



# crossvision Service Orchestrator

## Table of Contents

- Executive Summary**
- What is an ESB and what is its Purpose?**
- ESB Features**
- Who needs ESBs and what Business Problems do they solve?**
- crossvision Service Orchestrator**
- crossvision Service Orchestrator Uses**
- Legacy Integration
- Information Integration
- Business Process Management
- Composite Applications
- crossvision Suite**
- Conclusion**

## Executive Summary

- 1** Modern enterprises need to integrate applications and legacy systems to make critical business information accessible to their organizations. These interfaces may include
- 2** Web interfaces for internal and external users, exposing data to applications, enabling
- 3** mobile workforces and widely distributed business and partner networks. In order to
- 4** be more competitive, improve productivity, comply with new regulations and leverage
- 5** information to gain market share and profits, organizations must simplify data integration and messaging between a wide variety
- 6** of disparate systems. That may mean “ripping and replacing” older systems, which presents high-risk levels. It could also mean
- 7** integrating older, reliable workhorse systems and moving forward in an evolutionary, rather than revolutionary, fashion. Or it could mean staying put and avoiding the issue altogether – but that can be a short-sighted alternative.

If the first two situations apply to you, this white paper will help you understand your options and clear up many of the misunderstandings that surround various technologies/concepts like Enterprise Service Bus

(ESB), Service-Oriented Architecture (SOA) and Business Process Management (BPM). Despite the confusion, tangible solutions with measurable ROI have been emerging from leading vendors offering software in these categories. If you are looking to improve business processes, or want to incorporate your legacy applications into a more fully integrated and usable computing infrastructure, this white paper will interest you.

This white paper:

- > Introduces the concept and technology architecture of ESBs
- > Explores the business and technology problems they solve
- > Examines Software AG’s ESB solution, crossvision Service Orchestrator
- > Examines crossvision Service Orchestrator governance through Software AG CentraSite™
- > Further explores Software AG’s crossvision suite and its wider applications for SOA enablement

This paper examines what an ESB is and reasons for implementing one. The paper will then explore how an ESB fits into SOA and a tightly managed software development platform. The paper also gives real-world

examples of how enterprises are benefiting from using an ESB/SOA approach.

## What is an ESB and what is its Purpose?

An ESB is an event-driven, standards-based messaging engine. It acts as a conduit, central and intelligent bus for information that passes between lower-level systems and services in an SOA. Low-level systems could include legacy applications and databases, ERP application packages and any type of database. This layered pairing of services, in which disparate systems and applications must not need to know about other systems, is often termed "loose coupling."

ESBs are typically, but not always, designed around middleware infrastructure products built on open standards. The bus has persistence, meaning it can store and forward messages, and start subsequent operations before the current one is completed. The bus itself should not be confused with the term SOA. ESB can be used to implement SOA, but it has its own definition and can stand alone as a described entity. An ESB is a tool to help facilitate SOA.

## ESB Features

- > Supports messaging (synchronous, asynchronous, point-to-point, publish-subscribe)
- > Includes support for service orchestration and choreography
- > Includes intelligent content-based routing services (itinerary routing)
- > XML (eXtensible Markup Language) is its standard communication language
- > Supports Web services standards
- > Includes standards-based adapters (such as J2C/JCA) for supporting integration with legacy systems
- > Includes a standardized security model to authorize, authenticate and audit use of the ESB
- > Facilitates the transformation of data formats and values, including transformation services (often via XSLT) between the for-

mat of the sending application and the receiving application

- > Includes validation against schemas for sending and receiving messages
- > Uniformly applies business rules, enriching messages from other sources
- > Splits and combines multiple messages while handling exceptions
- > Routes or transforms messages conditionally, based on a non-centralized policy (e.g., no central rules-engine needs to be present)
- > Monitors for various SLA (Service Level Agreement) threshold message latencies and other SLA characteristics
- > Often facilitates "service classes," responding appropriately to higher- and lower-priority users
- > Supports queuing – holding messages if applications are temporarily unavailable
- > ESB is comprised of selectively deployed application adapters in a (geographically) distributed environment

Source: Wikipedia

Generally speaking, ESBs act as a centralized translation and routing center for all the different kinds of systems that could possibly exist within a network. They provide location transparency, protocol independence, data transformation and a consistent methodology for interacting with disparate systems. ESBs can facilitate the routing of code and data in an SOA or Web-services architecture. The business services created within an SOA are delivered to end-user applications (like an enterprise mash-up application leveraging Web 2.0 features) or to other systems that can utilize the services for different applications – machine to human or machine to machine.

One simple way to describe an ESB is to compare it to an e-mail rules filter. A message comes in (some raw data or a complex document described by metadata in the tradition of XML), the software verifies it, analyzes its content and then decides what to do with it. In very simple terms, that is what an ESB does. It takes data and decides what to do with it. More specifically, an ESB pro-

vides messaging, basic transformation and content-based routing. Another simple analogy would be an office receptionist. The receptionist receives calls, makes decisions about where they should be routed based on their content and requests, and then routes them appropriately. An ESB would extend this analogy by including the intelligence on the potential receiving end of the call – like when someone in the office communicates to the receptionist that he or she is not available for the call or indicates that they would like the call should be routed to voicemail.

ESBs include both consumers and providers of Web services, including rules and metadata about both. In our analogy, this includes the caller, the receptionist and the call receiver. Content-based routing, protocol transformation, message transformation are loosely coupled.

At the heart of an ESB are transformation services, typically powered by eXtensible Stylesheet Language Transformation (XSLT) that change data formats from the sending application to meet the requirements of the receiving application. For example, XSLT converts one XML document to another XML document format that uses a different set of XML tags (different schema). ESBs use XML as their standard communication language.

The power of transformation is a key to the success of an ESB. By design, an ESB helps different systems that have never known that the other existed work together. Each of these systems can and will have different ways of storing data. Even a time/date format will differ between systems. Transformation allows for the ESB to handle differences between systems and translate the data to allow it to be used by another consumer. Problems integrating different versions of applications, different operating systems or different languages start to disappear.

When the data gets sent to the ESB, it doesn't have to be in XML. There are different ways it can be transformed into XML,

but it doesn't matter where it is coming from. It will still be transformed into XML. Either the service itself or the ESB provides the capability to convert that data into the right format.

ESBs are designed to make application integration and legacy modernization simpler and less costly. They extract data from legacy systems, for example, without the need to rip-and-replace underlying systems. They help humans get at difficult-to-reach information, and, more importantly, they facilitate system-to-system exchanges that drive complex applications. Composite applications, like when Google Maps is combined with real estate data in a mash-up, are easily launched with an SOA that's facilitated by an ESB. If the real estate information comes from legacy county records systems, for example, the ESB and SOA can handle the data conversion cleanly and deliver the information to the Google Maps application with minimal friction.

## Who needs ESBs and what Business Problems do they solve?

ESB/SOA implementations solve a wide variety of business problems. Generally speaking, the loose coupling that an ESB enables allows the IT architect/developer domain to act independently of the business domain; even though the two are dependent. However, this separation of underlying data and business drivers/applications is what makes it easier for the two domains to solve real world problems. A platform-neutral, code-neutral, messaging-neutral and application-server-neutral bus enables any application to utilize any data. And, the data is described sufficiently to enable intelligent brokering and routing. If an application needs information X, it gets there via the bus. If an application needs several services that depend on data from stovepiped legacy systems, the data arrives via the bus. There is no need for stovepipes. With an ESB, the data is available, and large-scale, risky rip-and-replace projects are not necessary.

Consider an auto insurer that needs to gather data from a wide variety of legacy systems – both internal (claims information systems) and external (police records systems). In order to deal with customers efficiently, accurately and respectfully, customer service reps need information and functionality from customer data files, claims information systems, payment systems, billing system and, possibly, police and traffic records systems. Most insurers have to go through several manual processes in order to gather all relevant information in one place. The customer has to wait. Customer service reps are sent to various systems – logging in and out – searching for the right information. Some data is not current. Phone calls go back and forth. Paper is pushed in some instances. The customer becomes impatient, and the work is hardly productive.

With an ESB/SOA solution, the logic and data from those systems are exposed. The end-user application can complete typical functions, such as Get Customer Name, Query Customer Claims or Report Customer Claims and so on, hitting all the relevant legacy systems at once and assembling the full picture of the customer from within one application. Partner systems, like the police records database, are also in the mix. The service bus allows you to pull everything together. When a customer calls in a claim, the system looks up customer and claims information and the customer is paid all in one transaction. It walks through those different steps automatically, so the rep is more productive and efficient.

Some of the world's leading automobile manufacturers are using ESB/SOA implementations to connect parts catalogs and distribution systems for repair services. EDI and ERP systems need to coordinate the availability and physical distribution of parts. Car dealerships need to find out if the part is available, where it is coming from, how much it costs, and how long it is going to ship. All that information may reside in two, three, maybe even four different systems that need to communicate with each other

and deliver answers in real-time, with reliable information. The dealer might even want to integrate postal carrier information for delivery confirmations and notifications.

The ESB allows you do to those repeatable processes over and over again. Software AG helped Nissan Australia and Nissan UK do something similar. We will cover this in further detail as this paper progresses.

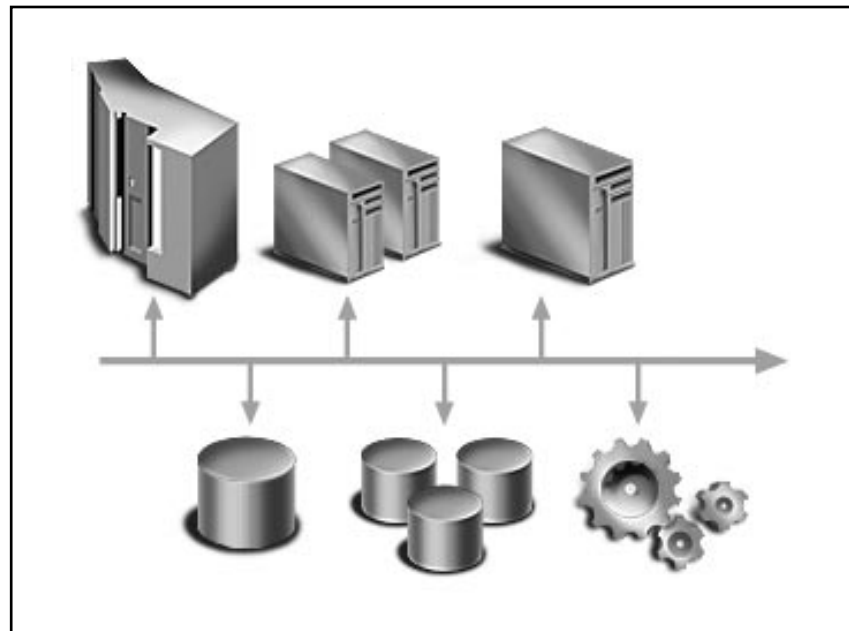
Companies that rely heavily on call centers benefit from ESBs by placing data and functionality directly into the hands of customers. Instead of leading customers through call center menus and connecting them with live help, companies are now Web-enabling customer data and offering customers access. The information supplied to customers online can come from any number of disparate databases, but everything comes together in one interface via the ESB. The strain on call centers is reduced, confusion and additional steps inherent to the live middleman scenario are eliminated, and customers can see what they want, when they want to.

## crossvision Service Orchestrator

Software AG's crossvision Service Orchestrator ESB directly addresses these business issues and offers simple, elegant solutions in even the most complex IT environments. crossvision Service Orchestrator is the foundation of the information integration and SOA enabling technologies that comprise the Software AG crossvision suite. crossvision Service Orchestrator is the result of evolving Software AG technologies and almost 40 years' experience integrating mainframe systems. It is a standards-based, flexible bus that ingests and transforms information, routes it and integrates it with other technologies, and then delivers it elsewhere.

crossvision Service Orchestrator is enterprise-ready and enterprise-class. It's a technology-neutral ESB with broad connectivity options and language support. In other words, it works with any virtually application server, programming language, database and messaging system. The ESB allows you to modernize and integrate legacy data and applications, integrate information across the enterprise, and develop BPM solutions from the services generated. Individual services are exposed from different servers (e. g. Get Customer Name, Query Customer Claims or Report Customer Claims etc.) and then combined as orchestrations that address a specific task. Legacy integration provides the fuel – in the form of data and rudimentary functionality – and business processes absorb the output as fully orchestrated services.

**Technology-Neutral ESB and BPEL Engine**  
Generally, there are a couple of different types of ESBs offered by the major software vendors. One is based on open standards and recommended as a strategic safe bet by the major analyst firms. The other is semi-open and requires organizations to buy into a proprietary operating system platform or application-server platform. The vendors of this second type of ESB say they offer the same benefits. But their products end up



forcing buyers to further commit to their particular technologies.

Software AG falls into the first category. Our ESB is operating-platform neutral, code neutral, messaging neutral and application-server neutral. It is flexible and works with anyone else's platform, OS, transport mechanisms, application server, messaging system or programming language. Our mission is to offer open-ended solutions.

This strategy applies to BPM as well. Software AG's SOA tools offer Business Process Execution Language (BPEL) integration that allow customers to orchestrate their BPM and workflow applications with existing Web services. Our BPEL engine facilitates composite-application architectures that loosely couple business processes with information systems. By exposing existing systems as Web services with BPEL, Software AG customers can access all their enterprise IT assets within the SOA, and reuse, combine and create new services as needs change and business drivers evolve. Software AG's BPEL engine is fully integrated with our crossvision Service Orchestrator ESB, so business functions drive service-oriented capabilities, rather than the other way around. Service Orchestrator Governance Through CentraSite

In complex development environments, service orchestrations need to be created, stored and managed in an efficient, intelligent manner. Software AG's CentraSite helps organizations manage all applications and services in one centralized place where developers can store them, share them, query them, and create new applications. Furthermore, through CentraSite Lifecycle Management and collaboration features, the development and deployment of these services and applications can be managed.

Governance and policy management features allow developers to work on the same project simultaneously while enabling you to discover different versions of code, roll back changes and troubleshoot diversions without losing work. If a development team is working on a search API, for example, CentraSite exposes all metadata, policies, procedures and the actual content in one place. It's a centralized, highly-manageable code marketplace, where efficiency, security and creativity reign. CentraSite allows you to reuse the same components over and over again, and keeps multiple artifact versions in one location – with unified workflow and approval processes. The repository directly integrates with the programming environment. crossvision Service Orchestrator is fully integrated with CentraSite, as are the other integration components included in



the crossvision suite, which we will discuss in detail later in the paper.

By providing maximum transparency on all SOA assets, CentraSite helps you to:

- > Manage the complete lifecycle of your SOA, based on open standards
- > Maximize the value of IT systems by promoting the reuse of new and existing assets
- > Achieve reliability through impact analyses prior to applying changes
- > Profit from improved transparency and enhanced collaboration

## crossvision Service Orchestrator Uses

### Legacy Integration

An ESB like crossvision Service Orchestrator plays a critical role in legacy integration efforts. Organizations of all sizes, shapes and structures are interested in modernizing their legacy systems. They want to modernize to get competitive, gain efficiencies, comply with regulations and utilize valuable information for profit. For some, that means risky rip-and-replace strategies that promise to solve many problems. For others, it means gaining access to information that resides in older, reliable workhorse systems with SOA – a preferable, less risky and less costly strategy.

An SOA approach to legacy modernization helps organizations combine existing functionality and quickly assemble run-anywhere software solutions. Java front-end applications, for example, can invoke back-end legacy processes that are appropriately optimized. You can record green screens and capture the interactions as complete user sessions for the purpose of Web or service delivery. The original source code does not have to be modified, so any developer can deliver a modernized system in a matter of days. For example, you can access COBOL, PL/1 or Natural programs directly via any .NET, Java, or Web services client, but those same host applications can also invoke services deployed on any Unix, Linux, or Windows server.

SOA helps you leverage existing IT assets and investments, and take full advantage of modern Web interfaces and new user-friendly Web applications, like those developed with Asynchronous Javascript XML (AJAX), for example. With the right integration software, data can stay in legacy systems while updated front-end programs access it via the service. If a database needs to be migrated at some point, it is not a problem. The services can be moved several at a time, and there's no effect on the front end application. It is isolated from the change in the back end. This allows you to port numerous transactions and applications

to the Web immediately, rather than three years down the line when the data migration and code customization has completed.

Software AG's crossvision suite, and the crossvision Service Orchestrator in particular helps organizations take a low-risk, calculated approach to legacy modernization. With Software AG, organizations can modernize with confidence and without rip-and-replace risks. Software AG customer Florida Community College, one of the largest and most comprehensive colleges in the United States, uses the crossvision suite to automate processes. The college has extended the value of their legacy applications by investing in SOA, rather than rewriting, converting and modifying them.

Nissan Australia Web-enabled its legacy applications and automated data exchange with their business partners using Software AG legacy modernization methodologies. Nissan dealers can now log into a centralized ordering Website and access legacy applications and data to query stock availability, place and track orders, transfer files and send emails. A rip-and-replace approach would have been risky and caused significant disruption to their business.

Nissan Motors in the United Kingdom created a similar system and cut their stock levels in half in 2004 – a move that saved an estimated 500 million Euros.

Oklahoma City-based American Fidelity Assurance Co. built business applications that exploit the company's core mainframe technology via SOA. They used tools from the crossvision suite to enable secure interoperability between enterprise applications via XML, Web services interfaces and standardized rules-based routing of documents.

### Information Integration

crossvision Service Orchestrator simplifies information integration projects, as well. For example, a manufacturing customer of Software AG's needed to reconcile 20 different databases in almost as many time zones

to streamline their manufacturing and delivery process. Their business was built through acquisitions of other manufacturers and each company had their own IT infrastructure. These companies had everything from mainframe databases running Adabas to Oracle on Unix to SQL Server and Microsoft Access on Windows. crossvision Service Orchestrator reconciled and integrated the data from each different source, pulling it all together in a centralized database using Software AG's Tamino XML database. The information integration component of Software AG's crossvision suite provided semantic integration, which creates and manages taxonomies (similar to metadata) for the data sources. The company layered business rules on top of the taxonomies, creating ontologies that map to data inside of existing databases. The information is queried and exposed as services. Now, all kinds of customer data can be pulled from numerous different databases and used as service orchestration sequences.

### Business Process Management

BPM builds on the promise of workflow applications by leveraging SOA. It's the next step in the evolution of SOA. Traditional workflow products are merely document routing systems that automate processes as documents flow within a company. With SOA, however, BPM blossoms into something more compelling. At each step of the workflow process, intelligence is inserted based on business rules and available data. Automated, machine-to-machine computation and intelligent routing create more value along the way. SOA and ESB automate the connections between a variety of different systems to achieve BPM.

Software AG customer The Pennsylvania State University implemented a BPM system based on crossvision crossvision Service Orchestrator that accesses data from back-end systems across the university. The new system, which replaced a 15-year-old workflow tool, covers more than 75 processes used by their 65,000 employees and students. The system allows users to access systems based on their roles and automates

hundreds of processes, easing IT maintenance requirements for 7,900 approval paths. Supporting documents can be attached to business processes, which eliminates approval delays caused by paper document trails.

### Composite Applications

Composite applications are fast becoming the "killer application" that organizations are looking to deploy to their users. No longer will users have to struggle with the application that has been deployed or have to copy and paste between two applications to get their work done. The composite application can deliver the right application to end users. In the world of SOA, composite applications are the face of SOA. While creating applications for end users from different data sources is not really new, composite applications, in the form of "Enterprise Mash-ups" deliver on the promise of SOA.

Apollo Optik, a large optical products vendor, deployed its Point of Sale (POS) system with Software AG's Adabas® and Natural® in 1990. This system, which provided order management, the sale of goods, information management and marketing, was design as a solution for the 480 retail locations, and was not integrated with the centralized data center. This resulted in redundant data and breaks in its business process chain.

As part of a large modernization and refactoring project, the data and logic of the POS systems at the individual retail locations were exposed as web-services. These services were then leveraged by the crossvision Service Orchestrator, to access information from the retail locations. Finally an AJAX end user application was created using crossvision Application Composer, creating the Enterprise Mash-up application needed by end users.

## crossvision Suite

Very few solution suites allow you to integrate legacy applications, take advantage of the latest technologies, incorporate a standards-based SOA strategy and maintain a central repository for your services infrastructure. Software AG has it offers a truly comprehensive range of capabilities with the crossvision suite.

Software AG's crossvision suite provides a comprehensive set of technologies, tools and the CentraSite SOA governance solution so organizations can develop SOA-powered applications like those mentioned in this paper. The suite helps organizations successfully leverage existing investments, reduce transaction costs, control existing business processes and integrate heterogeneous assets – all while protecting existing IT investments.

### crossvision Legacy Integrator™

Integrates existing assets and creates new functionality.

### crossvision Information Integrator™

Combines data from different systems into a single user-friendly view.

### crossvision Service Orchestrator™

Provides the ESB layer for SOA and Web enablement.

### crossvision Application Composer™

Builds new business applications from existing systems.

### crossvision Business Process Manager™

Coordinates the flow of business processes across your organization.

### CentraSite™

Manages and governs your SOA environment for maximum openness and transparency. It provides the registry and repository necessary to manage code and services while maximizing reuse and accessibility of valuable applications across the enterprise.

Instead of ripping and replacing your assets, crossvision transitions and modernizes your

existing IT landscape, to maximize the value of your assets. crossvision is based on open standards, enabling consistency with business partners and ongoing industry efforts.

## Conclusion

Software AG has more than 37 years of experience in developing proven software, and our forward-looking strategies are highly regarded by the analyst community. In fact, Forrester's Wave analysis named Software AG as a clear leader in ESB and SOA strategy, because we offer proven, highly-flexible solutions based on open standards.

Software AG's crossvision Service Orchestrator enables rapid implementation time frames for achieving short-term goals and its loose coupling allows you to flexibly meet future needs. Our crossvision suite of SOA tools, including CentraSite, are tested and widely successful across virtually every government and private sector vertical.

Software AG is optimally positioned to help you adapt legacy systems, leverage assets and connect information systems so you can adapt to business changes or new requirements as necessary. Our solutions allow you to focus on business issues and analyze process intelligence rather than agonize over technical hookups.

For more information ...  
[www.softwareag.com](http://www.softwareag.com) or  
[esbinfo@softwareag.com](mailto:esbinfo@softwareag.com)

Software AG  
Corporate Headquarters  
Uhlandstr. 12  
64297 Darmstadt, Germany

Tel: +49 6151 92-0  
Fax: +49 6151 92-1191  
[www.softwareag.com](http://www.softwareag.com)

