RTI Interoperability Issues - API Standards, Wire Standards, and RTI Bridges

03E-SIW-077

Len Granowetter
MÄK Technologies

lengrano@mak.com  http://www.mak.com
(617) 876-8085 x121
Local RTI Components (LRC’s) from different RTI implementations can not communicate
Federates Need to Agree on an RTI Implementation

I’m using the MÄK RTI. You need to switch.

No! I already bought the pRTI. You switch!
Overcoming the RTI Interoperability Barrier

- **Eliminate** the barrier
  - Use “wire standards” to allow different RTI implementations to communicate
- **Work around** the barrier
  - Use RTI bridges to translate between different RTI implementations
- **Lower** the barrier
  - Make it easy for federates to switch to the RTI implementation chosen by the federation
Eliminating the Barrier: The Allure of “Wire” Standards

- Wire standards (like DIS) specify the interface between an application and the network
- Applications not required to share any common software modules
  - As long as they can encode and decode the network data correctly, they can play
- Federate developers not constrained by platform or language choices made by others
If all RTI implementations agreed on a wire protocol, different federates could use different vendors’ LRCs, or could even provide their own LRCs.
Problem: Wire Standard Nullifies Many of HLA’s Advantages

- Both the HLA approach and wire protocols have their pros and cons
  - But mixing the two ideas is worst of all
- Usual argument: We can not have a wire protocol, since HLA must support shared memory, etc. in addition to networking
  - However even in the case of network-based RTIs, a standard for the interface between LRC and network is a bad idea
Wire Standards and HLA

- Wire standard is much more than just a scheme for encoding attribute values
  - Dictating a wire standard mandates a single approach to DDM, Time Management, reliable transport, advisories, need for rtiexec, etc.
- HLA interface specifically designed to allow multiple implementation choices
  - No single approach suitable for all federations
- Mandating implementation means federation developers can’t make their own tradeoffs
Example: Reliable Transport

- Fully connected approach has lower latencies
  - Each federate connected to every other
- TCP exploder allows RTI to return sooner
  - Federate sends only a single copy of a packet
- Relative importance depends on federation
- Even if we mandate fully connected
  - Who initiates connections?
  - How do we know who to connect to?
  - Tradeoffs between ease of configuration, bandwidth usage, time required to initialize
Example: DDM

- Even if we assume it’s OK to mandate a multicast-based scheme
  - Fixed grid approach is simpler, but must associate regions with groups a-priori
  - Distributed region approach more complex, but allows more accurate multicast filtering, and never sends a packet more than once
  - Which is best depends on federation - how often regions change, size of update regions, number of updates sent to each region, etc.
An RTI-to-RTI Bridge joins two or more federations, and passes data between them.
RTI Bridges

• Feasible approach for subset of RTI functionality
  – Several have been prototyped or built
    • MÄK, SAIC/CMU

• However, some important disadvantages
  – Loss of performance
  – Limited functionality
  – Difficult to debug a federation
  – For some RTI services, bridging not possible
  – Bridge rivals complexity of RTI
Limitations of RTI Bridges

• Much RTI internal state and functionality not available through RTI API
  – Bridge is not notified when a federate invokes each service
  – Therefore, can not “forward” that invocation

• Bridges are appropriate for hierarchical federations
  – Distinct separation is desired
  – Goal is not to approximate a single large federation
Limitations of RTI Bridges

- DDM – Information about region subscriptions/associations not available
  - Would be useless anyway, without control of underlying communications mechanisms
- Ownership transfer has been demonstrated
  - But cases exist where multiple federates can each think they’ve gained ownership
- Many MOM attributes would be misleading
- Even Declaration Management is problematic
  - Extra subscriptions are necessary
Lowering the Barrier: Making it Easier for a Federation to Choose a Common RTI
Remember the Meaning of RTI

The whole package: An RTI includes LRCs, plus any supporting tools necessary for their implementation.
The HLA Concept

- HLA defines an interface that federates use to exchange data
- Implementation of the interface left to RTI
- RTI choice is a federation-wide decision
  - Based on specific federation requirements
- A single federate may be reused in many different federations
- Asking how different federates can communicate using different “Run-Time Infrastructures” makes little technical sense
Analogy to OpenGL

- OpenGL provides a standard interface (API) to a 3D graphics system
- System includes a library that implements the interface (driver) and supporting tools (card)
- “Wire” (pin I/O) protocol is up to card vendor
  - API allows many different implementations
- Can’t switch cards without switching drivers
- But very easy to switch cards/drivers
  - Just swap shared libraries (DLLs) and go
  - A single video game supports many cards
Switching RTI Implementations

• Like OpenGL, switching RTIs must be easy
  – If this can be achieved, we can avoid the “No, you switch!” arguments
  – If not, users will be forced to look at bridging solutions, despite their limitations
  – Reasonable track record so far, but much room for improvement

• Making it easy to switch RTI implementations is not purely a technical problem
Lowering the Technical Barrier

- Different RTI implementations must be dynamic-link-compatible (swappable DLLs)
  - Allows end user to choose RTI without going back to federate developer (no source needed)
  - Allows tool providers to support many different RTIs with a single release
  - Allows federates not under active development to be used in new federations
- Reasonable degree of compatibility today
- SISO DLC HLA API PDG working out remaining issues
Non-Technical Barriers

• Supporting multiple RTIs is expensive
  – A federate developer that has invested in one RTI does not want to have to buy another

• Federation developers need to realize that an RTI is a piece of **federation** infrastructure
  – Like dedicated network rental, hardware, etc.
  – RTIs have never been free, just a hidden cost
  – If cost is budgeted by program, federate developers will not see un-funded requirement

• Important to compare the cost to alternatives
  – Bridging is **much** more expensive
Non-Technical Barriers

- What if no single RTI exists that supports all desired platforms?
- Expensive to contract RTI vendor to port
  - Porting an RTI is almost always cheaper (and always less limiting) than bridging
  - Situation is less and less common – three commercial RTIs support 5 or more platforms
Summary

• **Eliminating** the barrier through a wire standard nullifies most of HLA’s advantages
  – Locks end-users into one-size-fits-all solution

• **Working around** the barrier through bridging fails to achieve desired behavior
  – Appropriate for hierarchical federations, though

• **Lowering** the barrier is most promising
  – Through better standards, better budgeting, and better cooperation among stakeholders, we can make it quite easy for all federates in a federation to agree on a common RTI