Rational Software Architect
December 2004

Next-generation model-driven development

Gary Cernosek
Market Manager
IBM Software Group

Contents

Introduction

In October 2004, IBM launched a major release of the IBM® Software Development Platform, introducing a new generation of modeling and model-driven development (MDD) tools. Central to the design and construction category of that launch was IBM Rational® Software Architect, a comprehensive, integrated design and development product that supports MDD with the Unified Modeling Language™ (UML®) for creating well-architected applications and services.

Rational Software Architect supports application and Web development (static and dynamic) using modern software industry technologies, including:

- *Java™ 2, Enterprise Edition (J2EE™) and Web services technologies*
- *Object Management Group (OMG) Model Driven Architecture® (MDA®) initiative and Service Oriented Architecture (SOA) styles of development and their associated standards*
- *JavaServer™ Faces (JSF) capabilities that enable rapid application development*
- *Reusable Asset Specification (RAS) for supporting asset-based development*

Rational Software Architect product highlights, features, benefits and system requirements are captured in the product data sheet and can be found at http://www-306.ibm.com/software/awdtools/architect/swarchitect/index.html. The purpose of this white paper is to discuss the product from a more evolutionary perspective in the context of the IBM October 2004 launch and in the midst of today's marketplace for software development tools.

Rational Software Architect is indeed IBM's next-generation MDD product. It supports workflows and usage scenarios notably different from those practiced by users of our pre-existing MDD products. This paper highlights some of those differences, but does not serve as a "how-to" guide in terms of actually using Rational Software Architect. Such a guide is expected to be published in the near future as a companion to this paper.

This paper is intended for two types of readers. Existing customers experienced with IBM Rational products will find this paper useful in determining if and when they should migrate or trade up to Rational Software Architect (or a related product) from their current IBM Rational toolset. Less familiar readers will find this white paper helpful in considering IBM Rational products for their development environment.

Motivation

The Rational brand has a long history of developing modeling and model-driven development tools. Based on the highly diverse needs of various industries and types of applications, we created several distinct offerings, each optimized for meeting those needs. These tools continue to help our customers create better software faster.

The October 2004 launch added to our design and construction product portfolio. We have not replaced or discontinued support for any of our pre-existing modeling tools. So you may ask, "Why is IBM Rational releasing yet another line of modeling products? Why not just continue to evolve the existing product line?" Answering these questions is a major motivation for writing this paper.

Another motivation for this paper is to highlight the exciting fact that Rational Software Architect includes a comprehensive development environment, offering a fresh look at how the different aspects of software design and construction interact. For the first time outside our early history in supporting the Ada development market, customers can now consider using an IBM Rational-branded integrated development environment (IDE) alongside our modeling tools. The October 2004 launch also announced IBM Rational Application Developer for WebSphere Software, the next version and the renaming of the IBM WebSphere® Studio Application Developer product. Rational Software Architect includes all of the capabilities of this world-class IDE and augments them with the latest in MDD technology. The result: an integrated design and development solution packaged in a single product.

Finally, Rational Software Architect responds to today's global demands in software development. The role of the software developer today is changing from both technical and business perspectives. Modern programming frameworks such as J2EE and Microsoft® .NET are very powerful, yet also complex. In working with our customers, we see signs that conventional, code-centric methods for developing and managing applications in these environments are reaching their limits. On a business level, software development has become an international team sport, inherently global and distributed, which further adds to project complexity. Rational Software Architect responds to these industry challenges, allowing developers of all types to work at higher levels of abstraction—connecting them more closely with the business drivers of their organizations and making them more valuable employees in the process.

Background of Rational modeling tools

We take a moment here to review our history and evolution of modeling tools. The goal is to help our customers understand why IBM has chosen the current product development path.

Rational Rose

Rational pioneered visual modeling and development with UML in the 1990s and early 2000s. IBM Rational Rose® software has been and continues to be a market-leading visual modeling tool. Rational Rose is a stand-alone modeling tool that integrates at the application programming interface (API) level with market-leading IDEs to support a variety of programming languages and other implementation technologies. Rational Rose was an important step in bringing MDD closer to practicing software developers.

Through the years, we found that despite the success of Rational Rose and advances in UML modeling practices, only a small fraction of developers used modeling on a routine basis. When we asked our customers why, they told us that many developers still did not understand or embrace the value of modeling. Rational responded to that issue with a barrage of education and training. But even after successfully making the value proposition, we found that developers still had a key problem: they didn't like to leave their IDE. Developers wanted visual modeling to be integrated—not with their IDE, but rather inside their IDE.

Rational XDE

Rational responded to this need in 2002 with Rational XDE™ software, providing an extended development environment for the next generation of programming technologies emerging at the time: Java and Microsoft .NET. We characterized Rational XDE as the next generation of Rational Rose—not a new version of it (hence the name change), and not necessarily a replacement for Rose (since we purposefully restricted Rational XDE to support only a select number of IDEs and implementation technologies).

Architecting Rational XDE as a plug-in to IDEs made great strides in advancing the adoption of modeling and model-driven development by developers. Rational XDE also advanced the state of MDD by offering a very powerful patterns engine that enabled patterns-based development and took software design-level reuse to new heights. Special customizations were later added to provide early support for the MDA initiative.

Rational Rose XDE

In early 2003, Rational became part of IBM. We continued to sell both Rational XDE and Rational Rose software since many of our customers still preferred to work with a stand-alone modeling tool. But having multiple modeling products sometimes caused confusion for customers as to which tool to buy.

In October 2003, as part of IBM's Passport Advantage purchasing system, IBM consolidated the Rational Rose and Rational XDE product lines into a single family: IBM Rational Rose XDE Developer. This consolidation simplified the purchasing decision for customers—whether they wanted a stand-alone modeling tool or a modeling tool that was integrated directly into their IDE, they could purchase one package and install the tool that they needed.

Getting closer to Java and Eclipse

Even before the acquisition, IBM and Rational were partnering to develop a new and more powerful approach to integrating MDD capabilities into the Eclipse framework and the IDEs built on top of Eclipse. An early result of this work was a code visualization and visual editing feature added to WebSphere Studio Application Developer in 2003 to serve as a lightweight, highly efficient means for developing Java implementation-level models.

It is this latest MDD technology, based more broadly on the Eclipse framework, which forms the basis for IBM's approach to MDD tooling. Instead of simply integrating with Eclipse, we began building new MDD capabilities on top of Eclipse. This approach enables an unprecedented level of support for Java and C/C++ development, as well as a whole new dimension of capability and robustness for integrating with tools and other facets of the development life cycle.

The October 2004 launch

IBM's launch in October 2004 represented a significant implementation of the IBM Software Development Platform vision: to provide a complete and fully integrated platform for software development optimized for each role across the development team. IBM continues to promote software development as a core business process alongside others that are also critical to the success of the enterprise. The newly announced products enable a more business-driven approach to software development, one that transforms traditional development into a business process that delivers a competitive advantage and provides new revenue and market opportunities for our customers.

This launch introduced a number of new products and names to the marketplace, each with its own history. What is common to all of the new offerings is that the Eclipse open source platform provides the technology foundation for the IBM Software Development Platform. The role-oriented nature intrinsic to Eclipse allows us to show how all practitioners on a development team can, in effect, have their own software development platform customized with views particular to each staff member's specific responsibilities.

For the architect role, we announced our next-generation product for modeling and model-driven development: Rational Software Architect. Rational Software Architect is not the next version of Rational Rose or Rational XDE, but rather represents a fusion of select capabilities and development paradigms supported by Rational Rose, Rational XDE and WebSphere Studio Application Developer. Rational Software Architect takes features from these three tools, adds additional MDD capabilities, introduces new structural review and control capabilities and hosts it all on the Eclipse 3.0 platform. The result is a comprehensive design and development solution specifically targeted to the needs of software architects who also develop or otherwise touch code.

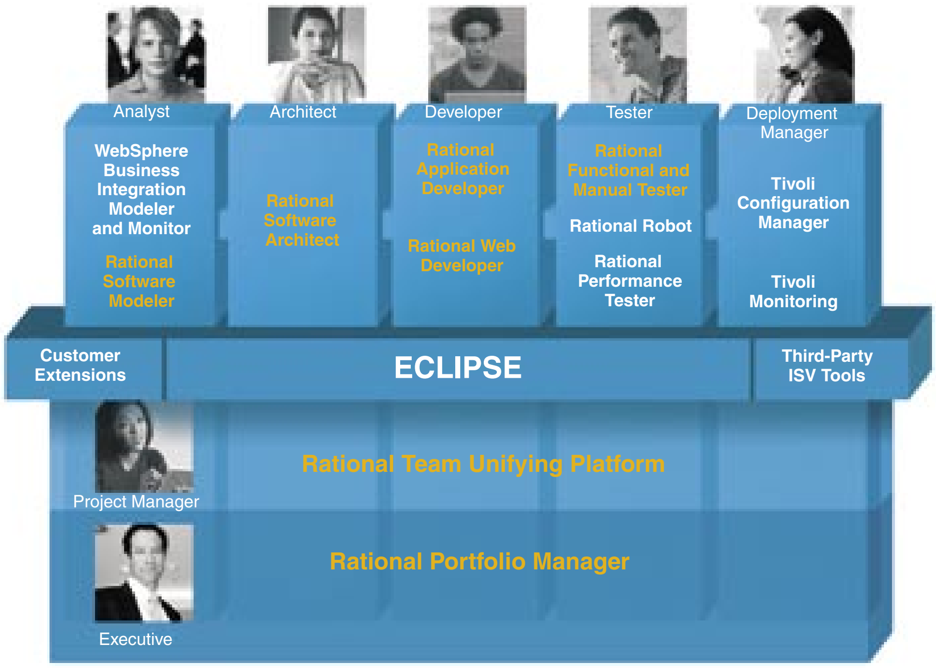

Analyst
WebSphere Business Integration Modeler and Monitor
Rational Software Modeler
Architect
Rational Software Architect
Developer
Rational Application Developer
Rational Web Developer
Tester
Rational Functional and Manual Tester
Rational Robot
Rational Performance Tester
Deployment Manager
Tivoli Configuration Manager
Tivoli Monitoring
Customer Extensions
ECLIPSE
Third-Party ISV Tools
Rational Team Unifying Platform
Project Manager
Rational Portfolio Manager
Executive

Figure 1: Eclipse provides the technology foundation for the IBM Software Development Platform

Why a new generation of modeling products?

As described earlier, UML-based model-driven development was first popularized by Rational Rose. Rational XDE introduced in-the-shell modeling as a means for bringing model-driven development closer to the developer. We also began adding runtime analysis to the packaging to support developer-level profiling and testing. With each addition of a tool or capability came another point-to-point integration requirement. As we continued adding more and more capabilities, we began reaching the practical limits of this style of tool integration. We realized that a new strategy was needed.

Eclipse as the foundation

Early trials with code visualization and visual editing inside Eclipse-based tooling were proving successful. For our next-generation MDD products, it was only natural to build additional model-driven development functions on top of Eclipse to form a more complete MDD tool. Rational Software Architect is the result of these changes, transforming the silos that previously defined modeling, development and code analysis into an integrated and unified design and development experience

Building our new products on top of Eclipse offers opportunities for deeper and wider integrations—deeper in how we can now leverage things like role-based user interfaces and tool extensibility and wider with new abilities to better integrate with other facets of the life cycle, such as requirements management and IDEs. The new IBM Rational modeling products combine our experience in market-leading UML modeling tools with an open, rather than proprietary, environment—making it easier for users of all types to customize them according to their needs.

Integrated design and development

One of the most important aspects of Rational Software Architect is that it is a complete design and development tool solution. Rational Software Architect includes all of the capabilities found in Rational Application Developer for WebSphere Software, which is the renamed new version of WebSphere Studio Application Developer. Rational Application Developer continues to include and improves upon the same code visualization and visual editing features found in WebSphere Studio Application Developer. This makes Rational

Application Developer a great entry-point for customers just starting out with MDD. Rational Application Developer includes all of the capabilities found in Rational Web Developer for WebSphere Software, which is the renamed new version of WebSphere Studio Site Developer.

Rational Software Architect starts with all of the features of Rational Application Developer and then adds full support for MDD, including UML Version 2 (UML 2) modeling, code generation, patterns and model transformations, and a new approach to implementing the MDA style of development. While Rational Software Architect is indeed a next-generation MDD offering from IBM Rational, it is not an entirely new product—it represents a natural evolution and merging of capabilities found in existing IBM Rational tools, packaged specifically for customers seeking to apply MDD in the broadest sense of the term.

Structural review and control

Rational Software Architect more than simply recasts IBM's MDD offerings into a new Eclipse-based framework and packages them with an IDE. We learned from customers that no matter how well an application might be initially architected and designed, its implementation will undergo code-level evolution which develops its own entropy. Left unchecked, the resulting architectural degradation can severely affect the quality of deployed software.

To combat this syndrome, software architects review existing code to assess its as-built architecture and code quality. In doing so, they find a variety of problems: incorrect mapping from design to code; code-level evolution resulting in design and architecture dependencies; coding standards, rules and style issues; and more.

To provide more automated assistance in these reviews, Rational Software Architect introduces a feature called "Java application structural review and control" built right into the tool. The premise is that application architecture is ultimately reflected in deployed code. Therefore, software architects must analyze the code to assess its maintainability and govern the evolution of the architecture with the assistance of rules.

Rational Software Architect supports template-based rule authoring to do just that. These new features enable users to discover the architecture for J2EE and Java 2, Standard Edition (J2SE™) implementations using high-level software visualization technology. In doing so, they will detect architectural weak spots, or "anti-patterns," such as cyclic dependencies, hubs or breakables that have crept into the application source code.

Structural review and control leverages elements of code visualization and developer-level testing to offer a new way for software architects to significantly improve the quality of the applications they design and deploy. We believe these are industry-changing facets of modeling tools and that they will start to change the way architects and developers think about their development process.

Integration with the runtime

Another thing that we have learned from our customers during this period is the important role that design and development tools play in enabling the runtime support environments in which the applications operate. Rational Rose and Rational XDE were developed to be more agnostic in this regard, and with that came pros and cons.

Rational Software Architect represents a significant step toward optimizing runtime support for Java applications on the WebSphere Application Server. At the same time, Rational Software Architect enables multiplatform execution support by virtue of Java Virtual Machines and data interoperability by virtue of its support for open industry standards. And since Rational Software Architect includes Rational Application Developer, and Rational Application Developer supports BEA WebLogic Server, applications deployed using Rational Software Architect are implicitly multiplatform as well.

Language support

The newly released modeling tools are not just about Java. We also expanded our support for C and C++ development to the enterprise via the Eclipse C/C++ Development Tool (CDT) packaged as part of Rational Software Architect. The Eclipse-based solution allows us to reuse some of the features in Rational Software Architect to support MDD across both types of languages.

Fostering a modern modeling ecosystem

The October 2004 launch also announced a variant of Rational Software Architect that supports only the modeling aspects of the tool. For users who do not need code generation or visualization, Rational Software Modeler serves as our next-generation visual modeling tool. Rational Software Modeler support all of the UML 2 modeling features found in Rational Software Architect. And by having Eclipse as its foundation, Rational Software Modeler offers advanced extensibility features via the Eclipse Modeling Framework (EMF).

The openness and robust extensibility of Rational Software Modeler enabled by its Eclipse foundation makes Rational Software Modeler well-suited to support a modeling "ecosystem" of customer and third-party extensions. Having such an ecosystem for Rational Rose has been a major contribution to that product's success in the marketplace. We envision Rational Software Modeler developing its own ecosystem in a similar fashion.

Part of a vision

Rational Software Architect and Rational Software Modeler are but two examples of the natural extension and evolution of the IBM Software Development Platform strategy to automate and integrate software development as a business process. Our vision is to have practitioners from all roles share a consistent user experience and a common definition and management of the assets they create or harvest. Every person on the team simply chooses the product that best matches their role and provides the kind of tool features and views of artifacts most relevant to their responsibilities and needs. This is what the IBM Software Development Platform is all about: team and organizational productivity.

Support for Model Driven Architecture

Those most advanced in applying modeling and model-driven development will be familiar with the OMG and its MDA initiative. MDA is an evolution of a number of OMG standards, including UML, along with an emerging philosophy for how best to apply modeling to the software development process. It is an initiative that formalizes the evolution of model-driven development by defining models at distinct levels of abstraction, and then defining the transformations that map and manage the relationships between those models and various implementation technologies.

What comprises MDA?

Here is a useful, albeit imprecise, definition of MDA levels of modeling:

- ***Computation-Independent Model*** *(CIM)—Addresses the environment and the requirements of the system without considering the system's structure or processing*
- ***Platform-Independent Model*** *(PIM)—Addresses the operation of a system without details related to a particular platform*
- ***Platform-Specific Model*** *(PSM)—Combines the PIM with the details related to a particular platform*
- ***Platform Model*** *(PM)—Defines, for use with a PIM, the technical concepts, parts, kinds of elements and services that compose a specific platform*
- ***Transformation Model*** *(TM)—Defines and specifies the transformation required to go from a specific PIM to a specific PSM*

While MDA itself is not a standard, it does explicitly call for the use of several OMG standards already in place. MDA specifies that:

- *Meta-Object Facility (MOF™) is used for defining the meta-models*
- *UML 2 is used to specify the application development models and transformations*
- *MOF Query/View/Transform (QVT) is used to specify the transformations (once it becomes formalized)*

Applying MDA with Rational Software Architect

Some of our customers might say that they have been applying the spirit of what is now called MDA for a long time. In fact, a number of IBM Rational staff members have been instrumental in working with those customers and the OMG in evolving MDA to its present state. Others view MDA as something different and new. In any case, Rational Software Architect supports both the principles of MDA as well as the standards upon which MDA is based.

Rational Software Architect actually supports a broad spectrum of code-centric, wizard-based and model-driven design and development paradigms, with MDA being but one of them.

Is Rational Software Architect right for you?

Rational Software Architect is IBM's most complete and robust solution for modeling and model-driven development. But is it right for you? This section will help answer that question.

Role orientation

In 1999, Rational Suite was launched with separate editions that spanned the stereotypical roles of analyst, developer and tester. We have since elevated the role of architect from underneath that of developer. There is also more granularity in the roles; for example, software architect versus data architect or J2EE developer versus embedded C/C++ developer. The IBM Software Development Platform is our designation for our overall software development solution, and that solution is more role-oriented than ever.

Role definitions and distinctions are intended to help our customers determine which tools are best for various staff members based on their activities and responsibilities. The October 2004 launch introduced a new product naming system that ties directly into this notion. But role names are inherently ambiguous and vary across industry, organization and even project lines. So our new product names should be considered as only a starting point for determining the applicability of the tools.

As indicated by its name, Rational Software Architect is intended for those performing in a software architecture and design capacity. Software architects are those members of a development team who are responsible for driving the major technical decisions—decisions that are often expressed in the form of an application architecture. In some cases, responsibility for this function may reside with only one person, who may be the sole proprietor of the vision of the system under development. This person is often referred to as the project's lead developer. In other cases, several staff members share this role, perhaps with one member acting as the chief architect. In still other cases, all of the developers on a team operate at one time or another in an architectural capacity—a trend we see growing. In this last notion, "architect" is more about what developers are doing than what title they hold.

Responsibilities of a software architect include identifying and documenting the architecturally significant aspects of the application, including requirements, design, implementation and deployment. These aspects require their own distinct views of the application, each of which is holistically related. There also is the distinction between an originating architectural design versus the as-built architecture—the one that is running in the code. We believe that software architects need a tool that helps them prescribe the desired architecture and discover the architecture that actually exists, ultimately helping them reconcile the two views.

Rational Software Architect helps users create and understand both perspectives. It includes capabilities that not only support the development and viewing of the implementation, but also support the architect in maintaining and controlling the implementation, and as a result, the implementation's as-built architecture.

While all of this may serve to describe the role of a software architect, relatively few staff members are given the title of architect. In most organizations, there will be staff members lacking the formal title who can realize value in the tool's features. Some additional characteristics to help determine the applicability of Rational Software Architect include:

- *Software architects who also develop code*
- *Anyone who needs to understand and work with both code and models*
- *Model-savvy developers who want to apply the full power of MDD*
- *Any developer on staff who reviews and validates existing architectures, or who wants to see how the implementation has forced the architecture to evolve*
- *C++ developers who want to apply MDD on top of Eclipse*

Keep in mind that Rational Software Architect includes all of the capabilities of Rational Application Developer, giving the Rational Software Architect user a full-scale IDE in which to work. Again, that is what makes Rational Software Architect unique for our traditional customers: a complete design and construction solution packaged into one product.

Rational Software Architect is for architects. Rational Software Architect is also for certain types of developers, as described above. For those architects who desire the full UML 2 modeling capabilities of Rational Software Architect but are not a developing architect, we offer Rational Software Modeler.

Degree of UML support needed

Full-scale MDD is not for everyone. Many developers still do not apply modeling to their development process. Or if they do, they use only a few diagram types, typically those most closely associated with the actual code. And in many of those cases, the diagrams are used only for visualizing the code as it has been built so that they can better understand the code and more efficiently affect changes to it.

It is for these types of developers that we chose to incorporate select UML features into Rational Application Developer. Rational Application Developer provides just enough UML capability to get developers started with the technology—a level we refer to as code visualization and visual editing. Rational Application Developer visualizes classes of code, their member data and functions and their relationships to each other. The product also allows you to visualize member function body information via UML 2 sequence diagrams. Rational Application Developer also supports the visualization of data in several notations (UML, IE or IDEF0) as well as the visualization of Web page hierarchies.

These powerful, yet easy-to-use features provide immediate value to the mainstream developer. And as such, many developers not yet ready for a full MDD solution will find Rational Application Developer to be an ideal starting point. And over time, as they find the interest or need to bolster development with additional MDD capabilities, they can upgrade to Rational Software Architect for a discounted charge.

Should I migrate or trade up from my current IBM products?

We encourage our customers to consider migrating or trading up to Rational Software Architect or Rational Software Modeler where and when it makes sense. Here are some of the benefits realized in doing so:

- *Eclipse integration*
- *Advanced model and tool extensibility via EMF*
- *Improved ease of use*
- *UML 2, the latest in modeling technology*
- *New structural review and control features (in Rational Software Architect)*
- *MDD with C++ on top of Eclipse*
- *Ability to create sophisticated custom transformations*
- *Ability to visualize code as static sequence diagrams*
- *Code review capabilities*
- *Use of both a Java IDE and an MDD tool (in Rational Software Architect)*

Depending on how your team currently uses Rational Rose or Rational XDE today, it might make sense to migrate to Rational Application Developer. The code visualization and visual editing features found in WebSphere Studio Application Developer have been upgraded for UML 2 and continue to reside in the Rational Application Developer product. While not applicable for general-purpose modeling or full-scale MDD, Rational Application Developer does support an entry-level use of UML. In particular, if current Rational Rose or Rational XDE users are using their tools primarily for extracting graphical documentation from their code, then this level of UML support may be sufficient for their needs.

Figure 2: Migrations and trade-ups to Rational Software Architect

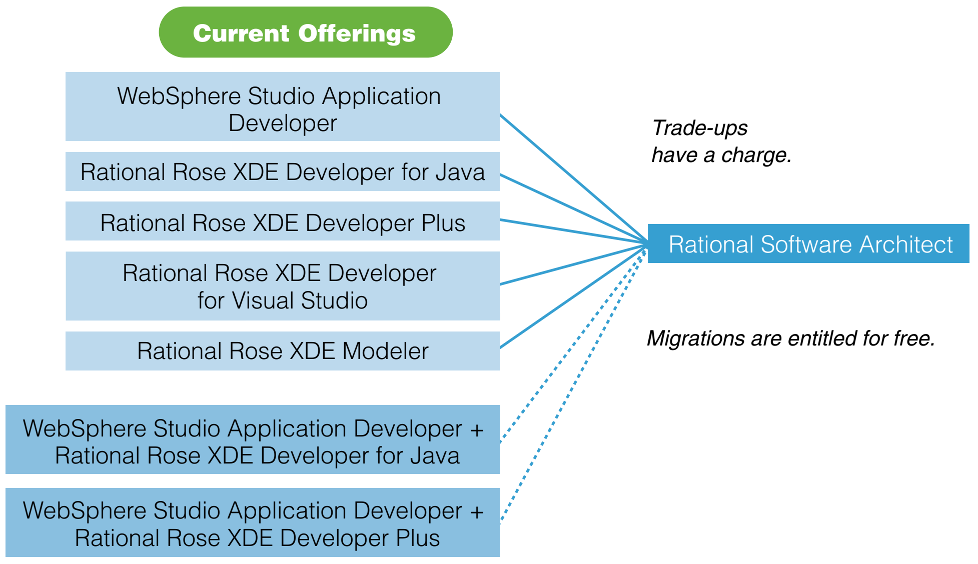
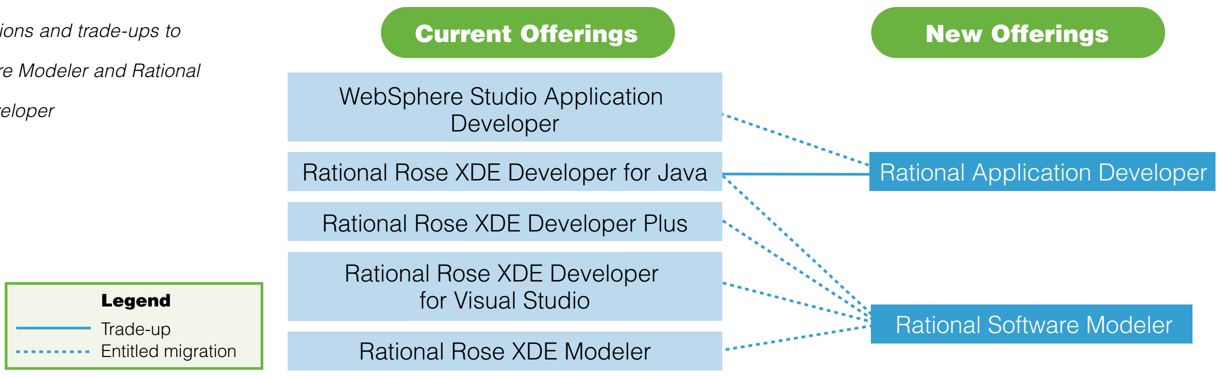

Current Offerings
WebSphere Studio Application Developer
Rational Rose XDE Developer for Java
Rational Rose XDE Developer Plus
Rational Rose XDE Developer for Visual Studio
Rational Rose XDE Modeler
WebSphere Studio Application Developer + Rational Rose XDE Developer for Java
WebSphere Studio Application Developer + Rational Rose XDE Developer Plus
Trade-ups have a charge.
Rational Software Architect
Migrations are entitled for free.

Figure 3: Migrations and trade-ups to Rational Software Modeler and Rational Application Developer


Current Offerings
New Offerings
WebSphere Studio Application Developer
Rational Rose XDE Developer for Java
Rational Rose XDE Developer Plus
Rational Rose XDE Developer for Visual Studio
Rational Rose XDE Modeler
Rational Application Developer
Rational Software Modeler
Legend
Trade-up
Entitled migration

For Java and Web development, we especially encourage our customers to transition from their current modeling products to Rational Software Architect. There are compelling technical advantages in moving to our Eclipse-based tooling.

To make these transitions more attractive, we offer a series of migration and trade-up paths, some shown in Figure 2 and Figure 3. See http://www-306.ibm.com/software/awdtools/architect/swarchitect/support/index.html for the latest in upgrade support information.

Additional information

If you have questions about IBM Rational Software Architect, visit http://www-306.ibm.com/software/awdtools/architect/swarchitect/index.html.

GC34-2485-03