CHAPTER 1

Introduction

1.1 WHY A HIGH LEVEL ARCHITECTURE FOR MODELING AND SIMULATION

The HLA is the glue that allows you to combine computer simulations into a larger simulation. For instance, you might have simulations of several different manufacturing machines and material-handling machines. The HLA helps you create a factory floor simulation from the pieces. You might want to combine simulations of air traffic control in several different regions of the country with simulators of individual aircraft. The HLA helps you combine these simulations into a single, comprehensive simulation. The HLA also helps you extend your simulation later, by adding, for example, another manufacturing machine or an additional airport.

The HLA defines some terms that will be used throughout this book:

- The combined simulation system created from the constituent simulations is a federation.
- Each simulation that is combined to form a federation is a federate.
- A federation execution is a session of a federation executing together.

A federation contains the following elements:

- Supporting software called the Runtime Infrastructure (RTI)
- A common object model for the data exchanged between federates in a federation, called the Federation Object Model (FOM)
- Some number of federates

A federate is a member of a federation, one point of attachment to the RTI. A federate could represent one platform, such as a cockpit simulator. Or a federate could represent an aggregate simulation, such as an entire national simulation of air traffic flow.
1.2 An HLA Federation Has Software and Data Components

Federates and the RTI are software. Typically the federation developer furnishes the federates and gets an RTI implementation from some other source. (An RTI implementation is furnished on the CD-ROM in this book.) The FOM is data created by the federation developer.

The FOM is a description of the kinds of and relationships among data that the federates will exchange in a federation execution. It expresses agreement about data between the federates. The FOM is a significant tool for communicating design decisions between federate and federation developers. Part of it is furnished to the RTI during federation execution, effectively parameterizing the RTI for the federation.

The relationship of the software components is depicted in Figure 1–1. Federates are shown in the figure as either simulations, surrogates for live players, or tools for distributed simulation. A federate is defined as having a single point of attachment to the RTI. A federate might consist of several processes, perhaps running on several computers. A federate might model a single entity, like a vehicle, or a federate might model many entities, like all the vehicles in a city. From the perspective of the HLA, a federate is defined by its single point of attachment to the RTI.

As shown in Figure 1–1, a federate might model some number of entities, or it might have a different purpose. It might be a collector and/or viewer of data, passively receiving data and generating none. A federate might act as a surrogate for human participants in a simulation. In this role, the federate might reflect the state of the larger simulation to the participant via some user interface, and might convey control inputs or decisions from the participant to the rest of the federation.

The RTI used by a federation may be implemented as many processes or as one. It may require many computers to execute, but conceptually it is one RTI. An RTI may support several federations executions at once.
THE HLA AS A STANDARD

1.3 THE HLA AS A STANDARD

The HLA is foremost a software architecture, rather than a particular implementation of its infrastructure or tools designed to work with it. The HLA embraces a variety of implementations. Consequently, it is defined not by software but by a set of documents. This book is written to reflect version 1.3 of the HLA specification as adopted by the U.S. Department of Defense (DoD). The full text of the specification is included on the CD-ROM. The HLA standard has three parts:

- HLA Rules
- Object Model Template (OMT)
- Interface Specification

1.3.1 HLA RULES

The HLA Rules are principles and conventions that must be followed to achieve proper interaction of federates during a federation execution. They are design principles for the Interface Specification and OMT. They also describe the responsibilities of federates and federation designers.

1.3.2 OBJECT MODEL TEMPLATE (OMT)

Every federation has an FOM. The OMT prescribes the allowed structure of every FOM. The OMT is the meta-model for all FOMs.

1.3.3 INTERFACE SPECIFICATION

The interface specification is the specification of the interface between federates and the RTI. The RTI is software that allows a federation to execute together. The interface between the RTI and federates is standardized. Implementation of the RTI could take a variety of forms.

1.3.4 HOW THE SPECIFICATION IS EVOLVING

At the time of writing, there are two parallel efforts under way to pursue the adoption of the HLA by standards bodies. One effort is through the Object Management Group (OMG), a consortium of software vendors and users pursuing standards for distributed object computing. Version 1.3 of the interface specification has been adopted by the OMG as a standard called the “Facility for Distributed Simulation Systems” [OMG 1998]. The other standards adoption effort is through the Institute of Electrical and Electronics Engineers (IEEE). The draft IEEE standards are P1516 (HLA Rules), P1516.1 (Interface Specification), and P1516.2 (OMT) [IEEE 1999].

1.4 YOU CAN RUN YOUR OWN FEDERATION

The accompanying CD-ROM contains all the software you need to run your own HLA federation:

- A version of the RTI suitable for the samples in the book and for limited experiments with federations of your own design
• A FOM for the restaurant federation to be described later
• Three kinds of federates

The three kinds of federates on the CD-ROM are:
• The basic federates developed in “The Sushi Restaurant Federation” and “Synchronizing the Federation.” We suggest you read your way through that part of the book before running them.
• The manager federate described in “Synchronizing the Federation.” This federate would be useful for your own projects as well as our sample implementation.
• A test federate, which allows you to invoke manually any HLA service. You can start any number of test federates at once to create a federation without any programming.

You can also create your own federates by modifying the sample implementations. You can also experiment by mixing your federates with instances of the test federate.