

## TECHNOLOGY AUDIT

# JNBridgePro v6.0

## JNBridge

**SUMMARY****CATALYST**

Few enterprises are pure .NET or Java application environments. Although service-oriented architecture (SOA) is often the default response for integrating diverse application stacks, in some cases direct interfaces remain the most practical answer. JNBridge has specialized in providing application programming interface (API)-level interoperability tools to bridge the .NET and the Java worlds.

**KEY FINDINGS**

- JNBridgePro provides integration between Java and .NET by managing the complexities of cross-platform interoperability.
- It provides full access to custom classes, exceptions, callbacks, and any broad object-oriented API.
- SOA approaches are not always the best answer for integrating heterogeneous application stacks.
- JNBridge's approach provides all of the advantages and drawbacks of API-level integrations. The chief advantage is that JNBridge delivers vendor-supported integration tools that reduce the maintenance burdens typically associated with point-to-point interfaces.
- JNBridge is expanding its tooling to support cloud platforms, which, at this point, are moving targets.

**OVUM RECOMMENDS**

For integrating heterogeneous application stacks, the conventional wisdom is that SOA is the most flexible and ultimately cost-effective architectural approach. However, there are exceptions that prove the rule. Firstly, many IT organizations lack the architectural expertise for designing web services-based SOA implementations that are truly loosely coupled. Furthermore, RESTful services do not provide the manageability that may be required for managing integration in heterogeneous environments.



Where existing code is highly stable or where transactional integrity matters, more traditional direct interfaces may still be the most practical, high performance, and maintainable approaches. JNBridge is useful for scenarios where direct API-style integrations between Java and .NET environments are necessary, and where development team skill-sets lack cross-platform programming capabilities.

### VALUE PROPOSITION

JNBridgePro's tooling architecture strives to abstract the complexities involved when linking .NET applications with Java counterparts and vice versa. For example, .NET graphical user interface components can be embedded in Java, and vice versa, or transactions with full two-phase commit can be executed between the two platforms. The product is designed to make applications on the opposite platform appear native, enabling development teams to code in the development environment, language, and platform that they know. Although no single tool can ever completely eliminate the complexities of maintaining cross-platform code, JNBridge's tools reduce the need for explicit cross-platform coding.

JNBridgePro's architecture supports deployment scenarios anywhere the underlying platforms can run, and allows multiple end points on both sides. This allows its tools to support instances that may be restricted to processes within the same physical machine to large server farms, with some clients hosting up to 30 servers.

Common use case scenarios for JNBridgePro include:

- Integrating existing Java and .NET components and systems.
- Leveraging investments in legacy components or systems with new development on the other platform.
- Migrating from one platform to another, where JNBridgePro is used to bridge between migrated and what remains on the original platform.
- Adding cross-platform support to an existing product or application, while maintaining a single code base.

### SOLUTION ANALYSIS

#### FUNCTIONALITY

JNBridge's goal is enabling Java and .NET developers to access and manipulate artifacts from the opposite platform from within their native coding environment. This entails allowing Java developers to access, explore, and expose .NET classes via a plug-in within Eclipse, while allowing .NET developers to access, explore, and expose Java classes from within Microsoft Visual Studio, also via a plug-in. This provides considerable productivity improvements, as most developers tend to specialize in one platform or another; even those developers with cross-platform knowledge will typically have stronger domain knowledge on either side.



Specifically, JNBridgePro is designed to provide full access to APIs on other platforms, including the ability to embed Java graphical user interface components in .NET apps, or Windows Presentation Foundation or WinForms in Java apps. The newest version, JNBridge 6.0, adds capabilities that help to manage licensing issues for deploying its interactions in the cloud, and drag and drop and wildcard capabilities for loading classes or generating proxies.

The drawbacks of JNBridge's approach are endemic to that of any API-style integration; the tightly coupled nature of APIs can leave them fragile to breakage whenever code artifacts on source or target are modified. Therefore, JNBridge Pro customers must manage change carefully. Another limitation is that, as yet, JNBridge does not address the rapid emergence of dynamic scripting languages that are in many ways becoming alternatives or extensions to Java and .NET environments. Ovum hopes to see JNBridge extend its coverage to this newly rising facet of the developer world.

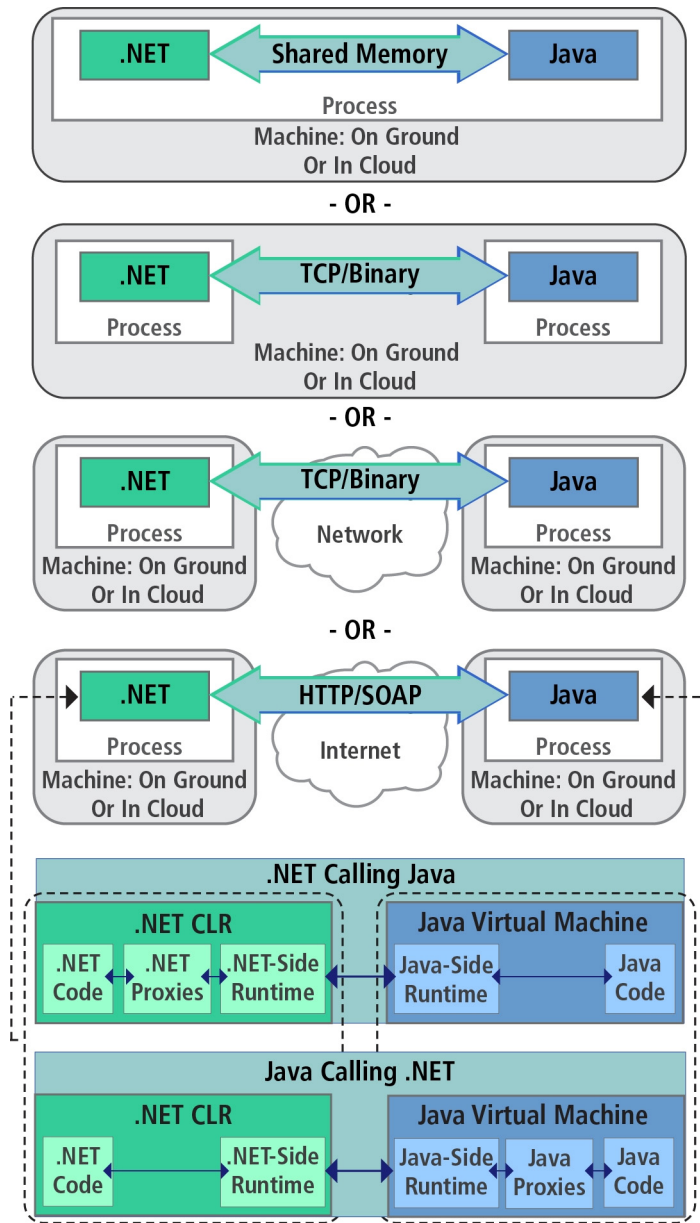
Nonetheless, API-level approaches make sense when there are a limited number to manage, the connections are mission-critical, and the application base relatively stable (although there will always be incremental changes that must be managed). In these instances, performance and reliability are critical, with the complexity introduced by SOAs a distraction. This is where JNBridge should shine.

JNBridgePro creates the interoperability bridge by generating a set of proxies that expose the classes' APIs and manage the communications. Once the developer selects which classes to expose, JNBridgePro automatically generates the proxies, and will optionally find and expose all the necessary supporting classes. After adding the proxies to the project, developers can then access the underlying Java classes from .NET, or the .NET classes from Java. When deployed, the .NET code runs in a .NET Common Language Runtime and the Java code runs in a Java Virtual Machine (JVM).

JNBridgePro's communications architecture offers a choice of three communication channels: a fast in-process shared-memory channel, a TCP/Binary protocol, or HTTP/Simple Object Access Protocol (SOAP). The latter could be considered an afterthought as, theoretically, using RESTful or web services should eliminate the need for delving at the API level, which is where JNBridge lives. The rationale for adding such connectivity is that it allows JNBridge to address hybrid environments that may use a mix of point interfaces and services.

JNBridgePro's practice is to support new versions of Java and .NET within 60 days of their release. Version 6.0 is the sixth major release in 10 years, and is focused on supporting deployment to the cloud. Upcoming releases will be focused on supporting additional cloud platforms and Java 7.

Figure 1: JNBridgePro architecture



Source: JNBridge

OVUM



## GO TO MARKET STRATEGY

JNBridge markets to three classes of customer: enterprise developers and development managers, systems integrators and consultants, and vendors that embed the tools under OEM agreements. JNBridge is a silver member of the Microsoft Partner Network as well as a member of the Visual Studio Industry Partner (VSIP) program, and works closely with Microsoft in integrating BizTalk Server with Java Message Service. While JNBridge has developed an Eclipse plug-in, its strategic ties with the Java community are looser, and reflect the more fragmented nature of the Java ecosystem.

## DEPLOYMENT

During the development phase, the user identifies the classes that need to be accessed cross-platform, and JNBridgePro generates proxy classes that mirror the classes on the other side. The user then programs directly against them, as if they were the underlying classes directly available on the other platform. Finally, the developer adds the JNBridgePro runtime components to both sides.

At deployment time, the developer determines where the .NET and Java sides of the application will physically reside and chooses which communications mechanism should be used. The user then deploys his or her application to the target machine or machines, and edits configuration files to specify the selected communications mechanism and ports to be used.

While the default integration might be simple, design decisions are still necessary for delivering the right performance. In other words, use of an integration tool does not provide a free pass to avoid making architectural choices, such as where logic resides, memory allocation, how to address values or references, or whether to use JNBridge's default garbage collection mechanisms.

### Global entertainment client

A global entertainment company that was launching a new attraction built an extremely content-rich and highly interactive website in ASP.NET that relied on a Java Enterprise Edition backend for content delivery. The content management system vendor's solution, a COM wrapper, failed during stress tests, so the company replaced it with a Java-.NET bridge using JNBridgePro. The company deployed it into a server farm prior to a large advertising campaign, and successfully supported over 181,000 visits on the day that its ad ran during the Super Bowl.

### Independent software vendor for semiconductor manufacturing

KLA-Tencor, a San Jose, California-based company that develops management and process control solutions for semiconductor manufacturers, was running a multi-tiered environment using Java and Microsoft .NET. The company's management decided it wanted to migrate to the .NET framework and move away from Java, but did not want to abandon the investment it had made in the Java tier. The company had about a million lines of Java code in use. Using JNBridgePro, KLA-Tencor saved \$1.6m in migration costs in the first year, and continues to expand its use of JNBridgePro in other projects today.



DATA SHEET

KEY FACTS ABOUT THE SOLUTION

Table 1: Data sheet			
<b>Product name</b>	JNBridge Pro	<b>Product classification</b>	n/a
<b>Version number</b>	6.0	<b>Release date</b>	n/a
<b>Industries covered</b>	All	<b>Geographies covered</b>	n/a
<b>Relevant company sizes</b>	All	<b>Platforms supported</b>	.NET platforms supported: Windows Server 2008, Windows Server 2008 R2, Windows Server 2003, Windows 7, Windows Vista, or Windows XP  Java platforms supported: any operating system that will support the underlying JVM or application server; Java 2 Software Development Kit, or Java Runtime Environment v1.3.1 or later
<b>Languages supported</b>	English	<b>Licensing options</b>	Perpetual
<b>Deployment options</b>	n/a	<b>Route(s) to market</b>	Direct, indirect
<b>URL</b>	n/a	<b>Company headquarters</b>	n/a
<b>European headquarters</b>	n/a	<b>North America headquarters</b>	JNBridge, LLC 3024 Jefferson Street Boulder, CO 80304
<b>Product name</b>	JNBridge Pro	<b>Product classification</b>	n/a

Source: Ovum

**OVUM**

Ovum's Knowledge Centers are new premium services offering the entire suite of Ovum information in fully interactive formats. To find out more about Knowledge Centers and our research, contact us:

**Ovum Europe**  
119 Farringdon Road  
London, EC1R 3DA  
United Kingdom  
t: +44 (0)20 7551 9000  
f: +44 (0)20 7551 9090/1  
e: info@ovum.com

**Ovum Australia**  
Level 5, 459 Little Collins Street  
Melbourne 3000  
Australia  
t: +61 (0)3 9601 6700  
f: +61 (0)3 9670 8300  
e: info@ovum.com

**Ovum New York**  
245 Fifth Avenue, 4th Floor  
New York, NY 10016  
United States  
t: +1 212 652 5302  
f: +1 212 202 4684  
e: info@ovum.com

All Rights Reserved

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior permission of the publisher, Ovum Europe Limited. Whilst every care is taken to ensure the accuracy of the information contained in this material, the facts, estimates and opinions stated are based on information and sources which, while we believe them to be reliable, are not guaranteed. In particular, it should not be relied upon as the sole source of reference in relation to the subject matter. No liability can be accepted by Ovum Europe Limited, its directors or employees for any loss occasioned to any person or entity acting or failing to act as a result of anything contained in or omitted from the content of this material, or our conclusions as stated. The findings are Ovum's current opinions; they are subject to change without notice. Ovum has no obligation to update or amend the research or to let anyone know if our opinions change materially.

© Ovum. Unauthorised reproduction prohibited

This report is a licensed product and is not to be reproduced without prior permission.

