HLA RTI Performance Evaluation

L. Malinga and Willem H. le Roux
Council for Scientific and Industrial Research[1]
Meiring Naude Road
Pretoria, 0001
South Africa
+27 12 841 2022/4867
{lmalinga | whleroux}@csir.co.za

Keywords:
HLA, RTI, Performance Evaluation, High Level Architecture, Modelling and Simulation

ABSTRACT *The performance of three Run-Time Infrastructures (RTI) is measured. Two open source implementations (Portico version 0.8 and CERTI version 3.3.0) and one commercial implementation (MÄK version 3.2) have been subjected to a specific test under ideal conditions using a dedicated Federate Object Model. This paper reports on the results of this test for each of the RTIs. We looked at the round-trip time for a federate to send an attribute update to another federate, and then to received the same update back. The attribute size was varied. This test is the first of a series of tests to be conducted on different RTIs to evaluate the performance under ideal conditions, and then later under more realistic conditions. The first test also entailed getting to know the interfaces of the different RTIs. This also contributed to the re-establishment of the use of HLA in our research unit. It is also planned to use more RTIs, from different vendors Initial results show that although the MÄK RTI implementation had a limitation in terms of the data size that can be transmitted for the best effort transport setting, it had a low round-trip time as compared to CERTI and Portico.*

1. Introduction

In this paper the performance of two open-source RTI implementations (CERTI version 3.3.0[2] and Portico version 0.8[3]) and one commercial implementation (MÄK 3.2[4]), is measured using a specific test. No specific criteria were applied in selecting the RTIs, merely access to the software. The commercial version was selected as the vendor supplies an evaluation license. It has the limitation that a maximum of only two federates can be used. The performance test was therefore designed to only use two federates.

A basic scheme is used. A simple FOM consisting of a single class containing a single attribute was defined. A variable number of bytes are published by a federate as the attribute of the object. Both Federates publish and subscribe to the same attribute. One federate sends an update and the other reflects back the update to the sender. The round-trip time is then measured over a number of updates. In this paper, Version 1.3 of the High Level Architecture (HLA) specification as adopted by the United States Department of Defense was used.

The MÄK RTI had exceptionally low latencies as compared to CERTI and Portico but it had a limitation in the size of the attribute. The maximum size of the attribute for the MÄK was limited to the maximum size of the UDP packet of the network, namely 64 KB, when using the best effort mode.

The performance analysis task of the different RTIs was undertaken for two reasons. The first is to re-establish a High Level Architecture (HLA) in our Research unit in the Council for Science and Industrial Research[1] (CSIR), and at the same time provide a mechanism for new employees to get *au fait* with HLA. Secondly, the possibility of switching to an open source RTI is considered, as well as the possibility to assist in further endeavours

1.1. Background

The High Level Architecture is a formal specification for the implementation of distributed computer simulation systems [1]. The components participating in the simulations are known as *federates*. The communication between federates is managed by a *Run-Time Infrastructure* (RTI). A collection of federates communicating through an RTI forms a *federation execution*. A *Federation Object Model* (FOM) is used to describe the data that is exchanged between federates.

[1] The CSIR is one of the leading scientific and technology research, development and implementation organisations in Africa. Constituted by an Act of Parliament in 1945 as a science council, the CSIR undertakes directed and multidisciplinary research, technological innovation as well as industrial and scientific development to improve the quality of life of the country's people. The CSIR is committed to supporting innovation in South Africa to improve national competitiveness in the global economy. Science and technology services and solutions are provided in support of various stakeholders, and opportunities are identified where new technologies can be further developed and exploited in the private and public sectors for commercial and social benefit. The CSIR's shareholder is the South African Parliament, held in proxy by the Minister of Science and Technology (www.csir.co.za).

[2] http://www.cert.fr/CERTI

[3] http://www.porticoproject.org

[4] http://www.mak.com/products/rti.php

Different RTIs realises the services of the HLA interface specification using different algorithms, techniques, architectures and implementations. The following is a list of some of the factors affecting the performance of an RTI [2].

- **Requirements levied by the HLA interface specification** (for example, reliable transport of attribute updates on a network. Due to the point-point nature of TCP, this requires that a message be sent for each participating federate on the network, hence high bandwidth consumption)
- **Choices of the RTI implementation** (different RTIs are optimised for certain simulation environments and programming languages)
- **Design of the federation** (for example, number of federates, scalability, etc)
- **Physical resources** (processing power, etc).

2. Evaluation Framework

2.1. Host Machines

The physical resources of the federation execution impose limitations to the RTI performance regardless of the implementation [2]. The experiments undertaken in this paper consisted of at least two host machines running Windows XP Professional Service Pack 2. One host was an Intel(R) Pentium(R) 4 CPU 3.20 GHz, 3.19 GHz with 2.00 GB of RAM. The other host was also an Intel(R) Pentium(R) 4 CPU 3.40 GHz, 3.4 GHz with 1.98 GB of RAM. The hosts were equipped with 100BaseT Ethernet and connected via a 1 Gb Ethernet switch. Figure 1 below shows the hardware architecture of the tests that were conducted

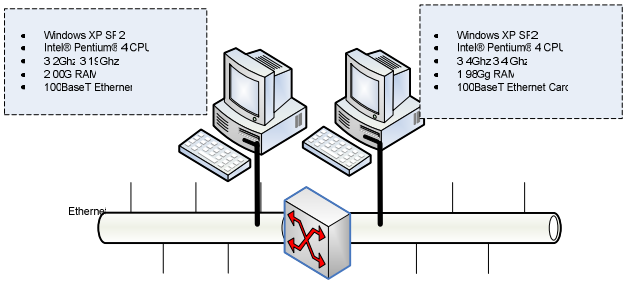

Figure 1: Hardware Architecture Set-Up

2.2. Federate Object Model

A FOM is used to describe the data that is exchanged between federates. A simple FOM consisting of a single class containing a single attribute was developed. The attribute was transported with both best effort and reliable transport mechanisms. The updates were configured to be processed on receive. Figure 2 depicts a generic form of the FOM used in the performance tests.

```
(FED
(Federation FederationName)
(FEDversion v1.3)
 (spaces)
 (objects
   (class ObjectRoot
     (attribute privilegeToDelete  best_effort receive)
     (class RTIprivate)
     (class BaseEntity
       (attribute AttributeName best_effort receive)
     )
   )
 )
 (interactions
   (class InteractionRoot  best_effort receive)
 )
)
```

Figure 2: Federate Object Model

2.3. Method

Each of the two hosts were running at least one federate. The RTI was started on only one host. The time it takes for a federate to receive an update was measured. This was achieved by one federate starting a timer and then publishing, ("sending") an attribute update. The publishing federate then waited for the reflecting federate to reflect the update. This was done to ensure that only one attribute update's round-trip gets measured. When the publishing federate received the attribute update back, it then stopped the timer. This procedure is illustrated in Figure 3. Although this procedure was repeated 100 times for an average measurement, each one was measured individually, so as to make sure that other factors do not influence the time performance. At least 1 second was allowed between round-trip attribute updates to ensure that other mechanisms do not interfere. Batching of a number of round-trip attribute updates to calculate a more accurate round-trip time will be conducted in a future experiment. The timer used is based on the high frequency performance counter accessible from C++. A time measurement accuracy better than 1 microsecond was achieved.

For each round-trip attribute update, the federation was created and destroyed before starting another measurement. This was done to ensure that there is no memory leakage and that the round-trip of each sample is independent from other test.

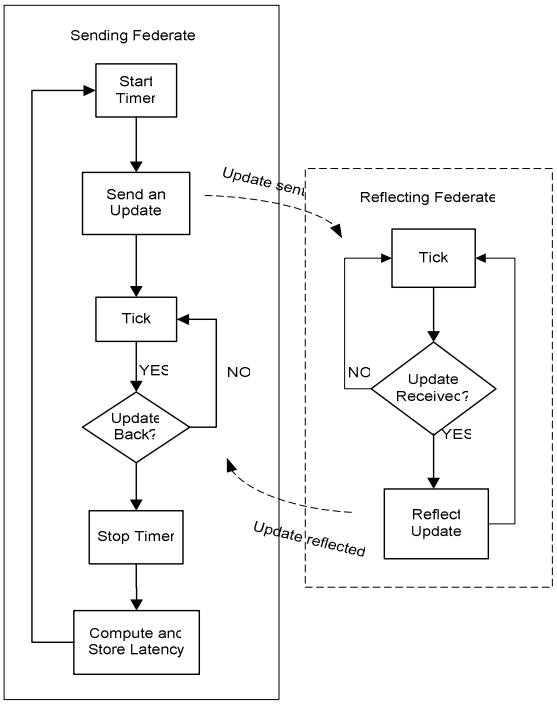

Figure 3: Publishing ("Sending") Federate and the Reflecting Federate

2.4. Selected RTIs

Two open source RTIs, CERTI 3.3.0 and PORTICO 0.8 and one commercial RTI, MÄK 3.2 were chosen for the experiments. The RTIs were only selected based on availability. Each one is discussed in the following paragraphs.

2.4.1. CERTI 3.3.0

CERTI is an open source RTI implementation for distributed discrete event simulation systems developed by ONERA licensed under General Public License (GPL). C++ is used for the used for the CERTI implementation and the development of federates. CERTI is built around an architecture of communication processes based on standard Unix libraries for process management. CERTI consist of the following components:

- A local process (RTIA)
- A global process (RTIG)
- A library (libRTI) linked with each federate

The CERTI architecture is shown in Figure 4 [3].

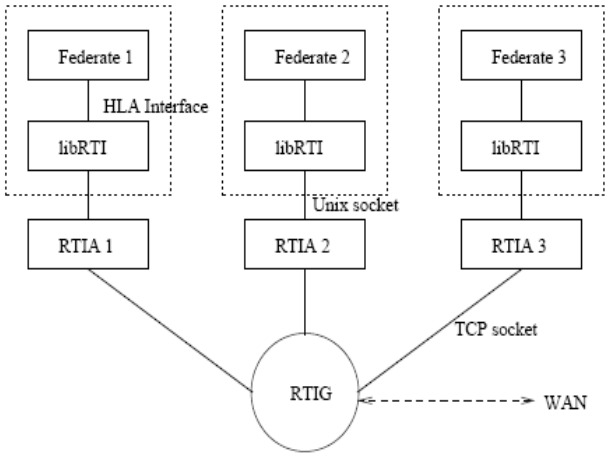

Figure 4: CERTI Architecture

A federate process communicates with the RTIA through Unix-domain sockets. The RTIA process exchange messages over the network with the RTI Gateway process, via TCP sockets or UDP in order to realise the services associated with the RTI. The allocation of CPU resources to the federate and the RTIA process is exclusively managed by the operating system. CERTI is multi-process oriented as opposed to multi-threaded.

There were no parameters to optimise CERTI 3.3.0 for performance, therefore experiments were conducted with the default configuration.

2.4.2. Portico 0. 8

Portico is a fully supported, open source, cross-platform HLA RTI implementation licensed under the terms of the Common Developer and Distribution Licence (CDDL) [4]. Portico is implemented in two languages; JAVA and C++. However the C++ version of Portico still requires the *Java Run-Time Enviroment (JRE).* C++ federates were implemented for this test. Portico consists of the following components:

- LRC (Local Runtime Component)
- Connection Binding
- RTI

The underlying communication infrastructure is isolated and hidden from the actual functionality of the RTI. The communication between the LRC and RTI is determined at run time by Portico. The Portico architecture is shown in Figure 5.

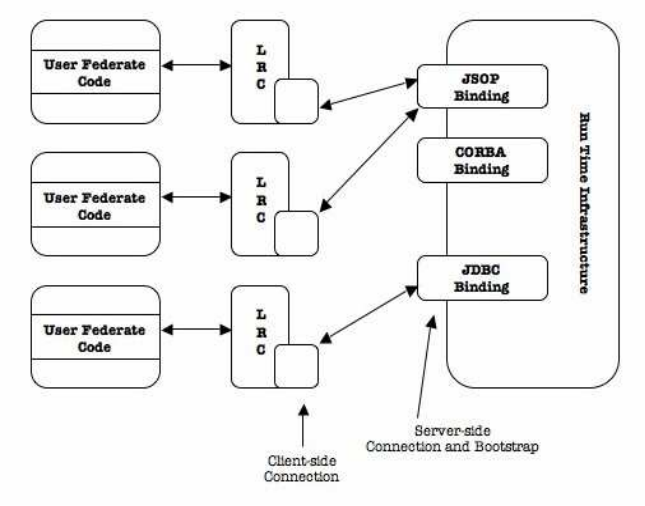

Figure 5: Portico Architecture

There were no parameters to tune to optimize Portico 0.8 as a result the experiments were conducted with the default configuration.

2.4.3. MÄK 3.2

The MÄK 3.2 RTI is a commercial software library with supporting executables that implements the HLA specifications in C++. MÄK has been verified by the Defense Modelling and Simulation Office (DMSO) as fully compliant with the HLA Interface Specification, version 1.3 and the IEEE 1516 version of the HLA Interface Specification (SISO DLC HLA API 1516: SISO-STD-004.1-2004) [5]. The MÄK RTI was designed with performance in mind [6].

The MÄK RTI was configured to allow asynchronous I/O (asynchronous reads and writes on the network) with tick() to avoid message drops [6]. This meant that an update was received by a federate when the tick() function was called.

The bundling feature in the MÄK RTI was turned off to ensure all packets are transmitted as soon as possible [6]. According to the MÄK RTI documentation, the *best_effort* transport mechanism does not support message fragmentation. Therefore, the maximum number of bytes of the update that can be transmitted before fragmentation is limited to the maximum number of bytes of a UDP packet, i.e. 64 KB. This property was set in the configuration file of MÄK RTI to allow the biggest data size for the tests that were performed.

```
(setqb RTI_asynchronousIO 1)
(setqb RTI_asynchronousCallbacks 1)
(setqb RTI_enablePacketBundling 0)
(setqb RTI_maxUdpPacketSize 70000)
```

Only two federates could be run at the same time with the MÄK RTI as the evaluation version was used.

3. Results

Figures 6 and 7 show the round-trip time in milliseconds as a function of the data sizes for the three RTI implementations for *best_effort* and *reliable* transportation respectively. For each size of the attribute, 100 updates are sent; the mean time is then calculated over the samples; i.e. over 100 samples. The mean time is for the round trip – from the time an update was published by the "sending" federate, the "receiving federate" receiving the update, and publishing the same update to back the "sending" federate, until the "sending" federate received the update.

The number of bytes per update was increased by a factor of 2. This was performed to determine how much data could be transmitted by the open-source RTIs.

In Figure 6, MÄK RTI maximum data size was 64 KB as this is the limit if *best_effort* is used. For the other RTI implementations, the data size was increased from 8 KB by a factor of 2 each time until 1024 KB.

Problems were experienced with Portico 0.8 after 63 updates when the data size was 1024 KB. According to the error, this was due to low memory availability. The value obtained for Portico 0.8 when the data size was 1024 KB, was therefore calculated over 63 samples instead of 100. CERTI and MÄK performed 100 updates of size 1024 KB without any problems.

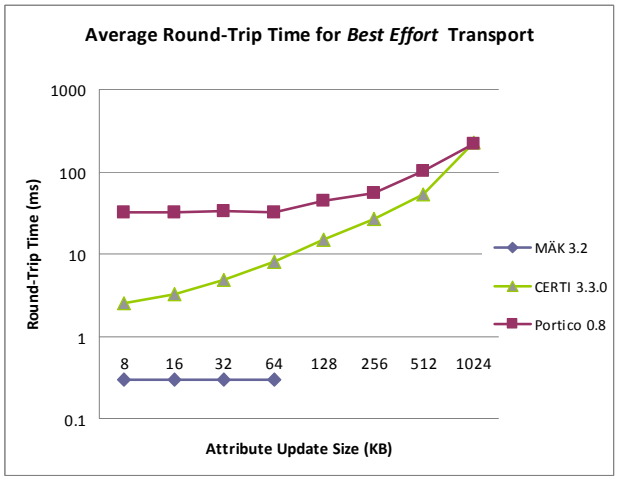

Figure 6: Round-Trip Time for *Best Effort* Transportation

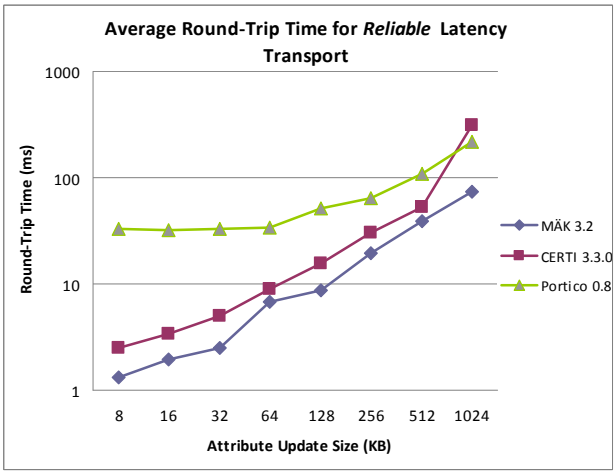

Figure 7: Round-Trip Time for *Reliable* Transportation

4. Discussions

The three RTI implementations have a similar behaviour for small data sizes; i.e. round-trip times for all three RTI implementations do not change that much for small data sizes.

Using *best_effort* transportation, the times for CERTI are lower than that for Portico for update sizes smaller than 512 KB. For update sizes, larger than 512 KB, the times for CERTI and Portico increased exponentially with CERTI the highest. MÄK could not be used for sizes above 64 KB as discussed elsewhere, but is the best performer for smaller updates, with a round-trip time of less than 1 ms.

Using *reliable* transportation, the times for MÄK are lower than the two open-source RTIs for all data points. For data sizes, smaller than 512 KB, CERTI has the lowest round-trip time. For 1024 KB, Portico has a slight advantage over CERTI.

5. Conclusion

Performance of the two open-source RTI implementations; CERTI 3.3.0 and Portico 0.8 and one commercial RTI implementation; MÄK 3.2 were measured. A basic FOM was defined consisting of a single class containing a single attribute. The round-trip time for an attribute update was measured. This was achieved by one federate sending an update and waiting in a loop for the second federate to reflect back the update (i.e. to send the update back) and recording the time it takes.

Although the MÄK 3.2 had a limitation of 64 KB in terms of the size of the attribute updates that can be transmitted using *best_effort*, it had exceptionally low latencies. For update sizes, larger than 64 KB, all three RTIs had an increase round-trip time, with MÄK at the lowest.

Future work includes more tests, namely to time a single attribute update, again with varying sizes. More federates; objects and attributes will also be used under laboratory conditions, before moving on to conditions that are more realistic. Finally, ways to validate the performance test results will be devised and applied.

6. References

[1] F. Kuhl, R. Weatherly and J. Dahman, "Creating Computer Simulation Systems: An Introduction to the High Level Architecture," Prentice-Hall, Upper Saddle River, 1999.

[2] B. Watrous, L. Granowetter and D. Wood, "*HLA Federation Performance: What Really Matters?*" In Proceedings of the 2006 Fall Simulation Interoperability Workshop, Stockholm, 2006.

[3] B. Bréholée and P. Siron, "CERTI: Evolution of the Onera RTI Prototype," In Proceedings of the Fall Simulation Interoperability Workshop, Orlando, 2002.

[4] The Portico Project Web Site, http://www.porticoproject.org, Accessed 4 February 2009.

[5] MÄK RTI 3.1.2 Release Notes, MÄK Technologies, 2007.

[6] B. Watrous and L. Granowetter and D. Wood, "The MÄK High-Performance RTI: Performance by Design," MÄK Technologies.

Author Biographies

LINDA MALINGA has been with the South African Council for Science and Industrial Research since September 2008 and is at present a Researcher in the Mathematical and Computational Modelling Research Group. Linda holds a Bachelor of Science degree in Electrical and Information Engineering from the University of the Witwatersrand.

HERMAN LE ROUX has been with the South African Council for Scientific and Industrial Research since April 1998 and is at present a Principal Engineer in the Mathematical and Computational Modelling Research Group. He is involved in Modelling and Simulation-based Acquisition Decision Support, specifically for the South African National Defence Force. Interests include enterprise information systems, information fusion, biometrics, artificial intelligence and software engineering. Le Roux completed a Masters Degree in Computer Engineering at the University of Pretoria in 1999 and is currently pursuing a PhD in Command and Control Modelling.