High Level Architecture
Module 2
Advanced Topics

Roy Crosbie
John Zenor
Simon Goberstein

California State University, Chico

High Level Architecture
Module 2
Advanced Topics

Data Distribution Management 1:
Fundamentals and Multidimensional Regions

Data Distribution Management (DDM):
1. Fundamentals and Multidimensional Regions

- This lesson and the next one are based on K. L. Morse and J. S. Steinman's paper "Data Distribution Management in the HLA", 1997 Spring Simulation Interoperability Workshop

The Goal of the DDM Services

- Reduce the transmission and the reception of ***irrelevant*** data in **large distributed federations** by limiting the messages received by federates to only those of interest, and thus reduce
  - The data set to be processed by the receiving federate
  - The message traffic over the network
- Provide this functionality in a straightforward and easy to use manner

The Issues to Be Addressed by the DDM Services

- Data Distribution Management services are specifically concerned with
  - **Efficiency**
  - **Scalability**
  - **Interfaces**
- Each of these issues will be discussed separately in the next three slides

Efficiency

- DDM services should involve minimum overhead in terms of ***computations***, ***messsage latencies***, and ***memory usage*** for all its services
- Expensive operations (string comparisons, costly or complex computations, etc.) should be avoided
- The incurred overhead cost must be justified
- An expensive service may be acceptable if it saves more than it costs

Scalability

- DDM services should scale in terms of:
  - **Computational complexity** for handling requests
  - **Message traffic** and/or bandwidth for distributing information
  - **Memory requirements** for storing attribute information
- The main parameters affecting scalability are:
  - The **number of federates** (or hosts) in the federation
  - The **number of stimulated entities** per federate
  - The **average complexity** associated with **each entity**
  - The **interaction rates between objects**
  - The **locality of objects**

Objects that are *local* (that is, those within the same federate) can interact without requiring messages to be sent through the network. In large applications, it is sometimes possible to decide which federates create which objects. It often makes sense for objects that stay within the same physical region during the course of the simulation execution to be created within the same federate to provide better locality for the federation.

Interfaces

- DDM services must support the ***right interfaces*** to provide the ***right filtering functionality***
- The interfaces should be made easy to use while supporting the essential DDM services
- The interfaces must also be sufficiently general so that different underlying implementations may support the same common interfaces

Declaration vs. Data Distribution Management

- Declaration Management (DM) services:
  - Provide information on data relevance at the ***class attribute level***
- Data Distribution Management services:
  - Add the capability to refine the data requirements at the ***instance attribute level***
- These differences are discussed in more detail in the next two slides

DM Services: Class-Based Filtering

- Recall the *publish* and *subscribe* mechanisms of DM services described in Part 5 of Module 1
- A federate subscribing to attribute values of a class will get ***all*** updates of these values for ***all*** objects of that class currently existing in the federation
- This allows the subscribing federate to eliminate delivery of attributes for ***whole classes*** of objects in which it has no interest
- This is called **class-based filtering**

DDM Services: Value-Based Filtering

- Class-based filtering is well suited for supporting performance and scalability for small federations
- Large federations require more refined filtering to improve their performance and scalability
- DDM services allow a federate to receive the subscribed attributes ***selectively*** based on values of characteristics of the publishing federate
- This refined type of filtering provided by the DDM services is called **value-based filtering**

Data distribution management services may be used by federates to **reduce** both the **transmission** and the **reception** of **irrelevant data**. Produces of data may employ DDM services to assert ***properties*** of their data in terms of user-defined spaces. Consumers of data may employ DDM services to specify their data ***requirements*** in terms of the same spaces. *The RTI distributes data from producers to consumers based on* ***matches*** *between these* ***properties*** *and* ***requirements***. In the remaining part of the lesson, this general scheme will be explained in much more detail.

Routing Spaces

- A **routing space** is a multidimensional coordinate system in which federates express an interest for either receiving data or sending data
- This concept is fundamental for supporting efficient data distribution across a federation
- The multidimensional routing space is subsetted to capture interest in receiving or sending data via:
  - **Update regions**
  - **Subscription regions**

Update and Subscription Regions

- An **update region:**
  - Defined in the corresponding routing space by a set of an object's coordinates changing dynamically over time
- **Subscription regions:**
  - The relevant discovery criteria, defined as bounded regions in the routing space, that describes the interests of subscribing federates which may dynamically change over time (these regions may change in size or position)
- An object is **discovered** by a federate when the object's update region overlaps the federate's subscription region

1) As the coordinate values of an update region change dynamically over time, one can conceptualize a trajectory in a multidimensional space representing an object's volume in the routing space as it evolves over time.

2) Update regions can be represented as n-dimensional rectangular volumes instead of points in a routing space. This can be useful for large objects, such as weather clouds, whose extents may overlap large routing space regions. Extended update regions can also be used to bound an object's true value in the routing space.

3) Regions in a multidimensional space do not necessarily map to physical geographical regions. A region in a routing space should be thought of as an abstract volume with possibly more than or less than three dimensions.

Federation's Routing Spaces

- To use routing spaces, *each federation defines the allowable routing spaces for its execution*, including the dimensions (coordinates) of the routing space
- Routing spaces are specified in the Federation Execution Data (FED) file with a name, the number of dimensions and additional parameters

The Routing Spaces and the Federates

- The federates specify the **types** of data they will send or receive using the DM services (either by class and attribute name or by interaction class)
- Routing spaces are used by the federates to specify the *distribution conditions* for the ***specific*** *data* they are sending or are expecting to receive
- Each federate decides which routing spaces of the federation are useful to it and defines their portions that specify **regions of interest** for that federate

Using Regions of Interest

- The federate uses its regions of interest to specify
  - Conditions under which it expects to receive the object state data and interactions specified using DM services (*Subscribe Object Class Attribute With Region*)
  - Conditions under which it is providing data (*Register Object Instance With Region* and *Associate Region For Updates*)

Using Update and Subscription Regions

- By specifying a **subscription region**, the federate tells the RTI to deliver data only from those update regions whose bounds overlap with the bounds of the region specified by that federate
- By specifying an **update region** and associating that update region with a particular object instance, the federate ensures that the characteristics of the object instance or interactions which map to the dimensions of the routing space fall within the bounds of the associated region when the attribute update or send interaction call is issued

It follows that the federate is monitoring these added characteristics for each of the attributes owned by the federate. As the states of the objects change, the federate may need to either adjust the bounds of the associated regions (*Modify Region*) or change the association to another region (*Associate Region For Updates*).

Routing Spaces and Regions, and Data Distribution by the RTI

- The RTI uses the routing spaces, regions, and the association data for distributing data:
  - When an update region and subscription regions of different federates overlap, the RTI ensures that attribute updates and interactions associated with the update region are routed to the federates whose *subscription regions* ***overlap*** the sender's *update region*
  - The subscribing federates each receive only the class attributes and interactions to which they subscribed

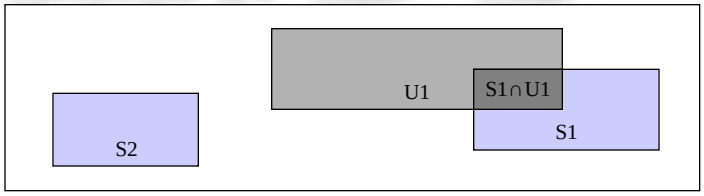
It must be noted that the *dimensions* of routing spaces **need not** necessarily **map** to *attributes* in the FED. The DDM services provide a federation with the capability to specify data distribution on characteristics of objects other than those exchanged as part of the federation execution. *By specifying routing spaces and regions using attributes specified in the FED, the DDM services provide a mechanism to* ***control data distribution*** *based on* ***values of attributes***.

Multiple Update and Subscription Regions

- Each federate can create multiple update and subscription regions
- Update regions are associated with individual objects and are registered with the RTI via the DDM service *Register Object Instance With Region*
- A federate might have a subscription region for each sensor system being simulated

An Example

- Consider a two-dimensional routing space with an update region, U1, and subscription regions, S1 and S2. Since U1 and S1 overlap, attributes and interactions associated with U1 will be routed by the RTI to the federate that created S1. On the other hand, U1 and S2 do not overlap and so attributes and interactions will not be routed from the federate that created U1 to the federate that created S2:



DDM and Associated Services

- DDM services work in cooperation with the HLA DM and OM (Object Management) services to provide value-based filtering
- OM is the group of RTI services dealing with the creation, modification, and deletion of objects and the interactions that they produce
- DM establishes the initial data availability between federates using the object class publication service

Thus:

1) OM services register the objects of interest and reflect their attributes;

2) DDM services ensure that the subscribing federates receive only the attributes of objects which are of interest.

The Functions of OM, DM, and DDM

- More specifically, OM, DM, and DDM services perform the following functions:
  - Create subscription and update regions (DDM)
  - Associate an update region with an object and its published attributes(DDM)
  - Subscribe to object (or interaction) classes (DM)
    - With Region (DDM)
  - Register or discover an object (OM or DDM)
  - Remove a discovered object (OM)
  - Update and reflect object attributes (OM)
  - Send and receive interactions (OM or DDM)
  - Modify or delete existing routing space regions (DDM)

The following slides will describe the DDM process and the flow and interactions of DDM, OM, and DM services in detail.

Object Discovery and Attribute Update

- The federation begins the process by defining the routing spaces (giving their names, the number and names of dimensions, the attributes to be routed) and creating the routing spaces with filter variables
  - The federates must agree on the use of each dimension
- The objects are created by the federates and are registered with the RTI using *Register Object Instance With Region*, so IDs are associated with each object

Objects and Regions

- Object subscriptions may be defined for each object but are associated with the object's federate
- When an object participating in any of the routing spaces is created by a federate, two filter-related quantities may be defined:
  - Its initial update region(s) in each of the relevant routing spaces (*Create Region; Register Object Instance With Region*) so it can be discovered
  - Subscription regions in the routing spaces (*Create Region; Subscribe Object Class Attributes With Region*) so it can discover others

1) Subscription regions do not necessarily have to be associated with an object.

2) As time progresses, the information about update and subscription regions can be updated using the *Modify Region* service in accordance with a consistent and correct strategy that is associated with a routing space. It is up to the federates to define how to use a routing space correctly.

Objects and Their Attributes

- When an object is created, its relevant published attributes need to be *associated* with update regions if filtering is used for this object class
- *Register Object Instance With Region* ties together
  - A routing space
  - The initial update region describing the object in the routing space
  - The corresponding set of published attributes (or its subset) when the object is first created

Object Discovery by Federates

- The RTI determines which objects should be discovered by which federates by **matching** *update* and *subscription regions*
- Federates are notified by the *Discover Object* service of objects that meet the federate's subscription requests
- Object discovery is provided *only once* by the RTI to a federate even when multiple locally-defined subscription regions overlap an object's update region

Attribute Reflection

- After registration, a federate may update the attributes of the object as needed
- The attributes are reflected in each of the federates that have discovered the object later in simulation time or wall clock time due to communication latencies via the *Reflect Attribute Values* service

1) If no federate has subscribed to a particular object, the RTI would perform the optimization of notifying the publishing federate not to update the object's attributes via the *Turn Updates Off* service. If the object is subsequently eligible to be discovered by some federate, *Turn Updates On* would be invoked again to tell the publishing federate to again update the object's attributes.

2) It is possible for federates to try to predict when attributes will change and then update them ahead of time. However, it is much more straightforward to simply update the attributes as they change without trying to predict anything (and live with the fact that the receiving federates will reflect those values later in time due to network latencies). The DDM services work correctly in both cases.

Modifying Regions

- As an object's state variables change over time, it might be necessary to modify subscription or update regions but *this does **not** have to occur **every** time an attribute is updated*
- When an update or a subscription region is modified, the changes are passed to the RTI (using the *Modify Region* service) and it makes a reassessment of the matches between the modified region and all complementary regions

1) When a federate uses the *Delete Object Instance* service, the RTI automatically removes the region's associations with its update regions from the RTI's internal structures and notifies federates reflecting the object to remove it via *Remove Object Instance*. However, a federate must break update region associations explicitly using *Unassociate Region For Updates* before it can delete an update region. Failure to do so will result in an RTI exception.

2) When a federate's subscription region no longer overlaps the update region of an object, the DDM services must coordinate actions to let the receiving federate know that its local representation of the object's attributes may no longer be up to date (if no other locally defined subscription region in any of the routing spaces overlaps the update region of the object). The RTI gives the federate this information via *Attributes Out Of Scope*.

Send/Receive Interaction Services

- The send/receive interaction services are similar to the update/reflect object services
- One difference is that instead of providing attribute value updates, the interactions themselves are bundled with a fixed set of well-defined parameters in the Send Interaction service
- The discovery mechanism of DDM services applies equally to objects and interactions but the latter do not persist, so there is nothing comparable to the *Update Attributes*, *Delete Object* or *Provide Attribute Value Update* services for interactions

It is implicitly assumed that the interaction will be discarded by each receiving federate after it has been processed.