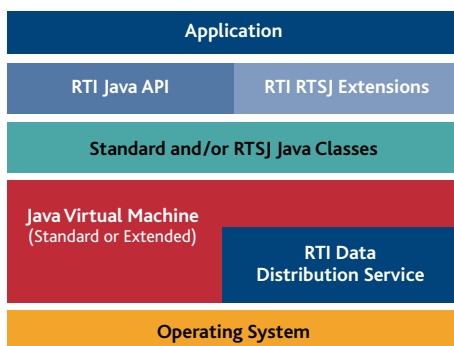


RTI RTSJ Extension Kit for RTI Data Distribution Service

True Real-Time Messaging for Java Applications

BENEFITS

- Provides end-to-end solution for real-time Java programmers needing scalable, high performance messaging solution
- Provides up to 25x better performance than JMS
- Brings the combined advantages of the Java language and DDS to real-time distributed systems.
- Works with real-time JVMs from multiple vendors



RTI now enables developers to write true real-time distributed applications in Java.

RTI RTSJ Extension Kit for RTI Data Distribution Service integrates real-time Java and high performance messaging. This combination provides an end-to-end solution and a much higher-performance alternative to Java Messaging Service (JMS) for Java developers building real-time applications.

RTI Data Distribution Service

RTI Data Distribution Service is high performance messaging middleware for the development and integration of applications that require low latency, high throughput, high scalability, deterministic responses and minimal consumption of network, processor and memory resources. Typical applications include:

- Mission-critical combat systems
- Financial transaction processing
- Air traffic control
- Roadway traffic monitoring
- Railways
- Industrial automation

RTI Data Distribution Service is an open-architecture platform that complies with the Object Management Group's (OMG's) Data Distribution Service (DDS) for Real-Time Systems standard.

Java and Real-Time

Java is the ideal language in which to write large complex distributed systems. It is a mature development and execution environment that is routinely used in both large and small applications and offers a more portable and safer alternative to C, C++ or Ada.

Now, with Java Virtual Machines (JVM) that implement the Real-Time Specification for Java (RTSJ), developers have a viable alternative to lower level languages for developing real time applications.

DDS implements a peer-to-peer model that does not need an intermediate broker, eliminating single points of failure, while providing up to 25x better performance than JMS.

What is RTI RTSJ Extension Kit?

Java Threads Support

The RTI RTSJ Extension Kit for RTI Data Distribution Service is a class library that allows developers to use real-time Java threads within RTI Data Distribution Service applications, providing significant reductions in jitter associated with Java garbage collection.

New DDS QoS Parameters

With this extension kit, real-time Java developers can create real-time threads, control their properties, and use those threads within RTI Data Distribution Service. Two types of real-time Java threads can be created:

- Real-time threads provide scheduling attributes better suited to real-time applications with support for asynchronous events and tools for minimizing or eliminating garbage collection
- No-heap real-time threads are not allowed to allocate memory from the heap, so they can interrupt any background garbage collection. No-heap real-time threads are useful for hard real-time applications that require minimum scheduling latency and deterministic behavior.

RTI Data Distribution Service can be used with both real-time and no-heap real-time threads.



About RTSJ

RTSJ or the Real-Time Specification for Java comprises Java class libraries that increase the determinism and predictability of Java for use in hard or soft real-time systems.

RTSJ introduces no syntax changes to the Java language; all legal Java programs are also legal RTSJ programs. Of course, without using the new RTSJ classes, one cannot get real-time behavior from ordinary Java classes.

Java has been enhanced in seven different areas: scheduling, memory management, synchronization, asynchronous event handling, asynchronous transfer of control, asynchronous thread termination, and access to physical memory. It is these new semantics that give Java its real-time capabilities.

Scheduling

RTSJ defines a priority-based, preemptive, base scheduler very similar to the usual schedulers in many RTOSs. Developers can specify precisely the characteristics of schedulable objects by changing a number of scheduling-related parameters.

Memory Management

Garbage collection has always played havoc with the determinism of real-time systems. RTSJ defines memory areas that are outside the purview of the garbage collector, thereby avoiding the unpredictable pauses usually introduced by a garbage collector.

RTSJ defines four kinds of memory areas:

- The usual heap memory that is garbage collected
- Scoped memory that contains objects whose lifetime is defined by scope
- Physical memory that is tied to specific physical memory regions and allows byte-level access
- Immortal memory that contains objects that can always be referenced

These four kinds of memory provide the flexibility and determinism needed by real-time Java applications.

Synchronization

Many resources in real-time systems only allow serial access by processes or threads. So, accesses to those resources may be synchronized. Unfortunately, naïve implementations suffer from priority inversion. RTSJ avoids priority inversion by requiring synchronized resources to implement an algorithm that prevents priority inversion. And in the case where it is not possible to avoid priority inversion, RTSJ implements a set of wait-free queue classes that can be used.

Asynchronous Event Handling

RTSJ generalizes Java's asynchronous events and how they are handled. Asynchronous event handlers are much like threads, and, like threads, are scheduled. Timers are an example of asynchronous event handlers. RTSJ was designed to deal efficiently with tens of thousands of asynchronous event handlers.

Asynchronous Transfer of Control

RTSJ supports the asynchronous transfer of control of another thread from one location to another when, say, the real world changes drastically, or when a timer expires. Asynchronous transfer of control is an extension of Java's exception mechanism.

Asynchronous Real-Time Thread Termination

RTSJ replaces Java's unsafe and deprecated mechanism for stopping threads with one based on asynchronous event handling and asynchronous transfer of control. This mechanism provides a fast but orderly cleanup and termination of a thread.

Physical Memory Access

Device drivers, memory mapped I/O, flash memory, battery-backed RAM and ROM all require access to a specific area of physical memory. RTSJ defines classes and methods that can be used to read or write bytes or objects tied to physical memory. (Those objects can't reference other Java objects

because that would allow Java's type system to be bypassed.) Only `get ()` or `set ()` operations on `byte`, `short`, `int`, or `long` data are allowed.

Other Enhancements

RTSJ enhances other areas in Java that, although they are not essential for providing predictability or determinism, are nonetheless useful in real-time systems. They are:

- The notion of time in Java has been extended to provide up to nanosecond precision (although accuracy might not be that high). Several new clocks or timers are provided that make use of the increased resolution of time.
- POSIX signals can now be bound to asynchronous events.
- New classes contain operations and semantics that affect the entire system.

About RTI

Real-Time Innovations (RTI) works in partnership with its customers to develop and integrate the world's most demanding real-time applications. RTI takes the risk out of distributed application development and system integration by providing deep expertise in real-time communications coupled with the highest performance messaging middleware. The company's software and services have been leveraged in a broad range of industries including defense, intelligence, simulation, industrial control, transportation, finance, medical and communications. Founded in 1991, RTI is privately held and headquartered in Sunnyvale, California. For more information, please visit www.rti.com.