of native libraries it should run on any Java compatible platform. It has only been tested on Windows 2000 and XP.
DCS	1.0	
SIMDIS Application	8.5.1 and 8.6	

3.2 Federate Description

The SIMDIS federate subscribes to all VMSA CompositeEntity objects and records their position and orientation over time to a properly formatted, complete SIMDIS ASI file. It also (optionally) communicates with clients via DCS to show this data as the simulation is unfolding.

ASI files and DCS clients need three basic types of information:

1. Header information about the scenario,
2. Header information about the platforms (CompositeEntity objects),
3. and data describing the position, orientation, and motion of the platforms (CompositeEntity objects).

The flow during execution is shown in figure 2.

SIMDIS needs to know the geographic location (latitude and longitude) to use as a reference for the scenario. This information is not directly available from the execution manager or the scenario file, so the federate uses the position in the first ReflectAttributeValues event that it receives from the RTI. While this is not ideal, it does ensure the reference scenario is in the proper vicinity.

The ASI file is only created once the federation has ended. While running, all platform data is written to a text file (the name is specified in the configuration file) and at completion the ASI file is made by writing out correct scenario and platform header sections followed by the data in the temporary file.

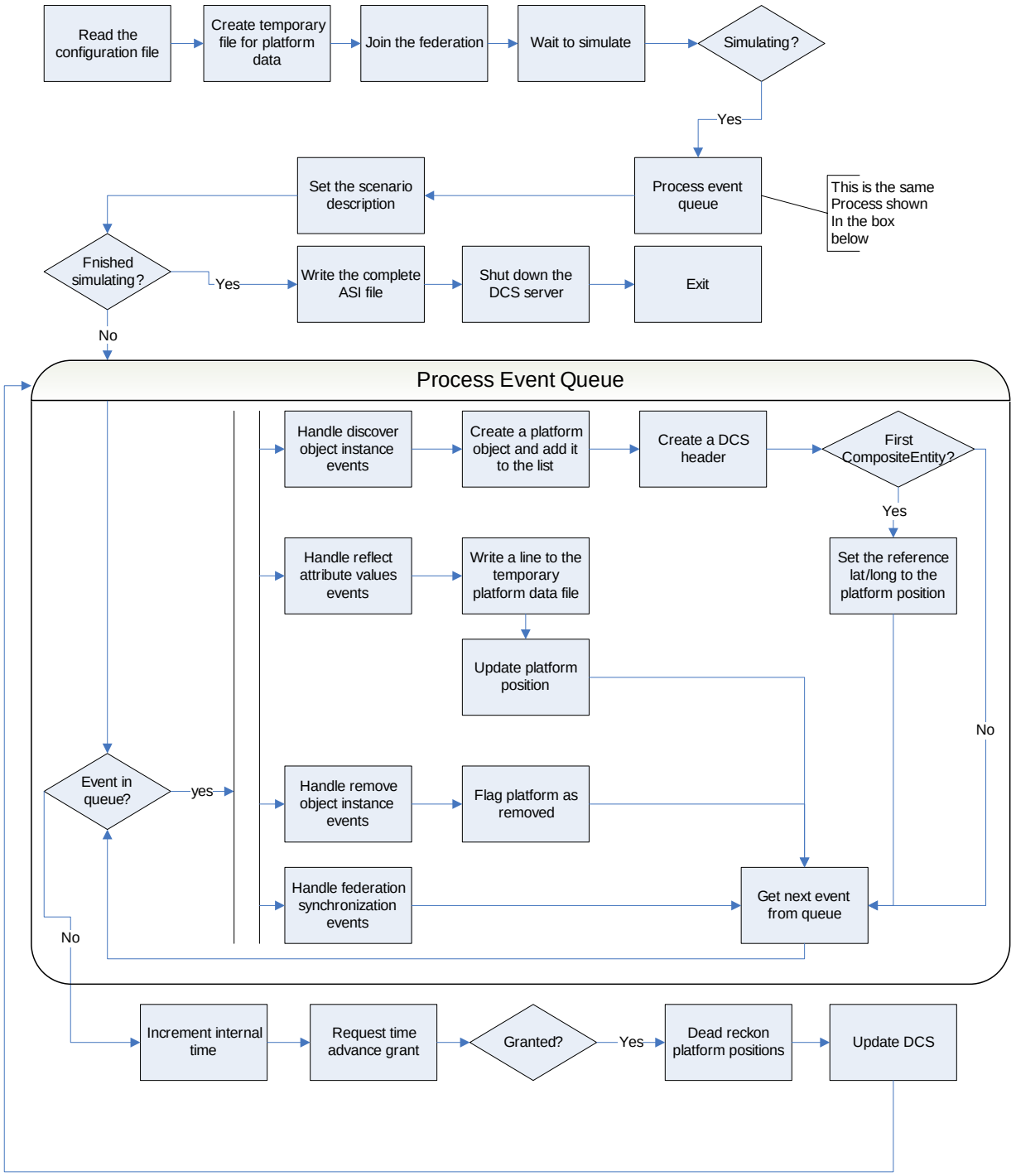


Figure 2. Program Execution Flow.

3.3 Functional Description of Models and Classes

Since this is a logging utility federate that performs no simulation, it has no models. The classes that implement the federate are described below, and a class diagram is provided in figure 3.

3.3.1 Simdis

This is the federate. It encapsulates the entry point to the application and defines the expected methods as specified by the VMSA architecture. It handles all communications with the RTI. It also creates the Scenario object, Platform Objects, Dictionary, and the DCSGateway (see below). In addition it deals with reading the configuration file.

3.3.2 Scenario

This class deals with capturing and formatting the scenario information. It provides a function to output a properly formatted ASI scenario header and it communicates with the DCSGateway as required.

3.3.3 Platform

The federate maintains a list of these objects. A new one is created for every DiscoverObjectInstance event involving CompositeEntity objects (ComponentEntity objects are ignored). These objects handle the nuts and bolts of maintaining the platform data and also communicate with the DCSGateway to create DCS platform headers and data updates when required.

3.3.4 PlatformSnapShot

This class is used to output properly formatted ASI data from what is known about a particular CompositeEntity. Every Platform object has a PlatformSnapShot object to handle these tasks.

3.3.5 Dictionary

The dictionary class is a utility for translating VMSA CompositeEntity attributes into correct SIMDIS 3D mesh files and type designations. It uses a dictionary XML file to store the data needed for this task. The SIMDIS federate creates a single instance of this and passes a reference to Platform objects.

3.3.6 DCSGateway

This class handles all communication with the SIMDIS application (or any other DCS client). It also provides functions that the Platform and Scenario objects use to pass scenario/platform headers and data. The SIMDIS federate creates this object and passes a reference to the Platform and Scenario objects. It also updates the time this object uses to stamp out-going data.

Figure 3. UML class diagram of the SIMDIS federate.

3.4 Integration and Testing

Since the SIMDIS federate only subscribes to simulation objects there is little chance that it will cause conflicts with other federates. However, proper functioning does depend on correct and complete data from other federates, so this was the focus of the VMSA portion of the integration and testing work. The other important area for testing was the ability of the SIMDIS federate to communicate with the SIMDIS application.

To date this federate has been tested in two demonstration federations. The first involved a Canadian Patrol Frigate (CPF), an ANZAC class frigate and several merchant ships in the straights of Malacca. The second was a general VMSA demonstration using most federates available to DRDC Atlantic at the time. The second federation included aircraft, submarines, and surface vessels.

In both federations, versions 8.5 and 8.6 of the SIMDIS application were tested.

3.4.1 First test federation

All vessels were created by the Motion federate [3] via an interaction passed by the Gameboard federate. All vessels were controlled by the Gameboard federate although the CPF was driven around with the helm included with the Horizon HLA Plug-in. Federates and tools used are given in table 13.

Table 13. *Federates and tools used in the first test federation.*

FEDERATE OR TOOL	VERSION
Virtual Marine Systems Execution Manager	1.8.1
Motion	2.3.0
Horizon HLA Plug-in	2.5
Gameboard	1.1
Sonar	2.0
SIMDIS Application	8.5 and 8.6

3.4.1.1 Results

The SIMDIS federate performed as expected, correctly showing the position, movement, and orientation of all CompositeEntity objects. Also, it did not cause any problems for the other federates.

3.4.2 Second test federation

All surface vessels, the submarine, and commercial aircraft were created by the Motion federate via an interaction from the JBoard federate [5]. A single Maritime Patrol Aircraft (MPA) was created by Motion via an interaction from the VMSEM federate. Except for the MPA and a single CPF, all changes in course, speed, altitude, and depth were controlled by the JBoard federate. The MPA was controlled using the IntAircraft federate and Microsoft Flight Simulator 2002, while the CPF was controlled using the Helm federate.

The complete list of federates and federation tools used is given in table 14:

Table 14. *Federates and tools used in the second test federation.*

Federation Spy	1.4
Federation Verification Tool (FVT)	1.3v4
JBoard	1.0.0
Gremlins	7.0
Helm	1.9
Horizon HLA Plug-in	2.5
IntAircraft	1.2.0.ttcp
Motion	2.3.0
Navigation	1.2
SIMDIS Application	8.5 and 8.6
Sonar	3.0.0
TMA	1.1
VMSEM	1.8.1
Virtual Maritime Systems Situation Display (VMSSD)	2.8

3.4.2.1 Results

Some difficulty was seen with the SIMDIS application when trying to connect to the DCS server before the federation time begins to advance. If the SIMDIS application is not connected (via DCS) before the time begins to advance there are no issues. We intend to address this in the next version.

3.5 Future Development

The next version of the SIMDIS federate will include more functionality and be compatible with the new versions of DCS and SIMDIS. In particular we would like to:

- Pass beam data, especially from Sonar 3,
- Pass gate data if it is available,
- Upgrade the DCSGateway to make it DCS version 2 compliant,
- And allow for non-multicast DCS connections.

References

1. SIMDIS Online Manual – ASI v4.0 ASCII File Format, https://simdis.nrl.navy.mil/print/ASI_ASCII_FileFormatPrint.asp

2. Canney, Shane A. (2002), Virtual Maritime Systems Architecture Description Document Issue 2.00. Edinburgh, SA, Australia: Defence Science and Technology Organisation.

3. Cramp, Anthony (2003), Motion Federate v2.3.0-vmsfom-3.0.0 Users's Manual. Defence Science and Technology Organisation.

4. Roger, W. A. (2003), An Enhanced Scenario Script Generator for the TTCP Virtual Maritime Systems Architecture. DRDC Atlantic TM 2003-007. Dartmouth, NS, Canada: Defence R&D Canada – Atlantic.

5. Gillis, Allan D. (2005), JBoard VMSA Federate: User guide and technical description. DRDC Atlantic TM 2005-027. Dartmouth, NS, Canada: Defence R&D Canada – Atlantic.

6. Cramp, Anthony (2003), VMSA Base Classes, Javadocs (included with the distribution of the VMSA base classes). Defence Science and Technology Organisation.

List of symbols/abbreviations/acronyms/initialisms

1.4.2 SE SDK	Version 1.4.2 of the Java Standard Edition (SE) Software Development Kit (SDK)
ASI	**AS**CII **I**nput. A text format for simulation data readable by the SIMDIS application.
DCS	Data Client Server
DED	Digital Elevation Data
DMSO	Defence Modelling and Simulation Office (US Department of Defence)
DND	Department of National Defence
DRDC	Defence Research and Development Canada
FCT	The file extension of SIMDIS' proprietary binary multimedia file format
GOG	General Overlay Graphics
IDE	Integrated Development Environment
NRL	Naval Research Laboratory
OPT	The file extension for SIMDIS' proprietary 3D files.
TTCP	The Technical Co-operation Program
VCS	The Virtual Combat Systems group at DRDC Atlantic
VMS-FOM	The Virtual Maritime Systems Architecture Federation Object Model
WVS	World Vector Shore-line
XML	Extensible Mark-up Language

Glossary

TECHNICAL TERM	EXPLANATION OF TERM
ComponentEntity	In the VMSA FOM these are the objects that represent individual weapon and sensor systems. CompositeEntity objects usually have one or more ComponentEntity objects attached to them.
CompositeEntity	In the VMSA FOM, these are the objects that represent ships, aircraft, submarines, torpedoes, or any real-world item that has multiple systems and/or sensors.
DiscoverObjectInstance event	In HLA, this event occurs whenever a new simulation object is created. Events are sent from the RTI to federates based on their subscription intents.
ObjectInstanceName	In the VMSA FOM, this is the unique name given to a particular instance of an object (in particular CompositeEntity and ComponentEntity objects).
ReflectAttributeValues event	In HLA, this event occurs whenever the attributes of an object change. ReflectAttributeValues events are sent to a federate by the RTI based on the federate's subscription intents.

Index

Distribution list

6 copies, DRDC Atlantic Library

3 copies, B. Dillman, VCS Group, DRDC Atlantic

1 copy, A. Gillis, VCS Group, DRDC Atlantic

1 copy, DRDKIM

1 copy, Maritime Synthetic Environment Coordination Office (MSECO)
CFMWC
P.O. Box 99000, Station Forces
Halifax, NS, B3K 5X5

DOCUMENT CONTROL DATA
(Security classification of the title, body of abstract and indexing annotation must be entered when the overall document is classified)

1. ORIGINATOR (The name and address of the organization preparing the document, Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.)

Publishing: DRDC Atlantic
Performing: DRDC Atlantic
Monitoring:
Contracting:

2. SECURITY CLASSIFICATION
(Overall security classification of the document including special warning terms if applicable.)

UNCLASSIFIED

3. TITLE (The complete document title as indicated on the title page. Its classification is indicated by the appropriate abbreviation (S, C, R, or U) in parenthesis at the end of the title)

SIMDIS VMSA Federate User Guide and Technical Description (U)
SIMDIS VMSA Federate: User guide and technical description

4. AUTHORS (First name, middle initial and last name. If military, show rank, e.g. Maj. John E. Doe.)

Allan D. Gillis

5. DATE OF PUBLICATION
(Month and year of publication of document.)

August 2005

6a NO. OF PAGES
(Total containing information, including Annexes, Appendices, etc.)

48

6b. NO. OF REFS
(Total cited in document.)

6

7. DESCRIPTIVE NOTES (The category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)

Technical Memorandum

8. SPONSORING ACTIVITY (The names of the department project office or laboratory sponsoring the research and development − include address.)

Sponsoring:
Tasking:

9a. PROJECT OR GRANT NO. (If appropriate, the applicable research and development project or grant under which the document was written. Please specify whether project or grant.)

11BK

9b. CONTRACT NO. (If appropriate, the applicable number under which the document was written.)

10a. ORIGINATOR'S DOCUMENT NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document)

DRDC Atlantic TM 2005−026

10b. OTHER DOCUMENT NO(s). (Any other numbers under which may be assigned this document either by the originator or by the sponsor.)

11. DOCUMENT AVAILABILIY (Any limitations on the dissemination of the document, other than those imposed by security classification.)

Unlimited distribution

12. DOCUMENT ANNOUNCEMENT (Any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11), However, when further distribution (beyond the audience specified in (11) is possible, a wider announcement audience may be selected.))

Unlimited announcement

DOCUMENT CONTROL DATA
(Security classification of the title, body of abstract and indexing annotation must be entered when the overall document is classified)

13. ABSTRACT (A brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual.)

(U) This federate provides a software gateway from Virtual Maritime Systems Architecture (VMSA) simulations to SIMDIS, the Naval Research Laboratory's (NRL) 3D visualisation software. The federate can both log VMSA simulations for later viewing with SIMDIS and communicate with SIMDIS for real−time viewing.

Since visual representation is not part of VMSA the federate uses an XML dictionary file to relate VMSA Composite Entities (ships, submarines, aircraft, etc) to SIMDIS platforms. The dictionary relates specific entities to particular 3D model files and SIMDIS types.

This document describes the federate software and how to integrate the federate into DRDC−Atlantic's VMSA execution system. As well, the software design and technical details are explained.

This document covers version 1.0 of the SIMDIS Federate.

(U) Ce logiciel fédérateur réalise une passerelle entre les simulations VMSA (Virtual Maritime Systems Architecture) et SIMDIS, le logiciel de visualisation 3D du Naval Research Laboratory. Il est en mesure d'enregistrer des simulations VMSA dans le but de les visualiser ultérieurement au moyen de SIMDIS, et de communiquer avec ce dernier afin de réaliser une visualisation en temps réel.

Comme VMSA ne comporte pas de composante de représentation visuelle, le logiciel fédérateur se sert d'un fichier dictionnaire XML pour établir un lien entre les entités composites de VMSA (navires, sous marins, aéronefs, etc.) et les plates formes SIMDIS. Le dictionnaire établit une relation entre des entités spécifiques d'une part, et des types SIMDIS et des fichiers modèles 3D d'autre part.

Ce document décrit le logiciel fédérateur et montre comment l'intégrer dans le système d'exécution VMSA de RDDC Atlantique. Il explique en outre la conception du logiciel et présente des détails techniques.

Ce document décrit la version 1.0 du logiciel fédérateur SIMDIS.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (Technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus, e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

(U) High Level Architecture; HLA; Virtual Maritime Systems Architecture; VMSA; SIMDIS; User Guide; Manual; Federate; Technical Description

This page intentionally left blank.

Defence R&D Canada

Canada's leader in defence
and National Security
Science and Technology

R & D pour la défense Canada

Chef de file au Canada en matière
de science et de technologie pour
la défense et la sécurité nationale

www.drdc-rddc.gc.ca