# Performance Model of a Distributed Simulation Middleware

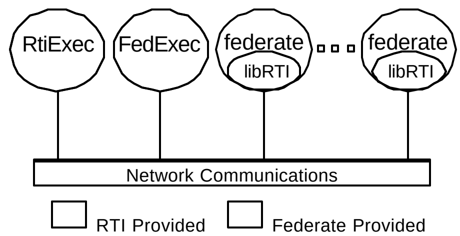
**CS672 Project Report**
**May 5, 2004**
**Roger Wuerfel**
**Yuhui Wang**

## Overview

- Introduction
- Project Goal
- System Description
- Workload Characterization
- Test Environment
- System Measurements
- Performance Model Development
- Performance Model Validation
- Summary

# Introduction

- The High Level Architecture (HLA) is a distributed simulation architecture specification
- A Run Time Infrastructure (RTI) is an implementation of this specification
- A mix of a peer to peer and a centralized distribution
- Objects are persistant and "updated"
- Interactions are events and are "sent"

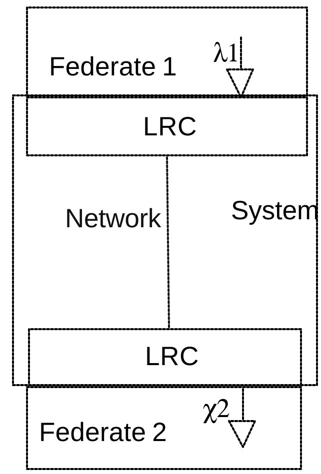



# Project Goal

- Create a model of two federates
- Validate the model

# System Description

- System is each federates the local RTI component (LRC) and the network
- The metrics are throughput and response time
- Response time is the time to send a message to the other federate




# Workload Characterization

- Sender is updating objects characterized by
  - Size
  - Frequency
- No bundling of message by the RTI
- Network sends the updates as messages
- Receiver receives the object updates

## System Measurements

- CPU usage at the sender and receiver
- Throughput of the system
- No disk I/O

## Test Environment

- Throughput benchmark program
  - Cycles of tight looping update calls (using 10,000 updates per cycle, for 10 cycles)
- RTI NG-Pro V1.0.1 implementation of the High Level Architecture v1.3
- Sender is laptop running Windows XP with a 1.2GHz processor and 1GB of RAM
- Receiver is laptop running Windows XP with a 700 MHz processor and 384 MB of RAM
- 100 Mbps switch
- Custom software to use the Windows performance counters to record CPU and disk usage at the sending and receiving sides

# Measurement Techniques

- Using Windows performance counters:
  - Total System % CPU to handle Interrupts
  - Total System % CPU for deferred procedure calls (DPC)
  - Per process % CPU
- Network service demand from procedure in “Capacity Planning for Web Performance”

# Example Data for 1024 byte Messages

| | #RAV | recv time( | Total % | %Int | %DPC | Proc % | % Priv | % User |
|---|---|---|---|---|---|---|---|---|
| | 10000 | 12848 | 97.87557 | 0.227618 | 70.10622 | 25.49317 | 6.525038 | 18.96813 |
| | 10000 | 13139 | 89.62597 | 0.352858 | 65.20819 | 23.57092 | 4.869443 | 18.70148 |
| | 10000 | 12999 | 90.57143 | 0.142857 | 66.35714 | 23.57143 | 5.071429 | 18.5 |
| | 10000 | 12679 | 93.88807 | 0.368189 | 67.89396 | 24.59499 | 5.596465 | 18.99853 |
| | 10000 | 12648 | 97.17988 | 0.838415 | 69.66463 | 25.83842 | 5.716463 | 20.12195 |
| | 10000 | 12768 | 93.38663 | 0.145349 | 68.31395 | 23.90988 | 4.941861 | 18.96802 |
| | 10000 | 12709 | 96.15094 | 0.301887 | 70.56604 | 24.5283 | 6.188679 | 18.33962 |
| | 10000 | 12699 | 99.76507 | 0.391543 | 72.59201 | 25.6852 | 5.481598 | 20.2036 |
| | 10000 | 12608 | 95.4955 | 0.15015 | 69.5946 | 24.47447 | 6.381381 | 18.09309 |
| | 10000 | 12598 | 98.67807 | 0.622084 | 71.53966 | 25.5832 | 7.387247 | 18.19596 |
| **Average** | 10000 | 12769.5 | 95.26171 | 0.354095 | 69.18364 | 24.725 | 5.81596 | 18.90904 |

# Measured Data Statistics for Primary System Measurement

| *Total Time* | |
|---|---|
| Mean | 12.8628 |
| Standard Error | 0.057279 |
| Median | 12.774 |
| Standard Deviation | 0.181133 |
| Sample Variance | 0.032809 |
| Kurtosis | 1.336247 |
| Skewness | 1.527657 |
| Range | 0.541 |
| Minimum | 12.709 |
| Maximum | 13.25 |
| Sum | 128.628 |
| Count | 10 |
| Largest(1) | 13.25 |
| Smallest(1) | 12.709 |
| Confidence Level(95.0%) | 0.129575 |
| Coefficient of Variation | 0.014082 |

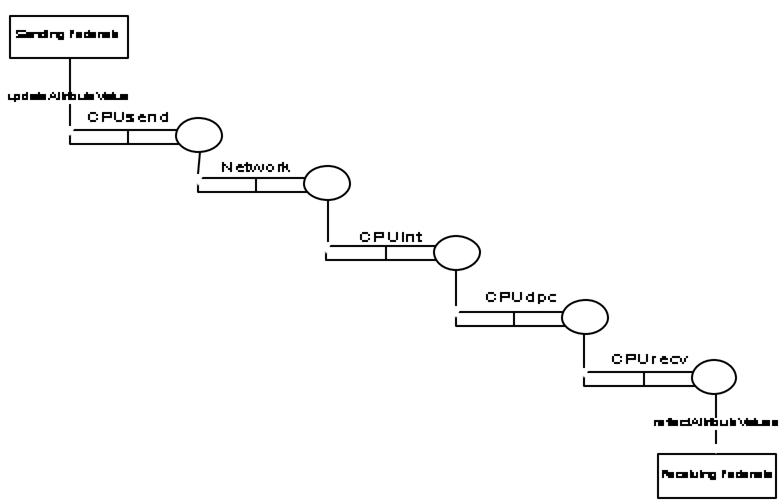
# Performance Model Development

- Open queing network – input is specified as arrival rate
- Initially had 3 queues (CPUs, Net, CPUr)
- Broke out into CPUs, Net, CPUint, CPUdpc, and CPUr
- Had to account for priorities
  - Ddpc is elongated by CPUint
  - Drecv is elongated by CPUdpc

# Performance Model Development (Inputs)

| **Description** | **% CPU** | **Sum higher** | **Elongation** |
|---|---|---|---|
| Interrupt | 0.351386 | | |
| DPC | 41.16812 | 0.35138606 | 1.003526251 |
| Application | | 41.5195099 | 1.709971988 |
| | | | |
| Throughput | 1677.692 | | |
| Dint | 0.000002 | | |
| DDPC | 0.000151 | | |
| Dcpu | 0.000182 | | |

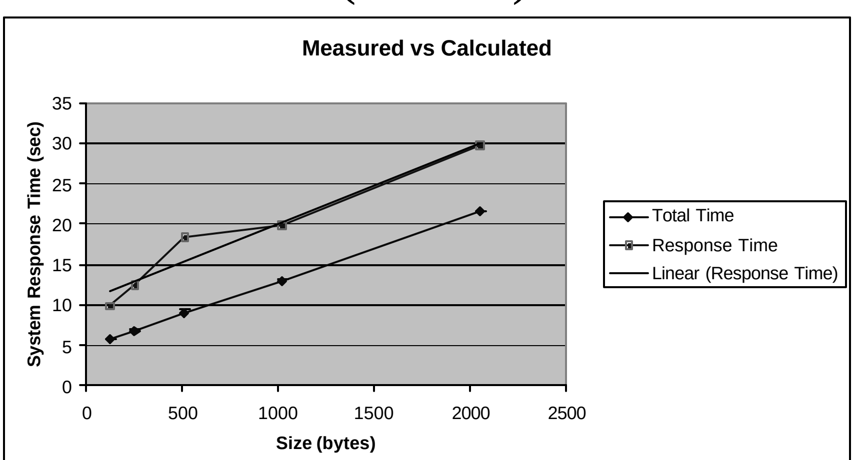
# Performance Model



# Performance Model Validation

- Use collected CPU information to calculate service demands
- Use OpenQN.xls to calculate response time for one message for each message size
  - 128, 256, 512, 1024, and 2048 bytes
- Compare with measured time to send all 10,000 messages

# Performance Model Validation (Results)

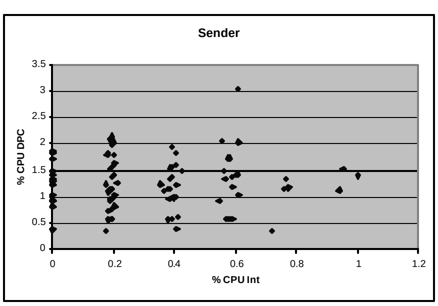


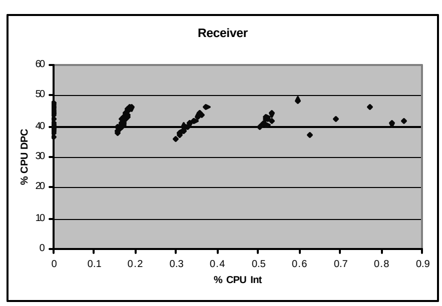
# Performance Model Validation (Comments)

- Models values are higher but follows the trend
- Potential Issues
  - Start and stop times are on different machines
  - Bundling of message in Windows
  - Relationship and interconnection of the interrupts and DPC usage of the CPU
  - Relationship between a single message and a single DPC processing is not clear

# Performance Model Validation (Problem Analysis)

- Clustering of % CPU for interrupts and % CPU for DPC suggests multiple classes



# Summary

- We able to model the performance of two federates
- Encouraged by the validation results but there are some discrepancies