High Level Architecture
Module 1
Basic Concepts

McLeod Institute of Simulation Sciences
California State University, Chico

The Society for Computer Simulation

Roy Crosbie
John Zenor

California State University, Chico
High Level Architecture
Module 1
Basic Concepts

Lesson 4
Basic Concepts of HLA Simulations
## Continuous vs. Discrete Simulations

<table>
<thead>
<tr>
<th>Continuous</th>
<th>Discrete</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Continuously advances time and system state.</td>
<td>• System state changes only when events occur.</td>
</tr>
<tr>
<td>• Time advances in increments small enough to ensure accuracy.</td>
<td>• Time advances from event to event.</td>
</tr>
<tr>
<td>• State variables updated at each time step.</td>
<td>• State variables updated as each event occurs.</td>
</tr>
</tbody>
</table>
Simulation Time Steps

Continuous

\[ \Delta t \quad \Delta t \quad \Delta t \quad \Delta t \quad \Delta t \]

time

Discrete

Event1 \quad Event2 \quad Event3

time
Simulation Objects

Object (Aircraft)

Attribute 1
(speed)

Attribute 2
(position)

Attribute 3
(altitude)
Simulation Structure

Federation

Simulation Programs (Federates)

Sim. for Objects of Class 1

Sim. for Objects of Class 2

Simulation Programs (Federates)

Sim. for Objects of Class 1

Sim. for Objects of Class 3
HLA Messages -- Updates

Updates

• Send/Receive new values of attributes at end of each time-step.
  – Send: Update Attribute Values
  – Recv: Reflect Attribute Values†

• Controlling unnecessary message traffic:
  – Update:
    • Enable/Disable Attribute Relevance Advisory Switch
    • Turn Updates On/Off for Object Instance†
  – Reflect:
    • Enable/Disable Attribute Scope Advisory Switch
    • Attribute In/Out of Scope†
Interactions

• Send/Receive Parameters describing the event when an event occurs.
  – Send: Send Interaction
  – Recv: Receive Interaction†

• Controlling unnecessary message traffic:
  – Enable / Disable Interaction Relevance Advisory Switch
  – Turn Interactions On/Off†
# Object and Interaction Registration

## Objects
- Publish/Subscribe
  - Publish Object Class
  - Subscribe Object Class
- Object Registration
  - Register Object Instance
  - Discover Object Instance
- Controlling Instance Registration:
  - Enable/Disable Class Relevance
  - Start/Stop Registration for Object Class

## Interactions
- Publish/Subscribe
  - Publish Interaction Class
  - Subscribe Interaction Class
Message Order

Two types of Message Ordering

- **TSO** (Time Stamped Order)
  - Messages delivered to federate in order of time stamp
  - RTI guarantees that no messages will be received from past

- **RO** (Receive Order)
  - Messages delivered to federate in order received
Regulating and Constrained Federates (1)

- To receive TSO messages in Time Stamped Order, Federate must declare itself **Time Constrained**.
- To send TSO messages, Federate must declare itself to be **Time Regulating**.
- By default, Federates are **neither** time constrained nor time regulating.
- To become time constrained, use RTI service *Enable Time Constrained*.
- To become time regulating, use RTI service *Enable Time Regulation*.
Time Constrained Federates subscribe to time stamped (TSO) data, with messages delivered in order of time-stamps.

Federate’s current logical time.

LBTS - Other federates will not send messages before this time.

Federate may only advance time within this interval
Time Regulating Federates

Time Regulating Federates publish time stamped (TSO) data, with messages delivered in order of time-stamps.
Requesting Time Advancement

- Continuous Simulations
  - To request an advance in time, use the RTI service: *Request Time Advance*.
  - RTI will notify when it's ok to advance time by calling: *Time Advance Grant* †

- Discrete event simulations
  - The RTI service: *Next Event Request (t1)*, requests time advancement to time of next event, or to *t1*, whichever occurs first.
  - RTI will notify when to advance time by calling: *Time Advance Grant* †, and will specify the amount of the granted time advance.
Federates That Are Both Time Regulating and Time Constrained

Federate’s current logical time.

Federate’s effective logical time.

Constrained:
Federate may only advance time within this interval

Regulating:
Lookahead
Federate will not publish messages within this interval (regulates other federates)

LBTS - Other federates will not send messages before this time.

t=0
Time Regulating and Time Constrained

- If a Federate sends and receives TSO data, in TSO order, it must be both time regulating and time constrained.
  - *Time constrained*: RTI prevents this federate from advancing time until it has received all messages that may be sent by other federates up to the requested time.
  - *Time Regulating*: RTI prohibits other federates from advancing time until this federate has sent all the data that it is going to send before the requested time.
When Are Messages Received?

Messages are only received when in a time-advancing state.

• A Federate is put into a time-advancing state by:
  – Time Advance Request OR
  – Next Event Request

• To enable receipt of RO messages at other times:
  – Enable Asynchronous Delivery (Prevents excessive delay for urgent events)