Fiorano
Enabling change at the speed of thought

www.fiorano.com

Message Driven SOA -- Enterprise Service Oriented Architecture

Component-based Business Process Composition

Providing Risk Mitigation, accelerated ROI and gets your EAI Project implemented in time

--- Atul Saini

AMERICA'S
Fiorano Software, Inc.
718 University Avenue Suite 212, Los Gatos,
CA 95032 USA
Tel: +1 408 354 3210
Fax: +1 408 354 0846
Toll-Free: +1 800 663 3621
Email: info@fiorano.com

EMEA
Fiorano Software Ltd.
3000 Hillswood Drive Hillswood Business Park Chertsey Surrey
KT16 0RS UK
Tel: +44 (0) 1932 895005
Fax: +44 (0) 1932 325413
Email: info_uk@fiorano.com

APAC
Fiorano Software Pte. Ltd.
Level 42, Suntec Tower Three 8
Temasek Boulevard 038988
Singapore
Tel: +65 68292234
Fax: +65 68292235
Email: info_asiapac@fiorano.com

Fiorano
Enabling change at the speed of thought™

COMPONENT-BASED BUSINESS PROCESS COMPOSITION

Providing Risk Mitigation, accelerated ROI and gets your EAI Project implemented in time

Executive Summary

Businesses that can react effectively to change will survive an increasingly difficult economy. Business users and IT developers are looking to deploy composite applications to enhance corporate decision-making and access to data, increase operational efficiencies and reduce overhead costs associated with maintaining application silos spread across the enterprise. Three key requirements are driving the next generation of distributed applications:

- Composite applications comprising of several individual applications loosely bound to each other using event-based messaging are being deployed over a widely distributed IT infrastructure. For example, as notions of grid computing begin to get popular, distributed computing on grids makes the situation worse with having to worry about heterogeneous computing environments for the programmer.

- Business managers need to securely modify existing composite applications in real-time in response to rapidly changing business drivers – without having to wait for programmers to re-program, test and deploy the new application.

- Businesses need to lower their costs and compress their schedules even as the need to deploy.

However, vendors positioning themselves as “solution providers” of composite applications have not been able to address the needs of real-time enterprises. Mired in their approach of integrating fine grained applications using brute-force low-level programming techniques, most incumbent vendors have created a patchwork of products that are adding greater complexity and costs in an attempt to provide solutions.

Component-based business process composition need never be this complex or expensive. This paper describes a simpler, affordable and scalable approach that enables Component-based Business Process Composition. Using the notion of coarse-grain programming techniques to create a palette of reusable components, the creation, prototyping, production and ongoing management of complex applications can be made as simple as manipulating numerical macros in a spreadsheet as used today.

Issues with current solutions

Unfortunately, the current notions of software components at the programming language level do not leverage the prevalent hardware topologies of internet-enabled networks to achieve scalable performance and flexibility.

Distributed applications running on today’s high-speed networks consist of one or more instances of individual software components running at each computational end-point, with component instances exchanging data in real-time across the network. Since all component instances typically run as separate processes in most real-world applications, the notion of a programming language level component (such as data-structure components) does not map well onto internet-enabled enterprise networks as shown in Figure 1

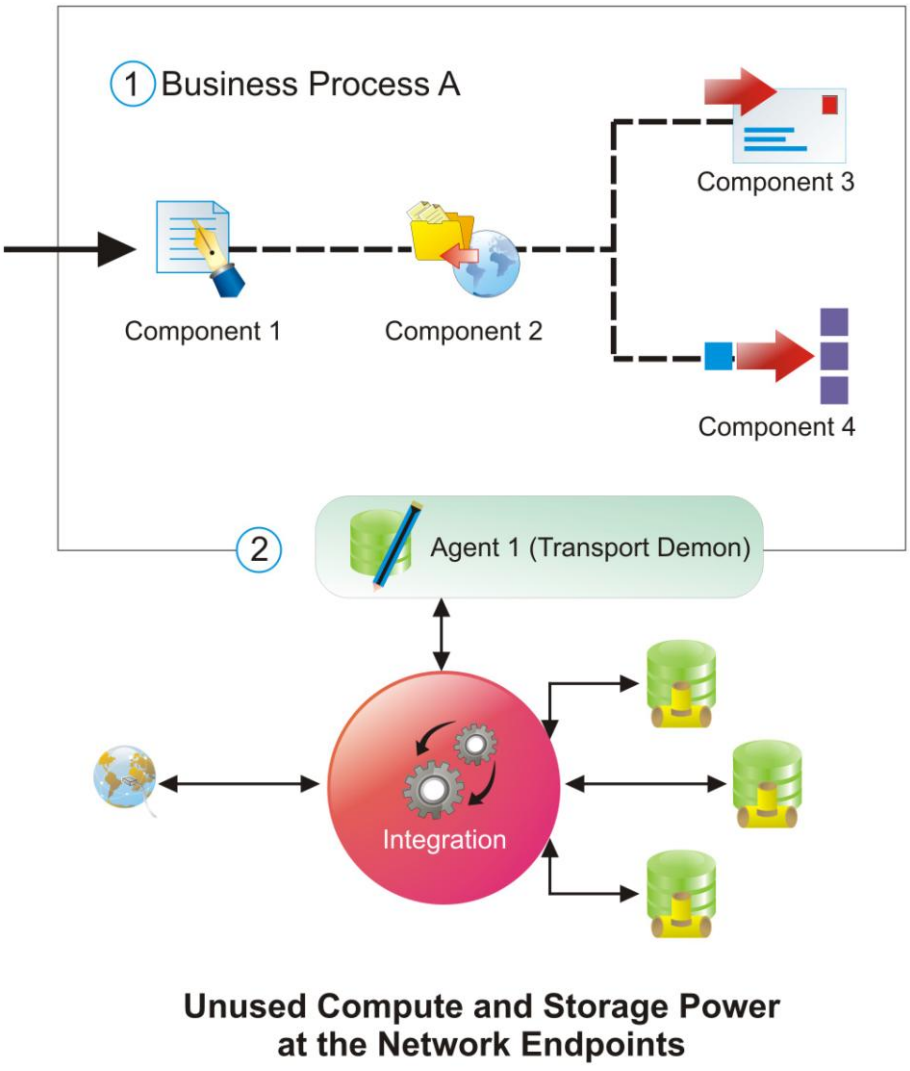

Figure 1: Incumbent composite applications do not leverage a distributed IT infrastructure

- Note 1 -Business Process A consisting of Components 1 – 4
- Note 2 -Agents, typically Transport daemons are distributed on client desktops
- Note 3 -A centralized Integration Server is mounted on a high-performance Server machine

Programmers tasked with implementing current EAI solutions have to deal with two critical areas:

Component-level programming includes the actual functionality of the component itself, for example, coding a Siebel adapter.

Routing data between components Incumbent solutions rely on a messaging bus for routing data between components, and each component contains hard-coded information about Topics and Queues, making it unsuitable for reuse in a distributed business process consisting of several applications. The component contains valuable information about the tasks that it is supposed to execute, but because of the embedded routing information, non-technical users cannot change the workflow using simple visual drag and-drop metaphors. Such a static business process enforces long delays between a need to change the business process and an implementation of the changes, at an additional cost of customized programming needed within each component.

Programmers with incumbent EAI solutions have to design and enforce a painful port nomenclature that enables inter-component routing at the global (business process) and the local (component) levels. A component-based programming model decouples the programmer from the business process integrator who needs to worry only about the routing between individual components. This simple paradigm forces a correct-by-design methodology for the integration project, since any global channel names at the integration level cannot be hard-coded into the component itself.

In order to manage the complexity of composite applications, many vendors resort to a centralized hub-and-spoke messaging and integration-bus oriented implementations. A centralized Integration Server (hub) is expected to manage the growing inter-component level programming detail. However, this architecture imposes severe performance and scalability bottlenecks, in addition to increasing complexity and deployment costs. Additionally, a centralized hub (or integration bus) becomes a single point of failure – a situation that can be remedied by adding redundant hubs, which in turn imply higher costs and management overhead. And finally, neither the hub-and-spoke nor Integration-bus oriented architectures efficiently leverage all the distributed computational power available at the end-points of an enterprise network.

Clearly, attempting to mask the low-level issues of programming components at a fine-grain level with expensive and unwieldy hub-and-spoke solutions can only lead to greater complexity and costs. Fortunately, by going back to the basics, some elegant frameworks and approaches have been suggested, and vendor-neutral consortiums, such as, have done well to suggest standards-based platform-oriented solutions.

Fiorano has assimilated rich developer-level feedback from over 300 global customers using its FioranoMQ JMS messaging servers, and worked on delivering coarse-grained component based business process solutions with Fiorano ESB (Fiorano Enterprise Service Bus) as discussed next.

Next generation Component-based Business Process Composition

Fiorano defines Component-based programming as the process of creating software applications by visually connecting objects together on a screen. Component-based programming is performed at a level of abstraction that is higher than the normal level of algorithms and data structures employed by programmers building solutions using various programming languages.

The power of synthesizing business processes from pre-built, pre-tested software components is akin to the power of using spreadsheets to create financial models.

Just as non-programmers find it easy to rapidly change spreadsheet formulae to create new numerical solutions, business managers should be able to create new business processes from existing ones by replacing software components with others from a palette of reusable software components. The result is a powerful software system that allows business processes to be created, modified, deployed and changed without needing the intervention of the IT department or any programmers.

The defining characteristics of next-generation component-based solutions are:

- **Coarse-Grained** Components consist of any computer process, written in any software
- **Language** As such, next generation Components are coarse-grained rather than fine-grained.
- **Two Primary Interfaces** Each Component has just two primary interfaces: one to read-in data each input port and another to write data to any output port. Typically a component receives data from one or more input ports, executes a task on the data and writes results on one of more output ports.
- **Independent of Data Routing** Components themselves are completely oblivious to the routing of data between component-instances. That is, there is no notion of a transport-layer data routing at the component level.
- **Standards-based (XML) Input and Output Assertions** optionally, each component may specify assertions in terms of XSD type-descriptors that are required of incoming data on each input port,

and outgoing data at each output port. Such assertions allow composite applications to be checked for correctness at 'compile time' in much the same way as strongly-typed programming languages (such as C++) support compile-time type-checking for greater productivity.

- **Built-in Security** Increasingly, role-based access control and standards-based transport layer security (HTTPS, SSL) need to be supported at the individual component and business process levels. A security administrator needs to be able to grant access rights down to each component level including disabling the creation, usage or binding of a component to a specific user, group of users or machines in the enterprise network.

A component-based architecture as described above can be used to create and deploy a business process as shown below.

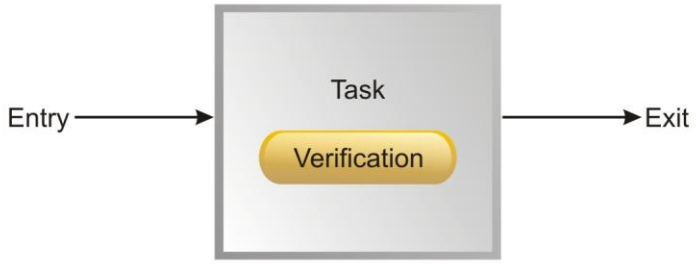

Figure 2: Component-based Business Process in Action

Components as shown above are termed as the ETVX (Entry, Task, Verification, Exit) components. Each component has pre-defined inputs; the data format is verified as defined by the XSD format, for example.

Next, a task encoded in standard programming languages such as Visual Basic, VC or .NET and the verified result is written on the output ports. Error conditions need to have a dedicated error output port.

A business process is simply composed by connecting an arbitrary number of components using their input and output ports, based on a rules-based connectivity of the routing between them. This is termed as conditional routing of data between components in the business process. Next generation Components based on the model outlined above have several benefits over the previous-generation.

- **Analysis** Model a prototype EAI project using existing business process templates and registered
- **Reusability** Instances of the same component can easily be reused in different distributed applications because each instance is completely independent of and has no dependencies on other instances of the same component. Data routing being external to the component instance itself, allowing easy reuse of components across applications.
- **Dynamic Deployment** Since each component is completely self-contained, it can be deployed automatically across one or more machines. The process of automatic deployment involves automatically transporting (after appropriate security and authentication checks) the resources of the component across the network, saving valuable time and resources compared to the manual deployment process of today.
- **Manageability** The simplified next-generation component model allows distributed business processes to be synthesized using visual tools to link different component instances on a screen. This makes it easy to dynamically change, debug and re-deploy a business process by simply replacing one component instance with another, or extending a given business process by adding another component instance.

The process of composing a component-based business process is illustrated below:

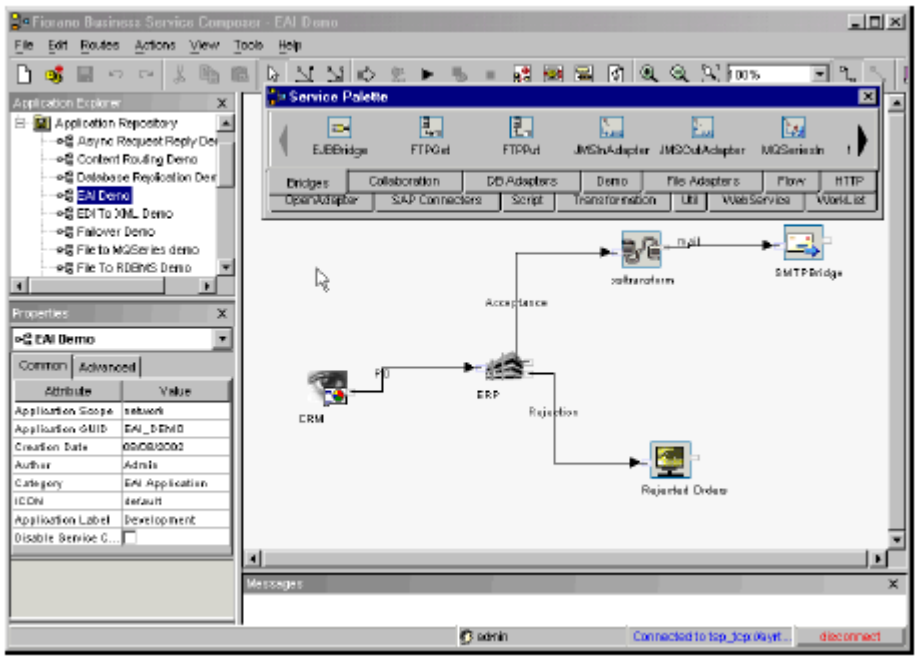

Figure 3: Composition of a component-based business process

The composition and deployment of a component-based business process can be broken down into the following well-defined tasks:

- **Analysis and Composition** Gather EAI requirements using the Fiorano Business Service Composer as in the preceding figure, simulate the business processes easily and in real-time, and create a robust prototype solution.
- **Secure Deployment** Configure the components and business processes on any remote machines within the enterprise network – inside or external to the corporate firewall. A prototype can be incrementally and securely deployed in order to manage the staging of a prototype into full- scale production.
- **Ongoing management** Once deployed, track all events and states of the business process using Fiorano tools. Given the massive scalability offered by the peer-to-peer Fiorano Peer Servers for data routing between components, tune the aggregate performance easily and in real-time. In case of peak-usage driven by external events, leverage all the network endpoints to meet flash traffic requirements in real-time, without adding any additional hardware. Trouble-shoot your Fiorano network using a rich set of distributed debugging features.
- **Easy Extensibility** Create, register and add a new component securely into an existing business process as easily as using a drag-and-drop operation on the FBSC. New service insertion or the addition of new versions of software can be easily performed on any in dual peering machine – or a set of machines – using the FBSC. As an imperative, ensure that only appropriately authorized users can perform these tasks only on designated machines using powerful role-based security models.

Easy Extensibility to emerging standards-based components

The next generation component-model outlined in the previous section is particularly suited for business-process composition using visual tools. For instance, it is possible to create a business process that accesses a web-service to retrieve a stock quote as show below.

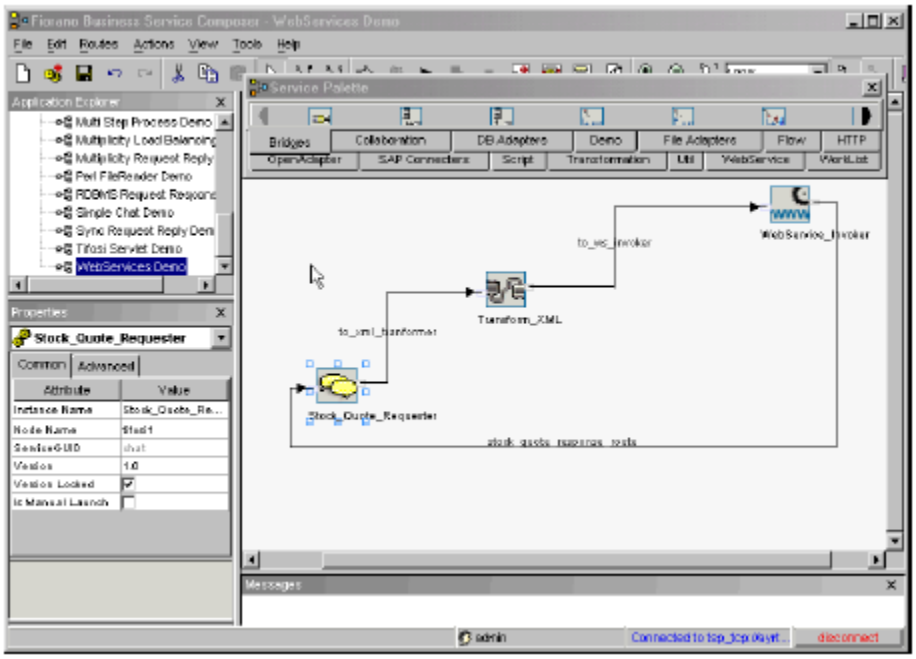

Figure 4: Extensibility to emerging standards such as Web Services

In Figure 4, the 'Stock Quote Request' component represents an input graphical user interface on a given machine, while the 'XML Transform' component performs a transformation, putting the data into the proper format for consumption by the 'Web Services: Stock Quote component'. Each of these three components can be reused in real-time within other business processes, which can also be synthesized using the same visual application composer.

Visual tools therefore allow multiple component instances to be 'loosely bound' to each other into a single 'distributed application'.

Developer defined protocols

One of the important benefits with the next-generation component-model is that it does not constrain developers to follow any complex API mandated by industry consortia. Instead, developers can easily reuse each other's components, provided that each developer exposes the interfaces of his/her components using XSD descriptors for input and output assertions. Thus, the simple use of *XML-based assertions* by component-developers allows effective and pervasive reuse of the application components in any number of applications.

Vision: A Components Exchange that lowers the TCO for Business Processes

Since components are self-contained business processes, components produced by a diverse set of developers will soon result in a very wide variety of reusable components becoming available. With the widespread availability of visual tools to compose and deploy distributed business processes from these existing components, the total cost of creating, deploying, managing and changing enterprise business processes will be reduced dramatically.

Specifically, a component-based business process enabler such as Fiorano ESB empowers business managers to compose and deploy business processes with the same ease with which they compose and execute spreadsheet.

Summary

Early approaches to component-based programming involved the componentization of programming language-level constructs such as data structures. These met with some success in the nineties, with reasonably rich sets of components becoming available to programmers. The lack of flexibility and generality of this first-generation component-model was underscored by the emergence of the Internet and the consequent creation of high-speed enterprise networks with vast amounts of computational power at each network endpoint. Specifically, it is not possible to efficiently create and deploy distributed business processes across internet-enabled networks using the older component-model.

The next-generation component model outlined in this paper maps well onto distributed, internet enabled networks and allows (through the aid of visual tools) the efficient synthesis and deployment of enterprise business processes by programmers and enterprise managers alike, while supporting a high-degree of reusability and flexibility – as summarized below.

Table 1 Benefits of Fiorano's Coarse-grained Component-based Business Processes

Enterprise Users	Benefits with Fiorano ESB	Weaknesses with current solutions
Business User	▪ Spreadsheet like simplicity ▪ Real-time Modifications of the business process ▪ Extend processes with new components when required	▪ Complex and expensive changes ▪ Fine-grain program level modifications ▪ Real-time extensibility
Component Programmer	▪ Focus on Intra-component level details (tasks, functions) ▪ Offer individual components as IP on a Services Exchange portal ▪ Testing and QA at the component level assuming standards-based interfaces	▪ Needs to worry about Intra-component AND Inter-component level data routing and assembly details ▪ Components are not reusable ▪ Testing and QA cannot be localized at component levels; adding risk into production deployments
Business Process Developer	▪ Focus on Inter-component level assembly details ▪ Remote composition of the business process ▪ Built-in Security at the component level enhances overall security ▪ Remote launch, monitoring and debugging of business process	▪ Weak security models require script overheads ▪ Real-time composition, deployment and, debugging are not possible; hence production deployments are risky, expensive and take a long time

References

- FOSTER AND C. KESSELMAN, *The Grid: Blueprint for a New Computing Infrastructure*, Morgan Kaufmann Publishers, 1995.
- J. STEWART AND H. EDWARDS, *The SIERRA framework for developing advanced parallel mechanics applications,* in Proceedings of the First Sandia Workshop on Large-Scale PDE Constrained
- Optimization, Springer Lecture Notes in Computational Science and Engineering, 2001.
- R. WESTON, J. TOWNSEND, T. EIDSON, AND R. GATES, A distributed computing environment for multidisciplinary design, in 5th AIAA/NASA/USAF/ISSMO Symposium on Multiple Disciplinary Analysis and Optimization, September 1994.
- T. EIDSON, *A Component-based Programming Model for Composite, Distributed Applications*,
- ICASE, NASA Langley Research Center, May 2001. Available on the internet at: http://www.icase.edu/Dienst/Repository/2.0/Body/ncstrl.icase/TR-2001-15/pdf
- CCA, Common Component Architecture Forum webpage, in http://www.cca-forum.org

Suggested Reading

- J. SIEGEL, CORBA*: Fundamentals and Programming*, John Wiley and Sons, ƒ C. SZYPERSKI, *Component Software: Beyond Object-Oriented Programming*, Addison- Wesley.
- Details on the Fiorano ESB architecture at: www.fiorano.com/products/fesb/products_fioranoesb.php

About Fiorano Software

Fiorano Software (www.fiorano.com) is a leading provider of enterprise class business process integration and messaging infrastructure technology. Fiorano's network-centric solutions set a new paradigm in ROI, performance, interoperability and scalability. Global leaders including Fortune 500 companies such as Boeing, British Telecom, Credit Agricole Titres, Lockheed Martin, NASA, POSCO, Qwest Communications, Schlumberger and Vodafone among others have used Fiorano technology to deploy their enterprise nervous systems.