

A Widely Deployable Web-based Network Simulation Framework using CORBA IDL-based APIs

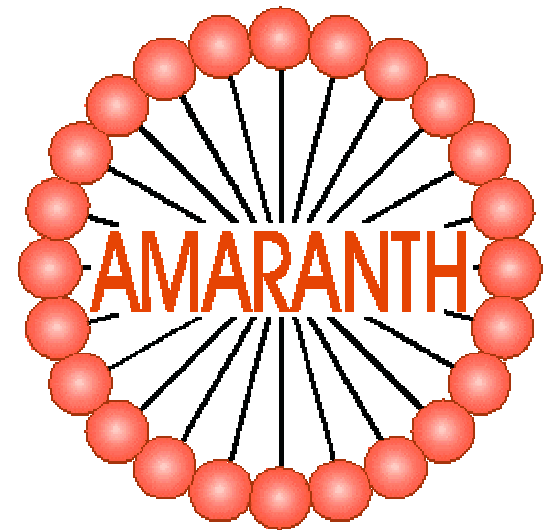
Arjun Cholkar

GTE Data Services, arjun.cholkar@ap.bdi.gte.com

Philip Koopman

Carnegie Mellon University, koopman@ece.cmu.edu

<http://www.ices.cmu.edu/amaranth>



**Carnegie
Mellon**



Institute
for Complex
Engineered
Systems

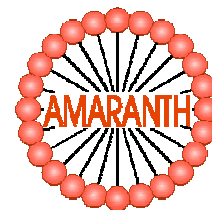


Electrical & Computer
ENGINEERING



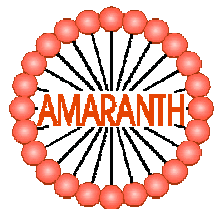
Overview

- **Wide Deployment**
 - Deployability of Current Network Simulation Frameworks
 - How does the Amaranth Network Simulation Framework Achieve Wide Deployability ?
- **The Amaranth Project**
- **ANSF: The Amaranth Network Simulation Framework**
 - ANSF Heritage
 - ANSF: The Big Picture
 - ANSF Internals
 - Communication in the ANSF
 - Facilities in the CORBA-IDL based APIs
- **Conclusions**



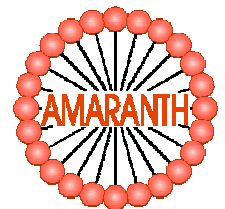
Wide Deployment ??

- **What is Wide Deployment ?**
 - *Dissemination of a software to a large number of platforms for the purpose of installation and use.*
- **Factors affecting Wide Deployment**
 - System capabilities.
 - User comfort and popularity.
- **Characteristics of a Widely Deployable Framework**
 - Easily accessible and available.
 - Platform-independent tools and user-interfaces.
 - Support for customization in a language-independent manner.
 - Extensible.
 - Includes pre-built fundamental simulation structures suitable for re-use.



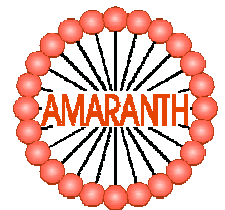
Deployability of Current Network Simulation Frameworks

Product	Language Independent	Platform Independent	Extensible	Deployability
REAL	No	No	Yes	Limited to platforms for which binaries are built.
REAL (Web Based)	No	Yes	No	Limited to users that prefer to run pre-built simulations.
ns2	No	No	Yes	Limited to platforms for which binaries are built.
NetSim ^Q	No	Yes	Yes	Limited to users that are comfortable with programming in JAVA.
Amaranth Network Simulation Framework	Yes	Yes	Yes	Deployable to all platforms that have CORBA support.



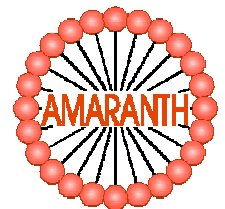
How does the ANSF achieve Wide Deployability ?

- **Characteristics of a Widely Deployable Framework**
 - Easily accessible and available - Use the World Wide Web
 - Platform-independent tools and user-interfaces - Use JAVA
 - Support for customization in a language-independent manner - Use CORBA
 - Extensible - Use Good Object Oriented (OO) Techniques
 - Includes pre-built fundamental simulation structures suitable for re-use - Package Stock Simulation Objects



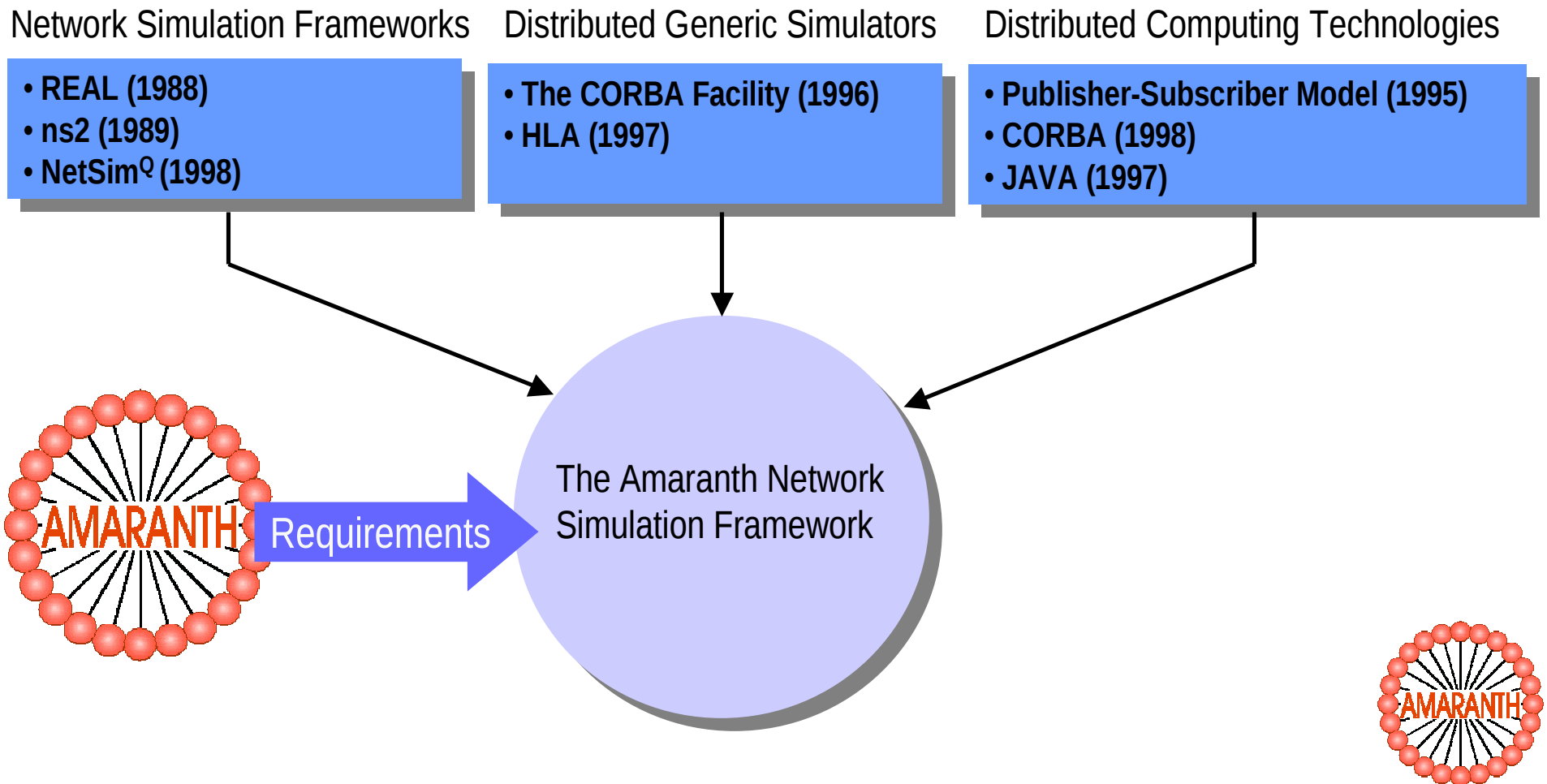
The Amaranth Project

- **Primary Focus**
 - Probabilistically Guaranteed Quality of Service (QoS) for Distributed Computing Systems.
 - Design and develop QoS “aware” policies rather than mechanisms.
- **Motivation for a Network Simulator**
 - Test and Compare performance of newly developed QoS “aware” policies.
 - Repeatability of Tests and Results.
 - Control and Visualization of Network Phenomenon affecting QoS.
- **Requirements for the Network Simulator**
 - Pre-existing collection of Distributed Computing Mechanisms.
 - Support to simulate faults and failures.
 - Relatively easy to use and availability over the WWW.

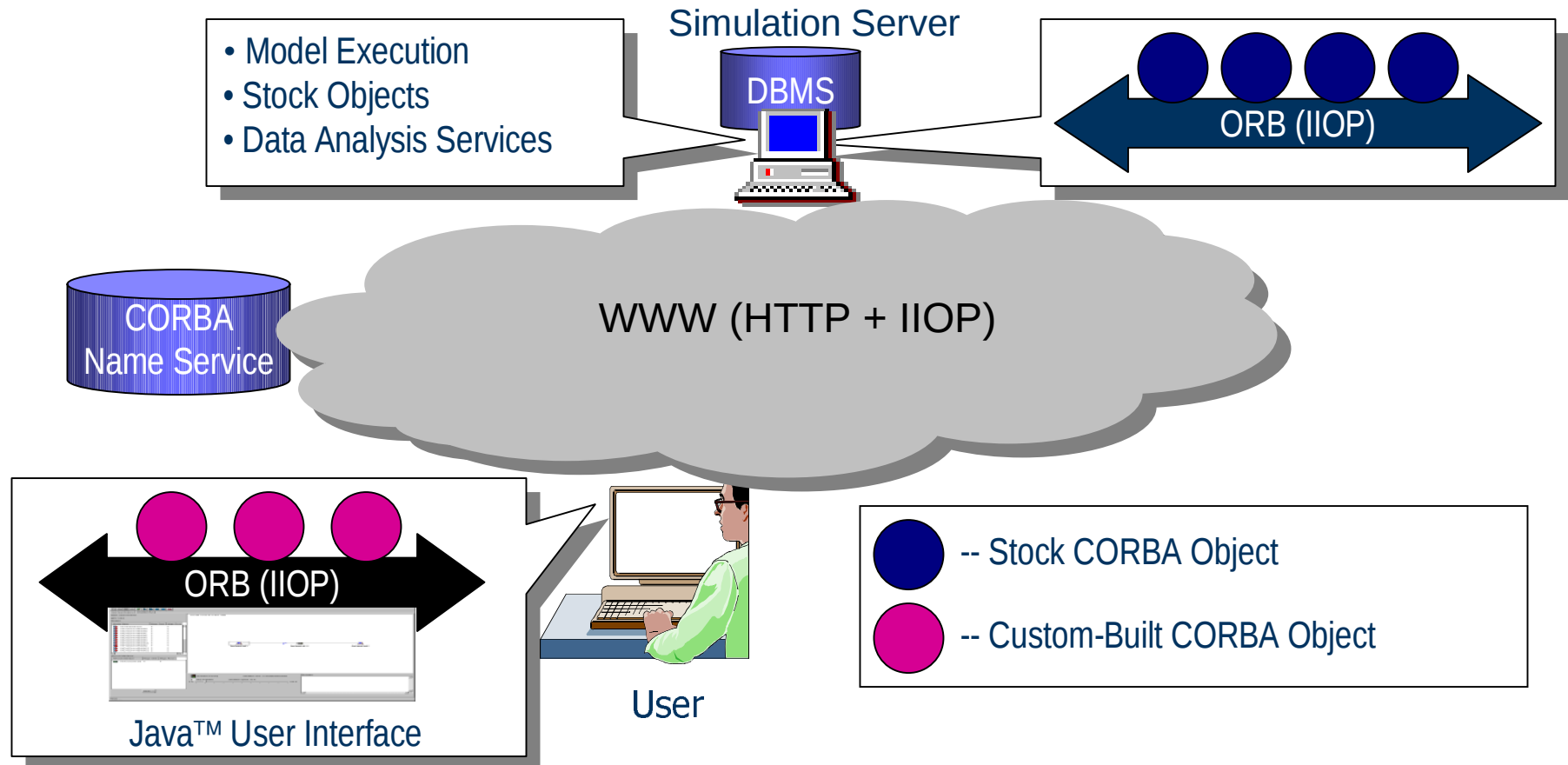


Amaranth Network Simulation Framework Heritage

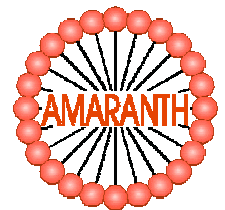
- The ANSF builds on work done in the Simulation and Distributed Computing communities to create an **Extensible, Language Independent and Platform Independent Network Simulation Framework**.



ANSF: The Big Picture

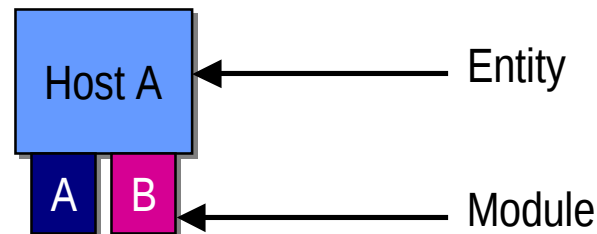


The ANSF can be Extended with Custom-Built Simulation Objects in a Language-Independent and Platform-Independent manner.

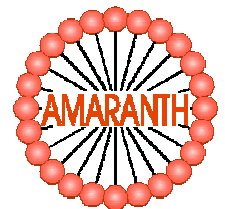


Amaranth Network Simulation Framework Internals

- **Entity** - *An object representing a hardware component such as either a Node or an Interconnecting Medium on the network. ex. A Host or a Link.*
- **Module** - *An object representing a software component which when added to an Entity changes the Entities behavioral aspects. ex. A “Routing Module” will change the behavior of a simple Host to a Router.*



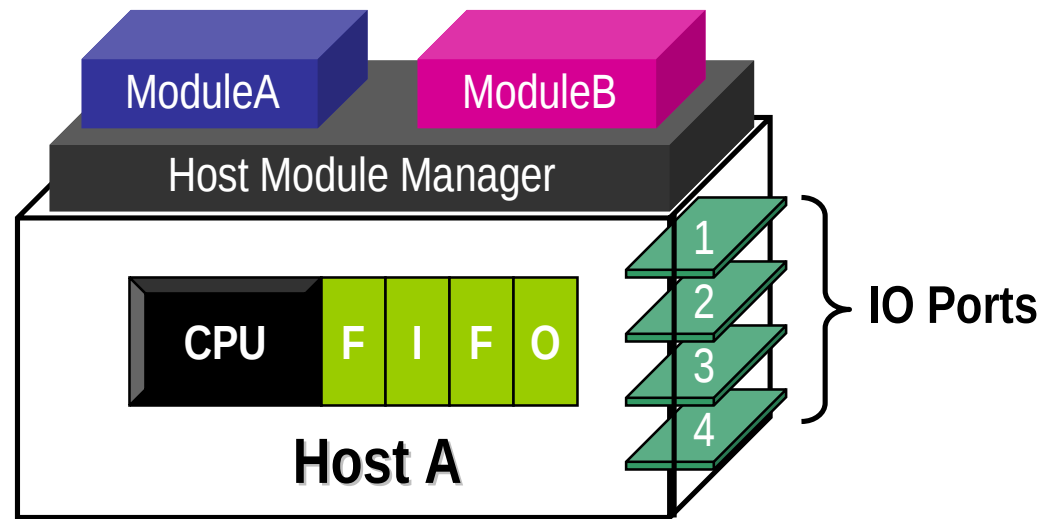
- **Key Idea - Simple Entities and Intelligent Modules.**
 - Entities can be dynamically composed.
 - Entities can be made polymorphic (change behavior over time).
 - Eliminates code changes for different behaviors in different simulation runs.
 - Enables simulation of faults and failures with relative ease.



ANSF Internals (Contd.)

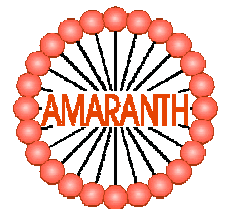
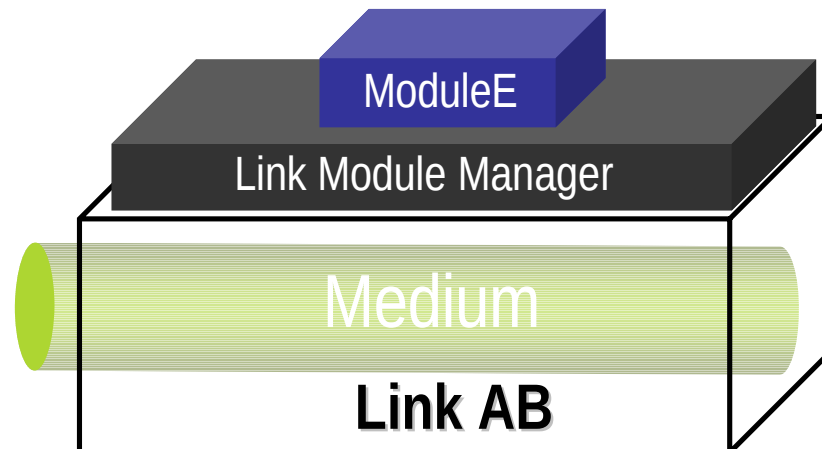
- **Constituents of a Host**

- A Module Manager
- A CPU with a FIFO Queue
- I/O Ports
- Modules



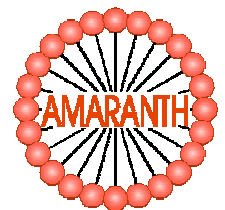
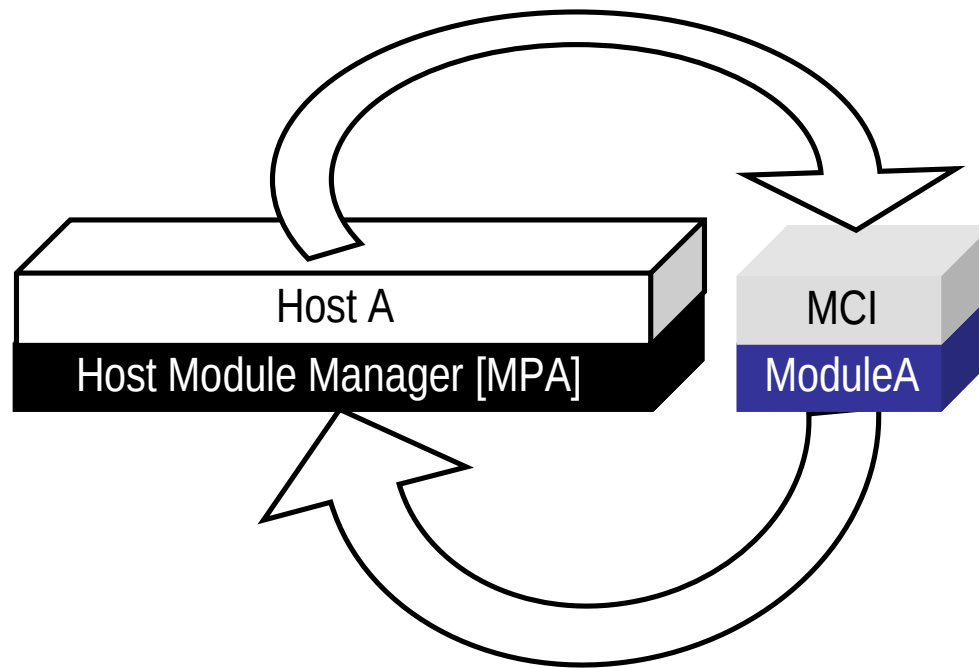
- **Constituents of a Link**

- A Module Manager
- A Medium
- Modules



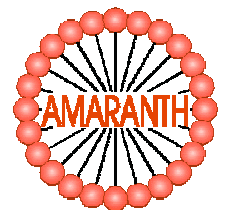
Communication in the ANSF

- **Two Standard CORBA-IDL based APIs are used**
 - The Module Plug-in API (Encapsulated by the Module Manager)
 - The Module Callback Interface (Encapsulated by the Module)
- **Module to Entity communication is achieved via the Module Plug-in API (MPA).**
- **Entity to Module communication is achieved via the Module Callback Interface (MCI).**



Inter-Module Communication

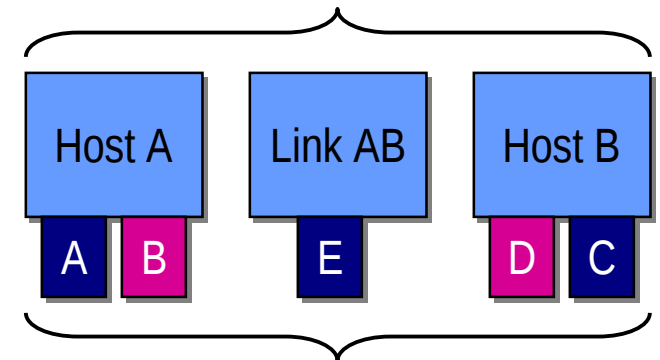
- **Publisher-Subscriber Model used for Inter-Module Communication.**
 - All registrations for publishing or subscribing are handled by the Entity.
 - De-couples communicating modules.
 - Reduces redundant messages in-case of multiple recipients.
- **Unified Messaging Format**
 - Eliminates an Entity from having to know the contents of the message
 - Facilitates relocation of modules without code changes.
- **CORBA-IDL based MPA and MCI**
 - Entities and Modules do not have to know the specifics of each other.
 - Allows Modules to communicate with Entities even if they reside on separate machines.
 - Allows Modules written in a variety of languages to communicate seamlessly.



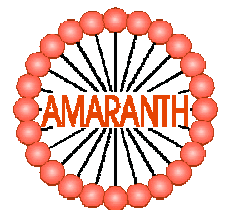
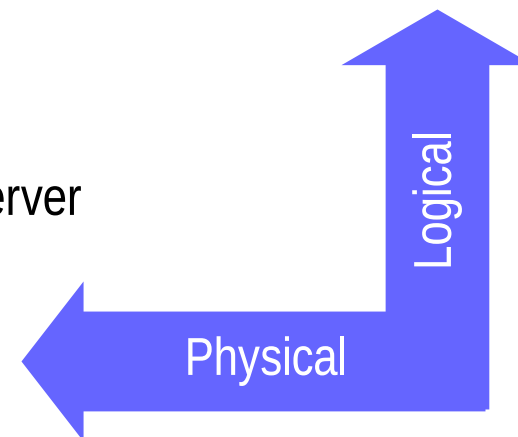
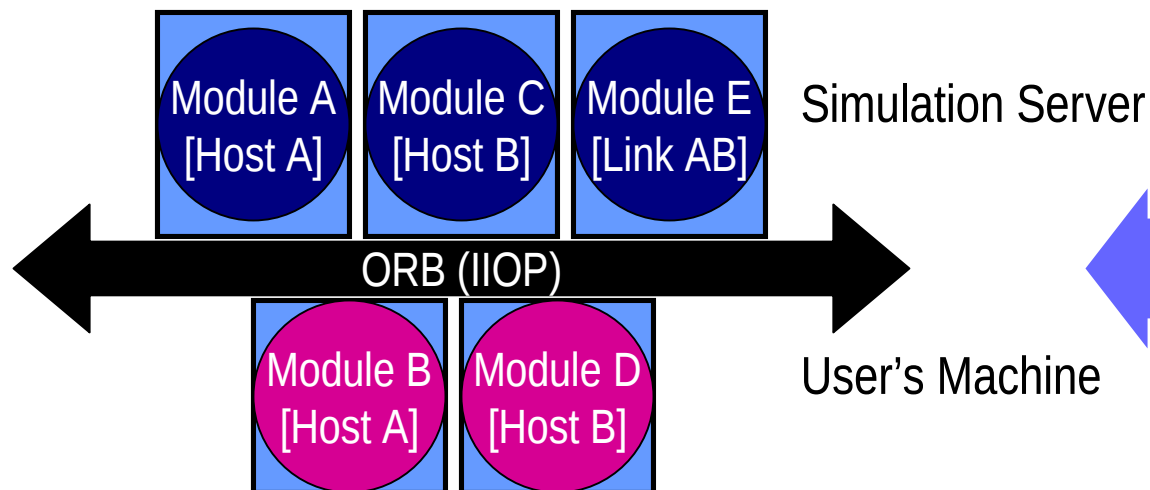
An Example Simulation Scenario

- Two Hosts connected by One Link.
- Host A
 - Module A (Stock Module)
 - Module B (Custom-Built Module)
- Host B
 - Module C (Stock Module)
 - Module D (Custom-Built Module)
- Link AB
 - Module E (Stock Module)

Simulation Entities

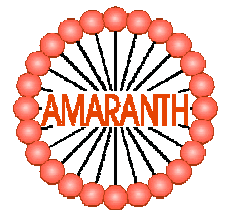


Modules



Facilities in the CORBA-IDL APIs

- **Communication**
 - Intra-Node and Inter-Node.
- **Life-cycle control of other Modules**
 - Useful to simulate faults and failures.
 - Used to provide polymorphic behaviors to Entities (behaviors that can change over time).
- **Triggers**
 - Timers and Interrupts.
- **Resource Control**
 - Control of CPU Access. ex. For CPU Scheduling.
 - Control of I/O Port Access. ex. For I/O Port Scheduling.
 - Control of Medium Access. ex. For Simulating Data Loss or Corruption.
- **Instrumentation**



Conclusions

- **Wide Deployment can be achieved without sacrificing extensibility.**
 - Leverage language and platform independent technologies.
 - Do not compromise on user comfort issues and factors.
- **The ANSF achieves wide deployability while remaining extensible and useful by -**
 - Providing powerful yet simple language and platform independent CORBA-IDL based APIs.
 - Providing users with the ability to develop their own modules in languages and on platforms of their choice and enabling these modules to be “plugged-in” to the simulation remotely from their platforms.
 - Using the “Simple” Entity and “Complex” Module paradigm and a publisher-subscriber model thereby giving users immense flexibility in composing various simulation scenarios.
 - Providing stock objects on the simulation server for rapid composition of simulations.
 - Providing platform-independent, all JAVA, simulation tools.

