An Introduction to the HLA
Part 1

Roger McFarlane
School of Computer Science
McGill University
Montreal, CANADA
Overview

- Introduction
  - What is the HLA?
  - Motivation
  - Goals
  - History

- HLA Components
  - The RTI
  - HLA Rules
  - Object Model Templates

- For Next Time …
Introduction

- What is the HLA?
- Motivation
- Goals
- History
What is the HLA?

- A general framework facilitating interoperability and reusability of distributed simulation components
- Developed by the Defense Modeling and Simulation Office (DSMO)
- Developed for the United States Department of Defence (DoD)
- IEEE Standard 1516-2000
Motivation

- Many large/complex simulations involve individual “sub-simulations” of components
- “Sub-simulations” are often heterogeneous (in the type of simulation and type of component)
- Simulators for the components may already exist
- Re-implementing or retrofitting a simulation system is risky and expensive
Goals

- **Reusability**
  - A component simulation may be used in different scenarios and applications over its lifetime

- **Interoperability**
  - Aggregate simulations composed of multiple component simulations
  - Aggregate simulations distributed across heterogeneous hardware and software platforms
  - Reuse without significant code change or development cost
  - Combine component simulations with diverse models of computation and representation
History

- DSB: Computer Applications to Training and Wargaming Study
- DoD Simulation Policy Study
- ALSP: linking of Service Wargames
- EXCIMS Functional Area Councils
- SBA Task Force
- Adopted by OMG

- New programs (JSIMS, JWARS)
- DoD VV&A Instruction
- IEEE 1516

- 1987
  - SIMNET
  - DARPA-SACEUR Distributed Wargaming System ACE-89

- 1988
  - limited scope simulation, little interoperability prior to 1988
  - DoD Dir 5000.59 M&S Management
  - DEPSECDEF Memo: EXCIMS formed & DMSO established

- 1990
  - DIS Standards Begun

- 1991
  - 1992
  - 1993
  - 1994
  - 1995
  - 1996
  - 1997
  - 1998
  - 1999

- 2000
  - 2001

- HLA 1.0 Released
- DoD M&S Executive Agents
- Architecture Management Group
HLA Components

- Definitions & Terms
- Technical Architecture
- HLA Rules
- Object Model Templates
- Run-Time Interface Specification
Definitions & Terms (1)

- **Federate**
  - An application which supports the HLA and is capable of participating in a simulation.

- **Federation**
  - A declaration between federates describing how and what will be simulated.

- **Federation Execution**
  - A run-time instantiation of a Federation; that is, an actual simulation execution.
Definitions & Terms (2)

- The HLA provides the **Federation** formalism by which **Federates** can be modeled such that the framework can support **Federation Execution**

- This is really no different from any other type of modelling and simulation application!
Technical Architecture

Support Utility
Live Player
Simulation

Run-time Infrastructure
Federation Management
Object Management
Time Management
Declaration Management
Ownership Management
Data Distribution Management
Software layer providing common services to federates

RTI Specification defines the interfaces federates must use to obtain services and interact with other federates

RTI Specification defines interfaces to be exposed by federates in order to be recognizable by the services and by other federates
Run-Time Infrastructure (2)

- Improvements on older standards
  - DIS
  - ALSP
- Provides efficient inter-federate communications
- Separate simulation concerns from communication concerns
- Language and platform independent
Service Groups

- Federation management
- Declaration management
- Object management
- Ownership management
- Time management
- Data Distribution management
- Support services
Federation Management

- Controls federation-wide activities during a federation execution

- Services offered:
  - Creation and destruction of federation executions
  - Joining and resigning of federates
  - Pause/Resume federation execution
  - Save/Restore federation execution
Declaration Management

- Manages the publisher/subscriber model for information exchange

**Services Offered:**
- Publish Object/Interaction class
- Subscribe to Object Class Attribute
- Subscribe to Interaction Class
- Control Updates
- Control Interactions
Object Management

- Manages the lifecycle and message passing for object instances

Services Offered:
- Register/Discover Object
- Update/Reflect Attribute Values
- Send/Receive Interaction
- Remove Object
- Manage Transport/Ordering
Ownership Management

- Supports cooperative modelling by allowing attribute ownership to be transferred across instances

- Services Offered:
  - Assume/Divest Attribute Ownership
  - Acquire/Release Attribute Ownership
  - Notification of ownership changes
Time Management (1)

- Coordinates federate time advancement along the federation time axis
- Attempts to preserve causality and ordering
- Mechanisms supported:
  - Conservative synchronization (with look ahead)
  - Optimistic synchronization (e.g., time warp)
  - Hybrid methods
  - Time-stepped
  - Real-time driven
Time Management (2)

- Federates request permission to advance their local time
- Services offered
  - Request Time Advance
  - Notification of Granting of Time Advance
  - Request Next Event
  - Notification of Granting of Next Event
  - Queue Management
Data Distribution Management

- Efficient data transmission between federates
- Uses routing spaces to direct data only to the interested parties
  - Publisher specifies the update region
  - Subscribes specify their interest region
  - Intersection define routing space
Support Services

- Miscellaneous functionality useful to joined federates
  - Name-to-handle transformation
  - Handle-to-name transformation
  - Setting advisory switches
  - Manipulating regions
  - RTI start-up and shutdown
HLA Rules

- Define the behaviour and capabilities of federates and federations
- Five rules for Federates
- Five rules for Federations
Federation Rules

- Must have an Federation Object Model (FOM) documented using the OMT
- All object representation occur in the Federates, not in the RTI
- Data exchange between instances of objects in different Federates occurs via the RTI
- Federates must interact with the RTI in accordance with the HLA Interface Specification
- During Federation Execution, an instance attribute may be owned by at most one federate at any given time
Federate Rules

- Must have a Simulation Object Model (SOM) documented using the OMT
- Must be able to update/reflect instance attributes and send/receive interactions as specified in their SOM
- Must be able to dynamically transfer/accept ownership of attributes during federation execution as specified in their SOM
- Must be able to vary the conditions under which they provide attribute updates as specified in their SOM
- Must manage their local time in a manner which allows them to coordinate data exchange with other federates
Object Model Templates

- Provide a mechanism for specifying data exchange and coordination within a federation
- Provide a mechanism for describing the capabilities of federate
- Facilitates design and implementation of common tools for building HLA compliant objects
Types of Object Models

- Simulation Object Model (SOM)
- Federation Object Model (FOM)
- Management Object Model (MOM)
SOM – Simulation Object Model

- Information exposed/consumed by a federate
  - Objects
  - Interactions
  - Attributes (of Objects and Interactions)
  - Parameters (of Objects and Interactions)
FOM – Federation Object Model

- Specifies data exchange between federates
  - Objects
  - Interactions
  - Attributes (of Objects)
  - Parameters (of Interactions)
- Provides the “information model contract” which governs the simulation
- Provides the foundation for interoperability
MOM – Management Object Model

- A predefined set of information elements to be included in the FOM
- Contains data relevant to Federation Execution
- Federates may also include referenced to the MOM if they may influence Federation execution.
OMT Components (1)

- Object model identification table
- Object class structure table
- Interaction class structure table
- Attribute table
- Parameter table
- Dimension table
- Time representation table
OMT Components (2)

- User-supplied tag table
- Synchronization table
- Transportation type table
- Switches table
- Datatype tables
- Notes table
- FOM/SOM lexicon
### Object Model Identification Table

- Describes object model’s identity
- Useful for developers seeking reusable object models
- Why the object model was constructed
- How the object model was constructed
- Who knows about the object model
- Where to look for more information
### Example – Object Model Identification Table

<table>
<thead>
<tr>
<th>Category</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Object Model Name</td>
</tr>
<tr>
<td>Type</td>
<td>“SOM” or “FOM”</td>
</tr>
<tr>
<td>Version</td>
<td>Version Identifier</td>
</tr>
<tr>
<td>Modification Date</td>
<td>Last Modified Date (YYYY-MM-DD)</td>
</tr>
<tr>
<td>Purpose</td>
<td>Why was this object model developed</td>
</tr>
<tr>
<td>Application Domain</td>
<td>Type of Application</td>
</tr>
<tr>
<td>Sponsor</td>
<td>Name of Sponsoring Organization</td>
</tr>
<tr>
<td>POC</td>
<td>Point of Contact’s Name</td>
</tr>
<tr>
<td>POC Organization</td>
<td>Point of Contact’s Organization</td>
</tr>
<tr>
<td>POC Telephone</td>
<td>Point of Contact’s Telephone Number</td>
</tr>
<tr>
<td>POC Email</td>
<td>Point of Contact’s Email Address</td>
</tr>
<tr>
<td>References</td>
<td>Where to look for further information</td>
</tr>
<tr>
<td>Other</td>
<td>Any other relevant data</td>
</tr>
</tbody>
</table>
## Object Class Structure Table

- Defines super/sub-class relationships
- For a SOM, classes may be tagged …
  - P: The federate is capable of publishing at least one attribute of the object class.
  - S: The federate is capable of subscribing to at least one attribute of the object class.
  - PS: Both publish and subscribe
  - N: The federate is neither capable of publishing nor subscribing to any attributes of the object class.
- For a FOM, the same tags indicate if least one federate is capable of publishing or subscribing to any attribute of the object class
## Example – Object Class Structure Table

| HLA Object Root (N) | Customer (PS) | | | | |
|---------------------|---------------|---------------|---------------|---------------|
| Bill (PS)           |               |               |               |               |
| Order (PS)          |               |               |               |               |
| Employee (N)        | Greeter (PS)  | Waiter (PS)   | Cashier (PS)  |               |
| Food (S)            | Appetizer (S) | Soup (S)      | Clam Chowder (PS) | Manhattan (P) |
|                     |               |               |               | New England (P) |
|                     |               |               | Beef Barley (PS) |               |
|                     |               |               | Salad (S)      |               |
|                     |               |               | Seafood (S)    | Shrimp (PS)    |
|                     |               |               |               | Salmon (PS)    |
|                     |               |               |               | Pasta (PS)     |
Interaction Class Structure Table

- Specific actions which a federate may perform
- Hierarchy similar to Object Class Structure Table
- SOM Interactions may be tagged
  - P: The federate is capable of publishing the interaction class
  - S: The federate is capable of subscribing to the interaction class
  - PS: Both publish and subscribe
  - N: The federate is neither capable of publishing nor subscribing to the interaction class
- Same tags used for a FOM meaning there does (not) exist a federate capable of publishing/subscribing to the interaction class.
## Example – Interaction Class Structure Table

<table>
<thead>
<tr>
<th>HLA Object Root (N)</th>
<th>Customer Transaction (P)</th>
<th>Customer Seated (PS)</th>
<th>Order Taken (P)</th>
<th>From Kids Menu (P)</th>
<th>From Adult Menu (P)</th>
<th>Food Served (P)</th>
<th>Drink Served (P)</th>
<th>Appetizer Served (P)</th>
<th>Main Course Served (P)</th>
<th>Dessert Served (P)</th>
<th>Customer Pays (P)</th>
<th>By Credit Card (P)</th>
<th>By Cash (P)</th>
<th>Customer Leaves (P)</th>
</tr>
</thead>
</table>
Attribute Table

- Properties of an object
- May be published by the object
- Other objects may subscribe to an attribute
- Declare how/when an attribute value changes
- Declares if attribute ownership may be transferred between objects
  - DA = Divest & Acquire
  - N = Neither
- The transport used to communicate the attribute
## Example – Attribute Table

<table>
<thead>
<tr>
<th>Object</th>
<th>Attribute</th>
<th>Data Type</th>
<th>Update Type</th>
<th>Update Condition</th>
<th>D/A</th>
<th>P/S</th>
<th>Available Dimensions</th>
<th>Transportation</th>
<th>Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root</td>
<td>PTDO</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>N</td>
<td>N</td>
<td>NA</td>
<td>HLAReliable</td>
<td>Timestamp</td>
</tr>
<tr>
<td>Employee</td>
<td>PayRate</td>
<td>Dollars</td>
<td>Cond.</td>
<td>Merit</td>
<td>DA</td>
<td>PS</td>
<td>NA</td>
<td>HLAReliable</td>
<td>Timestamp</td>
</tr>
<tr>
<td></td>
<td>Seniority</td>
<td>Years</td>
<td>Periodic</td>
<td>+1/year</td>
<td>DA</td>
<td>PS</td>
<td>NA</td>
<td>HLAReliable</td>
<td>Timestamp</td>
</tr>
<tr>
<td></td>
<td>Phone</td>
<td>Text</td>
<td>Cond.</td>
<td>Empl. Req.</td>
<td>DA</td>
<td>PS</td>
<td>NA</td>
<td>HLAReliable</td>
<td>Timestamp</td>
</tr>
<tr>
<td></td>
<td>Address</td>
<td>Text</td>
<td>Cond.</td>
<td>Empl. Req.</td>
<td>DA</td>
<td>PS</td>
<td>NA</td>
<td>HLAReliable</td>
<td>Timestamp</td>
</tr>
<tr>
<td></td>
<td>Manner</td>
<td>WaiterValue</td>
<td>Cond.</td>
<td>Perf. Rev.</td>
<td>DA</td>
<td>PS</td>
<td>NA</td>
<td>HLAReliable</td>
<td>Timestamp</td>
</tr>
<tr>
<td></td>
<td>State</td>
<td>WaiterTask</td>
<td>Cond.</td>
<td>Work Flow</td>
<td>DA</td>
<td>PS</td>
<td>NA</td>
<td>HLAReliable</td>
<td>Timestamp</td>
</tr>
</tbody>
</table>
Parameter Table

- Additional information to characterize an interaction
- Identify the transport used to deliver the parameter
- Identify the ordering constraints for the parameter
  - Timestamp
  - Receive (indeterminate order)
### Example – Parameter Table

<table>
<thead>
<tr>
<th>Interaction</th>
<th>Parameter</th>
<th>Datatype</th>
<th>Available Dimensions</th>
<th>Transportation</th>
<th>Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer Seated</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>HLAReliable</td>
<td>Timestamp</td>
</tr>
<tr>
<td>FoodServed. MainCourse Served.</td>
<td>TemperatureOK</td>
<td>ServiceStat</td>
<td>WaiterID</td>
<td>HLAReliable</td>
<td>Timestamp</td>
</tr>
<tr>
<td></td>
<td>AccuracyOK</td>
<td>ServiceStat</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TimelinessOK</td>
<td>HLABoolean</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Dimension Table

- Maps domain specific data values onto integer values ranging from zero to some upper bound
- Specifies the legal values which may be transmitted across the RTI
- Enables Data Distribution Management (DDM) and Declaration Management (DM)
- Used to specify update and subscribe regions to the RTI
### Example – Dimension Table

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Upper Bound</th>
<th>Normalization</th>
<th>Value If Not Specified</th>
</tr>
</thead>
<tbody>
<tr>
<td>SodaFlavour</td>
<td>flavourType</td>
<td>3</td>
<td>LinearEnumerated(Flavour, {Cola, Orange, Grape})</td>
<td>[0..3)</td>
</tr>
<tr>
<td>BarQuantity</td>
<td>DrinkCount</td>
<td>25</td>
<td>Linear(NumberCups, 0, 25)</td>
<td>[0..1)</td>
</tr>
<tr>
<td>WaiterId</td>
<td>EmpId</td>
<td>20</td>
<td>Linear(WaiterId, 0, 20)</td>
<td>Excluded</td>
</tr>
</tbody>
</table>
Declares the format used to represent time
  - For a federate
    - Across a federation

Declares the semantics of time
  - For a federate
    - Across a federation

Used by the RTI to coordinate federates during federation execution
### Example – Time Representation Table

<table>
<thead>
<tr>
<th>Category</th>
<th>Datatype</th>
<th>Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timestamp</td>
<td>TimeType</td>
<td>Floating point value expressed in minutes</td>
</tr>
<tr>
<td>LookAhead</td>
<td>LAType</td>
<td>Non-negative floating point value expressed in minutes</td>
</tr>
</tbody>
</table>
User-Supplied Tag Table

- Extensible mechanism for specifying auxiliary data
- Provides additional control and coordination of services provided by the HLA
### Example – User-Supplied Tag Table

<table>
<thead>
<tr>
<th>Category</th>
<th>Datatype</th>
<th>Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Update/Reflect</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Send/Receive</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Delete/Remove</td>
<td>HLAascii</td>
<td>Reason for deletion</td>
</tr>
<tr>
<td>Divestiture Request</td>
<td>PriorityLevel</td>
<td>High value for immediate transfer</td>
</tr>
<tr>
<td>Divestiture Completion</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Acquisition Request</td>
<td>PriorityLevel</td>
<td>High value for immediate transfer</td>
</tr>
<tr>
<td>Request Update</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>
Synchronization Table

- Provides a federate synchronization mechanism
- Federates declare the synchronization points they support
- Federations describe the synchronization points to be used
### Example – Synchronization Table

<table>
<thead>
<tr>
<th>Label</th>
<th>Tag Datatype</th>
<th>Capability</th>
<th>Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td>InitialPublish</td>
<td>NA</td>
<td>Achieve</td>
<td>Achieved when all classes are published and subscribed, and all initially present objects are registered</td>
</tr>
<tr>
<td>InitialUpdate</td>
<td>NA</td>
<td>Achieve</td>
<td>Achieved when instance attribute values for all initially present objects are updated</td>
</tr>
<tr>
<td>BeginTimeAdvance</td>
<td>NA</td>
<td>Achieve</td>
<td>Achieved when time management services are invoked</td>
</tr>
<tr>
<td>PauseExecution</td>
<td>TimeType</td>
<td>Register</td>
<td>Achieved when the time advance after the time in the user-supplied tag is attained; time advance requests should then cease</td>
</tr>
</tbody>
</table>
Transportation Type Table

- The RTI provides different mechanisms for transport of interactions and attributes between federates
- Allows a federate designer to describe the transports supported by the federate
- Allows federation designers to describe the transportation contracts between federates
## Example – Transportation Type Table

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HLAreliable</td>
<td>Provide reliable delivery of data in the sense that TCP/IP delivers its data reliably</td>
</tr>
<tr>
<td>HLAbestEffort</td>
<td>Make an effort to deliver data in the sense that UDP provides best-effort delivery</td>
</tr>
<tr>
<td>LowLatency</td>
<td>Choose the delivery mechanism that results in the lowest latency from service initiation to callback invocation at the receiving federate</td>
</tr>
</tbody>
</table>
Switches Table

- Configuration of RTI activities performed on behalf of a federate
- A few services are configured globally for the federation
  - Auto Provide, Convey Region Designator Sets
- Most services are configured per federate
  - Attribute Scope Advisory, Attribute Relevance Advisory, Object Class Relevance Advisory, Service Reporting
- Services may be either enabled or disabled
Switch Definitions (1)

- **Auto Provide**
  - (Global) Should the RTI automatically solicit updates from instance attribute owners when an object is discovered.

- **Convey Region Designator Sets**
  - (Global) Should the RTI provide the optional Sent Region Set argument with invocations of Reflect Attribute Values and Receive Interaction.

- **Attribute Scope Advisory**
  - Should the RTI advise federates when attributes of an object instance come into or go out of scope.
Switch Definitions (2)

- **Attribute Relevance Advisory**
  - Should the RTI advise federates about whether they should provide attribute value updates for the value of an attribute of an object instance.

- **Object Class Relevance Advisory**
  - Should the RTI advise federates about whether they should register instances of an object class.

- **Interaction Relevance Advisory**
  - Should the RTI advise federates about whether they should send interactions of an interaction class.

- **Service Reporting**
  - Should the RTI report service invocations using MOM.
### Example – Switches Table

<table>
<thead>
<tr>
<th>Switch</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto provide</td>
<td>Disabled</td>
</tr>
<tr>
<td>Convey region designator sets</td>
<td>Disabled</td>
</tr>
<tr>
<td>Attribute scope advisory</td>
<td>Enabled</td>
</tr>
<tr>
<td>Attribute relevance advisory</td>
<td>Enabled</td>
</tr>
<tr>
<td>Object class relevance advisory</td>
<td>Enabled</td>
</tr>
<tr>
<td>Interaction relevance advisory</td>
<td>Enabled</td>
</tr>
<tr>
<td>Service reporting</td>
<td>Disabled</td>
</tr>
</tbody>
</table>
Data Type Tables (1)

- Globally define data types referenced in other tables
- Basic Data Table
  - Name, Size in Bits, Interpretation, Endian, Encoding
- Simple (Scalar) Data Table
  - Name, Representation, Units, Resolution, Accuracy, Semantics
- Enumerated Data Table
  - Name, Representation, Enumerator, Values, Semantics
Data Type Tables (2)

- **Array Data Table**
  - Name, Element Type, Cardinality, Encoding, Semantics

- **Fixed Record Data Table**
  - Record Name, Field-{Name, Type, Semantics}*, Encoding, Semantics

- **Variant Record Data Table**
  - Record Name, Encoding, Semantics, Discriminant-{Name, Type, Semantics}*, Alternative--{Name, Type, Semantics}*

Notes Table

- Named annotations may be attached to any OMT entry
- A set of name/value pairs
- Value is free form explanatory text
- Name uniquely identifies the corresponding explanatory text
- Notes may be referenced multiple times
FOM/SOM Lexicon

- Name/Value pairs
- Dictionary tables associating every class, attribute, interaction, parameter, etc (by name) with a free form text description (value)
For Next Time …

- A deeper look at the RTI
References (1)

References (2)

- Roy Crosbie and John Zenor, “High Level Architecture, Module 1 – Basic Concepts, Parts 1-6.” California State University, Chico.  
  [http://www.ecst.csuchico.edu/~hla](http://www.ecst.csuchico.edu/~hla)

- <Steffen Strassburger's text>