
Simulation Foundations, Methods and Applications

Series editor

Louis G. Birta, University of Ottawa, Canada

Advisory Board

Roy E. Crosbie, California State University, Chico, USA

Tony Jakeman, Australian National University, Australia

Axel Lehmann, Universität der Bundeswehr München, Germany

Stewart Robinson, Loughborough University, UK

Andreas Tolk, Old Dominion University, USA

Bernard P. Zeigler, University of Arizona, USA

More information about this series at <http://www.springer.com/series/10128>

Okan Topçu · Halit Oğuztüzün

Guide to Distributed Simulation with HLA

Okan Topçu
Middle East Technical University,
Northern Cyprus Campus (METU NCC)
Kalkanlı, Güzelyurt, Mersin 10
Turkey

Halit Oğuztüzün
Middle East Technical University
Ankara
Turkey

ISSN 2195-2817 ISSN 2195-2825 (electronic)
Simulation Foundations, Methods and Applications
ISBN 978-3-319-61266-9 ISBN 978-3-319-61267-6 (eBook)
DOI 10.1007/978-3-319-61267-6

Library of Congress Control Number: 2017943250

© Springer International Publishing AG 2017

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

*To Tuğçe—the love of my life
and*

To Oğuz—the meaning of my life

Okan Topçu

*To Serpil, Çerağ and Ozan
My circle of love*

Halit Oğuztüzün

Foreword

I have always enjoyed driving boats, but this was a very special day. For the first time in my life, I was driving a huge ship, the 203 meter long, 2500 passenger cruise ship *Silja Symphony*. I was about to enter the Helsinki harbor when I noticed a small vessel crossing my ship's path. I turned sharply starboard, but nothing happened. I kept turning, and after five seconds the ship reacted and steered sharply toward a quay, completely out of control. In just a few seconds, the ship would crash, causing a major disaster.

Fortunately, all of this took place in a simulator in a ship's pilot school. My heart pounded while I drove the ship, immersed in the virtual environment. A few minutes later, I calmed down over a cup of coffee while the instructor walked through my decisions and mistakes.

Simulations let us experience dangerous situations, and let us learn how to handle them without risking our lives. Training in simulators saves time and money, instead of using real ships or aircraft. Simulators let us try many what-if scenarios, analyze the results, and take better decisions. Simulations take us places where we otherwise cannot go.

Simulation programs were initially developed as monolithic applications by an individual or a group. As simulation technology advanced, several simulation applications could be interconnected. Expertise from several groups can be combined, making it possible to build even more powerful, scalable, and interactive simulations. More complex scenarios can be simulated, scenarios that are more like the real world. Physical equipment, human decision making, sensors, environment, communications, and more can be integrated into one holistic approach, solving important problems in today's complex world.

For more than twenty years, I have been involved in developing distributed simulation systems as well as supporting the evolution of standards for simulation interconnectivity. I have visited a large number of countries, giving advice to distributed simulation projects in places such as Japan, Brazil, Canada, Taiwan, China, USA, and most European countries. Everywhere I've been, I've seen a big demand for know-how about distributed simulation.

It is an honor for me to introduce Drs. Topçu and Oğuztüzün's book, which provides an introduction to simulation as well as valuable insights into the design of distributed simulation systems. This book starts with a history and overview of the

High-Level Architecture (HLA), the leading modern standard for distributed simulation. HLA provides services for information exchange and synchronization between simulations that together form a federation.

To successfully apply HLA to a specific domain, be it pilot training, space mission planning, transportation, or manufacturing, a Federation Object Model, or FOM, is needed. The FOM describes the information that the simulations exchange. The second part of this book covers this interesting area. Today, there are many FOM standardization efforts ongoing around the world, for example, the Space Reference FOM, air traffic management FOMs, the Real-time Platform Reference FOM, road traffic simulation FOMs, and the NATO Education and Training FOM. This part of the book gives a foundation for anyone who wants to understand these FOMs and the architecture of such distributed simulation systems.

In part three, Drs. Topçu and Oğuztüzün show how applications, known as federates, use HLA and how they are structured and developed. While the HLA specification describes each service, this book shows how to put everything together. Various tools and frameworks facilitate HLA development. Drs. Topçu and Oğuztüzün introduce one such framework.

After covering some advanced topics in part four, Drs. Topçu and Oğuztüzün conclude this book with a complete case study. This should be a good time for the reader to gain an understanding of how everything fits together.

Drs. Topçu and Oğuztüzün's book will be very valuable to a large number of simulation developers, ranging from students to advanced practitioners. This book also fills a hole, since an up-to-date book on HLA has been missing for several years. I strongly recommend this book and hope that the reader will enjoy it as much as I did!

2017

Björn Möller
Vice President and Co-founder of Pitch Technologies,
Vice Chair of the IEEE 1516 HLA standard
Linköping, Sweden

Preface

Purpose

This is a hands-on guidebook on distributed simulation (DS), specifically on High Level Architecture (HLA), with a view toward software development. It contains a variety of examples to support learning by doing. It offers practical advice on real-world development issues for the novice engineers and programmers who are entering the field of distributed simulation.

Rationale

The book elaborates the implementation of an HLA federation covering all areas from the object model development to federate application development by exemplifying all the federate interface service areas. More importantly, the federate application development is based on the layered architectural style, which provides a clear separation of concerns. The implementation is based on the latest HLA standard, known as HLA Evolved, yet we point to the related topics with HLA 1.3 specification in the interest of backward compatibility. We also present a running example involving a great deal of variation to illustrate several practical issues of federate application development. The implementation is not based on a specific commercial runtime infrastructure (RTI) software, but rather based on generic and extendible tools that can be obtained freely. One is SimGe, a freeware, which is a fully dressed HLA object model development tool and a code generator. Second is RACoN, which is fundamentally an open-source .NET wrapper for HLA RTI application programming interface (API). The book serves as the only comprehensive reference for both tools by giving downloadable sample case studies with the source code. Moreover, this book contains a unique chapter that shows the practitioner how to employ HLA in (multi) agent-based simulation.

The prominent features are as follows:

- It comes with the tools RACoN and SimGe to enable the reader work at a conceptually high level.

- Numerous step-by-step examples and code snippets help the reader to understand the RTI concepts.
- It includes a running example involving a great deal of variation to illustrate several practical issues of federate application development.
- It includes a larger-scale case study involving multi-agents. The scenario is based on the maritime border surveillance with the help of a fleet of unmanned surface vehicles.
- It includes downloadable sample source code.
- It uses Microsoft .NET platform and the C# programming language for all implemented examples and case study.
- It provides a fast start-up on HLA federation development giving an implemented sample for each service area of the HLA federate interface specification.
- It includes a unique chapter to employ the HLA in multi-agent simulations.
- It covers not only federate application development, but also object model construction.
- Many chapters include questions for review and further study.

Book Overview

The book is structured as follows. Chapter 1 is a high-level introduction to the essential concepts of modeling and simulation, while highlighting DS as the focal area of interest. Chapter 2 introduces the fundamental concepts and the principles of HLA. Chapter 3 takes a look into federation development and presents development guidelines. These three chapters together lay the technical background for federation development.

The remaining chapters constitute three major parts of the book: object model development, federate implementation, and advanced topics. Chapters 4 and 5 present an introduction to object model development and then elaborate on the subject through a case study. Using the object model in the case study, Chaps. 6–8 elucidate the details of implementing a federate application. Last, Chap. 10 considers some advanced topics by discussing the connection between agent-based simulation and HLA. Then, Chap. 11 provides a complete case study to put it all together.

The chapters are structured in a layered way so that the initial chapters include more generalized topics such as M&S and HLA concepts. As we proceed, each chapter brings more specialized topics relying on the knowledge of the preceding ones.

Web Material

The software tools and examples used in the book are freely available on the web.

Audience

This book is intended for students of distributed simulation, based particularly on High Level Architecture. It can be used as a textbook or reference book for an upper undergraduate/lower graduate course, probably named as distributed simulation or distributed interactive simulation.

Final

We believe that the most prominent contribution of this book is to provide a starting point and all-in-one resource for HLA-based distributed simulation development without depending on any commercial tools. The book is complementary, in regard to its implementation slant, to model-based engineering approach for distributed simulations.

Edirne, Turkey
Ankara, Turkey

Okan Topçu
Halit Oğuztüzün

Acknowledgements

We would like to acknowledge our associate editor Simon Rees for smooth editorial assistance.

Okan would like to thank his parents Selime and Bekir for their boundless support.

Halit would like to thank all his graduate students, especially those who graduated from the MODSIM master of science program at METU.

For various chapters of this book, we have adapted parts of the following articles/chapters:

Topçu, O., Durak, U., Oğuztüzün, H. & Yılmaz, L., 2016. Distributed Simulation: A Model Driven Engineering Approach. 1st ed. Cham(Zug): Springer International Publishing. Parts adapted and reprinted with permission from Springer appear in Chaps. 1, 2 and 7.

Contents

Part I Introduction

1	Introduction	3
1.1	Modeling and Simulation	3
1.1.1	What Is Model?	4
1.1.2	What Is System?	5
1.1.3	What Is Simulation?	6
1.1.4	Model to Simulate and Model to Build	7
1.1.5	Simulation Engineering	8
1.1.6	Time and Change	9
1.2	Distributed Simulation	10
1.2.1	SIMNET	12
1.2.2	DIS Protocol	12
1.2.3	Aggregate Level Simulation Protocol	13
1.3	High Level Architecture	13
1.4	Tools	15
1.4.1	SimGe	16
1.4.2	RACoN	16
1.5	Introduction to Case Studies	17
1.5.1	Strait Traffic Monitoring Simulation	17
1.5.2	Maritime Border Surveillance Using Unmanned Surface Vehicles	19
1.6	Book Outline	22
1.6.1	Summary of Chapters	22
1.6.2	Typeface Conventions	23
1.7	Summary	24
1.8	Questions for Review	24
	References	25
2	High Level Architecture	29
2.1	Prelude	29
2.1.1	What Is HLA?	30

2.2	Basic Components	31
2.2.1	Federate and Federation	32
2.2.2	Runtime Infrastructure.	33
2.3	HLA Rules	34
2.3.1	Federation Rules	34
2.3.2	Federate Rules	35
2.4	HLA Data Model and Data Communication	36
2.4.1	HLA Data Communication Pattern	36
2.4.2	Object Exchange: Publish/Subscribe Pattern.	38
2.4.3	Data: Objects, Interactions, and HLA Classes.	39
2.4.4	Object Model Template.	42
2.4.5	HLA Object Models	44
2.4.6	Object Model Modularity	45
2.5	Interface Specification.	46
2.5.1	Federation Management	50
2.5.2	Declaration Management.	54
2.5.3	Object Management	59
2.5.4	Ownership Management	60
2.5.5	Data Distribution Management.	62
2.5.6	Time Management	64
2.6	Full Life Cycle of a Federation Execution	68
2.6.1	Initialization.	71
2.6.2	Operation	71
2.6.3	Termination.	71
2.7	Example Federation Deployment	72
2.8	Federation and Federate States	72
2.8.1	Federation Execution States	73
2.8.2	Federate States.	73
2.9	Summary	74
2.10	Questions for Review	75
	References	77
3	Federation Development At-A-Glance	79
3.1	Process Model	79
3.2	Federation Development and Guidelines	81
3.2.1	Federation Design.	81
3.2.2	Federate Design	87
3.2.3	Federate Implementation	89
3.2.4	Target Platform	89
3.2.5	Integration with Local Simulations/Games.	89
3.3	Implementation Steps	91
3.4	Summary	92
3.5	Questions for Review	92
	References	93

Part II Object Model Development

4	Introduction to Object Model Development	97
4.1	SimGe Overview	97
4.2	Managing SimGe Projects	98
4.2.1	The Main User Interface	98
4.2.2	Creating a Project.	100
4.2.3	Loading a Project.	100
4.2.4	Saving and Closing a Project	102
4.2.5	Project Start Page.	102
4.2.6	Project Settings	102
4.2.7	Options.	103
4.3	Federation Architecture Modeling.	104
4.3.1	Creating a FAM.	105
4.3.2	Federation Structure	106
4.4	Summary	108
4.5	Questions for Review	108
	References	109
5	Object Model Construction	111
5.1	Overview	111
5.2	Creating a Federation Object Model from Scratch	112
5.3	Loading an Existing SimGe Object Model.	112
5.4	Removing Object Model	113
5.5	Importing a FED/FDD File	113
5.6	Exporting the FED/FDD Files	114
5.7	Object Model Editor.	115
5.7.1	OME Toolbar	115
5.8	Table Editor	116
5.8.1	Object Model Identification	118
5.8.2	Objects	121
5.8.3	Interactions	122
5.8.4	Attributes	123
5.8.5	Parameters.	125
5.8.6	Dimensions	125
5.8.7	Time Representations	128
5.8.8	User-Supplied Tags	128
5.8.9	Synchronization	128
5.8.10	Case Study: Synchronization	129
5.8.11	Transportations.	129
5.8.12	Update Rates	131
5.8.13	Switches	131
5.8.14	Data Types	133
5.8.15	Case Study: Data Types	140
5.8.16	Notes	141

5.8.17	Interface Specification Services	141
5.8.18	OMT 1.3 Support.	142
5.9	Textual View.	143
5.9.1	Textual View for FED and FDD Files.	143
5.10	Validating the FDD File	144
5.11	MOM Integration	145
5.12	OM Explorer	146
5.12.1	Functionality for Traverse	147
5.12.2	Functionality for Modification	149
5.13	Report Generator	149
5.14	Summary	150
5.15	Questions for Review	151
	References	152

Part III Federate Application Development

6	Code Generation	157
6.1	Overview	157
6.2	User Interface	158
6.2.1	Code Explorer	159
6.2.2	Code Viewer	159
6.3	Architectural Style	159
6.3.1	Code Generation for Object Model	162
6.3.2	Code Generation for Federate	163
6.3.3	Code Generation for Data Type	163
6.3.4	Code Generation for MOM	164
6.4	Code Generator Configuration	164
6.4.1	General Settings	165
6.4.2	Callback Settings	165
6.4.3	Runtime Settings	165
6.5	Summary	167
	References	168
7	Federate Application Development Based on Layered Architecture	169
7.1	Federate Application Architecture.	169
7.1.1	Presentation Layer	170
7.1.2	Simulation Layer	170
7.1.3	Communication Layer.	171
7.1.4	Integration of Layers.	171
7.1.5	Encapsulation of Simulation Local Data Structures.	172
7.1.6	The Federation Foundation Library.	174

7.2	Development Environment Configuration	174
7.2.1	Prerequisites	175
7.2.2	Operating Environment Configuration	175
7.2.3	IDE Configuration	177
7.3	Case Study: Running the STMS Federation	179
7.3.1	User Interface	179
7.4	Summary	180
	References	181
8	Federate Implementation: Basics	183
8.1	Case Study: Federate Architecture	183
8.2	The Basics	184
8.2.1	Namespace	184
8.2.2	RACoN Methods	184
8.2.3	Creating the Simulation Manager	185
8.2.4	Creating the Federate Class	186
8.3	Implementing the Simulation Object Model	188
8.3.1	Defining an Object Class and Its Attributes	190
8.3.2	Defining an Interaction Class and Its Parameters	191
8.3.3	Connecting SOM and Federate.	191
8.4	Calls and Callbacks	192
8.4.1	Events.	192
8.4.2	Event Handling	193
8.4.3	Tracing the RTI	193
8.5	Federation Management	196
8.5.1	Federation Execution Creation	196
8.5.2	Joining the Federation Execution	197
8.5.3	High-Level Method for Initialization.	198
8.5.4	Finalization of Federation Execution	198
8.5.5	Connection Lost.	199
8.6	Declaration Management.	200
8.7	Object Management	201
8.7.1	Implementing Simulation Objects	201
8.7.2	Registering Objects.	203
8.7.3	Updating the HLA Object Attributes.	204
8.7.4	Discovering Objects	206
8.7.5	Reflecting the Attribute Values.	207
8.7.6	Request an Update for Attribute Values	208
8.7.7	Deleting and Removing an Object Instance	209
8.7.8	Sending an Interaction	210
8.7.9	Receiving an Interaction	210
8.8	Main Simulation Loop	212
8.8.1	Console Applications	213
8.8.2	Windows Forms Applications	213

8.8.3	Windows Presentation Foundation (WPF)	
	Applications	213
8.9	Parameter and Attribute Marshaling/Unmarshaling	214
8.9.1	Supported Data Types	215
8.10	Summary	215
8.11	Questions for Review	216
	References	217

Part IV Advanced Topics

9	Federate Implementation: Advanced.	221
9.1	Time Management	221
9.1.1	Time-Regulating Federates	221
9.1.2	Time-Constrained Federates	224
9.1.3	Time Advancement.	224
9.1.4	Queries	227
9.1.5	Changing Preferred Order Types	229
9.1.6	Sending and Receiving TSO Messages	229
9.1.7	Message Retraction.	231
9.2	Federation Synchronization	231
9.3	Federation Save-and-Restore	235
9.4	Data Distribution Management.	238
9.4.1	Case Study 1: STMS	238
9.4.2	Creating Regions	241
9.4.3	Subscribing an Object Class with Regions	242
9.4.4	Registering an Object Instance with Regions	243
9.4.5	Associating Regions for Updates	243
9.4.6	Case Study 2: Extended Chat Application	244
9.4.7	Subscribing Interactions with Regions	246
9.4.8	Sending and Receiving Interactions Using Regions	246
9.5	Ownership Management	247
9.5.1	Ownership Management Services	247
9.5.2	Pull Strategy	248
9.5.3	Push Strategy.	249
9.5.4	Case Study: Transferring Tracks Among Traffic Stations.	251
9.5.5	Querying the Ownership	252
9.5.6	Problems in OwM of HLA 1.3.	253
9.6	Handling Multiple Federation Executions	253
9.7	One Federate Application, Multiple Federates	255
9.8	Summary	257
9.9	Questions for Review	258
	References	259

10	Integration of Agents into HLA	261
10.1	Agent-Based Simulation	261
10.1.1	A Cognitive Agent Architecture	262
10.2	Integration of Agents into HLA-Based Simulations.	264
10.2.1	Architectural Approaches.	264
10.2.2	A Concrete Architecture	266
10.2.3	Using HLA as Agent Communication Medium	267
10.3	Summary	270
	References	271
11	A Complete Case Study	273
11.1	Prelude	273
11.1.1	Simulation Environment	274
11.2	Naval Simulation	275
11.2.1	Virtual Environment	275
11.3	Agent-Based Simulation	277
11.3.1	Goal Reasoning	278
11.3.2	Agent Manager	279
11.3.3	USV Agent	280
11.4	Object Model.	282
11.5	Federation Structure	286
11.6	Deployment and Execution	287
11.7	Federate Application with Intensive Graphics.	287
11.8	Main Simulation Loop	290
11.9	Graphics API Integration.	291
11.10	Summary	292
11.11	Questions for Review	292
	References	292
	Appendix A: SimGe Installation and Remarks	295
	References	301
	Index	303

Abbreviations

ACL	Agent Communication Language
AgentFdApp	Agent Manager Federate Application
ALSP	Aggregate Level Simulation Protocol
AMG	Architecture Management Group
AOR	Area of Responsibility
API	Application Programming Interface
ARPA	Advanced Research Projects Agency
C2	Command and Control
CGB	Coast Guard Boat
CGF	Computer Generated Force
CodeGen	Code Generator (of SimGe)
CogAgentLib	Coherence-based Agent Framework
COTS	Commercial off-the-shelf
CRUD	Create, read, update, and delete operations
D/A	Divest/Acquire
DDM	Data Distribution Management, Detailed Design Model
DeCoAgent	Deliberative Coherence Driven Agent
DIF	Data Interchange Format
DIS	Distributed Interactive Simulation
DLC	Dynamic Link Compatibility
DLL	Dynamic Link Library
DM	Declaration Management
DMAO	Distributed Simulation Engineering and Execution Process Multi-Architecture Overlay
DMSO	U.S. Defense Modeling and Simulation Office
DoD	U.S. Department of Defense
DOI	Digital Object Identifier
DSEEP	Distributed Simulation Engineering and Execution Process
DUB	Dimension Upper Bound
EnvFd	Environment Federate
EnvFdApp	Environment Controller Federate Application
FAM	Federation Architecture Model
FAME	Federation Architecture Modeling Environment

FAMM	Federation Architecture Metamodel
FAP	Federation Architecture Project
FDD	FOM Document Data
FED	Federation Execution Details
FEDEP	Federation Development and Execution Process
FedMonFd	Federation Monitor Federate
FedMonFdApp	Federation Monitor Federate Application
FFL	Federation Foundation Library
FIPA	Foundation for Intelligent Physical Agents
FM	Federation Management
FOM	Federation Object Model
fps	Frame per Second
GALT	Greatest Available Logical Time
GPS	Global Positioning System
GUI	Graphical User Interface
HLA	High Level Architecture
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
IF	Interface Specification
IP	Internet Protocol
ISBN	International Standard Book Number
KQML	Knowledge Query and Manipulation Language
LITS	Least Incoming Timestamp
M&S	Modeling and Simulation
MariSim	Maritime Simulation
MCV	Model-View-Controller
MDE	Model-Driven Engineering
MIM	MOM and Initialization Module
MOM	Management Object Model
MRU	Most Recently Used
MS	Microsoft
MSF	Maritime Surveillance Federation
NavySim	Naval Simulation
NSTMSS	Naval Surface Tactical Maneuvering Simulation System (pronounced “Nistmiss”)
OM	Object Model, Object Management
OME	Object Model Editor (of SimGe)
OMG	Object Modeling Group
OMT	Object Model Template
OOP	Object Oriented Programming
OOW	Officer of the Watch
OwM	Ownership Management
P/S	Publish/Subscribe
PDU	Protocol Data Unit
POC	Point of Contact

RACoN	RTI Abstraction Component for .Net
RB	Refugee Boat
RID	RTI Initialization Data
RO	Receive Order
RPR-FOM	Real-time Platform Reference FOM
RTI	Runtime Infrastructure
ShipFd	Ship Federate
ShipFdApp	Ship Federate Application
SimGe	SIMulation Generator
SISO	Simulation Interoperability Standards Organization
SL	Semantics Language
SOM	Simulation Object Model
SS	Support Services
StationFd	Station Federate
StationFdApp	Station Federate Application
STMS	Strait Traffic Monitoring Simulation
T/A	Transferable/Acceptable
TC	Time-Constrained
TC&TR	Both TC and TR
TM	Time Management
TMS	Traffic Monitoring Station
TR	Time-Regulating
TSO	Time Stamp Order
TSS	Tracking Sub-System
U/R	Updateable/Reflectable
UI	User Interface
UML	Unified Modeling Language
URL	Uniform Resource Locator
US DoD	United States Department of Defense
USV	Unmanned Surface Vehicle
VS	Visual Studio
WiX	Windows Installer XML
WPF	Windows Presentation Foundation
XML	Extensible Markup Language