



High Performance RTI-based Parallel Simulation: Past, Present, & Future

Pankaj Gupta and Dr. Ratan Guha
University of Central Florida



Presentation Overview

- Brief History
- State of the art
 - pRTI, FDK, DMSO
 - Applications
 - DoD & Industry scenario
 - Cluster Computing & Distributed RTI
- Future research directions



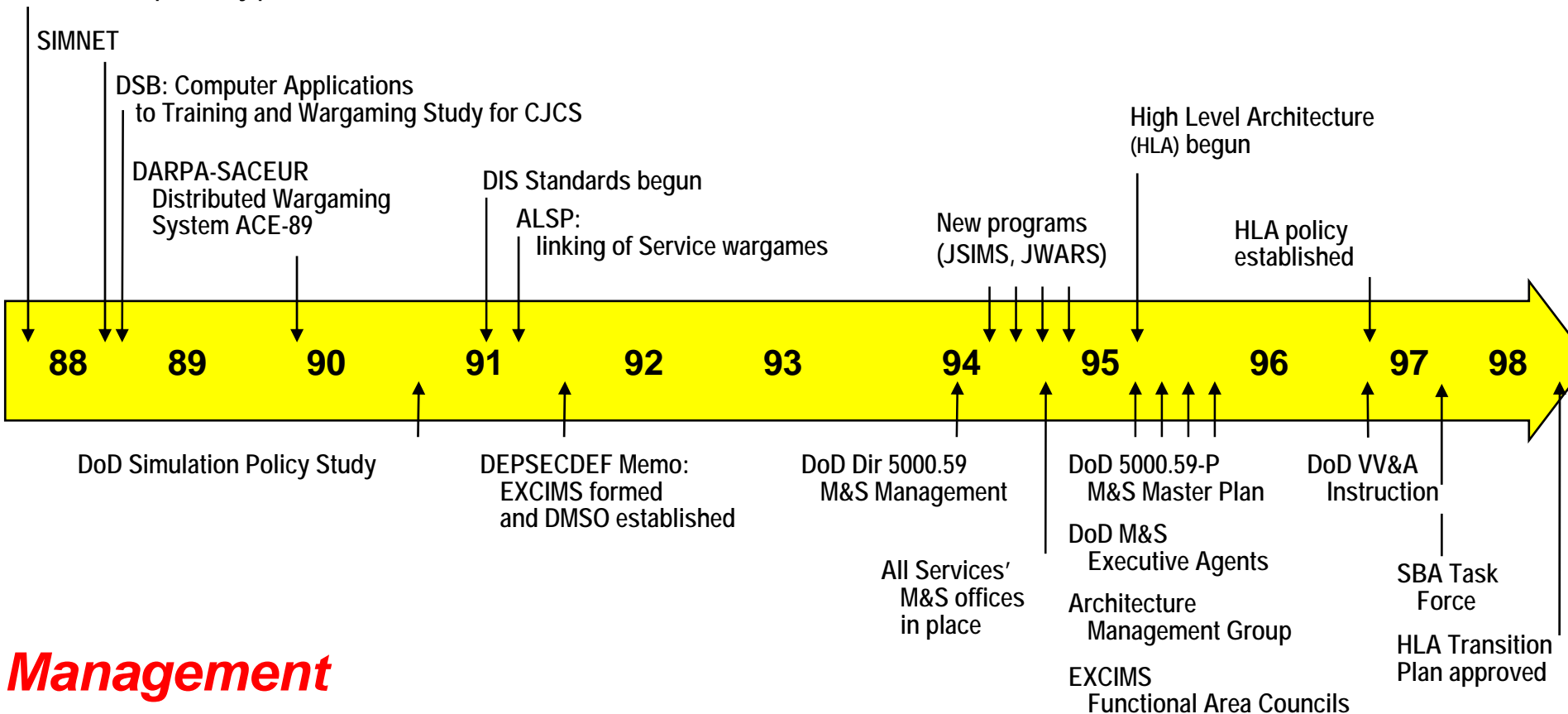
Why Distributed Simulation

- Simulation: an indispensable problem-solving methodology
- Simulation helps to
 - Hold costs down
 - Decrease errors
- No longer the “technique of last resort”

Recent DoD M&S *Technical progress*

Technical

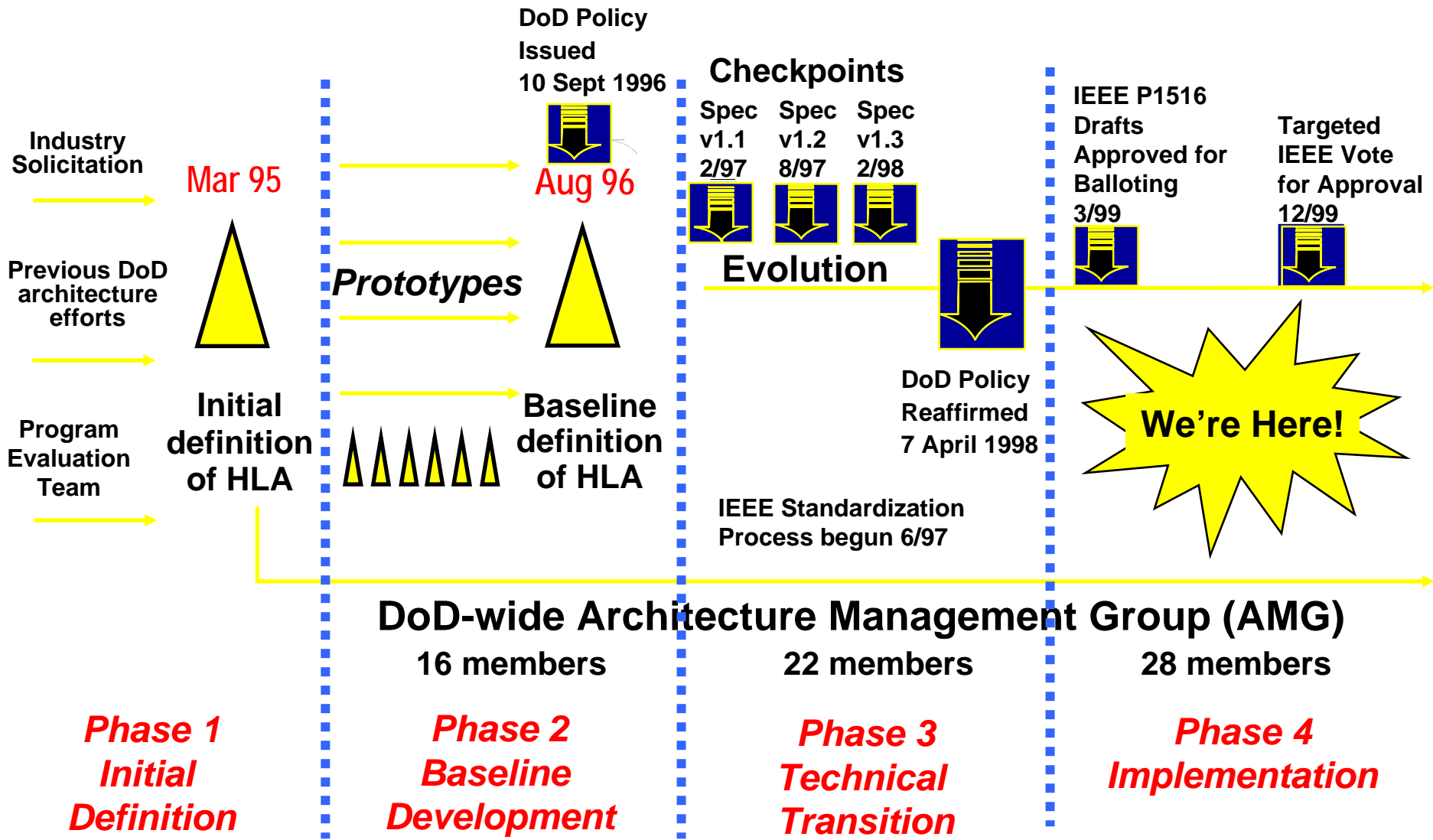
Limited scope simulations,
little interoperability prior to 1988



Management



Detail of HLA Development



Prototyping During HLA Baseline Development

- Over 25 different simulations
 - One Runtime Infrastructure (RTI) implementation
 - Training, analysis, and acquisition applications
 - Unit, platform, and weapon system component level granularity
 - Hardware-in-the-loop, human-in-the-loop, and closed-form simulations (live, virtual, and constructive)
 - Both real-time and fast-as-possible discrete event simulations
 - Both classified and unclassified federations
 - Local and wide area networks Run on Sun, Silicon Graphics, HP, and IBM workstations



Brief History of Distributed Simulation Standards

- SIMNET
- ALSP (Aggregate Level Simulation Protocol)
- DIS (Distributed Interactive Simulation)
- JSIMS
- HLA
- ModSAF
- OneSAF

Distributed Simulations

Applications

- Platform Architecture (vehicle computers, weapons, mobility, crew stations, electronics)
- C4ISR Architecture (sensors, battlefield C2, situational awareness)
- Modeling & Simulation Architecture (computer-based and embedded training, mission rehearsal, dynamic terrain)



HLA Overview

- HLA - Component-based software architecture to provide low-cost, high-capability simulation architecture
- Federation of individual simulations, or “federates”
e.g. cockpit simulator, fighting force
- RTI (runtime infrastructure)
- A FOM (federation object model) that facilitates data exchange between federates in federation



HLA Components

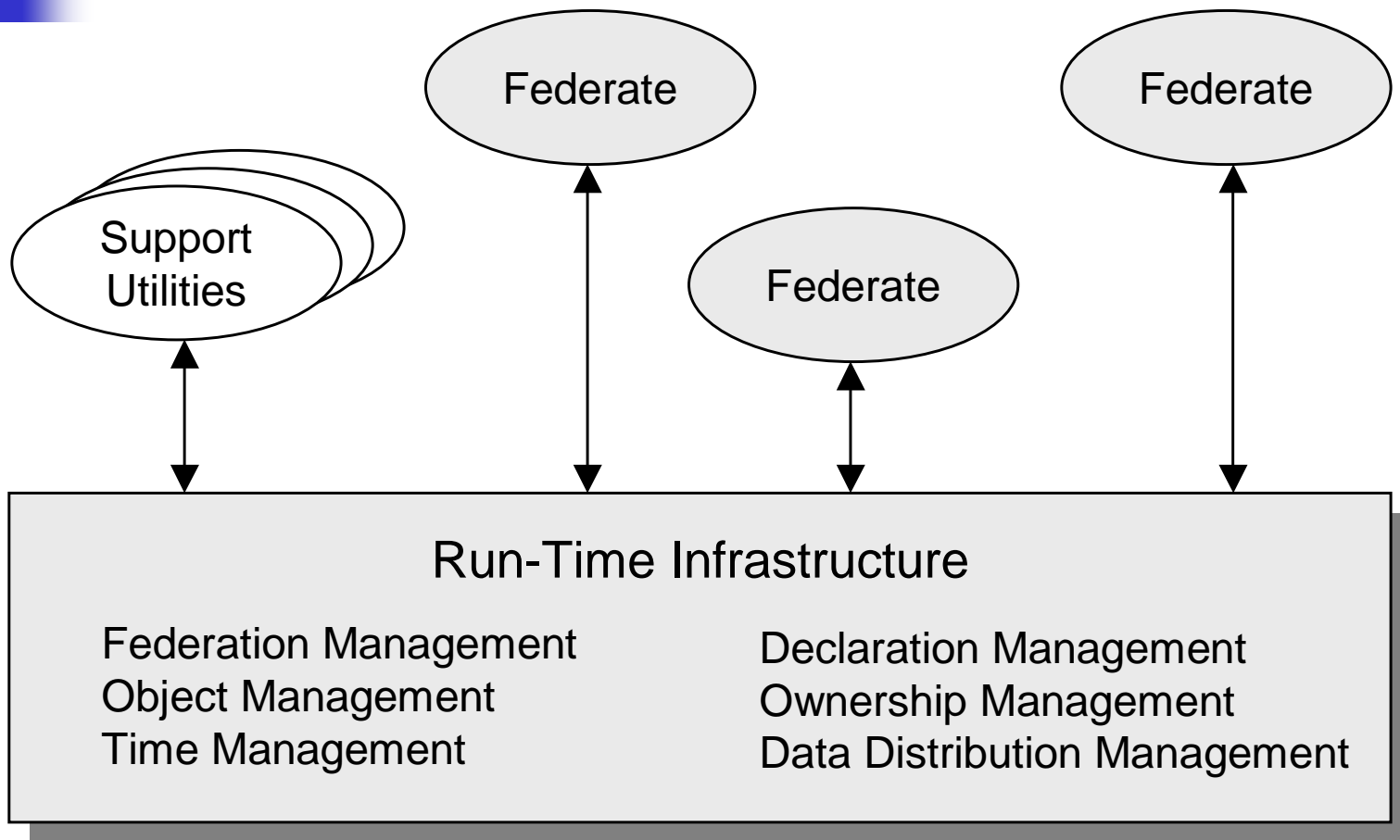
- Architecture specification consists of 3 components:
 - “Ten Rules” that define relationships among federation components.
 - Object model template (OMT) which specifies the form in which simulation elements are described.
 - Interface specification that describes the way simulations interact during execution. Defines access to RTI services as an API (C++, Ada 95, and Java).



HLA goals

- Facilitate the interoperability among simulations
- Promote reuse of their components

HLA Framework





Time Management API

```
while (not done)
  RequestedTime = INFINITE_TIME
  if (Event List Not Empty ) then
    RequestedTime = (Timestamp of earliest event)
  end if
  GrantedTime = NextEventRequest(RequestedTime)
  if (Event List Not Empty ) then
    if (Timestamp of earliest event <= Granted Time) then
      // Ok to process
      (Remove event from Event List)
      (Process Event)
    end if
  end if
end loop;
```



HLA Run-time Infrastructure

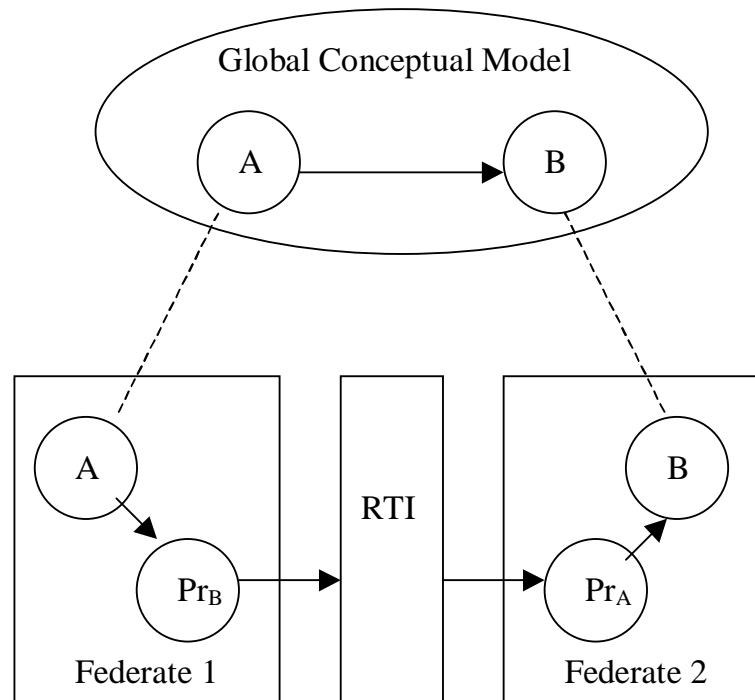
- Federation and object management
- Time management
 - Receive ordered
 - Priority ordered
 - Causally ordered
 - Totally ordered and casually ordered
 - Time stamp ordered



Data Distribution

- Distributing simulation events among producers and consumers
- Define publication/subscription communication services
- RTI ensures messages routing

Global Conceptual Model



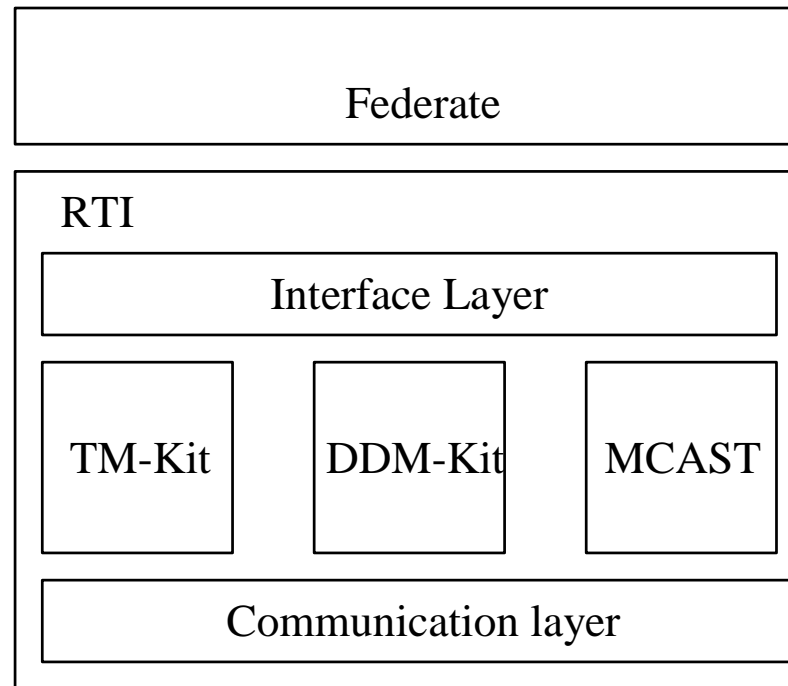
Mapping GCM to Federates. Pr_A is a proxy of A, and Pr_B is a proxy of B.

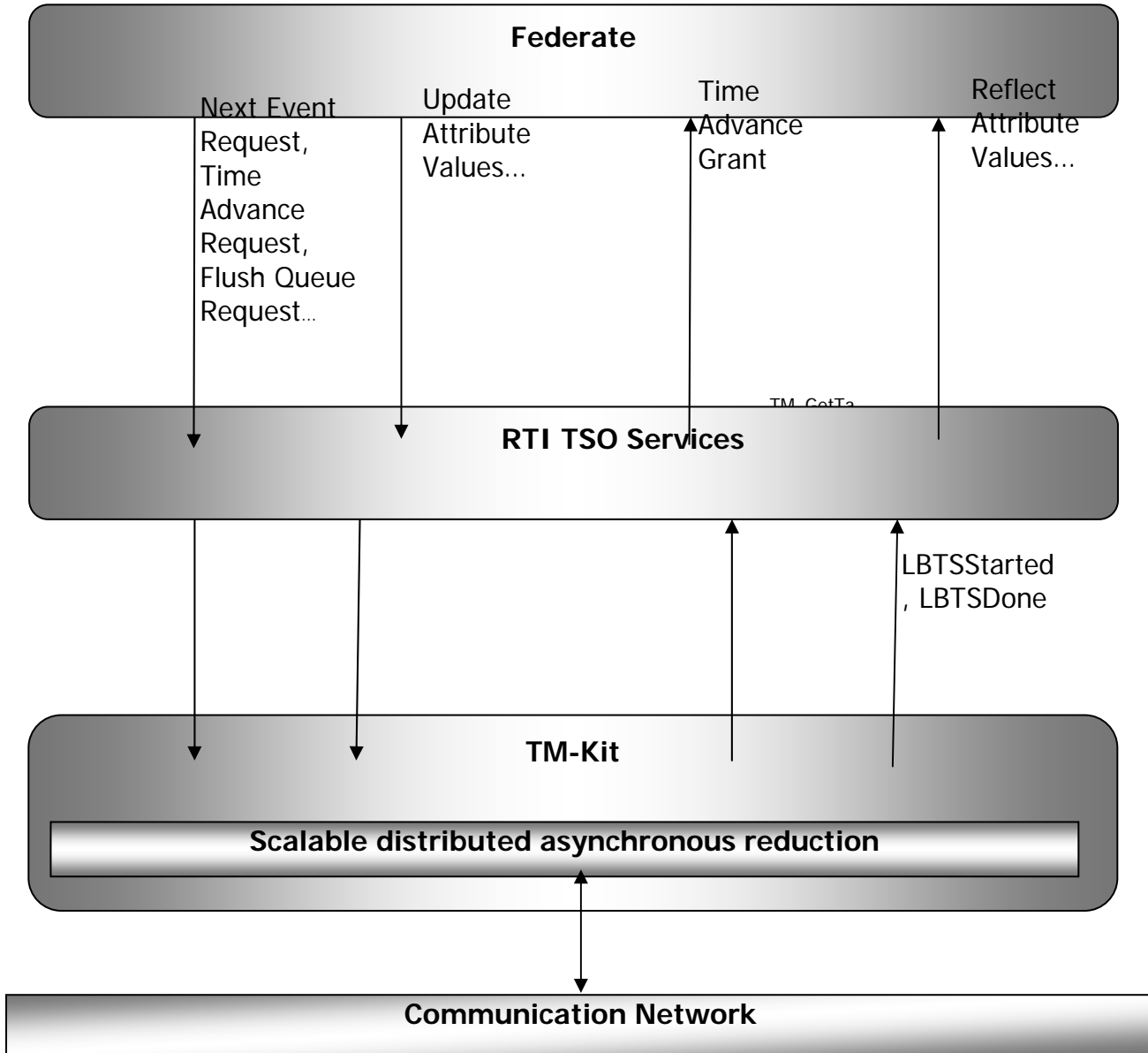


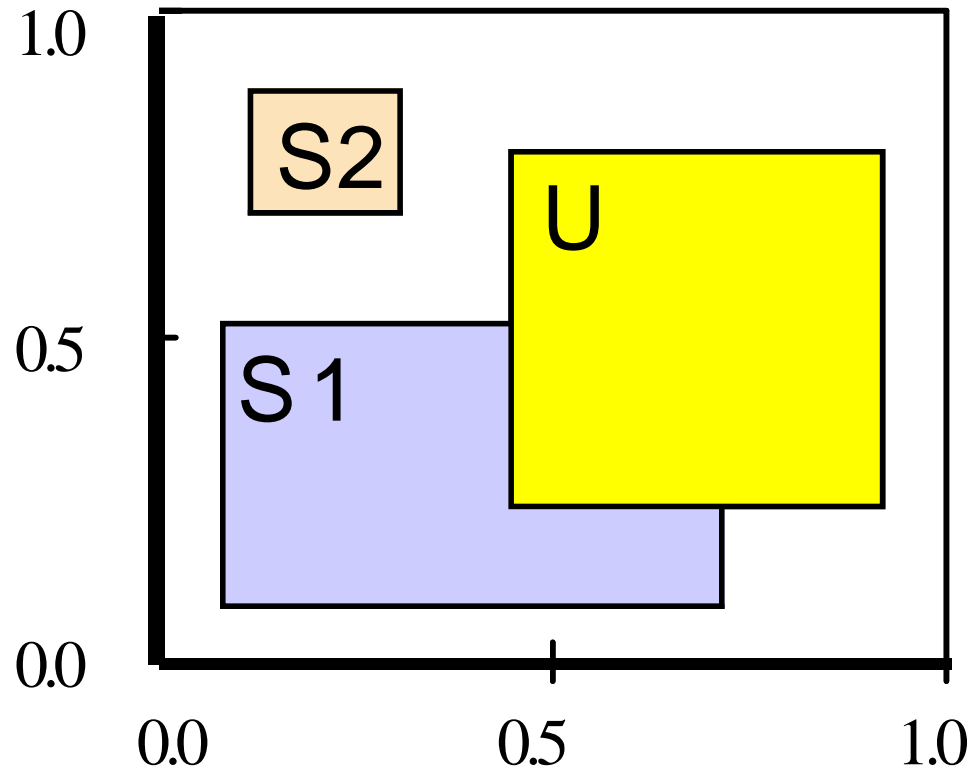
Performance Evaluation

- HLA vs. MPI
- Latency issues
- Comparison of metrics
- Scalability
- Portability issues

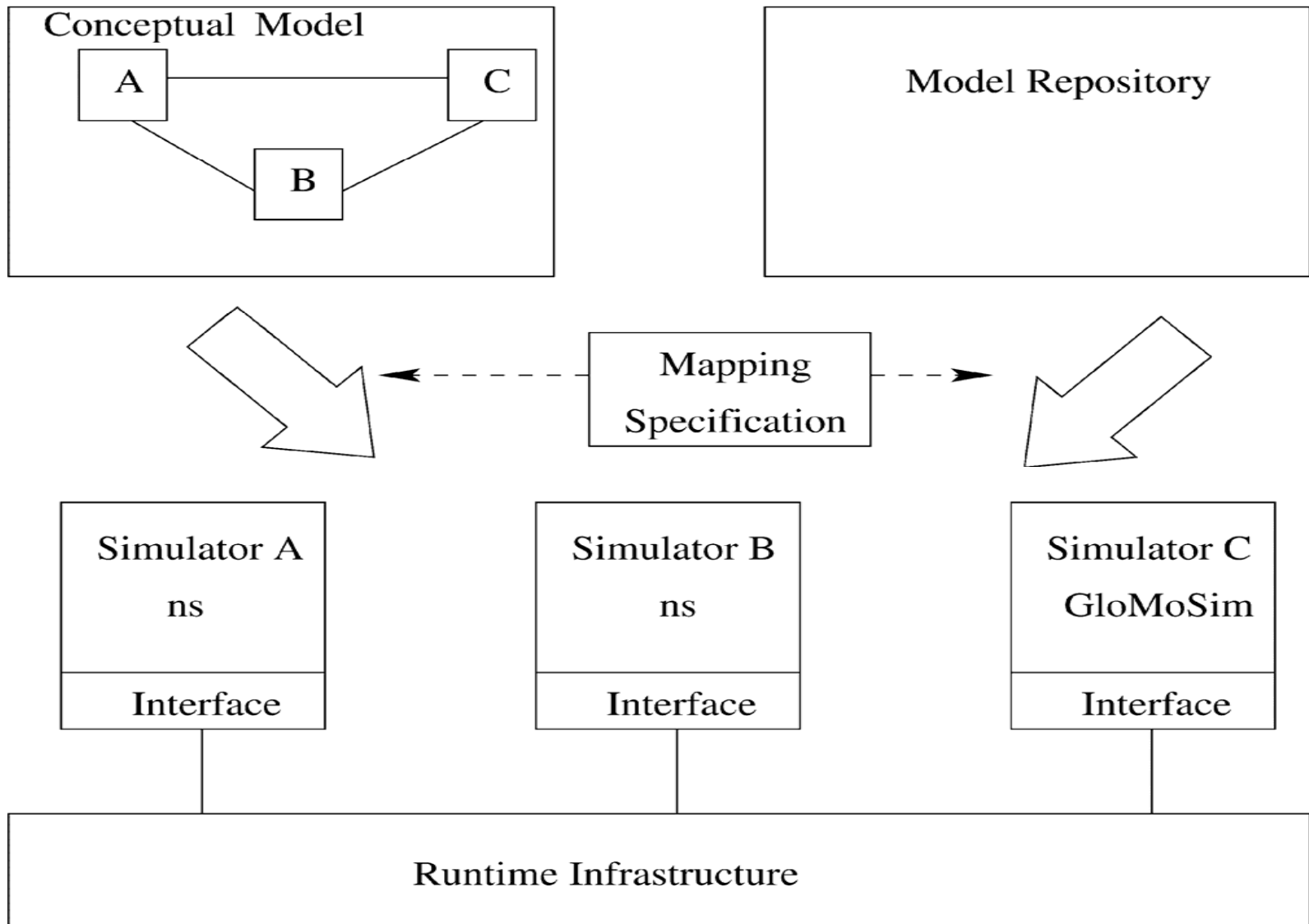
Federated Distributed Simulation Kit (FDK)







Two-dimensional routing space with subscription regions S1 and S2 and update region U



Conceptual Overview



High Performance RTI

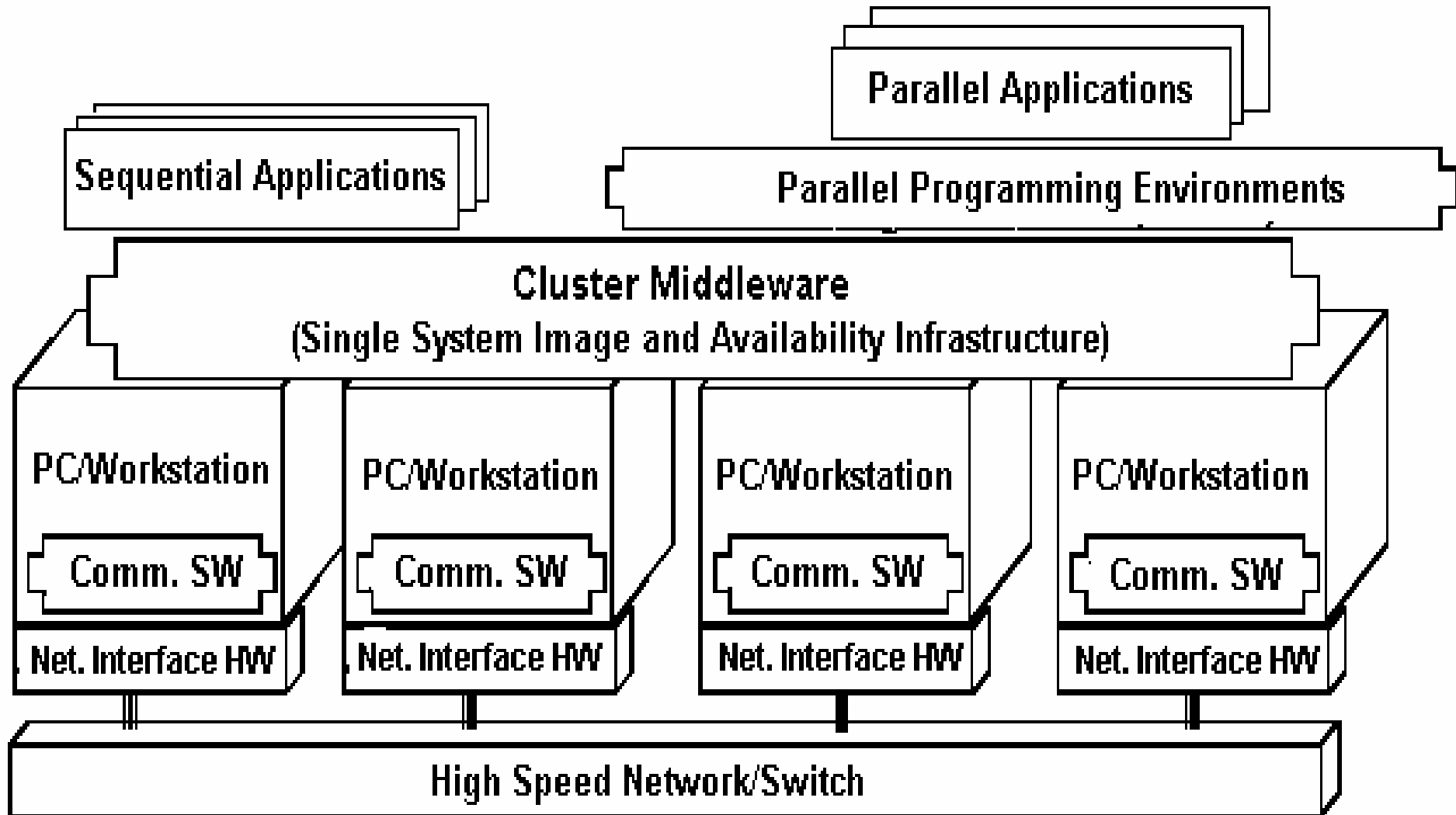
- Varied bandwidth, Often high latency, Flexible communication
- Requires more attention to reliability, security and routing
- Distributed computing
 - Communication overhead
 - Load balancing



Cluster Computing

- Parallel or distributed processing system
- Collection of interconnected stand-alone computers cooperatively working together as a single, integrated computing resource
- A typical cluster:
 - Network: Faster, closer connection than a typical network (LAN)
 - Low latency communication protocols
 - Looser connection than SMP

Cluster Computer Architecture





Cluster Programming Environments

- Message Passing Based
 - MPI
- Shared Memory Based
 - DSM
 - Threads/OpenMP (enabled for clusters)
 - Java threads (HKU JESSICA, IBM cJVM)
- Automatic Parallelising Compilers
- Parallel Libraries & Computational Kernels (NetSolve)



Applications

- Network Simulation
- High Performance Computing
- Portable, scalable paradigm
- Robust computing model



Future Directions

- OneSAF
- CGF
- Comparison with other distributed computing protocols
- Parallel Simulation approach
- CORBA-based HLA ?
- Alternative to Grid computing ?

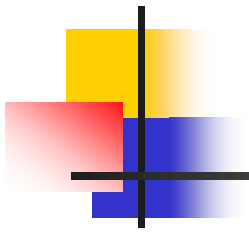


Conclusions

- HLA RTI simulation framework for distributed computing
- Cluster computing platform for development and testing
- Data distribution management for ensuring system scalability
- Applications to DoD problems

Thank You ...



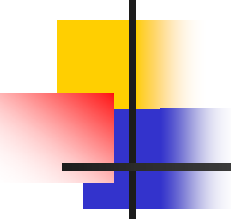


Backup Slides



Distributed Algorithms

- Communication overhead
- Load balancing

- 
-
- Data Distribution Issues
 - Distributed Network Simulation
 - High Performance RTI
 - High Performance Computation

