

## INSIGHT

---

### Achieving J2EE and .NET Platform Interoperability Today Using Borland Janeva

---

Rikki Kirzner

---

#### IDC OPINION

---

Companies have made significant investments using Microsoft and Java 2 Enterprise Edition (J2EE) technologies as well as CORBA and COM to build and deploy applications. Many environments deploy a mixture of these technologies and platforms. In these heterogeneous environments, the back-end servers are often running J2EE (and/or CORBA) to derive the benefits of scalability, security, portability, and transaction performance, particularly for mission-critical applications. Visual Basic, Windows, and/or .NET technologies are running on the front-end client devices. Companies with these types of environments need to find a more effective way to support and extend these environments and achieve interoperability between the platforms. IDC concludes the following:

- The risk associated with trying to port applications from either the J2EE or .NET platform to the other was too great, and the cost estimates were often prohibitive.
  - Companies don't have to wait for Microsoft or Sun Microsystems to use effective tools to achieve J2EE and .NET interoperability.
  - Borland Janeva provides seamless connectivity between Microsoft .NET applications and the J2EE and CORBA infrastructures.
  - Borland Janeva is a proven and effective technology for solving the interoperability issues between J2EE and .NET.
- 

#### IN THIS INSIGHT

This IDC Insight discusses the problems companies face while trying to integrate and support their investments in J2EE and .NET platforms when these technologies are deployed together. Companies that have implemented or that plan to implement Microsoft and J2EE technologies in their environment do not have to wait for Sun and Microsoft to finalize the specific details of this agreement or to produce the interoperability solutions to support the two platforms. Borland's Janeva is letting customers solve the interoperability problem today and allowing them to create heterogeneous environments that capitalize on the benefits of both platforms as well as their investments in CORBA and COM technologies. This Insight summarizes the current situation and provides a brief review of the key features of Borland's Janeva.

## SITUATION OVERVIEW

When creating the mission-critical applications that drive their business, most companies continue to select the best-of-breed technologies that are most appropriate to achieve their business needs and reduce the risk of vendor or technology lock-in. Consequently, significant investments have been made over the years using both Visual Basic and other Windows languages as well as Java to create applications.

The benefits of the Java language and J2EE have enabled developers to create reliable, secure applications with high performance and better scalability than other platforms. On the client, Visual Basic and other Microsoft languages and technologies have continued to grow in popularity due to ease of use, availability of good tools, satisfaction of a large installed base, and ability to easily create productive and smart desktop applications. Languages such as Visual Basic.NET and C# are now being used to create Web services and rich user interfaces for the Microsoft.NET framework.

Companies have used both of these platforms to create n-tier environments, where the typical back-end servers are often running J2EE (or CORBA) to derive the benefits of scalability, security, portability, and transaction performance, particularly in mission-critical applications. Microsoft Windows dominates the desktop, however, enabling .NET to become the preferred front-end user interface and/or presentation layer.

While this typical configuration affords companies the best capabilities from both platforms, it adds significantly to the complexity of their software environment. It also presents an incremental and often continuous expense and effort to achieve interoperability between the two disparate platforms.

Over the years countless organizations discovered that once their technology choices were implemented, they were now burdened with incompatible systems that required increased expense and effort to achieve interoperability between the heterogeneous systems. They also discovered the expense and difficulty of having to maintain these systems, to pursue further development, or to make any significant modifications. The risk associated with trying to port applications from one platform to another was too great, and the cost estimates were often prohibitive.

To solve this dilemma caused by interoperability between disparate technologies and platforms, many organizations turned to integration tools and solutions. Integration efforts were deployed as a way to ensure that data and information remained complete, consistent, and current across the entire collection of heterogeneous systems and applications.

So great was the need that integration became the priority of many businesses in the past two years because business initiatives and mandates demand consistency, particularly for those IT systems with which customers and partners interface. Failure to achieve success in this endeavor results in longer production cycles, increased development costs, lack of trust on the part of customers and business partners, and, ultimately, loss of business.

The effort to find a more effective way to solve integration was one of the key factors driving the interest in and early adoption of Web services. Web services are considered a viable means of integrating platforms and operating systems and finding a way for them to interoperate more efficiently. A recent IDC study determined that one-third of companies deploying Web services were using these services for integration purposes (see *Shining Lights on Web Services Projects: Findings from IDC's Web Services Investments Study*, IDC #29196, May 2003).

Companies eagerly embraced Simple Object Access Protocol (SOAP) and other emerging Web services standards as the panacea to resolve this growing problem of interoperability throughout the organizations. Unfortunately, there are scores of different implementations of the SOAP Specification, and each one has its own set of capabilities, features, and application specificity. Most of these implementations are still fairly immature or still under revision.

Furthermore, neither XML or SOAP can effectively handle the binary formatting or other requirements for tight, fine-grained integration needed by many mission-critical applications. Often a typical Web services deployment also requires additional hardware and software resources, increasing the cost (and sometimes the complexity) of the integration solution. Therefore, while Web services are useful for connecting loosely coupled Web-based applications, neither the technology nor the standards are always the most cost-effective or ideal way to solve key issues of integration or interoperability.

Some companies turned to bridging tools to try to improve the interoperability between J2EE and .NET technologies. However, the early tools that provided a way to bridge the J2EE and .NET technology just added to the complexity of the solutions already in place. These solutions typically required additional hardware and software and often became a single point of failure in a system or introduced additional performance bottlenecks.

At the point when it seemed there might never be a resolution to this incompatibility, a surprising event occurred. On April 2, 2004, Sun Microsystems and Microsoft signed a 10-year agreement to settle their outstanding antitrust and patent disputes and cooperate in a broad range of technology agreements. The agreement was met with broad industry support and accolades.

If carried out to its fulfillment, this agreement will eventually enable both Microsoft and Java vendors to produce technology solutions that may serve as the basis for allowing vendors to reduce software complexity, reduce integration issues between .NET and J2EE technologies, reduce the risk associated with trying to rewrite portions of existing Java or Windows applications, and allow developers to choose tools based on features, functions, and innovation rather than have the decision forced on them because of the choice of software stack or vendor platform.

Media articles speculated that at some time in the future Java and .NET technologies would be able to communicate at last, and companies could spend their time solving business problems rather than solving the issues of interoperability between the two popular platforms. The question being asked over and over again was: When can we expect to see this? The answer was not forthcoming — from the media.

## **Borland Janeva Addresses the J2EE, CORBA, and .NET Interoperability Problem**

Companies that don't want to wait for Microsoft and Sun to solve the J2EE and Microsoft.NET framework interoperability issues can turn to Borland. The company's message was always about platform neutrality and providing its customers with the freedom to choose best-of-breed technologies. Borland has been developing and delivering technology solutions that support both platforms as well as one that could deliver interoperability between them.

Borland's Janeva is a compiler and a set of libraries that allow Microsoft .NET applications to connect to back-end J2EE or CORBA applications. Unlike Web services and other J2EE and .NET connectivity software, Janeva provides a tight, fine-grained integration to J2EE and CORBA by communicating to these back-ends applications using the Inter-ORB Protocol (IIOP). Janeva provides seamless connectivity between Microsoft .NET applications and the J2EE and CORBA infrastructures.

Janeva does not force the .NET developer to become an IIOP, J2EE, or CORBA expert to use Janeva. The client-side developer can use the Microsoft.NET framework and never have to deal with the complexities of either J2EE or CORBA. With Janeva, the .NET developer can continue to develop the front-end client without worrying about the back-end J2EE or CORBA portion of the system because Janeva makes J2EE and CORBA transparent.

The Janeva libraries reside alongside the .NET application and are called by it. No additional hardware or software infrastructure is required. Included with the development environment are a Java-to-C# compiler and an IDL-to-C# compiler that generate C# stubs and assemblies. The Janeva libraries are fully J2EE and CORBA aware and handle the task of translating .NET remote calls to IIOP at runtime, thereby allowing a tight integration with the back-end applications. The net result is that .NET applications can be directly connected to high-performance servers, and the effort and training costs needed to achieve this are minimal.

Janeva supports Microsoft's Common Language Runtime and hence supports all the .NET languages, including C#, J#, Visual Basic.NET, and Visual C++.NET. Janeva is also tightly integrated with Microsoft Visual Studio and Borland C#Builder.

Java developers working with J2EE on the server do not have to learn Visual Basic or C# to continue to work on a heterogeneous system running both platforms. The separation of client side and server side is achieved by what Borland calls an "ease-of-use" layer. The client stub assembly comprises two layers. The bottom layer uses Janeva runtime to access the low-level server technologies, such as the Naming service, Transaction Service, and EJB objects. The .NET client can access this layer for a direct client mapping to the J2EE server technologies.

The ease-of-use layer provides a .NET-oriented mapping of the J2EE server technologies. In this layer, J2EE design patterns are mapped into the corresponding design patterns of the Microsoft .NET framework. Either type of developer can drill down into the details of the other language.

In many instances, Janeva may be a more effective or efficient solution than Web services for solving the interoperability issues between J2EE and .NET. Unlike Web services, Janeva supports peer-to-peer style protocols such as IIOP for greater efficiency and more synchronous communication ability. It also has intrinsic knowledge regarding both the client and the server, so it is capable of providing support to the following key requirements for enterprise-class applications:

- ☒ Stateful services through a distributed object model, which can support arbitrary server-sided components and arbitrary life-cycle requirements
- ☒ Complex data types to handle automatic data conversions with less error (It also converts complex Java data types to corresponding .NET data types.)
- ☒ Load balancing
- ☒ Fault tolerance
- ☒ The ability to propagate two-phase-commit transactions contexts across application boundaries
- ☒ Optimization of resource utilization

Janeva has already been put to use by many corporations. Borland customers, particularly those in the financial and telecommunications industries using Janeva, have already achieved success working with J2EE and .NET technologies residing within their heterogeneous environments. For one financial customer, Microsoft.NET was the only reasonable choice for creating new interfaces on its existing applications. However, that customer needed to be sure that it could make the necessary modifications to its existing systems without sacrificing its existing technology investments or being forced to rewrite existing applications. It knew whatever solution it chose had to preserve the current state of security and performance that it had achieved and not force the company to modify the CORBA back-end technology that was already in place.

Having already abandoned a Web services solution because it could not manage either its security or performance requirements using Web services standards and solutions, the company decided to consider Janeva. The Janeva solution allowed the company to achieve its business goal without having to make changes to its CORBA back end. The .NET developers did not have to learn about Java or CORBA to create the new user interface, and they were comfortable using Janeva since it integrates with Microsoft Visual Studio.

## **FUTURE OUTLOOK**

Until Microsoft and Sun establish effective interoperability solutions to support both J2EE and .NET, companies need to have products they can work with at their own comfort level. Developers who want the ability to use a mixture of .NET and J2EE (and CORBA) technologies in their environment should consider using Borland Janeva.

Janeva gives developers a choice of technologies without the burden of additional complexity and without waiting for Microsoft and Sun to implement the terms of their agreement before making their decision. Companies can use .NET on the desktop and deploy secure, high-performance, high-transactional, and scalable J2EE or CORBA-based servers.

Microsoft Windows or .NET programmers can produce clever, productive .NET applications and still be able to interoperate with enterprise-class applications that are deployed with Java Application Servers or CORBA servers. Developers can continue using products that let them work at their own skill level, and they don't have to learn another methodology or technology to be productive.

---

### **Copyright Notice**

This IDC research document was published as part of an IDC continuous intelligence service, providing written research, analyst interactions, telebriefings, and conferences. Visit [www.idc.com](http://www.idc.com) to learn more about IDC subscription and consulting services. To view a list of IDC offices worldwide, visit [www.idc.com/offices](http://www.idc.com/offices). Please contact the IDC Hotline at 800.343.4952, ext. 7988 (or +1.508.988.7988) or [sales@idc.com](mailto:sales@idc.com) for information on applying the price of this document toward the purchase of an IDC service or for information on additional copies or Web rights.

Copyright 2004 IDC. Reproduction is forbidden unless authorized. All rights reserved.

---

**Published Under Services:** Application Development Software; Leading Developer Environments