Evaluation of Run Time Infrastructure (RTI) Implementations

Ms. Pamela Knight
Mr. Aaron Corder
Mr. Ron Liedel
U.S. Army SMDC
P.O. Box 1500
106 Wynn Drive
Huntsville, AL 35807-38801
pamela.knight@smdc.army.mil, aaron.corder@smdc.army.mil, LiedelR@smdc.army.mil

Ms. Jessica Giddens
Mr. Ray Drake
Ms. Carol Jenkins
Mr. Paul Agarwal
COLSA Corporation
Advanced Research Center
6726 Odyssey Drive
Huntsville, AL 35806
Tel. 256-922-1512
jgiddens@colsa.com, rdrake@colsa.com, cjenkins@colsa.com, pagarwal@colsa.com

Keywords: Runtime Infrastructure, benchmark, HWIL, latency, throughput, HLA, distributed simulation

ABSTRACT: *The utilization of Hardware-In-The-Loop (HWIL) processing in a Modeling and Simulation (M&S) environment introduces especially critical latency and throughput requirements. Data provided by HWIL simulations must be transmitted and processed expeditiously to reduce the risk of information loss. Of particular interest are issues related to performance – namely, latency and throughput – of the High Level Architecture (HLA) federates interacting by way of a Runtime Infrastructure (RTI). This paper presents the benchmark design, test approach, and initial results reported from the evaluation of four commonly used and readily available HLA-compliant RTI implementations. The evaluated RTIs include the: (1) RTI Next Generation (NG) 1.3v3, (2) MÄK Real-time RT 1.3.3-ngc, (3) Pitch portable RTI (pRTI)1.0r5, and the (4) Georgia Tech Parallel and Distributed Simulation (PADS) Federated Simulations Development Kit (FDK) 3.0 Detailed RTI.*

The benchmark activity was conducted under the Wide Bandwidth Information Infrastructure (WBII) Program. The benchmarks employed selected computing resources from the Federation Analysis Support Technology (FAST) Laboratory, a part of the Space and Missile Defense Battle Lab (SMDBL) located at the Advanced Research Center (ARC) in Huntsville, Alabama. The FAST Lab is a shared Ballistic Missile Defense Organization (BMDO)/SMDBL community asset that assists programs achieve their M&S goals by leveraging its available high performance computing (HPC) and High Level Architecture support (HLA) resources to encourage the wider utilization of these distributed simulation technologies.

The benchmark's experimental design evaluates the effect of twelve independent variables on throughput and latency. The independent variables evaluated include: RTI, number of federates, distribution of federates, Data Distribution Management, network transport mode, objects per federate, attributes per object, interactions per federate, parameters per interaction, attribute buffer size, interaction buffer size, and data bundling. Latency and throughput measures were then evaluated to determine if the four RTIs exhibited statistically significant performance differences. The benchmark results are intended to provide assistance to Modeling and Simulation personnel in HLA federation design and optimization.

This effort was performed under the Congressional Wide Band Information Infrastructure (WBII) initiative. The Space and Missile Defense Command is the executing agent of the WBII and SAIC is the Prime Contractor.

1. Introduction

The Department of Defense (DOD) High Level Architecture (HLA) has become a standard for models and simulations that interact with each other in a distributed fashion. It was mandated under DOD directive 5000.59 as the “standard technical architecture for all DOD simulations” and it has since evolved to become Institute of Electrical and Electronics Engineers (IEEE) standard 1516. The main purpose of HLA is to promote reusability and interoperability of simulations. It was developed to satisfy the requirements of simulations in a wide variety of areas including analysis, testing, training, hardware-in-the-loop (HWIL), and other engineering functions.

A HLA federation is primarily comprised of one or more federates, the federation object model and the Runtime Infrastructure (RTI). HLA is a software architecture that permits objects in one simulation to exchange data with objects in another simulation through services provided by the RTI. HLA is predicated on the concept of a federation, that is, a composable set of distributed models or simulations interacting with each other. Each member of the federation is called a federate. A federate can be any of a wide variety of entities including a computer simulation, an interface to a radar, a data collection utility, or an interface to a live player. A federate presents (publishes) data for other federates to input (subscribe). It is within the federates that all objects are represented. Interactions between federates are not conducted directly, but through the functions provided by the RTI. In addition, the RTI carries out support services required for federation management. Thus, the RTI is a distributed run-time interface for the whole federation.

The Wide Band Information Infrastructure (WBII) Interoperability Integrated Product Team (IPT) investigates issues related to simulation interoperability and systems interoperability. An important concern is the impact that data exchange overheads has on the performance of simulation systems. Another consideration is the use of real-time HWIL simulations because they have much more critical latency and throughput requirements. Data provided by HWIL simulations must be transmitted and processed expeditiously to reduce the risk of information loss.

The Federation Analysis Support Technology (FAST) Laboratory located at the Advanced Research Center (ARC) in Huntsville, Alabama is chartered to provide assistance and technical support to the missile defense modeling and simulation community concerning issues related to distributed-simulation technology using HLA. Problems related to new and legacy simulations desiring HLA compliance, as well as HLA migration analysis, development, and testing are in the direct purview of the FAST Lab. Several computer platforms are specifically dedicated as FAST Lab assets. They provide diverse cross-platform environments for the development, testing, and analysis of distributed simulations. The ARC also has available a multiplicity of additional computing platforms dedicated to National Missile Defense (NMD) and Space and Missile Defense Command (SMDC) work.

2. Purpose

The purpose of this RTI evaluation activity is to evaluate the performance of currently available RTI implementations. Of particular interest are the response variables of throughput and latency. This is because of their criticality to HWIL participants in HLA exercises.

The performance of a Federation is affected by many factors. These include the RTI capabilities, operating system, local and wide area network (LAN/WAN) environments, federate behavior and characteristics, hardware platforms, and network interface cards, among others. The RTI is crucial to all interactions between federates and its poor performance can sharply penalize overall federation performance. RTI evaluation and selection should be considered during the design phase of a federation because there are many performance, functional and compatibility tradeoffs that impact design of the overall simulation.

3. RTI General Observations

- **DMSO RTI-NG 1.3v3** - The DMSO Run-Time Infrastructure Next Generation 1.3. This is the reference implementation. It was sponsored and developed by DMSO and is available free of charge to all qualified federation developers. It corresponds to the HLA Interface Specification version 1.3 (http://www.dmso.mil). It provides a collection of common services that can be accessed through a standard programming language API. It supports C++, Java, Ada 95, and the CORBA Interface Definition Language. This RTI is HLA compliant.
- **MÄK Real-time RTI** - The MÄK Real-time RTI is developed by MÄK Technologies (http://www.mak.com/rti.htm). It is currently available free of charge to the simulation community. It can be configured to use point-to-point, broadcast, or multicast communications for flexibility across different network architectures. In addition, the company indicates the MÄK RTI minimizes CPU and memory requirements, simplifying the architecture of HLA-compliant simulations. However, this RTI does not implement all DMSO RTI services. Thus, all simulation requirements might not be met for some federates. MÄK Technologies also has developed several tools that can operate with its Real-time RTI.
- **Pitch portable RTI (pRTI)** - The Pitch portable RTI is developed by Pitch Corporation in Linköping, Sweden (http://www.pitch.se/prti). It is a platform-independent implementation of all services documented in the HLA Interface Specification version 1.3. It is implemented in Java. The pRTI runs on Windows NT4/95/98, Sun, SGI, RedHat Linux and other platforms, providing full-scale interoperability on multiple platforms including C++ bindings. This is also an HLA compliant RTI.
- **RTI-Kit Developed RTIs (FDK-DRTI)** - The RTI-Kit is a collection of libraries designed to support development of RTIs for parallel and distributed simulation systems, especially federated simulation systems running on high performance platforms. Each library is designed so it can be used separately, or together with other RTI-Kit libraries, depending on the functionality required by the user. The specific RTI tested was the Detailed RTI (DRTI), which is a sample TCP/IP implementation of the RTI based on the RTI-Kit. It is part of the Federated Simulations Development Kit developed at Georgia Tech by Prof. Richard Fujimoto (http://www.cc.gatech.edu/computing/pads/tech-highperf-rti.html). The developers claim this software contains composable modules to build RTIs from which different simulations can be integrated with each other. It is designed so that RTI developers can pick and choose from the set of FDK modules that are most appropriate for developing their particular RTI implementation. Developers can use the ready-made modules, saving the time required to develop them on their own. The RTI-Kit, however, does not implement all HLA services.

Some overall observations collected during Benchmark Testing are provided in Table 1.

RTI	HLA Compliant	Transport Mode	rtiexec required	Notes
DMSO RTI NG	Yes	Reliable and Best Effort	Yes	Most Stable of RTIs Tested
MÄK RT RTI	No	Static Reliable and Best Effort	Yes	Sensitive to Frequency of Tick; Manual Cleanup of Processes Required
Pitch pRTI	Yes	Reliable and Best Effort	Yes	Least Stable, Developed for NT Benchmarked in IRIX, Multi-Threaded RTI (Limited Evaluation Time)
FDK Detailed RTI	No	Reliable	No	Second Most Stable, Easiest to Run Once Environment Variables are Set, Objects Cannot Be Destroyed

Table 1. Comparisons of RTIs Show Advantages of Each

4. Summary of Test Runs

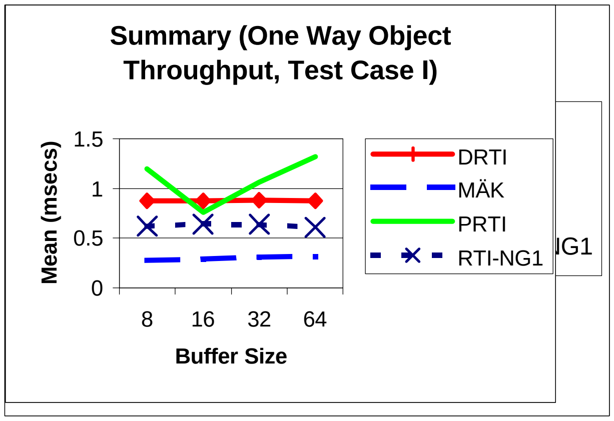
Over 9000 tests were initially identified to cover the entire experiment space. To date, over 1900 tests have been conducted. Due to RTI anomalies, benchmark test redesign and software modification was required. The Pitch RTI was modified as required to accommodate differences in WindowsNT and IRIX operating systems. Other unforeseen benchmark complications included manual intervention to terminate tests, which, in turn prevented batch test processing. These issues resulted in the benchmarking becoming increasingly time consuming. Initial tests were modified (from 10 samples of 100 updates [One-Way-Object-Throughput] to 1 sample of 1000 updates) so that more runs could be made in the time available.

The mean of throughput was evaluated for buffer sizes 8, 16, 32, 64, 128, 256, 512, and 1024. Results of these tests were compared using a student's t statistic with the DMSO benchmark results for One-Way-Object-Throughput (OWOT) with results indicating the overall means were approximately the same (with 98.4% probability). This provided anchoring for our benchmark results. The rationale being that if our results were comparable with the DMSO standard it added validity to our results. The following are graphical representations of results. Figure 1 represents initial One-Way-Object-Throughput test-runs (100 replications of 10 samples) across all four RTIs. Figure 2 shows results after implementing test modifications (1000 replications of 1 sample). Here (as in Figure 3), pRTI is not included due to the expiration of the RTI evaluation license. Figure 3 depicts One-Way-Interaction-Throughput results. As each figure shows, MÄK RTI appears to be the fastest of the four RTIs under the specific tests discussed.

Preliminary Qualitative Results
One Way Object Throughput (OWOT)

Figure 1.

Figure 2.

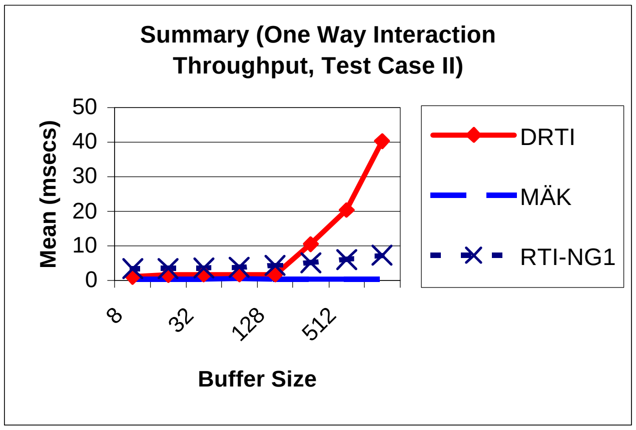


*Pitch Not Included Due To Evaluation License

Expiration

Preliminary Qualitative Results
One Way Interaction Throughput (OWIT)

Figure 3.



*Pitch Not Included Due To Evaluation License Expiration

5. Conclusions

The selection and configuration of the RTI and federates affects the performance and stability of the Federation execution. When configuring the RTIs for performance, it is important to note that settings tuned to optimize throughput often do so at the expense of latency and vice versa. So, a decision has to be made to optimize for latency, throughput or a balance between the two. Federates residing on the same system may exchange data more effectively if placed on separate systems if they are CPU bound and the network is adequate. Conversely, if the federates are network bound, the data exchange may be improved if the federates are placed on the same system. With several federates, a combination of the two approaches is usually the most effective.

Within federates, the use of the RTI “tick” function can have a dramatic effect on performance and stability. Conceptually, the tick function is one of the easiest RTI functions to use. The RTIs may use the call to perform different tasks internally, but all synchronous implementations use this function to actually deliver data that has been queued for delivery or received by the federate in the local RTI component. It can be very difficult to find a balance between optimization and stability. Ticking the RTI too often is a waste of CPU cycles and impacts performance. Ticking the RTI too little often causes instability or anomalies to occur. Furthermore, what is optimal for one RTI is not necessarily optimal or even minimally acceptable for another.

All RTI implementations provided the required level of functionality to complete many of the test cases. The three most problematic areas in dealing with multiple RTIs are the differences in configuration, startup, and effective use of the “tick” function. Automated tests consisting of multiple federation executions are more difficult to implement for RTIs that require an additional executive process. The most significant performance enhancement may be experienced with all RTI implementations by reducing the number of object types, instances, and interactions.

Additional benchmark analysis is required to complete the collection and evaluation of latency and throughput data outlined in this paper. Further results will be published for the simulation community in future conference proceedings when evaluations are complete.

-For more information on the evaluation of RTI implementations, refer to ***“Independent Throughput and Latency Benchmarking for the Evaluation of RTI Implementations”*** *published in SISO Fall 2001 Conference Proceedings – paper SISO-01F-SIW-033.*