.NET Boosts Homeland Security

Kingdomware Technologies used .NET Web services to build a notification system that provides warnings and instructions regarding national, local, and business security.

by Lee Thé

Print Article

Tech Ed 2003 Issue

The 9/11 disaster drove home the importance of being able to issue emergency alerts to the public quickly and efficiently. It also highlighted a less obvious lesson: Businesses need a comparable capability to alert their employees, and to give them specific information and instructions relevant to their lines of work. Generic alerts sent out to the public at large can't do that.

Husband and wife development team Tim and LaTonya Barton, of Kingdomware Technologies Inc. (KT), tackled both needs with a pair of alert-issuing applications—one for government agencies, the other for private enterprises. Any IT team trying to develop Web services that disseminate information to a variety of user interfaces and devices, some connected intermittently, can benefit from the Barton's experiences.

The Bartons had noticed that public alerts issued by the government provided little to no information or instruction to the business community. Moreover, the ways in which employees received public alerts affected when they would receive them—some from the radio, others from

Executive Summary

Company
Kingdomware Technologies Inc. (KT), a software engineering and management consulting firm that specializes in providing custom software solutions to information technology problems.

Project
Develop an instant alert system for executive corporate management that can immediately disseminate both government and business-specific alerts with information and/or instructions. The ultimate goal was to protect a corporation's human capital and minimize its loss during an emergency.

Legacy
KT was an early adopter of the .NET technology, converting existing proprietary Web applications solutions and 32-bit software solutions to .NET technology.

Solution
The Homeland Security Notifier System-Business Version (HSNS-

visiting various Web sites, while others might only hear about an alert from fellow employees.

So, the KT team set out to design a business system to provide immediate notification of security alerts, along with useful information and instructions. They focused on the pros and cons of current technologies, starting with e-mail- and telephone-based systems. These technologies are widely used, but in both cases delivery relies on a user's actions—either checking e-mail or voicemail. For maximum effectiveness, the system couldn't rely on user action or any other technologies except an Internet connection. Next, the KT team focused on the requirements and functionality needed to make a serviceable application:

- Speed: When an alert is sent, the user must receive it within 60 seconds.
- Attention: The alert must be able to get the user's attention, both visually and audibly.
- Mobility: The alert must be able to reach users at their desks or in the field.
- Visual vigilance: Users should remain aware of the national or state threat-level at all times, and have access to more information supporting the alert.
- Bridge to the government: Company alerts must be tied to the external environment, so users can receive national and state threat-level changes as well as national, state, and local alerts.

BV), which immediately blasts alerts with information and/or instructions to employees' computers and/or corporate cellular devices that support short messaging service (SMS). It can send alerts to multiple locations in multiple states.

Tools

• Visual Studio .NET
• Visual C++ .NET
• Visual Basic .NET
• ASP.NET
• .NET Web services
• Microsoft SQL Server 7
• Windows 2000 Server SP3
• T-SQL
• SQL Profiler
• SQL Performance Monitor
• Internet Information Services (IIS) 5.0
• ADO.NET
• GSM modem (using Global System for Mobile Communication, which sends SMS text messaging to cellular devices)

Challenges

• Testing different cellular devices to ensure the SMS feature worked
• Testing multiple operating systems for same functionality

The next decision centered on whether to build the app as a Web service or as an app installed locally. Web service technology won out for a raft of reasons. First, it allowed KT to centralize all equipment in a secure location. It also meant reduced setup and training costs, the ability to keep proprietary technology in-house, and the facilities for releasing updates easily.

If Web Service, Then .NET

.NET was architected to support Web services, so choosing Visual Studio .NET (VS.NET) as the development platform was a no-brainer—especially because the Bartons have been using it since the first beta release in early 2000. They were comfortable with its built-in technology, such as Web service support, and Tim Barton is a Microsoft Certified Solution Developer. They decided to use VS.NET to create all components, Web services, data

access, and ASP.NET pages. At the same time, they chose to limit the use of third-party components to the ASP.NET pages. This gave them total control over the client software source code and let them test all aspects of the application for performance and stability.

Choosing .NET for development also meant they could support all Windows versions after Windows 95 (it doesn't support .NET). Windows let the Bartons exploit .NET's capabilities without adding any special software or components. When and if Microsoft releases .NET runtimes for Unix and the new Macintosh operating system, they'll consider adding support for them, writing the code in Visual C++ .NET. Waiting for the runtimes reduces the amount of code rewriting that's needed.

Any Web service alert system needs a database, and it should come as no surprise that the Bartons chose Microsoft SQL Server for theirs. The alert system needed to be able to scale up the number of users over time, without service interruption. They knew SQL Server would support such scaling because they'd used it successfully with other high-volume applications they'd developed. They also looked at Oracle, but the Web service architecture meant they maintained all servers in-house, so they didn't need to support databases systems a customer might have.

Security was also a priority, so the Bartons decided to encrypt all alerts with MD5 encryption. This is one of the strongest encryption algorithms on the market. And because the algorithm is available in VS.NET, they were able to use it without buying third-party tools.

The TK team began development with a tried-and-true system for mapping out an application's architecture: a white board. They listed some early functionality, then made additional changes on the white board as the design progressed (see Figure 1). Tim was able to translate the white board design into the necessary requirements to take things to production.

The team decided to call it the Homeland Security Notifier System-Business Version (HSNS-BV), architecting it as a .NET-based Web service application residing on an Internet Information Services (IIS) server. The Web service retrieves alerts from a SQL server database, encrypting them before they go to the client. The client uses a Win32 Winform application that sits in the client computer's system tray and decrypts alerts arriving from the Web service. The Web service also sends alerts through a series of GSM modems to cellular devices that support SMS messaging. A designated official or administrator fills in and maintains the list of designated recipients, specifying how each is to be contacted (using the Internet or text messaging).

The database's structure lay at the heart of the new application. KT already had a working

service (the government version) that informs the public of national, state, and local threats and alerts. The KT team wanted to incorporate those public alerts into the business version. They linked the two services so that when a Homeland Alert is sent, a national or state threat level changes, or when a state or local alert is sent, businesses are alerted immediately of these changes as well.

The Bartons began designing the public system in February 2002 and completed it that October. Once they decided to develop the business notifier, they began developing it that December, completing development the following February. The business version shares a lot of code with the public version, which reduced development time. Specifically, the business version uses the public version's alert module and change in threat status module. Both systems combined took one year to develop, involving more than 83K lines of code, written in VB.NET and C++.NET.

Stay Lit With Two Servers

The Bartons set up a server farm for the Web service, using dual-processor SCSI drive systems with 1 GB RAM. Users receive alerts on a normal computer display or as text messages on supporting cellular devices. The KT team put the database on two dedicated servers, which enables them to take one server offline for maintenance and activate the backup without affecting end users.

The database uses six stored procedures, 15 tables, and four views. The stored procedures increase the performance of the application because KT anticipates frequent updates. The views let them join related tables and get results without doing joins in code. "I recommend putting your business rules in the database and not in the code," Tim Barton says. "This greatly increases overall system performance, especially if you have the database on a dedicated server."

After creating and testing the database, they used SQL Server's performance monitor and profiler to check for any bottlenecks or table locks to the system. This let the KT team see if they needed to change the way they updated or retrieved data. They then started developing the business version of their Web service, using VB.NET. The Web service resides on an Internet Information Services (IIS) Web server, which the Bartons knew from previous experience could handle heavy traffic (100,000 or more hits per day).

The Web service also exposes a series of functions, such as retrieveThreatLevel and retrieveAlertStatus. The

Bartons control access to these functions by requiring users to provide business information, such as the corporate manager's username and password. If a user provides incorrect information, the application takes measures to block the intruder from accessing Kingdomware's servers, using the firewall to disable the IP address-also the MAC address if available.

Timothy and LaTonya Barton of Kingdomware Technologies Inc. used .NET technology to create a Web service to provide notifications regarding alerts from the office of Homeland Security or breaches to a company's internal security system-or anything else that might occasion the need to get the word out to employees right away.

Whatever the purpose of a Web service, the KT team believes they should incorporate some type of security to ensure that users don't abuse it. The security can be as simple as a username or password, or as advanced as controlling access through IP or MAC addresses. "You should also make sure that your Web service doesn't pass a huge amount of data back to the user unless you know a user's connection speed," Tim Barton adds.

The KT team tested the security of their new Web service by logging onto it, entering the required information, and getting encrypted information back. Then they decrypted that information to make sure it was correct.

A key part of the user interface was the administrative section businesses would log onto to create and send alerts. In the .NET world such UIs are usually done in ASP.NET with VB.NET as the programming language, and this was no exception. Alerts are emergencies, so this UI had to be self-explanatory and in no way impede a nontechnical business official or administrator who might be in a big hurry.

You can't assume that such people would know HTML. Consequently, the KT team incorporated an HTML editor into the ASP.NET page, giving nontechnical authorized administrators the ability to add text, upload pictures, add tables, and change font colors and sizes. This also obviates the need for customers to install special software or components. They simply log in, type the alert, and send it. The UI includes a "send to" listbox for specifying who gets what alert, which a user can fill in and edit as needed.

Provide For Intermittent Connections

At this point the KT team considered the circumstances under which employees would receive alerts. They soon realized they couldn't depend on users being at their desks. Field personnel might not have a desk, and employees who are in meetings periodically would be away from theirs.

Fortunately, these days most managers and field employees are issued cellular devices that support text messaging or Short Messaging Service (SMS), such as Nextel,

BlackBerry, and Verizon. So the KT team added cellular support with a couple of GSM (Global System for Mobile communication) modems to send SMS text messaging to cellular devices. They used the modems' supplied API and some Visual C++.NET coding to build a UI and functionality to let users send text message alerts alongside Web-based alerts.

That completed the server-side portions of the application. Next the KT team had to create the part of the application that resides on end-user computers. This would apprise users of the national or state threat level, as well as display business-specific alerts (see Figure 2).

By and large this was a straightforward task. However, they did have a problem with the color orange. Windows 2000 and its predecessors support only 16 colors in the system tray, and none of them are orange. But to be consistent with the colors used in the national security threat-level indicators, the Bartons had to figure a way to display it. They did so using .NET graphic classes and invoking custom system messages to the computer to get it to display the desired colors.

They also had to come up with a way to cope with dropped or unavailable Internet connections. They decided that if the Internet connection drops, the client module should halt until the connection is restored. The system then checks the Web service for updates, and changes the threat level or displays an alert as required.

The KT team wanted to give the business alert app the option of receiving national, state, or local alerts—or not. Once disabled, only business alerts would reach users, along with the current national or state threat level. They also decided that once an alert was sent the app should require employees to acknowledge each alert by closing the alert window. Finally, they decided to give employees an alert print option.

Trust—But Verify

After testing the app thoroughly in-house, KT decided to conduct a public beta. They went to a number of newsgroups such as alt.business and alt.computer and asked for volunteers. Nearly a thousand put the beta client on their machines and activated it. Then the Bartons sent daily dummy alert messages and random threat level changes.

The beta test revealed a row-locking problem produced by some SQL statements. Modifying those statements boosted database performance tenfold. Another bottleneck showed up with users who had dial-up connections. When an alert contained graphics, the picture came up on their systems slowly. So, they decided to put a 500K limit on pictures. This won't affect most JPEGs, but might require resizing some GIFs or converting them to JPEGs.

The last test came when they took the business alert system offline for an hour to see if it would make end-user systems hang or display the dreaded Blue Screen of Death. They held their breaths, but it didn't.

As befits a Web service, KT used the Web for release and distribution of the system. That was not only convenient, but it also ensured that anyone who acquired the application had the Internet connection it requires. However, they did decide to permit some customers to install the whole system locally, even though that would complicate updates and enhancements. Another option they decided to provide was a silent install feature in their installation package to install the application on multiple systems without user intervention.

We live in perilous times; and computer viruses and worms have contributed to the peril. It's nice to see computers play a role in heading off disaster before it occurs as well. And it's promising to see current state-of-the-art technology enable you to take business-employee communications to the next level.

About the Author

Former *Visual Studio Magazine* executive editor Lee Thé writes occasional technology pieces to pay for scuba diving trips for his wife and himself while he works on a science fiction trilogy. Reach him at aw66@att.net.

Close Window