COMPONENT-X ™ White Paper

The open standards-based Java-XML solution for worldwide computing

XML is quickly becoming the defacto-standard for enterprise level Web Services, Component-X provides drag-and-drop assembly of Java-XML components based on open standards **-** *Component-X is positioned to become the component model for XML web services.*

Component-X embraces the visual and intuitive "**drag and drop**" style of **component assembly** and configuration, which has been so successful for user interface "Widgets". Component-X applies this style of assembly to server-side business components and enables these components as Web Services supporting standard and proprietary protocols such as ebXML, HTTP, Soap, .NET, Corba, J2EE, MQ-Series, Java Messaging and others. Due to its open nature, Component-X works equally well for legacy systems, new applications and services that integrate new and legacy capabilities.

An open marketplace in Enterprise Business Components; the ability to build on and integrate "Best of Breed" application components traded in an open marketplace has been a shared vision a long time in coming. The recent advances in web services technologies, XML and emerging standards have set the stage for this to become a reality. Component-X is positioned to be the way XML business components are produced, packaged, assembled and delivered. A network of "Component-X Marketplace ™" vendors will provide trading of XML business components in on-line and direct marketplaces. Enterprises are also encouraged to implement an internal Component-X marketplace to facilitate component reuse.

"Open components for web services is one of the last key pieces to enable the vision of an open, connected enterprise and an open marketplace in business components," said Cory Casanave, President and CEO of Data Access Technologies, Inc. *"We are on the edge of the next Internet Revolution – The Connected Enterprise."*

1.0 release now available

The 1.0 release includes the Component-X Standard edition platform. This platform is available for free on: www.enterprise-component.com

Focus on business requirements:

Business today is rapid, dynamic and integrated. The enterprise needs efficient business processes that can span divisions, suppliers and customers and needs these processes deployed quickly to meet new opportunities, solve problems and respond to changes. Business partners and their information systems may be "inside" one day and spun-off the next, what was once a contractor may become a division. Companies need their business processes integrated with their suppliers, their customers across diverse systems, geography and divisions. Customers increasingly want direct access to the enterprise information system via web pages and web services. The information system has become the backbone of the modern enterprise and must be able to respond quickly, efficiently and reliable to these business initiatives.

The multi-tier web services architecture is quickly becoming accepted as the only way to facilitate such a dynamic and integrated environment. A component assembly platform is the only way to deliver these web services quickly, inexpensively and reliably.

Integration of processes, applications and services is essential to meet these business goals. The challenge here is two-fold – to get legacy systems exposed as web services and then to integrate these web services to support new business initiatives. Component-X provides two ways to integrate existing applications, as services and via Java. Due to it's open "adapter" architecture, Component-X components can work with almost any standard or proprietary distributed protocol. An adapter is written to communicate with that protocol and turn the information into XML. These adapters are Component-X components, which can be used with any other Component-X component. Basic capabilities can also be added via Java – any Java IDE can be used to create new components which use the Java native call capability to directly communicate with other applications or services.

Getting a web service will only get you half way – Each application will have it's own data structures, processes and technology requirements. Meeting new business initiatives requires integrating these web services and creating new processes and services from them. Component-X provides libraries of components for process control, mapping database information to XML, transforming and integrating document information and creating new services, applications or web pages. This library is "open ended" and will be augmented with the Component-X Marketplace. Integration and adaptation is the key to open systems.

A key advantage of drag-and-drop component assembly is simplicity. Component-X provides a visual environment that the average business developer or analyst can easily understand and use. Components are " wired" together in the visual environment and configured for the local requirements. Document and business process transformations and DBMS integration are easily instrumented using other components that are provided as part of the platform. External services or legacy systems are wrapped as components that can be used and integrated with minimal exposure to the details and complexities that may be behind the component. Components can be integrated with middleware by simply wiring middleware adapters to business components, making the entire process both business focused and intuitive.

Technologists who understand the details of the middleware and back-end services can provide technology components that provide access to these vital technologies without burdening the application assembler with their details. New components can be build from existing components, from web services or in any Java ™ IDE.

The entire Component-X platform runs interactively (without the need to "compile"), allowing components and processes to be tested, debugged and traced immediately in Component-X studio. The end result of this power and simplicity is key business needs met quickly and reliably with minimal development or deployment expense.

"Component-X represents the next generation of application development technology built on robust specifications and industry standards. This generation will provide the ability to compose enterprise applications using purchased components and quickly implement custom components from robust platform-independent specifications. This will greatly improve enterprise flexibility and the ability to re-deploy applications as new technologies and products offer economic advantage or enhanced functionality." Said Fred Cummins, Enterprise Consultant, EDS

Component-X Partners

Data Access Technologies is inviting others to participate in the Component-X platform and marketplace. The partner program is being launched to invite five types of partners:

- **End User Partners** – companies or organizations that will utilize and deploy Component-X solutions to achieve their business vision.
- **Marketplace Partners** –will offer Component-X platforms, components and related products to solution providers and end users.
- **Solution Provider Partners** –application developers and consultants who will use the Component-X platform and the Component-X Marketplace to produce or help customers produce solutions.
- **Component Developer Partners** – who will develop and market business and technology components and applications through a Component-X Marketplace. Component developers will have some type of application or technology expertise to offer customers as components.
- **Infrastructure Partners** – who will integrate the Component-X platform with their application server, web service or distributed computing infrastructure.

" We have long recognized that the development, delivery and ownership cost structure for software are incompatible with the demand for new software solutions. While PC hardware technology has consistently sustained an annual doubling of productivity and performance gains, similar significant productivity gains

in software have not materialized. The software industry needs to capitalize on the proven concepts of specialized, application-independent, encapsulated, units of functionality - components. Component-X technology from DAT is the first software development environment that enables us to describe "software ICs" and wire them together in various industry standard ways. This dramatically improves the sharing and reuse of these components thus collapsing the development cycle. In addition, the resulting application can be targeted to a wide variety of platforms and middleware, further enhancing the value of business logic embodied in the component.." Said David A Schramm, President, Standard Systems

Component-X is based on current and emerging standards:

Component-X as designed to be open and standards based - an alternative to proprietary and technology dependent solutions. Some of the standards driving Component-X are:

- **XML** (www.w3c.org) – The basis for Component-X is standard XML. Component-X components produce and consume XML to communicate and integrate. The Component-X platform also provides direct support for **XSL** transforms and adapters to use **HTT**P to transport XML messages.
- **ebXML** (www.ebxml.org) – Component-X is targeted to be the component platform for ebXML. Data Access Technologies has been instrumental in producing the "**ebXML Business Process Specification Schema**" – the specification for the way XML web services will be used for business processes. Component-X will directly support the business specification schema and provide drag-and-drop assembly of ebXML services. Direct support will be provided for ebXML infrastructures, as they become available. DAT has been a key player in the ebXML process and participated in the proof of concept.
- **Unified Modeling Language "UML"** (www.omg.org) – Component-X is based on the **"Enterprise Distributed Object Computing**" (**EDOC**) profile of UML which is currently under development in the Object Management Group. Data Access Technologies has been instrumental in producing EDOC and has been the editor for the "**Component Collaboration Architecture**" (CCA) part of EDOC which specifies how components are used recursively, to build components and applications at every level of granularity; From large enterprise components to small technology widgets. The UML profile will allow Component-X modes to be interchanged with and between generic and Component-X specific UML tools. . Due to the active participation of Cory Casanave from Data Access Technologies and Karsten Riemer from Sun Microsystems, the OMG EDOC and ebXML standards will be interoperable – allowing EDOC components to implement ebXML business processes.
- **J2EE ™** (www.sun.com), **Corba**, (www.omg.org), **Soap** (http://www.microsoft.com/)– Due to it's open architecture, Component-X can support a variety of standard and proprietary middleware protocols and application server platforms with add-on adapters. Component-X components are not bound to any middleware or container. An open adapter architecture provides the connection from a business component to a specific middleware technology. Data Access Technologies and other Component-X partners will be filling out the adapter component library over the next few months.

Pricing and Business Model

The "Component-X component builders' platform - standard edition" is provided at no charge by Data Access Technologies. Deployment licenses for components, services or applications, which include Data Access Technologies infrastructure or components are available from $800 to $25,000 (Current USA List price) per server. Data Access also requires that Component-X components containing their components or infrastructure be sold through an authorized Component-X marketplace, which charge a royalty on each sale (The amount of the royalty is up to the Marketplace Partner). The authoring component developer will price business Components and Technology components - business components are expected to be in the same price categories as business applications.

Component-X Technology

What is a Component-X component?

The basis of Component-X is, of course, the component. A component provides some process or service (business or technology) that communicates with other components via “ports”. The information flowing into and/or out of these ports is always XML. It may be a simple XML element or a complex protocol implementing a business process, but the essential unit of interaction is always XML.

Components are executed inside some kind of container. That container may either be some kind of application server *or another component*. This ability for components to “compose” other components is one of the essential features and capabilities of Component-X. This allows building components from components and then deploying “top level” components on application servers.

There are two things you do with a component when you use it – you “wire” it to other components and you configure it. For Example, when you wire a “pricing component” to a “soap adapter” you have just made that pricing component into a web service. When you wire that same pricing component into the “inside” of an order generator component you have just used it as a component of the order generator, but not necessarily as an independent web service.

Components can also be “configured” with properties. Configuration allows the behavior of a component to be specialized based on how it is used within its container. For example, the pricing component may be configured to use a given pricing model or the location of a price database resource.

While many interactions between components are a single messages or events – some require a two-way “conversation”. This is provided for with “interfaces” that composes multiple XML interactions. Future versions will also allow interactions to be choreographed in accordance with the ebXML business process specification.

These central concepts; Components having ports which produce and consume XML and can be configured and wired together form the basis of the Component-X paradigm. This style of component assembly is based on the in-progress OMG “Component Collaboration Architecture” standard.

What is a Web Service?

Web services allow systems, applications and people to interact on the web. The standard web page is a great way for an individual to interact with an enterprise system, providing both information and applications. But, web pages are not suited for another computer system or application to interact – making it useless as a means to integrate applications, customers and suppliers *in an automated way*. To support automated integration a service must be provided that is suitable for this kind of system to system interaction. For example, you may have a web page with a price list. But, if a customer wants to automatically integrate that price list into their ordering system, how do the do it? With a web service. Applications servers for web page support also use web services.

Capabilities for systems to interact with systems have been around for quite some time – these are the “legacy protocols”. What differentiates a web service is that it requires no special infrastructure to use it – all that is required is off-the-shelf web capabilities. This makes it easy and inexpensive for a client to use a web service. Web services are also very “loosely coupled” and make very few assumptions about the system or technology on the other end, usually exchanging data as “Documents”. Finally, web services tend to not require distributed transactions, since distributed transaction support requires a lot of infrastructure. So, web services provide many of the capabilities of the pre-web technologies without the heavyweight infrastructures.

The end result of this is that web services will become the backbone of integration, inside and outside the enterprise.

What is the Component-X Studio ™?

Component-X studio provides the development and configuration environment for Component-X. It comes with a built-in component library and the ability to build new components and connect existing components together – all visually.

Component-X studio also includes the capability to import and define new XML types and the basic platform includes a very useful set of components to get you started (See below). Also included with the studio is an XML editor, debug and trace facility and the ability to watch and produce component events.

The studio is also a "lite" application server – you can send messages to components to see how they respond, trace messages and debug the application. Component-X Studio is the "IDE" for Component-X. Using the studio is fast, effective, open *and fun*!

What is in the standard Component Library?

The standard edition platform included the studio and a component library. The component library is intended to support building and integrating web services, it does not include any business components. The types of components found in the standard platform include:

- **Control components** – to define the business process by routing information among components. This includes conditional logic for making choices, iterator for sets, expressions (XPATH and Java) and, of course, error handling.
- **Filters** – for transforming information in a variety of ways, including XSL, Java Expressions, Summation and the ability to do document transformations visually, by wiring components.
- **Database** – for mapping XML document types to SQL database systems. This includes the capability to query and update complex XML documents based on a XML request.
- **Adapters** – Connect components to distributed protocols. The "First look" release only includes adapters for HTTP. The next revision will include Soap and Java Messaging (JMS) support. Additional adapters will be provided in the Component-X Marketplace.
- **Utility Components** – for basic processing needs and system interfacing.
- **Templates** – for business process elements and common component "patterns"

It is important to recognize that assembling components does not need to do everything. Things that are better done in Java should be done in Java and made into a component. Existing functionality should be wrapped and exposed as components. Legacy applications will provide much of the base functionality of a web service. The Component-X Marketplace will provide a reusable library of advanced functionality and legacy wrappers produced in a variety of ways.

How do you deploy components?

Component-X business components are not an "application server" or "web service" on their own. A Component-X deployment platform is built into an application server. It is the combination of a set of components and an application server that creates the web services. The first application servers supported by Component-X will be J2EE ™ compliant servers, others will be supported in the future – based on user requirements.

The component developer provides a Component-X deployment package as a "CAR" file (Component Archive – a kind of JAR file). This package is installed in the application server and configured as required. The components will then respond to events coming from external systems through the application server.

Since the Component-X platform is pure Java, it may be installed on any execution environment. Component-X is designed to support multiple distributed technologies and to integrate into multiple "application server" environments – in this way it complements rather than competes with most existing infrastructure. Note that most other tools lock you in to a specific technology.

What are Adapters?

Many existing component models are bound to a particular kind of technology, such as Corba components, EJB components or COM components. It is natural and necessary to have such technology components. However, technologies change – particularly distributed technologies. And, companies frequently need to support multiple technologies and have the parts integrate. Sometimes you don't even want to distribute components, but you do want to be able to assemble them and keep them as independent "parts" of the system. For these reasons, Component-X components do not "distribute" directly. Component-X components communicate by sending XML elements over "local" Java events, within a single "Virtual Machine". While these components are not directly distributed, the are "distribution capable" since they communicate via XML events.

Adapters are used to connect business components to a distribution technology, such as HTTP, Soap, ebXML, Corba or MQ-Series (the list is open, use your imagination). Adapters are just Component-X components – so they are "wired" to the business components like any other component. Once a business component is wired to an adapter it becomes a web service. But, unlike technology dependent components, they can become another kind of web service by a simple re-wiring.

This separation of distribution concerns from business logic has proven to be very successful. Adding adapters is very ease and makes the platform totally open. In addition, components can be used for "smaller things" – like process flow control or document transformation, without suffering the overhead of a distributed system. Component-X is Very light weight and efficient.

About Data Access Technologies, Inc.

Data Access Technologies (DAT) " Where Business Meets Technology" provides business-focused products and services for enterprise computing. Their specialties are distributed Internet systems, B2B, E-Commerce and automated development tools. DAT is a leader in standards efforts required to make enterprise components open and interoperable.

DAT is chartered with developing a new generation of products and services for enterprise integration based on distributed object and web technologies, They provide products for developers, technologies to industry business partners and services to enterprise clients. DAT is privately held.

Data Access Technologies is a "spin off" of Data Access Corporation, a privately held corporation with almost 20 years experience in software development tools. Data Access Corporations primary product, DataFlex ™, (an application development system) has sold over 400,000 licenses worldwide and has produced thousands of custom and packaged applications. See: www.dataaccess.com

Much of the technology offered by Data Access Technologies is a direct result of a grant from the Advanced Technology Program of the US National Institute of Standards and Technology. This technology development grant specifically targets the automated development of enterprise components.

Contact

Data Access Technologies
(305) 238-0012
info@enterprise-component.com
www.enterprise-component.com