Data Distribution

Static Data Distribution

Richard M. Fujimoto
Professor

Computational Science and Engineering Division
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0765, USA

http://www.cc.gatech.edu/~fujimoto/

Outline

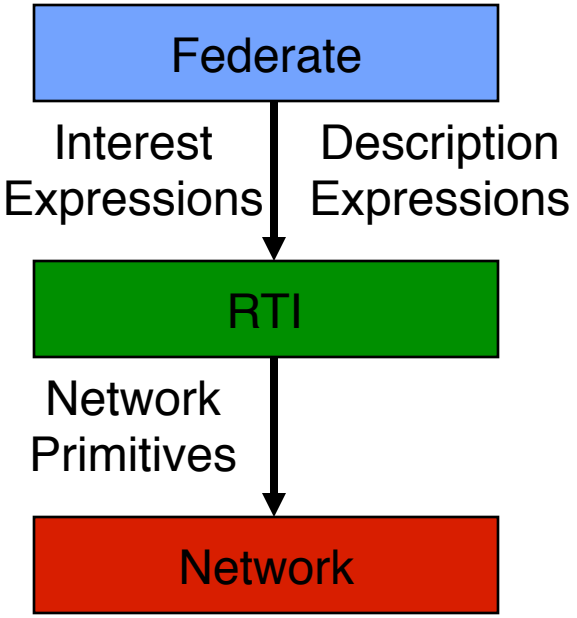
- Fundamental concepts
  - Name space
  - Description expressions
  - Interest expressions
- Static Data Distribution: HLA Declaration Management
  - Class-based filtering

Background

- Basic question: When a simulator generates information (e.g., state updates) that may be of interest to other simulators, who should receive the message?
- Example: moving vehicles in a virtual environment
  - Moving vehicle sends “update” messages indicating new position
  - Each vehicle that can “see” the moving vehicle should receive a message
  - How does the sender/RTI know which other federates should receive the message?
    - Data distribution is essentially a message routing problem

RTI Software

- RTI Interface to the application (federate)
  - Name space
  - Interest expressions
  - Description expressions
- Network primitives
  - Unicast
  - Multicase
  - Broadcast
- RTI design issues
  - What should the federate interface look like?
  - How is the interface mapped to the underlying communication mechanisms



Communication Primitives

- Unicast
  - One sender, message received by one destination
- Broadcast
  - One sender, message received by all destinations
- Multicast
  - One sender, message received by multiple (but not necessarily all) destinations
  - Operations (analagous to newsgroups)
    - Join group
    - Leave group
    - Send message to group
  - Can be implemented by unicast, or network multicast
  - Best effort vs. reliable multicast

Data Distribution in SIMNET and DIS

- Question: Who receives each message that is sent?
- Approach: Broadcast each message, receiver responsible for filtering (deleting) unneeded messages
- Drawbacks
  - $O(N^2)$ messages with N federates; can use large amount of communication bandwidth
  - Time spent receiving and filtering unwanted message becomes a bottleneck

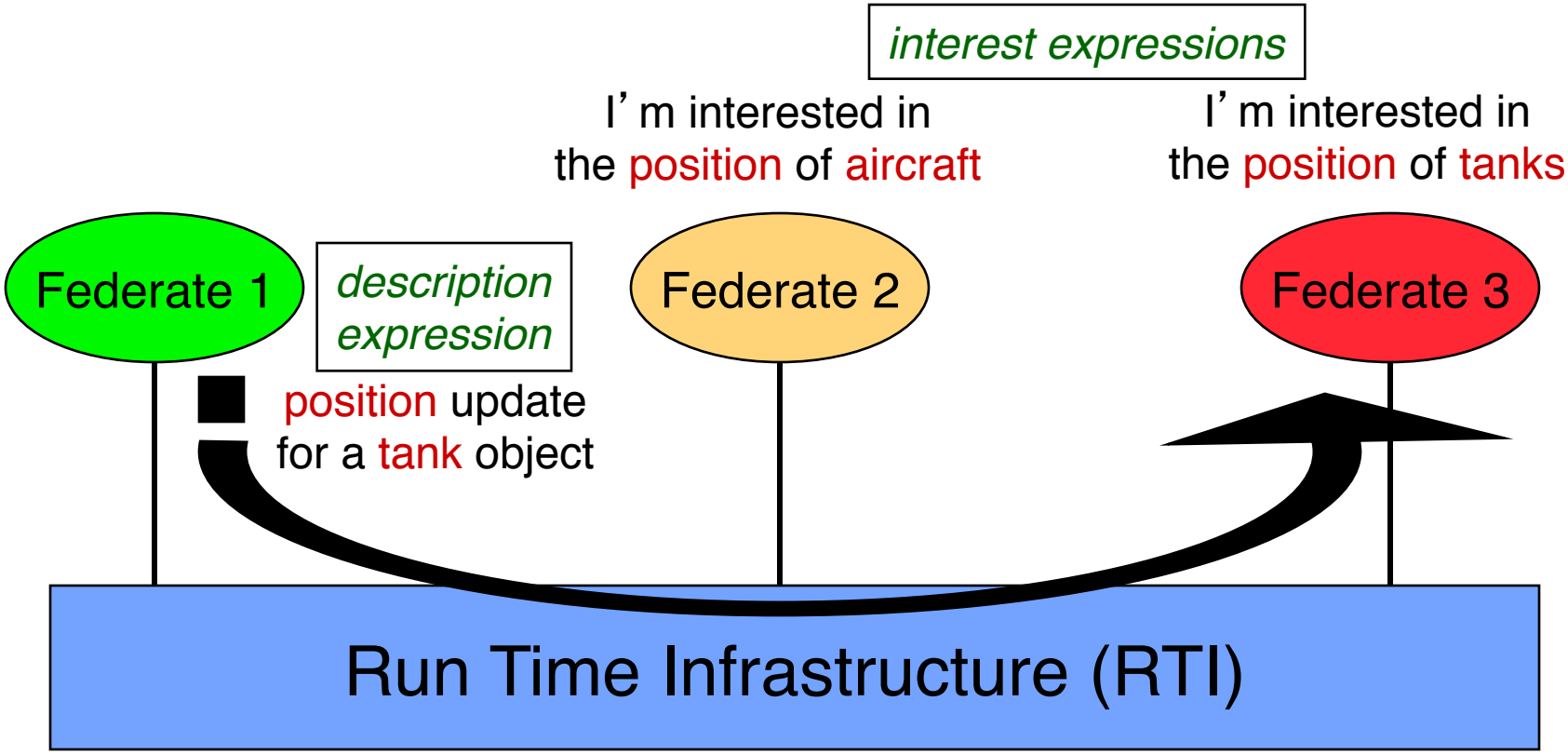
Data Distribution

Goal: route data produced by one simulator to those simulators that are interested in receiving it (and ideally, not to simulators that are not interested in receiving it)

This implies there is

- Some way for a simulator to specify what data it is interested in receiving (interest expressions)
- Some way to describe the data that is produced (description expressions)
- A common language (vocabulary) to specify description and interest expressions (name space)

Example



Name space: position, tank, aircraft

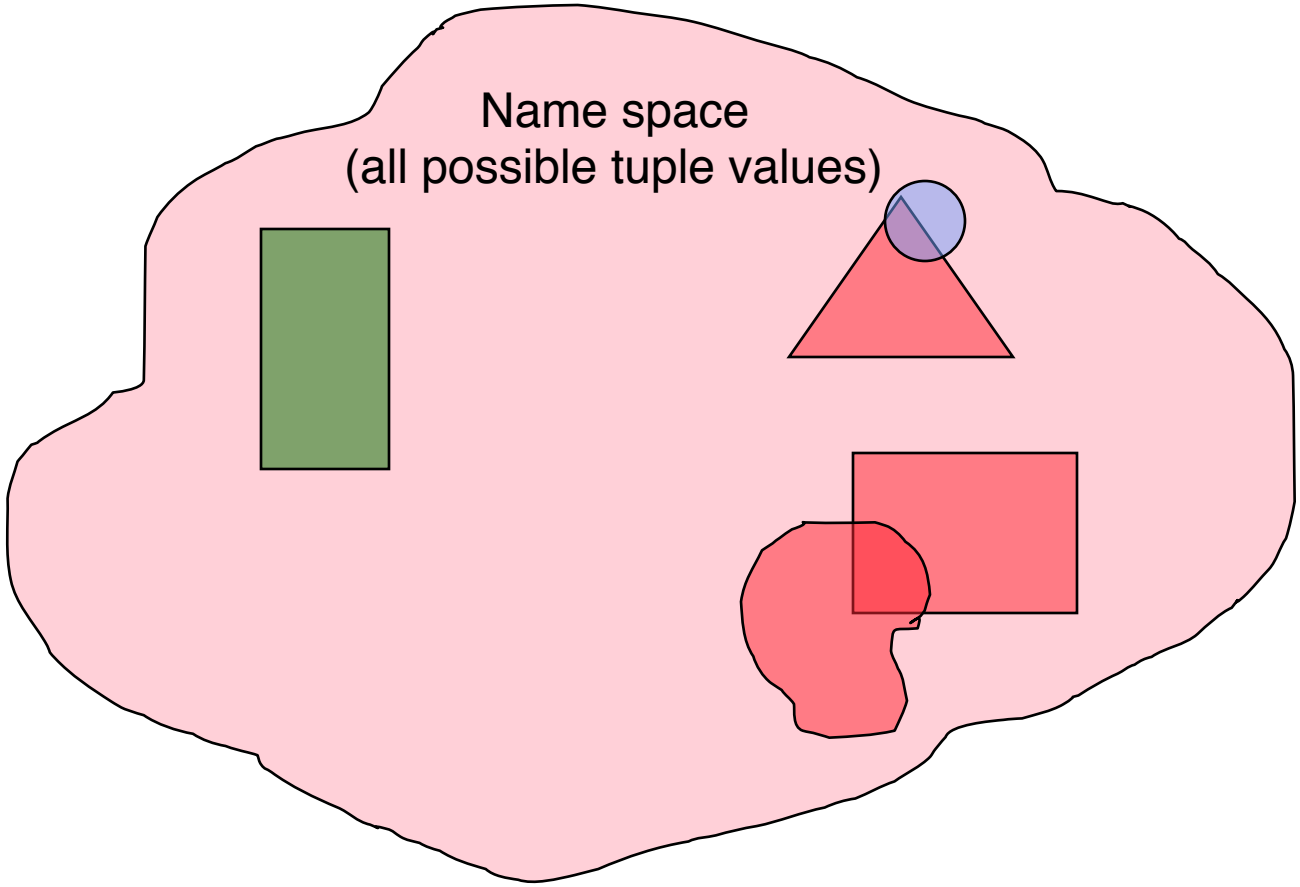
Name Space

- "Vocabulary" used to create
  - Data description expressions
  - Interest expressions
- A name space is a set of tuples (V_1, V_2, … V_N) where V_i is a basic type or another tuple
  - Example: (class, location)
    - Class: enumerated type <tank, aircraft, ship>
    - Location: tuple (int: X-coordinate, int: Y-coordinate), where $0 < X < 1000$ and $0 < Y < 1000$
  - Values in name space: (tank,(30,200)); (aircraft,(10,20))
  - Values of the tuple space may correspond to
    - State variables of simulation (e.g., values of object attributes), or
    - New construct defined just for the purposes of routing data (HLA: routing space regions; newsgroup name)
- May include static properties of objects (e.g., class names, attribute names) or dynamic properties (attribute values)

Interest & Description Expressions

- Interest expression: subset of name space
  - Interested in all aircraft
    - (aircraft, (X, Y)) for any X and any Y
  - Interested in tanks that are "close by"
    - (tank, (X, Y)) where $10 < X < 20$, and $130 < Y < 150$
- Description expression: subset of the name space
  - (tank, (15, 135))
  - (aircraft, (X, Y)) where $35 < X < 38$ and $98 < Y < 100$
- Data routing
  - A simulator receives a message if the message's description expression overlaps with the simulator's interest expressions

Data Distribution Concepts



Interest expressions, Simulator 1

Interest expressions, Simulator 2

Description expression for a message

The message is routed to Simulator 2, but not to Simulator 1.

Static vs. Dynamic Data Distribution

- Static Data Distribution
  - Name space only includes static properties that do not change during the execution
  - Example: Declaration Management in the HLA
    - Class-based data distribution
    - Filter based on data types
    - "Give me updates to the position attribute of all objects of class tank"
  - Cannot filter based on dynamically computed quantities
    - Cannot say: "Give me updates to tank objects that are close to me"
- Dynamic Data Distribution
  - Name space includes dynamic quantities that may change during the execution
  - Value-based data distribution
    - "Give me updates to tank objects that are close to me"
  - Example: Data Distribution Management in the HLA
    - Routing spaces
    - Subscribe to position attribute of tank objects that are in my sector of the battlefield

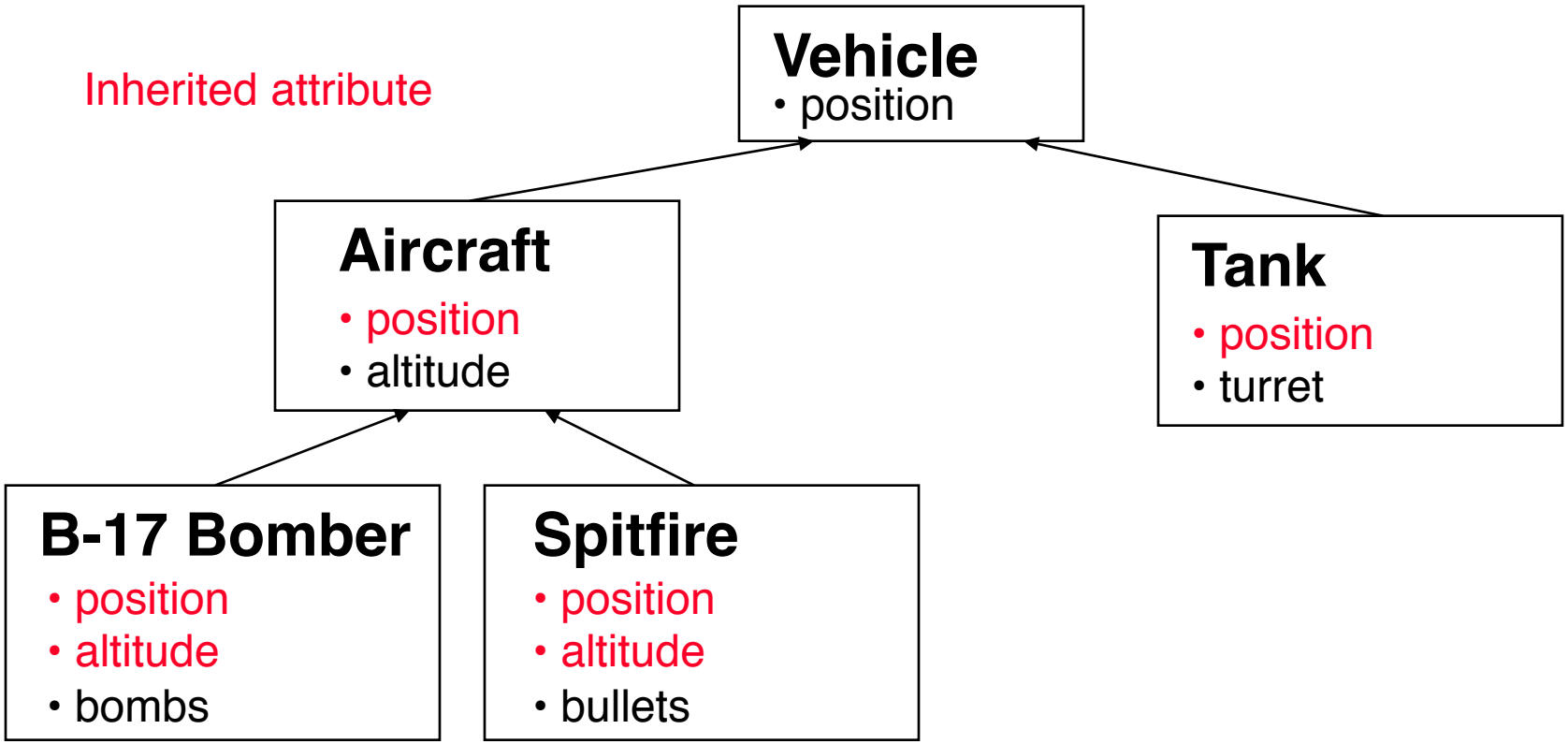
Outline

- Fundamental concepts
  - Name space
  - Description expressions
  - Interest expressions
- Static Data Distribution: HLA Declaration Management
  - Class-based filtering

Class-Based Data Distribution

- Declaration Management services in the HLA
- Federation Object Model (FOM) defines an object class hierarchy describing all data exchanged among federates
  - Object classes
  - Attributes
- Description expressions and interest expressions specify points in the object class hierarchy

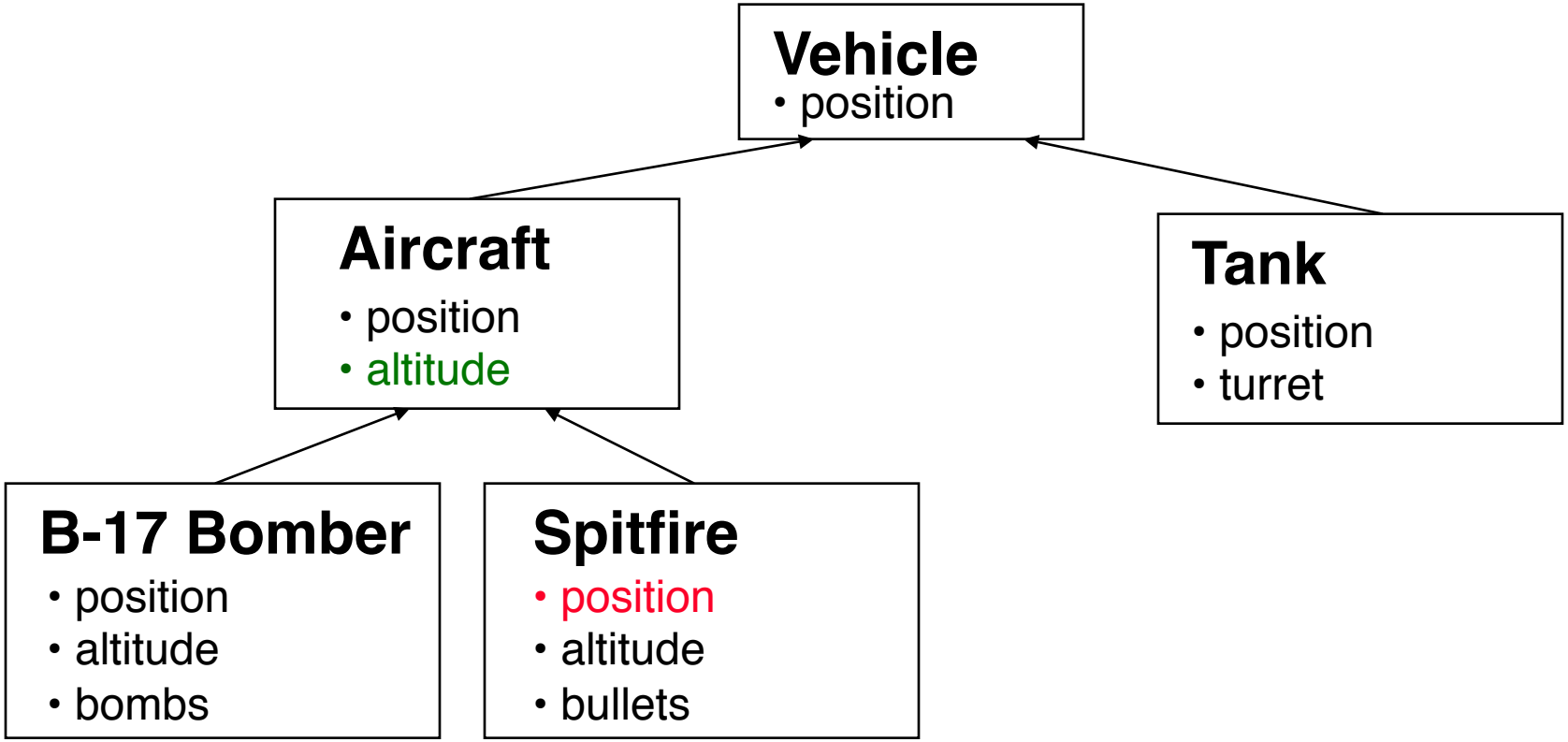
Class Hierarchy Example



- Each class inherits attributes from parent class
- Name space: <class, attribute>

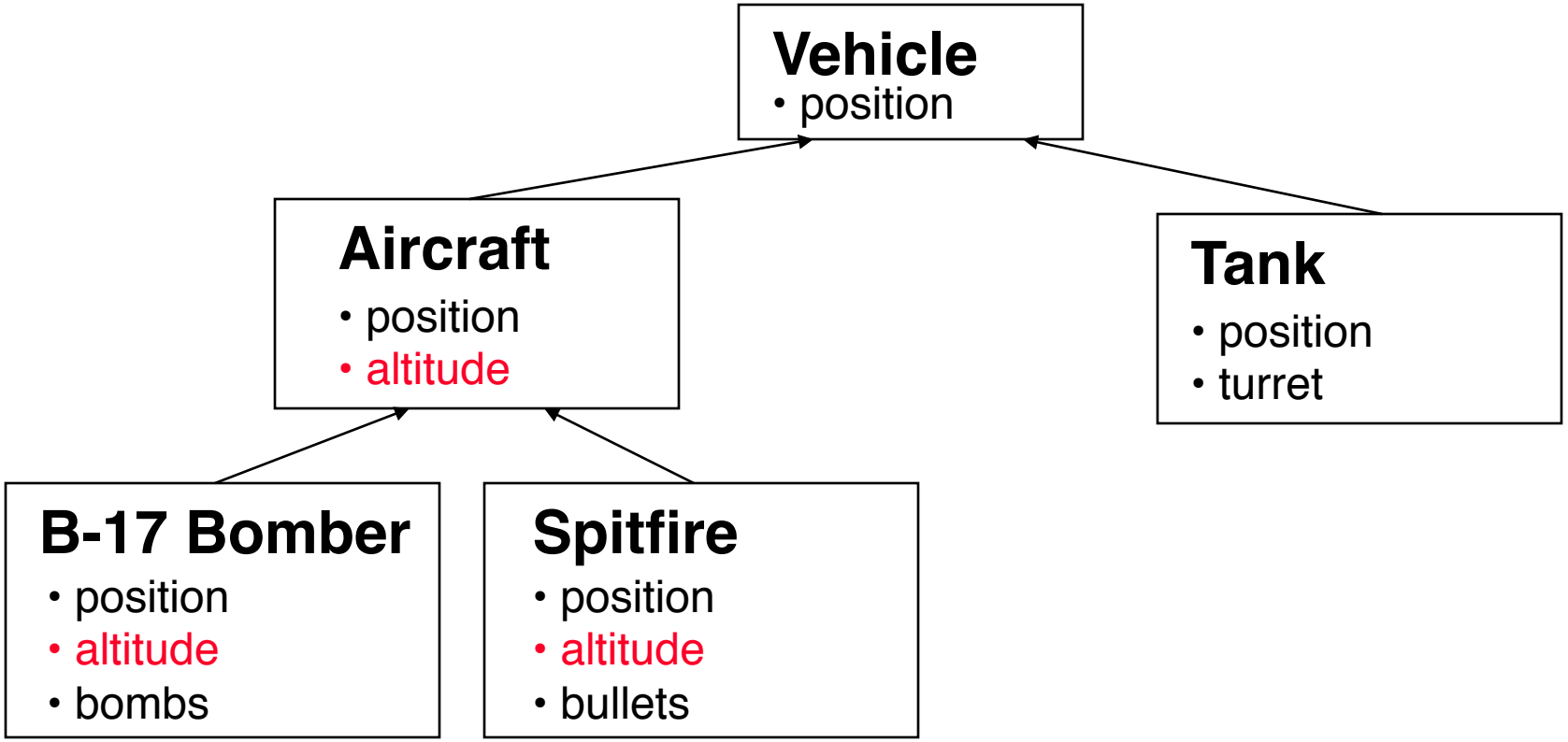
  <Vehicle,position>, <Aircraft,position>, <Aircraft,altitude>, <Tank,position>, <Tank,turret>, <B-17,position>, <B-17,altitude>, <B17,bombs>, <Spitfire,position>, <Spitfire,altitude>, <Spitfire, bullets>

Description Expressions



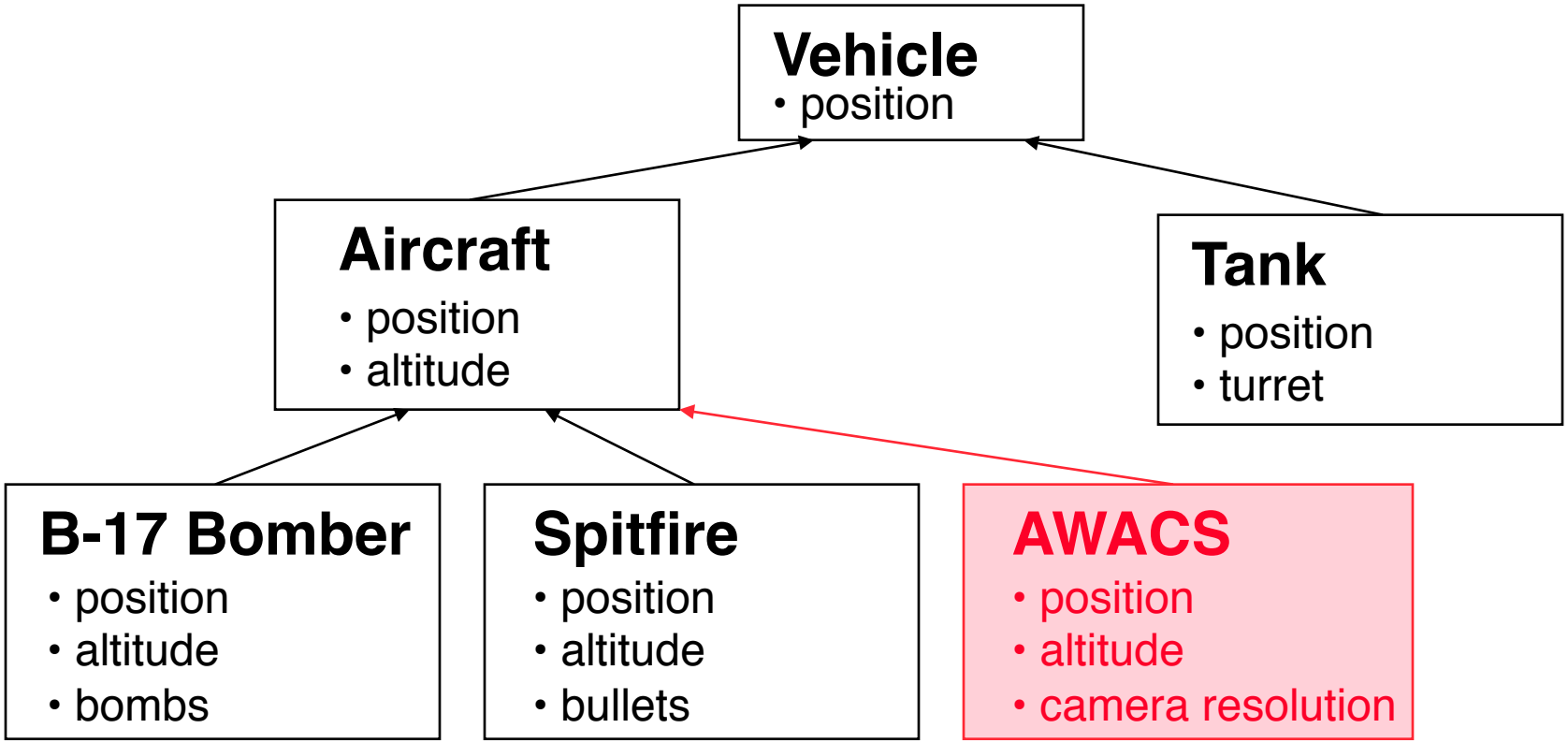
- Update Attribute Values service sends a message
- Description expression: an attribute of an object instance
  - Single <class attribute> point in the name space
  - Examples: <Spitfire, position> or <Aircraft, altitude>

Interest Expressions



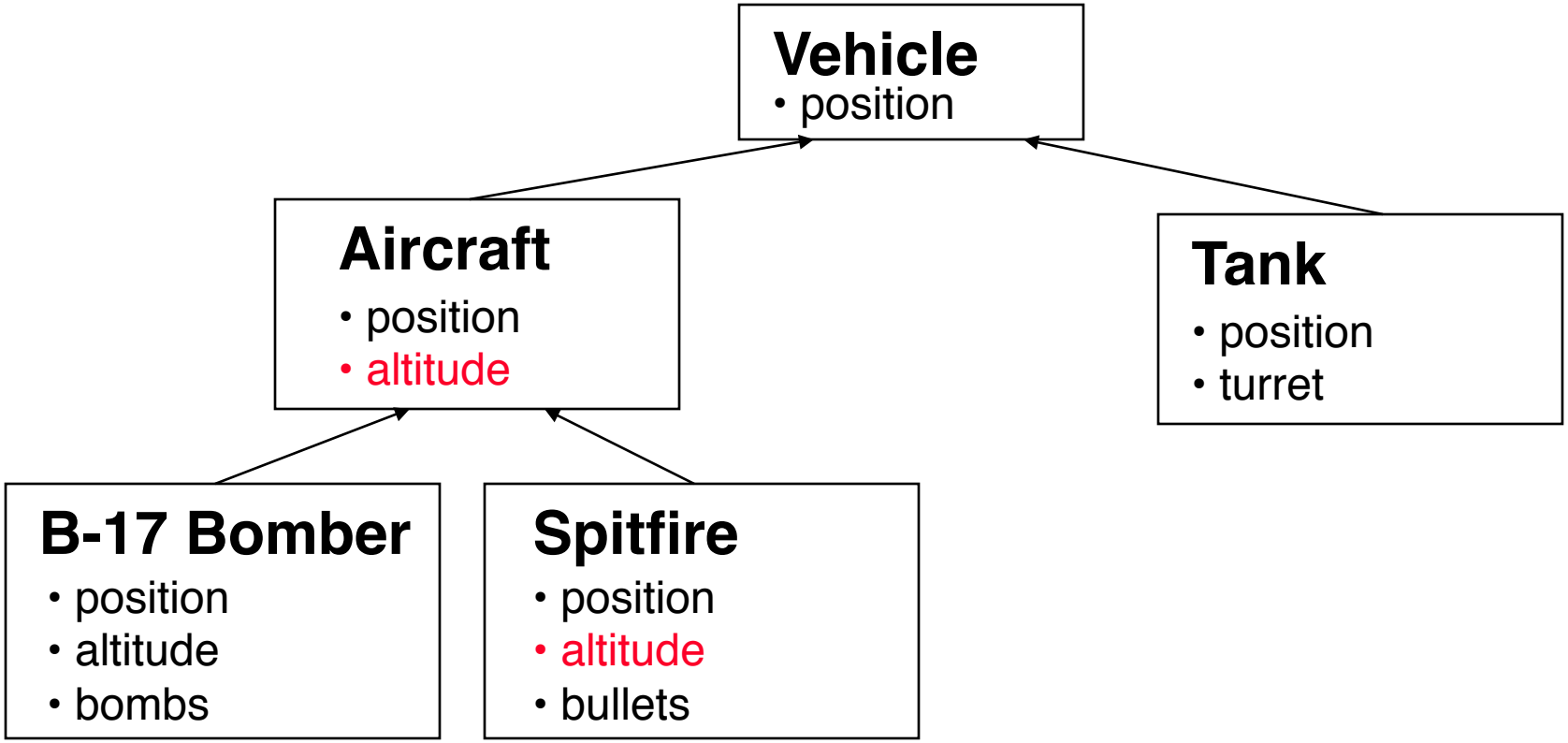
- Subscribe Object Class Attributes [class, attribute(s)]
- Interest expression: Subtree rooted at subscription point
  - Subscribe (Aircraft, altitude): receive updates to altitude attribute of Aircraft, B-17, Spitfire objects
  - <Aircraft,altitude>, <B-17 Bomber, altitude>, <Spitfire, altitude>
  - In all cases, message appears as an update to an aircraft object

Extensibility



New subclasses can be added without requiring modification to subscribers at higher levels in the class hierarchy

Another Approach



- Interest expression: single point in tree
  - <Aircraft, altitude>
- Description expression: Path up tree to point where the attribute is defined
  - Update (Spitfire, altitude): <Spitfire,altitude>, <Aircraft,altitude>

Summary

- Data distribution mechanisms are needed to avoid broadcast communication
- Fundamental concepts
  - Name space
  - Interest expressions specify information simulator wants to receive
  - Description expression describes data contained within the message
- RTI must “match” interest expressions and data description expressions to route data