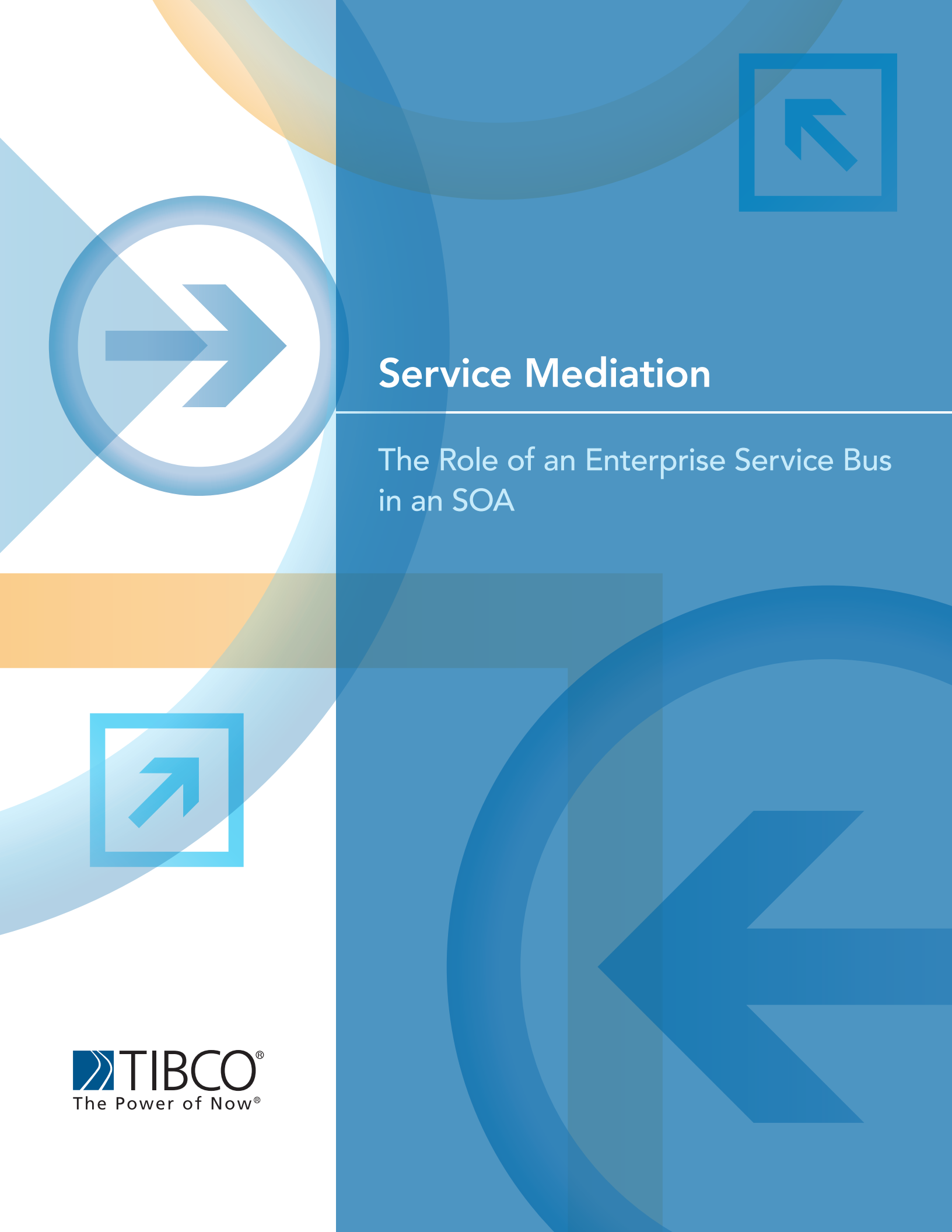
Service Mediation

The Role of an Enterprise Service Bus in an SOA



TIBCO®
The Power of Now®

TABLE OF CONTENTS

Executive Summary *Business agility—the ability to quickly adapt to changing needs—is increasingly becoming a key goal for businesses operating in the highly competitive global marketplace. Business managers need rapid resource and process reconfiguration to effect such change. To facilitate this change, IT managers are moving away from siloed processes and large monolithic enterprise software; they are examining ways to loosely couple applications together and provide business functions that can be turned into services and reused across the enterprise.*

A service-oriented architecture (SOA) is a way of building an enterprise IT infrastructure out of loosely coupled components known as "services" that perform discrete functions. Composite applications are a key element of an SOA environment. These applications are created by invoking and orchestrating multiple services, events and models in such a way that they collectively perform a higher-order business function. This functionality increases business agility by enabling IT departments to reuse components that have already been tested in production and have known scalability and quality-of-service characteristics. Such reuse can help reduce time to market and lower IT development costs.

An enterprise service bus (ESB) is a distributed, message-based integration solution based on open standards. The role of an ESB is to facilitate reliable communications between IT resources such as applications, platforms and services that are distributed in multiple systems throughout an enterprise. As IT departments increasingly focus on designing SOAs to lower development costs and increase business agility, ESBs are a key first step in setting up an enterprise SOA. ESBs form the foundation of the SOA and can be complemented by additional productivity capabilities such as service orchestration and registries. This paper discusses the requirements for an ESB that can address the needs of an enterprise-ready SOA.

Trend: Web services adoption standardizes SOA messaging

A key challenge for developers has been integrating multiple systems that use different languages and formats. However, the rise of Web services has provided service-oriented architectures (SOAs) with a standardized messaging format—SOAP—that enables different systems to interact. Web services also use Web Services Description Language (WSDL) to describe the endpoint's application program interface (API), which enables applications written on different platforms, such as C++, Java or .NET, to communicate with common interfaces.

1. The Road to Web Services and ESBs

SOAP, Web Services Description Language (WSDL) and HTTP have solved a thorny issue that plagued previous attempts to create standards for distributed systems such as Common Object Request Broker Architecture (CORBA) or the Distributed Computing Environment (DCE)—namely, widespread acceptance from the two largest application technology camps (.NET and J2EE) as well as from a number of application vendors. While this is no mean feat, the acceptance was gained by oversimplifying the standards to a core, agreed-upon set of technology statements. This design principle is simultaneously the Web services standard's greatest virtue and its biggest limitation.

In short, no enterprise-class implementation can run on SOAP, WSDL and HTTP alone. As Web services and SOA rapidly gain acceptance in the IT community, they face the same challenges that previous generations of distributed systems have encountered, only on a larger scale due their inherent decomposition principles. Foremost among these challenges is how to address the scaling of point-to-point connections, also know as the M*N connection problem.

Figure 1 below illustrates the connection explosion problem that point-to-point integration approaches introduce. For every application that is added, the number of connections grows exponentially as each application connects to every other application. TIBCO invented the Information Bus™ paradigm, illustrated on the right side of the figure. In this model, each application connects only once to a common backbone—the bus. This minimizes connections and provides a centralized location for administering the connections and managing integrated systems and architectures.

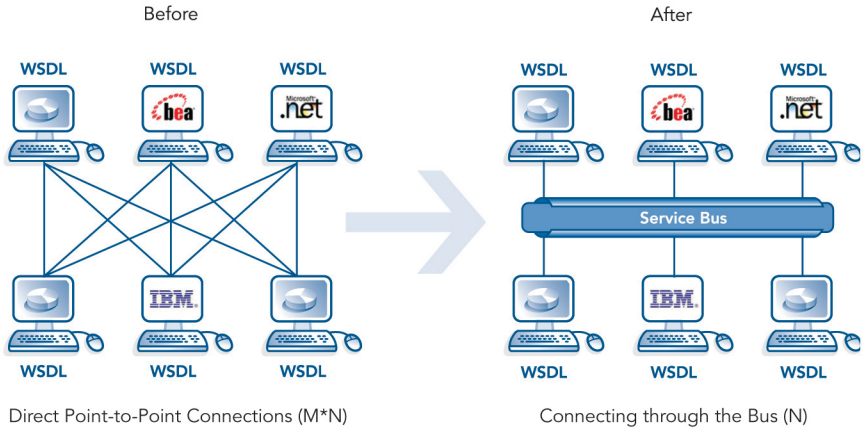


Figure 1: Reducing connection complexity with the ESB

To manage the complexity of how a service client connects and communicates to a service provider, the SOA needs a backbone that can go beyond the traditional distributed messaging to provide complex transformation, routing and loosely coupled connectivity in a heterogeneous IT environment, regardless of the platforms used. This reliable backbone provides an enterprise-class service bus that is true to all parts of the moniker ESB.

2. Enterprise-Class Requirements for an ESB

Let's take a look at the enterprise-class capabilities of the service bus. Each of the following functions is an essential element of successful integration in an SOA. Together they address the challenges facing service clients and service providers in an SOA environment.

Distributed messaging. At the heart of the ESB is a message-oriented middleware foundation such as TIBCO Enterprise Message Service™ software. This foundation provides a reliable, distributed transport mechanism using a store-and-forward mechanism that ensures message delivery even in the face of network failures.

Location transparency. With service mediation, a service client invoking a service provider only needs to be aware that the service exists; the client does not need to know where the service is running. The ESB locates the service when it is invoked, providing a level of service virtualization and localization transparency so that if a machine goes down or a service provider has to be moved, individual service clients do not need to be notified of the change. This can significantly lower IT management costs and minimize risk.

Transport transparency. In traditional, point-to-point integration approaches, components and objects are all very tightly coupled. In SOA, services are located throughout the IT environment and are less tightly coupled, due to location transparency. While relying on location transparency to connect service clients and providers, the ESB also provides physical transport protocol bridging to allow communication between services using different transports.

Multiprotocol support. Because the HTTP transport model contains inherent reliability issues and only functions well for synchronous message exchange patterns (MEP), it does not satisfy the requirements of every application or service.

For example, Java Message Service (JMS) contains asynchronous characteristics as well as improved transport reliability in comparison to HTTP. To support disparate application behavior, some systems use SOAP over JMS to achieve their desired effects. Other types of transport models are being used as well, including proprietary transport systems from major enterprise resource planning and systems vendors. Therefore, ESBs need to be able to support many types of transport systems to effectively integrate disparate systems and manage complex communications at the transport level.

Quality of service. For enterprise applications, quality of service (QOS) mainly pertains to service reliability. Message delivery and reliable service invocation are mission-critical functions of any system. Yet Web services alone do not provide guaranteed delivery. An ESB, on the other hand, can provide high service reliability by ensuring end-to-end message delivery that is beyond the reliability transports such as JMS can provide. Moreover, the way the high QOS is achieved must be standards-compliant, such as supporting the WS-ReliableMessaging specification.

Message exchange patterns. Most ESBs today operate on a request/reply paradigm using SOAP over HTTP, meaning the service client issues a request message to the user and waits for the response. This is also known as a synchronous MEP.

However, in the publish/subscribe MEP, the service client can send a message and subscribe to the response, rather than wait for it. The publish/subscribe MEP can respond more efficiently to events within an enterprise, particularly when the lifecycle of a service action takes place over long periods of time. An ESB needs to be able to address both paradigms.

Content-based routing. There are two types of routing within an ESB. The first, service routing, occurs when a service invocation enters the ESB and the ESB routes the request to the appropriate service provider, without requiring the service client to know the location of the service provider. This is how the location transparency discussed earlier is achieved.

The other type, content-based routing, introduces a set of rules or business logic that is applied to the content of the message at the routing stage and enables the ESB to route messages to specific service providers based on their content—for example, prioritizing orders from certain customers or flagging large orders for special treatment. This is a valuable service for businesses because it can

help lower information management costs, ensure adherence to service-level agreements and enable companies to focus on customer satisfaction.

Transformation. While the task of an ESB is to route messages from one service to the next, there are sometimes going to be cases where the data formats are mismatched. Hence, the ESB needs to be capable of transforming data from one format into another.

3. Additional Evaluation Criteria

In addition to evaluating the previous features when deciding on the best integration tool for an SOA, special attention should be placed on the following criteria:

Open standards. Open standards, such as SOAP, WSDL and Java Business Integration (JBI), are an integral requirement of an enterprise SOA. Therefore, these open standards should be supported both by the ESB solution components (runtime container, messaging infrastructure, integration services and design-time notations) and by the mechanisms for integrated resources to participate (attach, request and respond) on the bus.

Scalability and high availability. The ESB must be able to handle a large volume of messages to meet enterprise needs. In addition, high availability is essential to ensure uninterrupted business operations. If one element in the ESB fails, it should not necessarily stop services from communicating.

These criteria help IT departments ensure that the ESB is capable of handling the necessary transaction load quickly, reliably and with room for future growth—an essential element of business agility.

4. The TIBCO ESB for SOA

POWERFUL EXTENSIONS TO A SIMPLE CONCEPT

With TIBCO BusinessWorks™ software, introduced in 2001, TIBCO offers a mature ESB product that provides complete functionality for designing an SOA. TIBCO has specialized in integration for more than 20 years, pioneering event-driven architecture (EDA) with the introduction of its service-oriented Information Bus architecture in the 1980s. In fact, the philosophy at TIBCO is that enterprises need

a single architecture that supports both services and events so IT departments can expose information and applications as reusable services across the enterprise and enable the real-time flow of event-driven information.

With this vast experience, TIBCO understands that an SOA is not just about Web services. While most ESBs presume everything is a Web service, pure Web services standards are not enough to ensure the integration of all applications and interfaces.

BusinessWorks is an extensible SOA enablement platform for integrating enterprise applications and developing and deploying Web services. Its bus-based architecture can be extended to accommodate a wide range of integration capabilities, providing an effective tool for organizations with complicated integration issues, and its "configure rather than coding" approach can help reduce total cost of ownership.

BusinessWorks goes beyond ESB functionality by providing an integration backbone that effectively creates, orchestrates and deploys services and assets in an enterprise SOA. The following features and functionalities in BusinessWorks can help IT departments realize the full benefits of an SOA.

5. Asynchronous Invocation Enhances SOA Flexibility

Built on decades of integration experience, BusinessWorks provides a stable, high-performance integration platform for a variety of applications and service endpoints. In addition to supporting Web services, including SOAP over HTTP, BusinessWorks also offers SOAP over JMS. This enables asynchronous MEP and increases message reliability, which is an important aspect of quality of service. For businesses with services that do not need to incorporate Web services as part of the SOA, the JMS transport support still provides benefits, because it is inherently more reliable and offers better quality of service than HTTP.

6. Multiprotocol Support Streamlines Communications

Although ESBs support Web services, not all have multiprotocol transport support or provide the same level of multiprotocol support as BusinessWorks. TIBCO's software is specifically designed to support multiple protocols, including SMTP and FTP. This support provides a greater level of flexibility within an SOA and improves connectivity between heterogeneous systems.

7. Service Mediation Virtualizes Systems

As an integration platform, BusinessWorks has many different ways to bring systems together. A key element of that connectivity is the use of adapters. Since BusinessWorks is an independent platform and not linked to any one hardware or application server vendor, it is not tied down to any one approach or technology. BusinessWorks includes adapters for most major business systems, including mainframe applications and common business process software, such as Oracle and SAP products, which sit on top of the ESB transport and provide the additional special logic required to ensure complex connectivity.

8. Complex Routing and Transformation Optimizes Business Processes

In addition to connecting applications and systems within a heterogeneous environment, BusinessWorks provides a very powerful transformation engine. While most ESBs provide simple transformation, BusinessWorks supports complex transformation. Through an easy-to-use GUI, developers can design very complex transformations without having to spend hours in tedious coding (see Figure 2). By applying business logic and rules to messages, business functions and services can be more efficiently managed and composed, reducing design risk and increasing productivity.

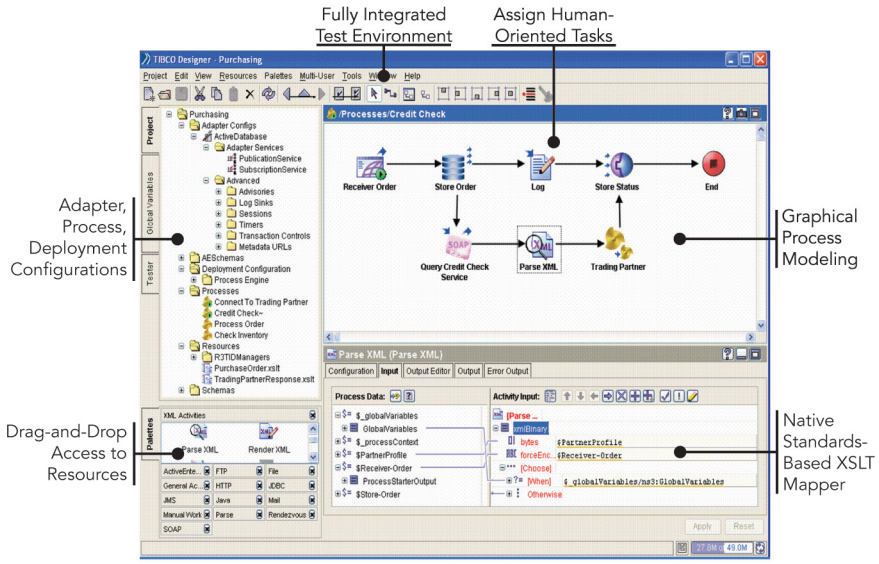


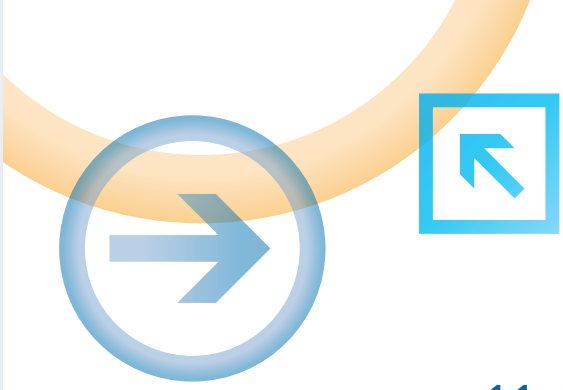
Figure 2: The TIBCO BusinessWorks no-coding approach employs a rich GUI for easy use. In the top half of the screen, orchestration is made simple with drag-and-drop operations. The bottom half shows the mapping of complex transformations.

9. Scalability and High Availability

BusinessWorks is highly scalable and reliable. Because it does not have to be deployed on an application server, it does not have to rely on the scalability and reliability of an application server. Instead, multiple instances of the software can be run on many different servers that communicate with each other. This provides active availability: if one machine disappears or starts slowing down under a heavy workload, another can take over. These operating system platforms may be from multiple vendors, a significant benefit of a vendor-independent ESB. Deploying multiple instances provides scalability and ensures high availability for mission-critical transactions.

10. Process Orchestration

Although orchestration is very much part of an overall business requirement, mainstream ESBs do not typically include orchestration services. BusinessWorks takes the ESB concept one step further and includes features that can orchestrate different business processes within the enterprise and compose those services into applications.



11. TIBCO Streamlines a Service-Oriented Approach

There are almost as many functional definitions for ESBs in the market today as there are vendors. In describing their products, however, a number of these vendors confuse the word "simple" with a lack of functionality; some products are missing such basic functions as adapters for connecting legacy and packaged applications to the bus, while others lack a proven messaging infrastructure. TIBCO BusinessWorks provides an ESB that is simple to use, as well as a no-coding approach to developing, deploying and running integration projects and building SOAs. By simplifying some of the more complex implementation issues that are critical for the glue and sequence between connected systems, such as process orchestration and transformation, BusinessWorks serves as a robust, enterprise-class messaging backbone and powerful ESB.

12. About TIBCO

TIBCO Software Inc. (NASDAQ: TIBX) is a provider of infrastructure software for companies to use on-premise or as part of cloud computing environments. Whether it's optimizing claims, processing trades, cross-selling products based on real-time customer behavior, or averting a crisis before it happens, TIBCO provides companies the two-second advantage™ – the ability to capture the right information at the right time and act on it preemptively for a competitive advantage. More than 4,000 customers worldwide rely on TIBCO to manage information, decisions, processes and applications in real time. Learn more at www.tibco.com

TIBCO®
The Power of Now®

Global Headquarters
3303 Hillview Avenue
Palo Alto, CA 94304

Tel: +1 650-846-1000
+1 800-420-8450
Fax: +1 650-846-1005

www.tibco.com