Microsoft Visual Studio 2008
Customer Solution Case Study

Insurance Company Improves Time-to-Market with Enhanced Rating Engine

Overview

Country or Region: United States
Industry: Financial services—Insurance

Customer Profile
Headquartered in Columbus, Ohio, Grange Insurance offers automobile, life, home, and business insurance protection to policyholders in 13 U.S. states. It employs 1,500 people.

Business Situation
To maintain its competitive standing and its reputation among agents for being easy to do business with, Grange Insurance needed to keep its rating engine working at top performance.

Solution
Using Microsoft® Visual Studio® Team System and Visual F#, the company recoded its rating engine to take full advantage of parallel computing and multicore hardware.

Benefits
- Dramatically faster what-if modeling, rating, and other processes
- Time-to-market that is weeks shorter
- Significant IT efficiencies
- Greater competitiveness
- Strong positioning for the future

"With this streamlined development cycle, we can rapidly deliver more powerful solutions to agents—and they can deliver more choices and better service to policyholders that much faster."
Glenn Watson, Associate Vice President, Personal Lines, IT, Grange Insurance

For nearly 75 years, Grange Insurance has offered competitive products and services to policyholders in more than a dozen U.S. states. To maintain its well-earned reputation and standing, the company decided to enhance its rating engine—a software tool for rating policies and performing what-if modeling, impact analyses, and other vital activities. Working with the Improving Enterprises and using the Microsoft® Visual Studio® Team System development environment and Microsoft Visual F# programming language, Grange Insurance parallelized its rating engine to take better advantage of multicore server hardware, and in so doing garnered significant performance benefits. Processes that used to require hours now take just minutes, enabling the company to trim time-to-market by weeks and making it far easier for independent agents to sell and service Grange products.

"In IT we consistently find that using the latest technology helps us provide the most powerful and easiest-to-use systems for our customers. For that reason we decided to use Visual Studio Team System for this project."

Michael Fergang, Chief Information Officer, Grange Insurance

Situation

Founded in 1935 and headquartered in Columbus, Ohio, Grange Insurance offers automobile, home, life, and business insurance protection to policyholders in13 U.S. states. Employing some 1,500 people and working with more than 3,000 agencies and 16,000 licensed agents, the company earned 2008 gross revenues of U.S.$1.3 billion and today holds $2 billion in assets. Grange Insurance consistently receives an "A" (Excellent) rating from A.M. Best and is ranked among the top 75 insurance companies in the country by independent agents.

To maintain its outstanding industry position and reputation, Grange Insurance is committed to developing a variety of competitive, affordable products and delivering them to market rapidly and cost-effectively. Grange Insurance is equally committed to making it easy for both policyholders and independent agents to do business with the company. In a 2008 survey of policyholders who had filed a claim, more than 95 percent of respondents said that based on their claim experience, they would recommend the company to friends and family. In surveys conducted by the Professional Insurance Agents associations in Ohio and in Indiana, Grange Insurance consistently receives a number one rating.

To achieve both objectives—delivering the products that customers want and creating a positive experience for them and the agents who serve them—Grange Insurance strives to ensure that the software tools supporting those products are powerful and effective. One of the most important tools at Grange Insurance is the rating engine, which is used by internal staff and independent agents alike. Grange pricing analysts and actuaries use the rating engine to perform what-if modeling and to generate rules, proposals, premiums, and other essential components of various insurance-policy types. Independent agents use the rating engine to generate policy premiums for policyholders.

Because of the complex mathematical calculations it must perform, the rating engine at Grange Insurance, as at most insurance companies, ran traditionally in a mainframe environment. With the emergence of high-performance, multicore hardware, the company rewrote the engine in the Microsoft® Visual C#® programming language and, in 2006, began running it in a server environment based on the Microsoft .NET Framework 3.0. That environment included an intranet accessible by the pricing analysts and actuaries and an extranet accessible by the independent agents. Although both sets of users were generally satisfied with the rating engine, Grange IT executives began to see that they could make it even more powerful and effective by taking further advantage of the performance capabilities of the multicore hardware. To that end, they decided to rewrite the rating engine in parallel code.

Solution

Migrating sequential code to a parallel form can be a daunting task for programmers due to the greater potential for problems in communication and synchronization among concurrent tasks. Grange IT executives were determined to use the most advanced technologies available for supporting the development and deployment process—which led them to the Microsoft Visual Studio® Team System 2008 development system. "In IT we consistently find that using the latest technology helps us provide the most powerful and easiest-to-use systems for our customers," says Michael Fergang, Chief Information Officer, Grange Insurance. "For that reason we decided to use Visual Studio Team System for this project."

"We liked that Visual F# is tailored to highly mathematical problems, helping programmers work more closely to the problem domain and enabling actuaries and other nonprogrammers to review the code as they would a math formula."

Eric Zechman, IT Supervisor, Grange Insurance

Eric Zechman, IT Supervisor, Grange Insurance, explains, "The company had used Visual Studio Team System for about three years and had recently upgraded to the 2008 version. This was largely because of the application lifecycle management enhancements in Visual Studio Team System 2008 Team Foundation Server, particularly for continuous integration and tying automated testing to multiple builds."

For creating the parallel code itself, Zechman and his colleagues decided to use another product in the Microsoft Visual Studio and .NET Framework product family: the Microsoft Visual F# programming language. Microsoft Visual F# began as a Microsoft Research project, then was available as a separate download for Visual Studio 2008, and is scheduled to be included in the release of Microsoft Visual Studio 2010 as a full .NET language.

"We liked that Visual F# is tailored to highly mathematical problems, helping programmers work more closely to the problem domain and enabling actuaries and other nonprogrammers to review the code as they would a math formula," Zechman explains. "We liked the .NET Parallel Extensions available through F# for managing parallelism automatically through the runtime system. We also liked that Visual F# is an integrated part of the Visual Studio tool set and the .NET Framework. That meant our programmers could use the same IDE [integrated development environment] and powerful .NET libraries they knew from having developed the earlier version of the solution in Visual C#."

From Proof of Concept to Product Release in Two Weeks

For overseeing the project, Zechman and his colleagues enlisted the Improving Enterprises, a Microsoft Gold Certified Partner with lengthy experience in Visual Studio technologies and the .NET Framework, as well as a growing reputation for expertise in Visual F#. Led by Daniel Gray, Managing Partner of the Improving Enterprises, development began at Grange Insurance in early 2009 with a team of three Grange developers dedicating 50 to 100 percent of their time to the project. Development progressed from proof of concept to a finished and working rating engine for one state in just four weeks, and to finished and working rating engines for 12 additional states in just two more weeks.

Today, the parallelized rating engine runs in a .NET Framework 3.5 environment that includes the Windows Server® 2008 operating system, Microsoft SQL Server® 2005 data management software, a communications portal based on Microsoft Office SharePoint® Server 2007, and a Microsoft ASP.NET–based Web site for policy management. The rating engine also interacts with a statistical software package used by actuaries and pricing analysts to perform what-if analyses and determine rates. Deployment of the new rating engine is proceeding state by state and, within each state, one line of business at a time. Grange IT executives estimate that deployment will be completed in early 2010.

Process Efficiencies Through Compatibility

According to Zechman, the Visual Studio Team System environment helped the team expedite and simplify the development process through support for continuous integration, highly automated testing for builds, sophisticated source-control mechanisms, and continuous tracking of work items and bugs. In the past, Zechman would manually compile data on work items and bugs, aggregate it in Microsoft Office Excel® spreadsheet software, and publish it biweekly to a SharePoint portal,

"Unlike working in J2EE or another infrastructure besides the .NET Framework, working with Visual Studio Team System meant that programmers didn't have to rethink how they did things or worry about the underlying layers of the architecture."

Glenn Watson, Associate Vice President, Personal Lines, IT, Grange Insurance

but this chore is one he no longer must do. "Today, work items and bugs are tracked automatically and published immediately through Visual Studio Team System Team Foundation Server," he says. "The people who need the information can access it at any time through the SharePoint portal."

The interoperability and compatibility of Visual Studio Team System with the .NET Framework also made matters easier, according to Glenn Watson, Associate Vice President of Personal Lines, IT, at Grange Insurance. "Unlike working in J2EE or another infrastructure besides the .NET Framework, working with Visual Studio Team System meant that programmers didn't have to rethink how they did things or worry about the underlying layers of the architecture," he says.

Other efficiencies emerged because of the compatibility of the programming language—Microsoft Visual F#—with the .NET Framework environment and the larger Visual Studio Team System, according to Gray. "Using the .NET Framework CLR [Common Language Runtime], developers were able to introduce C# objects from the prior code base into F#, which reduced development time by half," he points out. "Testing was easier, too, because the Visual F# runtime enabled side-by-side testing of components that would remain in C# and those that we were rewriting in F#."

Benefits

In addition to the process efficiencies experienced during the development of the rating engine, Grange Insurance anticipates that it will see major business benefits once the solution is fully deployed, both internally and to the independent agents. These benefits—largely the result of significant performance increases—include a shorter time-to-market, more efficient IT operations, and improvements in the way Grange Insurance does business with independent agents.

Gains in Performance, Productivity, and Competitiveness

As Watson tells it, the new rating engine is so much faster than its predecessor that on viewing the results of an early test report, Grange pricing analysts and actuaries could hardly believe their eyes. "After the first prototype was released, we performed typical what-if modeling on a book of 15,000 policies, and the test report noted the execution time in milliseconds—but at first people read that as seconds," he reports. "When they understood what the report actually said, they figured out that a process that used to take hours to do could now be done in just minutes."

Considering that these very same pricing analysts and actuaries regularly conduct similar what-if modeling up to a half-dozen times before releasing a new product, the efficiency improvement will be significant. Watson adds, "We'll be able to cut our time-to-market by weeks, an enormous advantage both in staff productivity and in the marketplace."

Major IT Efficiencies

Other performance gains in the rating engine will make a difference at the IT level, Watson explains. "In the past, after using the C#-based rating engine to determine the rating on a book of policies, pricing analysts and actuaries had to wait for the QA [Quality Assurance] group to test the rate validity on a mainframe because the prior rating engine required up to six hours to perform this task," he says. "Now, we can use the engine itself to do the validity testing, typically in about nine minutes for a book of 50,000 policies. This will relieve QA of having to get involved and enable us to retire the mainframe-based validity testing

"Using the .NET Framework CLR [Common Language Runtime], developers were able to introduce C# objects from the prior code base into F#, which reduced development time by half."

Daniel Gray, Managing Partner, Improving Enterprises

altogether, freeing up one-quarter of our current mainframe developers to work on other, more lucrative projects."

Such efficiencies support a fundamental philosophy within the Grange IT group, according to Brent Wyrick, Associate Vice President of Application Development, Grange Insurance. "We prize the idea of using one system to accomplish something, rather than two or three," he says. "By moving to a faster, server-based rating system, we are realizing that idea."

Still other IT efficiencies will result from the ease of maintenance inherent in the concise nature and mathematical structure of the F# code. "The C# language is excellent for many challenges requiring an object-oriented application, but for the highly parallel nature of a rating engine, F# provides developers a more natural way to express a solution," Zechman says. "In our case, by moving the rating engine to F#, we reduced the rating-code size by nearly 75 percent, which makes the application far easier to maintain."

Moreover, Zechman adds, because F# expresses the algorithms in such a mathematical form, pricing analysts and actuaries can read and follow the code more easily than they can read other programming languages. "This enables nonprogrammers to collaborate with programmers on troubleshooting and implementing enhancements," he explains. "The result is greater functionality for less cost."

Greater Ease of Doing Business

From whatever the vantage point, the IT efficiencies gained from the new rating engine and the use of F# for development and maintenance translate directly into a strong business advantage, according to Watson. "With this streamlined development cycle, we can rapidly deliver more powerful solutions to agents—and they can deliver more choices and better service to policyholders that much faster," he says. "Agents will be closer to offering same-day service from the time an application is submitted to the time a policy is approved."

This, in turn, makes life easier for the agents. "Ease of doing business is central to the company's mission to its customers—the independent agents and policyholders as well," Zechman explains. "This rating engine supports that mission fully."

In the future, agents will have an even easier way of serving policyholders, according to Watson. "Because the rating engine is so fast and because it is built on the overall Microsoft .NET Framework, it lends itself well to integration into a presentation model we are now building based on Microsoft Silverlight™," he reports. "Through that presentation model, we will be able to offer ideas, options, and online quotes to prospective policyholders through our Web site in a visually compelling way, with links to the independent agents who can serve the customer directly. This will help generate new business for the agents and Grange alike, which is an essential part of the company's push to expand aggressively and successfully into new market segments."

For More Information

For more information about Microsoft products and services, call the Microsoft Sales Information Center at (800) 426-9400. In Canada, call the Microsoft Canada Information Centre at (877) 568-2495. Customers in the United States and Canada who are deaf or hard-of-hearing can reach Microsoft text telephone (TTY/TDD) services at (800) 892-5234. Outside the 50 United States and Canada, please contact your local Microsoft subsidiary. To access information using the World Wide Web, go to:
www.microsoft.com

For more information about Improving Enterprises products and services, visit the Web site at:
www.improvingenterprises.com

For more information about Grange Insurance products and services, visit the Web site at:
www.grangeinsurance.com

Microsoft Visual Studio 2008

Microsoft Visual Studio 2008 is the world's most popular development environment for designing, developing, and testing next-generation Windows®-based solutions and Web applications and services. By improving the development experience for Windows, the Web, mobile devices, and Microsoft Office, Visual Studio 2008 helps organizations deliver a variety of solutions more productively than ever before. Visual Studio Team System expands the product line with new software tools that enable greater communication and collaboration throughout the development lifecycle. Interaction between developers and designers is enhanced through the use of Visual Studio 2008 and Microsoft Expression® Studio design software. With Visual Studio 2008, businesses can deliver modern service-oriented solutions more efficiently.

For more information about Visual Studio 2008, go to:
www.msdn.microsoft.com/vstudio

Software and Services

- Microsoft Visual Studio
  - Microsoft Visual F#
  - Microsoft Visual Studio Team System 2008 Team Foundation Server
- Microsoft Office
  - Microsoft Office Excel 2007
  - Microsoft Office SharePoint Server 2007
- Microsoft Server Product Portfolio
  - Microsoft SQL Server 2005
- Windows Server 2008 Enterprise
- Technologies
  - Microsoft .NET Framework 3.5
  - Microsoft Silverlight 2

Document published September 2009 (rev. April 2011)

Microsoft