

WHITEPAPER

Modeling RTI Connex[®] DDS Performance on Your Platforms

BEST PRACTICES FOR BENCHMARKING AND DESIGNING PERFORMANCE TESTS

Are you or your team planning to benchmark the performance of your system? Then there is an important thing to consider: How do you ensure that the tests are realistic?

DEFINING REAL-WORLD PERFORMANCE

Performance does not just mean “fast.” Performance requirements tie your system to the real world by imposing time-related constraints. For example, latency requirements may look like “*shall do X functionality in N milliseconds,*” whereas throughput requirements may look like “*shall handle M thousand Ys per second.*” Combining these gives loaded latency requirements, such as “*shall do X in N milliseconds while the system is handling M thousand Ys per second.*” Often, the variation of these measures is also of interest – a good example of this is the jitter on latency. Together, these measures allow us to drive toward a predictable system with deterministic behavior.

MODELING PERFORMANCE

Why do we call it “modeling performance” and not “measuring performance?” Because, unless you are able to run tests on the actual applications running on the actual hardware for a real system, the best that you can do is to create a model of your system and run tests on the model. The model should mimic the operational requirements of the real system as closely as possible. With respect to communications, this means using the same data streams and data sizes (if not the actual data types), as well as the data rates that are expected in a real system. Often, there are many different operational modes or use cases that a system must support, so different models will have to be developed to test the use cases independently. Measuring performance is usually not running a single test, but running multiple tests against different scenarios.

GETTING STARTED WITH PERFORMANCE TESTING? WATCH OUT FOR THESE COMMON MISTAKES

As part of their decision-making process, 90% of customers who evaluate our products list performance as an important criterion for selecting a middleware solution. However, what customers initially may consider trivial to benchmark often ends up being more complex and non-trivial than expected. Benchmarking can become even more difficult when testing different technologies in a head-to-head comparison.

There are organizations that do this full time and are paid to conduct fair, unbiased, benchmarks of middleware. One example is STAC, <https://www.stacresearch.com/>.

Surprisingly, testing “best” performance is not as useful as you might think. The highest throughput or the lowest latency is usually only achieved by running a system until a limit is reached, often consuming 100% of a CPU or using up 99% of the network bandwidth. At that performance level, there are no resources left over to execute the user’s own processing logic or share the network with other protocols in a real system. So realistically, a system that actually does something useful would never be able to run at the maximum measured performance of any middleware.

Another thing to consider is that 99% of the use cases for applying the Data Distribution Service™ (DDS) standard don’t actually require DDS to perform anywhere near the limits of which DDS is capable. However, 100% of the customers who use DDS do require DDS performance to meet or exceed the performance requirements of the system that they need to build.

In looking at testing code from a variety of use cases, we often see the following four issues:

1. The test is not measuring what they think it is measuring

- The test logic doesn’t reflect how the API would actually be used in real applications.
- Latency measurements erroneously include the execution of the logic of the test framework itself, and not just the time spent by the middleware.
- Throughput measurements do not account for the overhead introduced by the test itself.

2. Timestamps from clocks on different computers are being compared

- Unless system clocks are synchronized to within a microsecond or better, it’s impossible to obtain accurate measurements of time values that are on the order of microseconds.
- End-to-end latency across hosts is incredibly difficult to measure accurately without specialized hardware.

3. The middleware itself has not been configured to run optimally

- Often, users are testing a technology that is new to them. They may not be aware of how to configure the technology for best performance. And the configuration may need to change depending on the test scenario; 1-1, 1-to-many, data sizes, lowest latency versus maximum throughput, etc.
- Most middlewares are not tuned for the best performance of any type out-of-the-box. Just as a Tesla doesn't start up in Ludicrous mode.

4. Using different testing frameworks to test different middleware

- Even though many tests claim to measure throughput and latency, it does not mean that the results from different tests can be compared directly. Often, we see that the test frameworks themselves affect the results. Comparing results from different tests can lead to wrong conclusions.

3. Gain a thorough understanding of the middleware being tested

Configuration parameters can vary between different middlewares, so it is important to understand how best to configure the middleware you choose for best throughput or best latency. As stated previously, middlewares may not be configured for optimal performance by default. Also, different configurations may be required to maximize different performance metrics such as throughput versus lowest latency. Typically, test applications can be configured to use multiple network interfaces, and default settings may use transports such as using shared memory or intra-process transports. If the intent is to test performance over certain network interfaces, first ensure that the test application is configured to exercise the intended network interface(s).

4. Measure averages and variability

The average measurement for any metric is informative, but only considering this measurement alone can be misleading. Taking into account the standard deviation of maximum/minimum measure values or 90th, 99th percentile values will provide a better understanding of how a middleware behaves.

BEST PRACTICES FOR DESIGNING PERFORMANCE TESTS

Before you jump into testing, here are four best practices to remember when designing performance tests

1. Create a performance test scenario that represents the aspects of the application in operation

Waiting until the system is operational is too late to discover any performance problems. Pre-defining performance goals such as maximum response times and acceptable performance metrics help to better interpret the results of tests.

2. Create realistic tests

The test setup must reflect the actual payload sizes that are representative of the application architecture, and run on the same or comparable hardware platforms and operating systems. The configuration of the Network Interface Card (NIC) and the network switch, as well as the maximum network throughput and the CPU, all have an impact on the final performance results. So, to make a relevant comparison, you must: a) include the impact of the network device configuration in the performance model; and b) run the benchmarks on the same hardware you use to create a performance model.

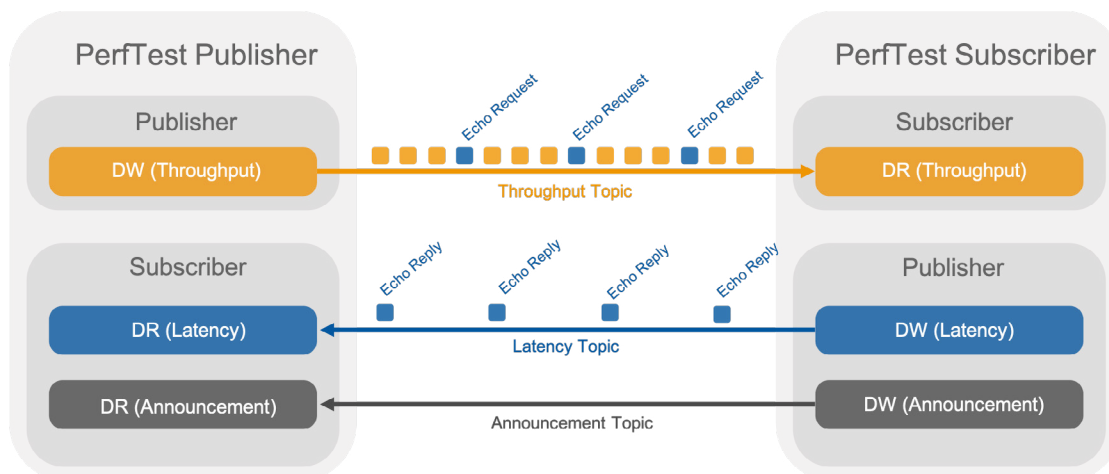
RTI PERFTEST

Official benchmarking tool for RTI Connext

RTI PerfTest is used to benchmark the performance of RTI Connext. The results for the latest versions of the products can be found here: www.rti.com/products/performance.

With PerfTest, customers can capture definitive performance measurements. Though PerfTest is not a full-blown performance testing environment, it's an ideal way to successfully test specific use cases of a system with extreme accuracy. Extrapolating results to the final system performance has to be done carefully. An important advantage of using PerfTest is that it provides a reliable and time-saving alternative to developing your own performance model, as it is highly configurable and already has telemetry capability.

Performance test scenarios are most useful when they are showing comparative results. When comparing results between Connext and other DDS implementations or other middleware technologies, great care has to be taken to ensure that the comparison is "apples to apples." This means that the assumptions made in the models are as similar as possible.



RTI PerfTest supports the swapping in of other communications solutions, enabling the framework for the performance tests to remain the same.

GETTING STARTED WITH PERFTEST

RTI PerfTest bundle is provided in 3 different ways:

1. Clone and compile from the official GitHub repository:

```
git clone https://github.com/rticomunity/rtiperftest.git
```

2. Download and compile from: github.com/rticomunity/rtiperftest.
3. Download the executables for popular platforms from: github.com/rticomunity/rtiperftest/releases in the “Binaries” section of the RTI PerfTest release page.

To learn more about the supported platforms and tutorials, please refer to the [Documentation](#).

NEED HELP WITH PERFORMANCE ISSUES OR TESTING?

Do you want to accelerate your performance benchmarking? To get assistance with performance testing and/or other concerns, contact RTI Support or RTI Professional Services

[RTI Professional Services](#)

[RTI Support](#)

LINKS TO RESOURCES

RTI provides a number of resources to assist customers in dealing with performance considerations.

Benchmarks

RTI regularly benchmarks RTI Connexx® Professional to check for potential degradation in performance. These benchmarks are published to give customers a first data point in the performance debate. Customers are encouraged to use PerfTest to repeat these benchmarks in their own environment.

[Pre-run benchmark numbers](#)

[Running your own benchmarks](#)

DDS Performance FAQ

[Discovery performance](#)

[OS specific configuration](#)

DDS Performance Basics

The basics of performance measurements are described more fully in the following knowledge base article on the RTI Community Portal:

[The basics of getting performance with your DDS applications](#)

Performance Tools

RTI provides tools to assist customers designing for performance and understanding performance issues. PerfTest is a DDS application built upon the Connexx libraries. By contrast, Monitor is a library that can be used to augment these Connexx libraries to make some of the internal behavior visible. It can be added to a completed system under test and will provide throughput numbers.

[Monitor](#)

[PerfTest](#)

ABOUT RTI

Real-Time Innovations (RTI) is the largest software framework company for autonomous systems. RTI Connexx® is the world's leading architecture for developing intelligent distributed systems. Uniquely, Connexx shares data directly, connecting AI algorithms to real-time networks of devices to build autonomous systems.

RTI is the best in the world at ensuring our customers' success in deploying production systems. With over 1,700 designs, RTI software runs over 250 autonomous vehicle programs, controls the largest power plants in North America, coordinates combat management on U.S. Navy ships, drives a new generation of medical robotics, enables flying cars, and provides 24/7 intelligence for hospital and emergency medicine. RTI runs a smarter world.

RTI is the leading vendor of products compliant with the Object Management Group® (OMG®) Data Distribution Service™ (DDS) standard. RTI is privately held and headquartered in Sunnyvale, California with regional offices in Colorado, Spain and Singapore.

Download a free 30-day trial of the latest, fully-functional Connexx software today: <https://www.rti.com/downloads>.

RTI, Real-Time Innovations and the phrase “Your systems. Working as one,” are registered trademarks or trademarks of Real-Time Innovations, Inc. All other trademarks used in this document are the property of their respective owners. ©2021 RTI. All rights reserved. 50050 VO 0521