Microsoft Robotics Studio

Tyco Security Products Ensures Real-Time Alarm Delivery Using Microsoft Robotics Studio

Tyco Security Products provides world-class security and access-control systems to customers in more than 100 countries, helping to protect 300 airports, 80 of the world's top 100 retailers and more than 5 million other facilities. As Tyco's Software House® brand, which offers security-critical control solutions, prepared to release its new C•CURE® 9000 security and event management application, its developers needed a better way to manage the real-time concurrency and coordination challenges inherent in a system that at times handles hundreds or more notifications per second. When Software House developers heard about Microsoft® Robotics Studio's Concurrency and Coordination Runtime (CCR) library, they inserted the code into their solution. CCR worked seamlessly, and the developers found immediate benefits, including 100 percent faster thread processing and linear scalability.

CASE SUMMARY

Location:
United States

System:
Software House:
C•CURE 9000

Situation

Around the globe, when it comes to deploying world-class security and access-control systems, the answer is often found with Tyco Security Products (a division of Tyco International Ltd.), which employs 90,000 people and generates annual revenue of more than US$11 billion. Tyco Security Products helps protect 300 international airports, 80 of the world's top 100 retailers and more than 5 million businesses, public buildings, power plants, roadways, ships, hospitals and homes
in over 100 countries—from the fire system in Berlin's Parliament building to the security system for the World Bank headquarters in Washington, D.C., and the fire detection and suppression system for Australia's Sydney Harbor Tunnel.

Tyco Security Products's Software House® brand of security solutions is used in some of the world's most sensitive and secure facilities, including nuclear power plants, major government buildings, and large corporations. Software House's C•CURE® 800/8000 product line provides an access-control software platform that is highly regarded for its reliability and performance. More than 6,000 customers and nearly half of all Fortune 500 companies use Software House solutions to secure, monitor and control access to facilities.

As Tyco Security Products prepared to release a major upgrade, C•CURE 9000, it adopted the Microsoft® Application Platform, taking advantage of the latest technology advancements from Microsoft. C•CURE 9000 was developed using the Microsoft Visual Studio® 2005 development system. The C•CURE server at the heart of the system is deployed using the Windows Server® 2003 operating system and Microsoft SQL Server® 2005 database software. In addition, Tyco Security Products used the Microsoft .NET Framework 3.0 to offer customers a flexible, best-of-breed security and event management solution that performs well over a wide area network. Using the Microsoft Application Platform helps assure Software House customers that they will be using the most current, secure, Internet-based communications protocols.

Software House developers take pride in C•CURE's ability to provide real-time responses to monitored events. This task can be demanding in large deployments, in which the C•CURE server might receive hundreds or more notifications per second that must be distributed to more than 100 C•CURE client monitoring stations. Many of these notifications are internal events used by the C•CURE system to maintain the state of all the system's components. Such information is still important and is logged for corporate records and auditing needs. However, the highest-priority notifications are system events that security personnel need to know about—ranging from someone swiping an ID card across an electronic card reader to unlock a door, to significant alarms, such as when a door is forced open or otherwise accessed without authorization. Software House developers needed a way to efficiently prioritize such security-critical events.

"To get the real-time response times we wanted, we needed a technology that would resolve the concurrency issues that a monitoring system inherently faces," says Stephen Tarmey, Architect, Software House. "And we needed the ability to prioritize messages out to the C•CURE clients being used by security personnel at the monitoring stations."

Solution

Software House developers found the solution in Microsoft Robotics Studio's Concurrency and Coordination Runtime (CCR) library. Microsoft Robotics Studio was initially developed as a software platform that the robotics community could use to develop an array of applications across a wide range of hardware. CCR makes it easier to write programs that handle asynchronous input from multiple robotics sensors and output to motors and actuators.

But as Software House developers discovered, the CCR technology has value far beyond robotics. In fact, Tarmey says he heard about the CCR technology almost by accident. "We went to the 2007 Microsoft ReMIX developers conference in Boston," he says. "At the time, I hadn't heard about Microsoft's robotics work and frankly wouldn't have cared because we don't work with robots. But one of the lunchtime speakers was a professor from MIT who was trying to solve the asynchronous I/O problems created when a huge amount of data is sent to a repository for processing."

That caught Tarmey's attention. "I thought: 'He's talking about our problem!'" Tarmey recalls. "When the professor spoke about solving the problem using the Concurrency and Coordination Runtime library from Microsoft Robotics Studio, I made a note and immediately followed up. When you find someone who has already solved a problem that you are still facing, you tend to pay really close attention."

Tarmey faced a quandary, though. His team was already nearing completion of C•CURE 9000, having used a thread pool solution for dealing with concurrency and coordination. He knew CCR would provide a better solution, but he was concerned how much time would it take to integrate the new code.

The quandary didn't last long. "I just dropped the CCR into our code and it worked," he says. "It integrated so simply that it really facilitated my unit testing."

Architectural Notes

C•CURE 9000 was created using the Microsoft Application Platform and has a multi-tier architecture that includes:

- **Hardware devices.** Software House provides a number of smart card readers, including readers that support multiple technologies and protocols. These devices, mounted on or near doors, read cards that are swiped through, or brought into close proximity to, the reader. The reader transmits the data to a controller for user authentication and logging.

- **Controllers.** The Software House iSTAR eX controller is connected to card readers and uses an internally-stored database to determine an individual's access privileges. When a card is swiped through a reader, the data goes to the controller. If access is granted, the controller sends an "open" command to the door and access is granted. If access is not granted, the door remains closed and locked. Large deployments can include multiple controllers.

- **C•CURE server.** Each controller sends notification messages to the central C•CURE server to apply specific business logic to system activity, process data persistence and handle server-to-client notifications. All events are stored on the C•CURE server using SQL

Server 2005 running on the Windows Server 2003 operating system. System administrators use the C•CURE server to add and remove access permissions and control access rights for C•CURE clients.

- **C•CURE clients.** C•CURE client software, hosted on personal computers running Microsoft Windows Vista® or earlier Windows® operating systems, provides real-time information, including alarms, to personnel at control stations. A large organization may have more than 100 C•CURE clients receiving information from the C•CURE server.

Security Validation

Software House developers used the .NET Framework 3.0 to build encryption, authentication and other key technologies into its product suite.

Before its release, C•CURE 9000 attained the U.S. Government's Federal Information Processing Standards Publications (FIPS) 197 validation. FIPS 197 validation ensures compliance with the government's Advanced Encryption Standard encryption algorithm. It is the preferred cryptographic method used by U.S. government agencies due to its high-level cryptographic key strength. FIPS 197 provides third-party assurance of security claims on any product containing cryptography that a government agency might purchase.

Benefits

Software House developers gained significant advantages by incorporating the CCR library from Microsoft Robotics Studio into its C•CURE 9000 application. Software House has found that CCR provides faster queue processing and can provide its customers with linear scalability. Its developers benefited from the ease of CCR integration with their existing application code as well as from Microsoft research R&D breakthroughs.

Faster Queue Processing with CCR

Prior to using Microsoft Robotics Studio's CCR technology, Software House developers had created a solution using a thread pool of 75 threads. The initial solution was effective, but in a high-demand scenario it couldn't ensure the real-time message delivery that the developers wanted. Fortunately, CCR provided the real-time performance they sought.

"The speed of the CCR in processing queue data is far superior to what we had with our thread pool," says Tarmey. "We were amazed. CCR just does more, and it does it faster. CCR was immediately 100 percent faster, and it did this with just two message threads instead of the 75 threads we had been using. And reducing the number of threads makes for cleaner processing."

Using CCR, one thread is for normal messages and the second thread is reserved for high-priority messages such as door alarms.

"CCR processes the threads using a round-robin method, meaning that if there are 5,000 notifications in the normal queue when a high-priority notification arrives, it will be the next one handled," Tarmey says. "Previously, notifications could be handled only on a first-in, first-out basis, so the high-priority message would have to wait for the 5,000 normal-priority messages to be sent."

Linear Scalability with CCR

Software House developers were impressed by the linear scalability supported by CCR. While other solutions can incur management overhead related to adding processing cores, C•CURE 9000 has attained direct scalability because CCR automatically takes advantage of additional cores without incurring overhead.

"Normally when you add processors you see improvement, but if you double the number of processors you don't see a doubling in performance," says Tarmey. "With CCR, we see linear scaling. If you double the number of processors, you will see a doubling in performance. That is impressive."

Tarmey credits the linear scalability, at least in part, to the CCR's efficiency in handling threads and its ability to gracefully serve two threads from one CPU.

"With our 75-thread solution, we encountered a lot of overhead because of the context-switching required when moving threads between cores," says Tarmey. "CCR automatically takes advantage of additional cores without incurring the overhead of context switching. If I upgrade to a quad-core computer from a dual core, we see a linear scale of performance. CCR just automatically takes advantage of new resources when you scale your hardware."

The linear scalability of CCR makes life easier for Software House and its customers. "With CCR we can provide a product that is perfectly linear in scalability," Tarmey says. "That's really a big deal because hardware is easy to upgrade. Software is not so easy."

Ease of CCR Integration

The simplicity with which CCR could be incorporated into the existing C•CURE 9000 application code made it easy for Software House to take advantage of the new functionality without affecting its development schedule.

"I came back from the Microsoft conference, and two days later I had removed our thread pool code and we were running with CCR," says Tarmey. "You can't beat that."

Tarmey also likes working with the Microsoft Application Platform because it provides tight integration. "From the development tools to the operating system, database and other elements, everything just works very well together," he says. "It was still surprising, though, to see how easy it was to drop CCR into our product and see such immediate results."

Use of Microsoft R&D Breakthroughs

As a developer, Tarmey is happy to take advantage of Microsoft's software innovations, something he had already been doing with the .NET Framework 3.0. "There are just so many features that come standard in the .NET Framework 3.0, such as encryption, authentication on the channels, serialization, extension points and more. That's a lot of code that I don't have to write."

He feels the same about the efficiency of CCR. "I want to take advantage of the engineering efforts that come out of the developers in Redmond," Tarmey says. "I've looked at what was done with the CCR, and you can just see that the Microsoft developers were really focused on making sure there is no blocking anywhere. The code is optimized so it doesn't seem as if there are ever any wasted cycles in the CCR. Every ounce of processing power is being used all of the time."

Tarmey praises Microsoft for taking a fresh look at a classic problem: coordination and concurrency in dealing with multiple threads. "It is brilliant code," he says, noting that had CCR not come along, Software House would have spent a lot more time delivering a high-quality product to market.

"Something like CCR wasn't in our frame of reference," says Tarmey. "It is a solution that people in the regular working world probably wouldn't have had the time to contemplate and create."

Microsoft Robotics Studio

Microsoft Robotics Studio was developed as a software platform that the robotics community could use to develop an array of applications across a wide range of hardware. The Concurrency and Coordination Runtime (CCR) feature of Microsoft Robotics Studio is a managed code library that addresses the need of service-oriented applications to manage asynchronous operations, deal with concurrency, exploit parallel hardware and deal with partial failure. The Decentralized Software Services (DSS) feature of Robotics Studio supports a simple services oriented application model, enabling developers to create program modules that run as dynamically inter-operable units that can communicate using a simple, open protocol. CCR and DSS can be used to manage multi-thread and multi-core processing.

"I just dropped the CCR into our code and it worked. It integrated so simply that it really facilitated my unit testing."

Stephen Tarmey
Architect, Software House
Tyco Security Products