Rational® software

IBM Business Partner Morpheus automates gap insurance processing using IBM Rational Enterprise Generation Language software

Overview

- ***The Challenge***
  The insurance division of a leading banking and financial services company in the U.K. had been using a manual system to process automotive insurance sales. Faced with competitors that had already deployed Web-based solutions, the bank needed to develop an automated solution and position itself for Web-based development initiatives.
- ***The Solution***
  The bank worked with IBM Business Partner Morpheus Limited to develop an automated system for insurance sales and processing. Using IBM Rational Enterprise Generation Language and IBM Rational Application Developer software, as well as IBM System i technology, Morpheus delivered a Web-based solution that integrates with the bank's legacy systems.
- ***The Benefits***
  Morpheus completed development in 100 days, enabling the bank to meet its deadline. Already serving several major automotive manufacturers, the bank is maintaining its competitive advantage. And Morpheus has built its own in-house expertise.

Motor vehicle insurance is a key growth area for one of the United Kingdom's largest mortgage and savings providers. Return-to-finance (RTF) and return-to-invoice (RTI) gap insurance policies are an important part of the company's financial services offerings. RTF and RTI policies protect a motorist when a vehicle becomes an insurance write-off by supplementing the motor insurance payout to clear any outstanding finance or returning to the customer the full invoice price that paid for the vehicle.

Until recently, the bank relied on manual systems to process sales of gap insurance. Many automotive dealerships offer such policies to customers when they buy a new vehicle. However, because the system was paper based, dealers tended to process applications in batches — a practice that ran counter to regulations set forth by the Financial Services Authority (FSA), the agency that regulates all financial services providers in the U.K. In addition to being labor intensive, the manual process also led to costly fulfillment errors.

The bank wanted to automate the process with a Web-based application that would integrate directly with its existing systems. In addition, it needed to deploy the system rapidly to keep pace with competitors that were beginning to roll out their own automated systems. Although the bank's development team had significant experience in IBM Report Program Generator (RPG) development, it had relatively little expertise in Web and Java™ technology-based development. Seeking to deploy the new system as quickly as possible, the bank turned to IBM Business Partner Morpheus Limited to develop, maintain and host the application, named Online Gap.

As an IBM certified for e-business Business Partner, Morpheus designs and delivers Internet business solutions for business-to-business (B2B) and business-to-consumer (B2C) environments. The success of Online Gap depended on the Morpheus team's ability to not only address the regula-tory requirements of FSA, but to deliver a secure, intuitive application. "It was important to create a simple, easy-to-use site that would encourage dealerships to use the solution rather than fall back on the manual processes. Security of the

Key Components

Software

- *IBM Rational Enterprise Generation Language*
- *IBM Rational Application Developer for WebSphere Software*
- *IBM WebSphere Application Server*
- *IBM DB2*
- *IBM i5/OS*

Hardware

- *IBM System i*

Services

- *IBM Rational*

site was also vital to ensure dealer pricing, and commission rates were kept confidential," explains Bleddyn Williams, director of Morpheus. "From a technical perspective, one of the biggest challenges was that there was no existing site in place, so Online Gap and the branding had to be built from the ground up—and it all had to communicate with our client's existing RPG-based premium calculation system over the Web."

Selecting IBM Rational EGL software

The bank did not specify a particular technology for the production of Online Gap; instead, it left the decision to Morpheus. After considering several options, Morpheus decided to build Online Gap using IBM Rational® Enterprise Generation Language (EGL), IBM's strategic rapid development technology. Although Morpheus planned to maintain Online Gap in the near term, it wanted to use a technology that would be easy for RPG developers at the bank to extend or modify if needed. EGL was a perfect fit because its development paradigm can be easily understood by skilled RPG developers. In addition to fulfilling that need, EGL enabled Morpheus developers with wide-ranging skill sets and backgrounds to apply Web and service-oriented architecture (SOA) technologies as they delivered an advanced solution.

Because the Morpheus development team has substantial experience in Java development, it could have produced Online Gap with a strictly Java implementation. But the company recognized the project as an opportunity to build its in-house expertise in a technology that could be leveraged for not only this client's project but for many other clients' projects as well. "Many of our clients are [IBM] System i™ customers. Because they are our primary focus, we are always looking for ways to speed up Web development for people that are RPG developers. And, because we had a free hand in selecting the technology to be used on the project, we decided to use EGL so we could get a good understanding of how it works; then take that knowledge and apply it to other customers," notes Williams. "EGL offers us a number of advantages, including integration with existing RPG applications, Web-enabling legacy applications, enabling clients to accelerate the development of Java skills, and a way to support our clients as they move further along the Java roadmap."

The project team

The Online Gap project team consisted of two seasoned Java developers and a third developer with no real-world Java development experience. Despite this disparity, the less experienced Java developer quickly became productive using EGL. "One of our developers came into the project having only used Java at university. He was able to make a rapid transition to working with EGL. When I can staff projects with people that don't necessarily have five years of Java experience, then that makes a difference in project cost," says Williams.

He adds, "EGL has worked quite well for us. We are now working with a number of other customers who are RPG based, and because of all we've learned on this project, we can answer the questions they have right away. When we can use EGL as a quick way to get some of our customers going for small projects, then it is a good option."

"EGL has worked quite well for us. We are now working with a number of other customers who are RPG based, and because of all we've learned on this project, we can answer the questions they have right away. When we can use EGL as a quick way to get some of our customers going for small projects, then it is a good option."

—Bleddyn Williams, director, Morpheus

Starting on the right path

Before Morpheus developers began working on the project, they received training from IBM Rational to ensure that the team started with a solid understanding of EGL and its capabilities. "The first step for us was to bring in IBM EGL experts to sit down with us and explain how things would work from an EGL point of view," comments Williams. "That was important for us, because although we definitely had a way we would have built Online Gap in Java, we wanted to make sure that when we built it with EGL it was going to be done the right way. We wanted to start off on the right path, and not try to work it all out ourselves."

In addition to initial training, Williams notes that support from IBM enabled the team to work with confidence throughout the project. "When we decided to use EGL, I wanted to make sure we had a good level of support. Some of our questions had immediate answers, and others took longer to resolve, but there was always a good level of support during the project," Williams adds.

An entirely new user interface

Because Online Gap was replacing a manual process, the Morpheus development team needed to develop a completely new user interface. To accelerate this part of the project, the team used EGL together with JavaServer Faces (JSF) technology, a set of Java classes and JavaServer Pages (JSP) tag libraries. "Building the presentation side of the application — putting the pages and the flow together — is much easier with JSF," says Williams. "We started by putting together flat pages of the site with really just the screen and the inputs and outputs that needed to go on that screen. After that, we began building the logic behinds the screens. Some of the pages looked quite simple but were very complex behind the scenes, and many of them were customized for specific automotive manufacturers. The combination of EGL and JSF was a big help in putting everything together."

"We wanted to quickly get to the point where the customer could see the application and begin testing functionality. … EGL and IBM Rational solutions helped us do that."

—Bleddyn Williams, director, Morpheus

By rapidly developing functional screens, Morpheus was able to get feedback and approval from the customer in the early stages of development. This reduced the risk of developing a solution that was not aligned with customer needs, and helped ensure that any differences were identified and resolved early, when they are easiest to fix. "Because we were on a fairly tight time schedule, our goal was to rapidly produce a testable application. We wanted to quickly get to the point where the customer could see the application and begin testing functionality. It was important for us to get the customer involved in the process as early on as possible. EGL and IBM Rational solutions helped us do that," comments Williams.

Building the application

The Morpheus development team used the IBM Rational Application Developer for IBM WebSphere® Software application to automatically generate Java code from EGL and construct the rest of Online Gap. "After we had set up the core flow through the application, we started to build up the pieces that then sat around it.

The Java code we generated using Rational Application Developer was very useful, and we tended not to touch it after it was generated. We were not just developing components that are deployed into IBM WebSphere Application Server, but also components that were run as Java programs to interface with the back-end RPG system," explains Williams. "In addition, it was very easy to write some parts of the application directly in Java and just plug them in with the rest of the application. The interoperability of EGL and Java worked well; no problems whatsoever."

Rational Application Developer also helped the team in developing the presentation tier. Williams adds, "We also used Rational Application Developer to build a site map, which we in turn used to build navigation components in EGL. That was quite helpful to us."

"The Java code we generated using Rational Application Developer was very useful, and we tended not to touch it after it was generated. … We also used Rational Application Developer to build a site map, which we in turn used to build navigation components in EGL. That was quite helpful to us."

—Bleddyn Williams, director, Morpheus

Delivering a compliant system in 100 days

Morpheus developed and deployed Online Gap in just 100 days. The IBM Business Partner also maintains the application that it deployed on IBM WebSphere Application Server. The production environment also includes an IBM DB2® back-end database and an IBM System i server running the IBM i5/OS® operating system.

Before Online Gap was made available for production use, the bank conducted a security audit of the application, including penetration tests, and certified the system as a banking application in compliance with FSA regulations. Because FSA compliance was a primary requirement of the Online Gap development effort from the beginning, Morpheus incorporated compliance requirements in the software specification and maintained a focus on them throughout development. For each insurance policy sold through Online Gap, the system produces a complete set of records to document the transaction.

"Our use of EGL for Online Gap has led the bank to consider using EGL internally. After seeing the speed at which RPG programmers and other developers can learn EGL and produce solutions, we are quite optimistic about EGL going forward."

—Bleddyn Williams, director, Morpheus

A win-win situation

According to Williams, the use of EGL and IBM solutions in developing Online Gap benefited both Morpheus and its client. Not only was the project successfully delivered on time, but both organizations are better positioned for future development initiatives.

"From Morpheus' perspective, we see EGL as giving us another option in addition to developing directly in Java. If we are working with a customer that has RPG developers and wants to move into developing Web applications, we can introduce EGL. We've noticed that open-minded RPG developers can pick up the concepts of EGL very quickly. For many of our customers, there is no need to become Java programmers—it is too much of a leap for them. With EGL they can produce applications, and a lack of Java expertise is not going to stop them from getting things done," says Williams. "As a business partner, we can show our clients that building Web applications is not as hard as they think, because EGL provides an easy way to start."

Williams adds, "Our client got the Web-based system they wanted to replace a manual process. A significant amount of labor is saved due to reduction in errors and paper-based processes. In addition, the bank is in a better position to move toward SOA. They are looking at ways to modernize some of their System i applications and expose them as Web services in their internal infrastructure. We have been talking to them about EGL from an SOA point of view—and showed them how they can take a premium calculation routine, for example, and expose it as a service. Our use of EGL for Online Gap has led the bank to consider using EGL internally. After seeing the speed at which RPG programmers and other developers can learn EGL and produce solutions, we are quite optimistic about EGL going forward."

For more information

To learn more about IBM Rational Enterprise Generation Language software, contact your IBM representative or visit the IBM developerWorks® Web site:

ibm.com/developerworks/rational/products/egl

SWM10207-USEN-00