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 2
Basic Structure
of HLA Simulations
Logical View of RTI Components

RTI Components At-a-Glance

RtiExec
FedExec
Federate(s)
Federate(s)

Inter-Process Communications

RTI Provided
Federate Provided
FedExec - The Federation Executive

- One running process per executing federation
  - Created by first federate to successfully join federation
- Manages multiple federates joining and leaving the federation execution
  - Assigns unique handles to each federate
- Facilitates data exchange between federates
- Console interface for manual operations
rtiExec - The RTI Executive

- Manages the creation and destruction of multiple federation executions (with different names)
  - Ensures that each FedExec has a unique name
- Global process executes on one platform
- Listens to a well known port
- Console interface for manual operations
libRTI - The RTI Library

- Makes HLA service methods available to federates
  - Methods communicate with rtiExec, FedExec, and other federates through them
- Written in C++ with interfaces in C++, Java, CORBA IDL, Ada
What’s in a Federate?

The Federate’s Code provides internal functionality.

The Federate’s code must define the abstract RTI::FederateAmbassador class.

White Federate

Federate Code

Federate Ambassador

LRC

RTI Ambassador

The Local RTI Components (LRC) provide external functionality as specified by the IFSpec.

The LRC include the methods for the RTI::RTIAmbassador class.
RTI and Federate Ambassadors

RTI and Federate "Ambassadors"

libRTI

Federate Ambassador

"Various RTI Objects"

Federate Code

"Ambassador Implementation"

"Various Federate Objects"
The Big Picture
When a federation is run, the RTIExec is started first.

Then a federate, acting as a manager, creates a federation execution by invoking the RTI method “createFederationExecution” on its RTI Ambassador. The RTIAmbassador then reserves a name with RTIExec, and spawns a FedExec process, and that FedExec registers its communication address with RTIExec. The federation execution is underway.

Once a federation execution exists, other federates can join it. That RTIAmbassador consults RTIExec to get the address of FedExec, and invokes joinFederationExecution on FedExec. Additional federates can join via the same process.
Content of an IF Specification

- Interface Name and Brief Description of Service
- Supplied Arguments
- Returned Arguments
- Pre Conditions
- Post Conditions
- Exceptions
- Related Services
Sample RTI Service Request

```cpp
try {
    rtiAmb.timeAdvanceRequest(requestTime);
} catch (RTI::Exception& e) {
    cerr << "FED_HW: ERROR:" << &e << endl;
}
```
Possible Exceptions Thrown

• The federation time is invalid.
• Federation time already passed.
• The TimeAdvanceRequest is already pending.
• The federate is not a federation execution member.
• Save in progress.
• Restore in progress.
• RTI internal error.
Using “Tick”

- Tick used To Wait for Service Completion
  - Allows RTI a chance to execute and respond to request
  - Waits for not more than 1.0 sec’s, nor less than .01 sec’s

- `timeAdvGrant` is a global variable, initialized to false, set `true` in callback routine `TimeAdvanceGrant`

```cpp
    timeAdvGrant = RTI::RTI_FALSE;

    while (!timeAdvGrant)
    {
        rtiAmb.tick(0.01, 1.0);
    }
```
HelloWorld Code Responsibilities

• 1. Create and destroy the federation.
• 2. Join and Resign from the federation.
• 3. Declare data to be published and subscribed to by the federation.
• 4. Send/Receive data to/from other federates.