DI Systemer Accelerates Development using Model Driven Architecture and IBM Rational® Software

Overview

■ ***The Challenge***

To compete effectively in an increasingly aggressive accounting software market, DI Systemer needed to respond rapidly to customer requests for added functionality to their product line. The company wanted to automate routine development tasks, increase productivity, and reduce time to market while improving the quality of its software.

■ ***The Solution***

DI Systemer began using IBM Rational® Rose® Modeler and the Unified Modeling Language (UML) to create comprehensive business models and applications models. The company then adopted a Model Driven Architecture (MDA) approach to further increase productivity, artifact reusability, and its focus on business goals.

■ ***The Benefit***

DI Systemer has cut iteration time by 90% for server-side development. The MDA approach, based on Rational Rose Modeler, has enabled the company to concentrate more on business needs and business modeling, and less on semantics and infrastructure code details. The results are higher software quality, shorter time to market, and improved responsiveness to customer needs.

Based in Trondheim, Norway, DI Systemer AS develops, sells, and supports software for the accounting market, including invoicing and payroll systems. Over the past several years, competition in this market has increased, along with customer expectations of more frequent updates and additional functionality.

With its existing product line built in COBOL, DI Systemer was finding it difficult to keep pace. At the same time, the company's development teams were growing, and management needed a way to facilitate better communication and manage the increasing complexity of its development efforts.

First Steps

DI Systemer began working on a new product family based on a client-server architecture developed in Delphi. The goals of this initiative were to increase code reusability, shorten time to market and reduce development costs. To help drive the initiative towards these goals, the company adopted IBM Rational Rose Modeler for business and application modeling.

Support for the industry standard Unified Modeling Language (UML) in Rational Rose Modeler was seen as a key factor for the company moving forward. Arnstein By, Manager of Development at DI Systemer, explaains, "Our COBOL systems were very large and had very long life spans — some were started more than 20

years ago. So for us, it was clear how important good design is when you are building systems that can last for a long time. From that experience, we saw the value of modeling. UML is a standard all over the world, and the ability to work in accordance with the UML principles is very important to us. Rational Rose Modeler enables us to realize the benefits of UML."

He adds that using UML and Rational Rose Modeler improved communication and smoothed the transition to larger development teams at DI Systemer. "In the past, we would have only one programmer working on a project, but it is not like that anymore. When we have five people working together, we need the UML models to work effectively and hand off the design to another developer. We can use UML diagrams to communicate with different people and project groups. Also, when we bring new people in, there is a good chance that they already know UML, which helps them be productive quickly. For us, there are many benefits to using the UML and Rational Rose modeler," says By.

Extensibility Enables Automation

One of the primary goals of DI Systemer's new development approach was to automate as many development tasks as possible. Specifically, the company wanted to be able to automatically generate Delphi code from the UML models it developed. To ensure that models and code remained synchronized, the development team also wanted to reverse engineer code changes back into the model.

DI Systemer developers used the extensibility interface of Rational Rose Modeler to build tools for forward- and reverse-engineering Delphi code from UML models. This openness of Rational Rose Modeler has enabled the company to extend and customize the capabilities of Rational Rose Modeler and to automate many develop activities at DI Systemer. "Another factor in our decision to select Rational Rose Modeler is that it is open and it is possible to automate with it. That is important to us. We looked at other tools, but they did not have the capabilities we needed. Using Rational Rose Scripting and the extensibility interface of Rational Rose Modeler we can automatically generate code and database structures from UML models, and also reverse-engineer the code to update the models," reports By.

A New Vision and a Bigger Opportunity

Although DI Systemer had already achieved significant gains in productivity and development speed using Rational Rose Modeler, the development team saw an opportunity for even greater improvements. Because Component Object Model (COM) details could not be automatically generated with their current tools, the team still had to code them by hand, a task which was both tedious and error-prone. In addition, the system models had a high level of interdependency between classes, which made implementing some changes overly complex. Lastly, the product line was locked into the Delphi platform, which exposed the company to the risks and limitations common to all platform-specific implementations.

DI Systemer embraced Model-Driven Architecture (MDA) as the foundation of a new vision for development at the company. Thor Saether, System Architect for DI Systemer, explains, "Although Rational Rose Modeler was working very well for us and was very stable, I was still spending a lot of time coping with the tedious COM details. One of the main goals was to find a way of developing applications that would allow us to focus more on the business problem. We were looking for a way to focus on the business, further decrease our iteration times, build more robust and reusable business models, and prepare for different platforms. We

have achieved that with Model Driven Architecture."

The MDA Transformation

MDA is an approach to application design and implementation that encourages efficient use of system models in the software development process. Incorporating a number of Object Management Group (OMG) standards, including UML, MDA is an initiative that formalizes the evolution of model-driven development by defining models at distinct levels of abstraction. MDA also defines the transformations and relationships between those models and various implementation technologies.

By adopting the MDA approach, DI Systemer is building on its earlier successes with Rational Rose Modeler and UML. Architects perform analysis and design in a platform independent model (PIM) in Rational Rose Modeler. This model consists mainly of class diagrams and encapsulates the majority of the business modeling. The team then transforms the PIM to a platform-specific model (PSM) using automated tools which they built in-house. From the PSM, developers use their existing forward-engineering tools to automatically generate source code for Delphi and their database schema. After further development of the source code, the team can reverse-engineer changes back in to the PSM.

Lars Ofstad, Senior Developer at DI Systemer notes, "After UML modeling, MDA was the next step for us. We can separate the platform independent aspects in the models and we can now automatically generate all the COM interfaces and housekeeping code. Not having to write that code is a huge benefit."

Improved Speed and Quality

According to Saether and By, DI Systemer has already seen significant improvements in development speed and software quality as a result of following an MDA approach supported by Rational Rose Modeler. "We now have a framework that enables the developers to use their creativity while using UML profiles to ensure consistency across the business. That eliminates many errors because the profiles are implemented in the same way all over the system. In the past, you could do it in a million different ways. The framework is more strict, and that uniformity also helps when other developers need to do subsequent work on the model," says By. Saether agrees, "By automatically generating as much code as possible, we reduced the error frequency to a high degree. As a result, we have source code with very few errors".

Accelerating the development iterations for its server software has also enabled DI Systemer to reduce time to market. Saether explains, "On the server side, we have shortened our iteration time dramatically. Our systems include a middle tier and the user interface as well. But server-side development powered by MDA is very effective — we have seen a ten-fold increase in speed."

Focusing on Business Needs and Managing Complexity

DI Systemer is now better able respond effectively to customer needs because its developers can concentrate on business problems and manage the complexity inherent in large accounting systems. From a business perspective, these improvements have proven to be just as valuable as increased speed and quality.

"MDA and visual modeling with Rational Rose Modeler brings the programmer away from the source code and the details. As a result, we are able to focus more on the business logic and business needs,"

say Saether. “Our business model itself is quite complex. Before MDA, the models were complicated and highly interdependent. MDA enabled us to create a far more component-based model in Rational Rose Modeler. This model is more viable and robust, and gives us a better basis for extending it or restructuring it to meet business needs.”

With a model driven architecture and Rational Rose Modeler, DI Systemer is able to undertake changes that would have otherwise involved too much risk. Saether explains, “As the business model grows, we must be able to keep it clear and understandable. Building a model with a lower degree of interconnectivity makes it more expandable and extendible. Without MDA, this would have been impossible because it involved too much restructuring of the model, too much work to implement the results and make the code run well. It would have been too risky.”

Next Steps

Using MDA, DI Systemer has been able to separate business-based decisions from platform decisions. This separation has enabled the team to create other platform and language-specific models from its platform independent model—paving the way for the company to more readily consider alternative platforms moving forward.

The DI Systemer development team is also planning on extending its code generation capabilities to allow more of the presentation- and middle-tier to be generated from UML models. In addition, the team’s development process, which already includes several best practices from the IBM Rational Unified Process®, is being updated as well. As part of that initiative, DI Systemer is evaluating IBM Rational RequisitePro® for requirements management and IBM Rational Software Architect for integrated design and development. “Our plans are to extend our process and make it more formalized. We are looking at Rational RequisitePro as part of that. We are also interested in Rational Software Architect to help us further integrate the process and the modeling work,” says By.

A Real World Success

DI Systemer is using the MDA approach to drive the development of its next generation application suite, DI Office. Development of the CRM application is proof of the effectiveness of using MDA and Rational Rose Modeler for real world development initiatives, says Ofstad. “I think that is important to remember that this is not a theoretical practice. We are making real solutions, real computer systems. We use Rational Rose Modeler, UML and MDA daily. We had a very practical approach from the very beginning, and we kept the focus on building systems, and making it easier to build them.”

The results — increased speed, productivity, flexibility and quality—have enabled DI Systemer to position itself to respond to market demands more rapidly and more efficiently. This is an achievement, Saether notes, made possible by Rational Rose Modeler and model driven architecture. “Without Rational Rose Modeler and MDA, this would have been very difficult, if not impossible,” he concludes.

G210-2116-00

IBM Corporation
Software Group
Route 100
Somers, NY 10589
U.S.A

Printed in the United States of America
09-05

This case study is an example of how one customer and Business Partner use IBM products. There is no guarantee of comparable results.

References in this publication to IBM products or services do not imply that IBM intends to make them available in all countries in which IBM operates.

G210-2119-00