Softonomy™

Web Services

28th April 2002

Softonomy Ltd
Growcorp Innovation Centre, 3015 Lake Drive, Citywest, Dublin 24, Ireland.

For queries or further information, you can reach us by:

Email:	**info@softonomy.com**
Telephone:	+353 (0)86 821 0917
Fax:	+353 (0)1 284 6381

Our website is located at: **www.softonomy.com**

Table of Contents

1. Introduction

Softonomy Ô, a company launched in 2001 and headquartered in Dublin Ireland, are builders of **Software Solutions** for businesses. We design and develop product-class software solutions, each one **tailored** to meet the specific needs of our Clients.

We build and implement solutions via an efficient no-nonsense software development approach. Our customised software can be based on pre-built software and business components and our solutions enable Clients to realise real improvements in their business performance and deliver a real return on investment.

What we provide to our Clients is powerful, yet really quite straightforward - we build tailored software solutions, that:

- provide a custom and **optimal solution** for each Client's business
- deliver a **return on investment**
- **integrate** smoothly with users, business processes and systems
- facilitate enhancement when/if needed to **meet changing business demands**
- are **business driven** and focussed
- may be based on pre-built **cost-effective** software and business components
- utilise **proven technologies**

We provide expertise for all the necessary steps when creating software solutions. We work closely with our Clients and become involved in the success of the solution. We use streamlined techniques to capture our Clients requirements, to design the appropriate solution framework, to iteratively develop and test the solution, to provide training and knowledge transfer, and where requested, change and project management.

Our software solutions are used in a variety of business situations, such as:

- **Strategic** or **Management** information systems (incl. real-time dashboards)
- **Operational** information systems and Process Driving/Workflow applications
- **Wireless** applications (mobile/cellular, 3G)
- **Web**, **Internet**, **E-business** and **Digital Media** applications
- **Integration** solutions, between front-ends, back-ends, ERP, CRM, SCM
- **Add-on** solutions for ERP, CRM & SCM packages
- **Informational** Systems (databases, data warehouses, knowledge systems)
- **Security** or **Risk** applications
- **Compliance** applications

This white paper on **Web Services** outlines how software solutions from **Softonomy** relate to both the proven component-based software paradigm and the emerging so-called Web Services area. It outlines in plain business terms what Web Services are, the current stage of the development of the associated technologies, what way this area is likely to develop and how businesses can exploit the advantages that this technology may bring to software.

2. Softonomy Research Programme

We have been involved with investigating the vast bulk of software techniques, methodologies, languages, architectures and technologies for over 20 years. We have used all relevant information systems and software development paradigms.

We have developed real-world, robust and functionally complex software applications for major Clients. Many of these software applications have been used internationally and subjected to demanding users and business environments. We've also developed applications for less complex businesses, but nonetheless equally demanding in terms of expectations and quality requirements.

The Clients we have provided solutions to span across a range of sectors such as Financial Services, Pharmaceutical, Telecoms and Hi-Tech. Our collective software development experiences have helped us to understand what works and what doesn't.

Over 2 years ago, we embarked on a continual **research programme** that will sustain our understanding of software and technologies and their linkage to business benefit. During this ongoing programme we are conducting extensive secondary research using our access to academic research sources in Ireland, the US and elsewhere and we complement this by our direct primary research (*see Figure 1*).

The data we collect is analysed and selectively distributed and published on our website: www.softonomy.com. Our research programme allows us to know how software and technologies can be optimally applied for business exploitation. We follow and evaluate all upcoming trends and developments on behalf of our Clients and for the upkeep of our commitment to deliver product-class and effective software.

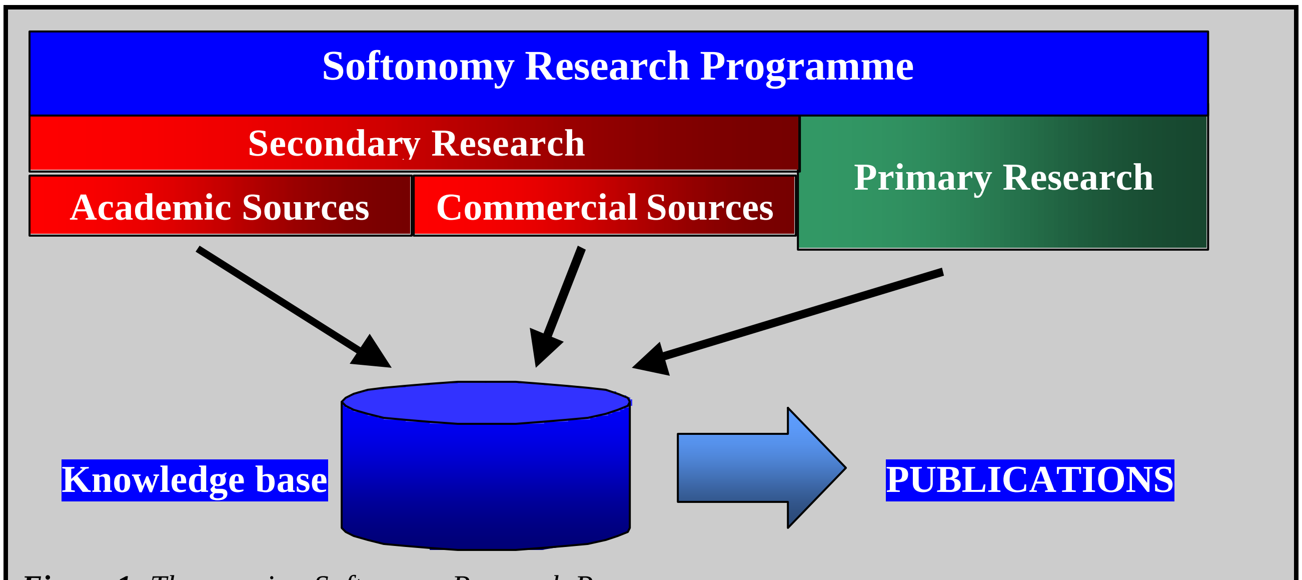

Figure 1: *The ongoing Softonomy Research Programme*

3. Opportunities for Businesses

Our extensive research is showing that there is still very much to automate within businesses. In 2001, the *American Management Association* reported that one of the most important skills businesses need today is the ability to use information to address business challenges. We concur that having the right information, at the right level, delivered to the right function and at the right time is key to operating an efficient and performing business in a dynamic market environment.

Many Clients are using this as a **business opportunity** to create increased automation and information system "intelligence" that allows businesses to be more effective and to realise efficiencies when interacting with **Customers**, when interfacing with **Suppliers** and **Partners**, and when executing business processes **internally** (*see Figure 2*).

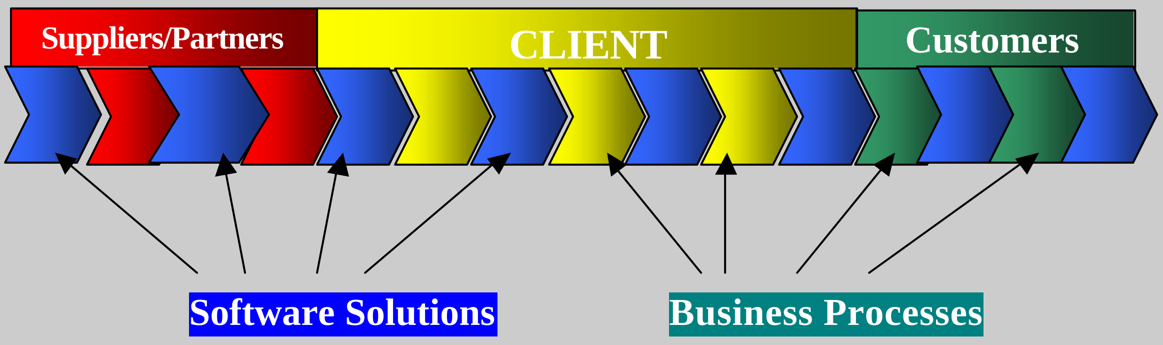

***Figure 2**: Relationship between Software Solutions & Business Processes* © Copyright 2002, Softonomy Ltd

Our Clients are using new software applications to support new or **enhance business processes**, to streamline business operations, to improve the performance of interactions with their Customers, to optimise the supply chain, to improve margins and profitability and to generate and support revenues. Technology is a key tool to enhancing businesses and maintaining competitiveness. Targeted and focussed investments in software solutions today are proven to deliver results and returns in the short term.

4. Component-based Software

The background to **component-based software** and its successful adoption is outlined in: "***The Emergence of Components***". This can be downloaded from the white papers section at www.softonomy.com. Some key points are:

- the idea that a software provider can provide **solutions via the re-use of components** is not a new concept. Ever since "shared software libraries" were put forward by *Doug McIlroy* (1968), and then popularly re-iterated with "object integrated circuits" by *Brad Cox* (1986), software re-use has been recognised as an attractive idea with a direct payoff. Building software solutions from tested, proven, high-quality software components certainly saves both the costs and the time of redoing work and overall enhances solutions.
- the **technical infrastructure** framework (*see Figure 3*) necessary to support component-based development is a relative recent advance maturing in the last couple of years. For example, platforms such as J2EE and .NET both support complex components and the emergence of standard protocols such as XML has made it easier for builders to specify components and put them together.
- Clients are aware that packaged software is cost-effective but they realise that they have **unique business attributes** that distinguishes them from their competitors and towards their Customers. Businesses know that **tailored solutions** are ideal but they are unable to acquire them due to high development and support costs *when* those custom applications are built from scratch. However, with component-based development it is not necessary to build software solutions from nothing. The re-use of components allows Clients to acquire tailored solutions cost effectively and makes good business sense.

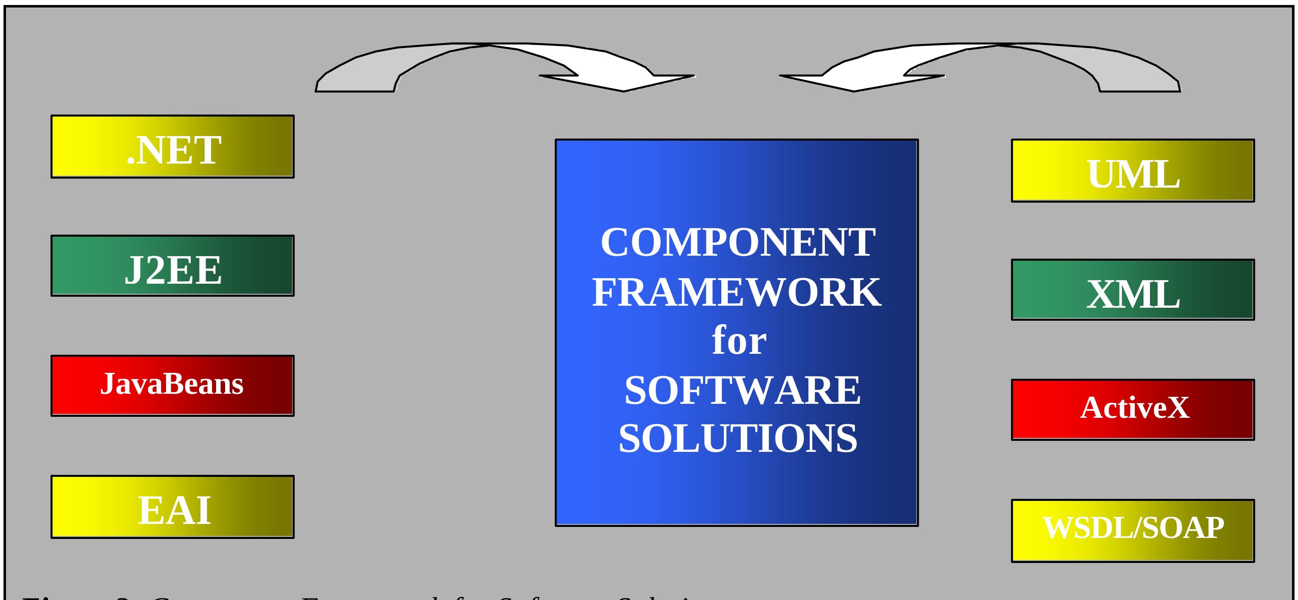

***Figure 3**: Component Framework for Software Solutions*

5. Web Services

This section outlines in plain business terms what **Web Services** are, the current stage of their development, which way Web Services are likely to develop and how businesses can exploit the advantages that this technology may bring.

5.1 What are Web Services?

There is much being talked about in the press and within the computer industry about Web Services. The term has a particular meaning in 2002 (this should not be confused with the same term that was being used in 1997, which related to delivering services to customers over the web). In fact, it could be said that the current definition of Web Services is somewhat of a *misnomer* as it is attempting to define a set of technologies which can be used to build software applications, many of which will have nothing to do with HTML and the World Wide Web. So, what exactly are Web Services? In its simplest definition, Web Services are "**a set of standards that can be used to build software applications**". These standards are:

- **WSDL** (web services description language
- **SOAP** (simple object access protocol)
- **XML** (eXtensible mark-up language)
- **UDDI** (universal description, discovery, integration)

And there will be others! The catch is that the "standards" are very new, with incomplete functionality and are still evolving, so for all intents and purposes they are *not* finalised standards and are far from being mature and proven.

The Web Services application model is a very simple model with a **client** and a **server** and a **directory**. In fact, this model is so simple that it is the same as that seen in the early 1980's - not exactly a software revolution, but one that definitely works.

The basic steps (*see Fig 4*) when running a Web Services application are as follows:

- a **web service** registers itself with a **UDDI** directory
- a **web service client** queries a **UDDI** directory to locate a **web service**
- a **web service client** accesses the **WSDL** description of a **web service**
- a **web service client** invokes a **web service** via **SOAP/XML**

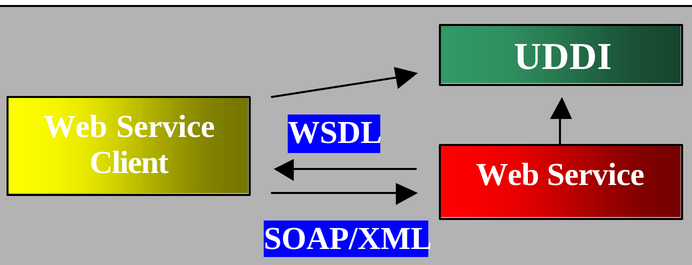

***Figure 4**: architecture of Web Services*

5.2 Current State of Development

Web Services are based on standards. However, the "standards" that they are based on are neither fully functional for the requirements that businesses need nor are they fully fledged standards. This becomes problematic for businesses that are looking for deployable technologies and is exasperated when technology supply companies are talking about Web Services standards when they are **"immature" standards** in the true sense of the word. Establishing standards can be a slow process and although there is continuous progress the standards issue will need to be rectified before there will be widespread deployment of Web Services-based solutions.

Where and how can Web Services-based solutions be deployed? In general, such types of technologies can be used to integrate *islands* of applications, data and information. This is the classic **enterprise application integration (EAI)** type of deployment. A Web Services-compatible interface can be built for each *island*, a series of which is sometimes referred to as a *software bus*. Then, when each *island* wants to communicate with each other (ie: when businesses want to build software applications that will use data across two or more sets of *islands*) the application talks to the single *software bus* integration point that all *islands* have exposed.

Web Services can be used to expose those integration points in what is termed **Services Oriented Architecture (SOA).** There is however nothing new or unique about this approach as current technologies are being used to do the same thing ie: build a single interface point for each *island* and use a SOA framework (see Figure 5).

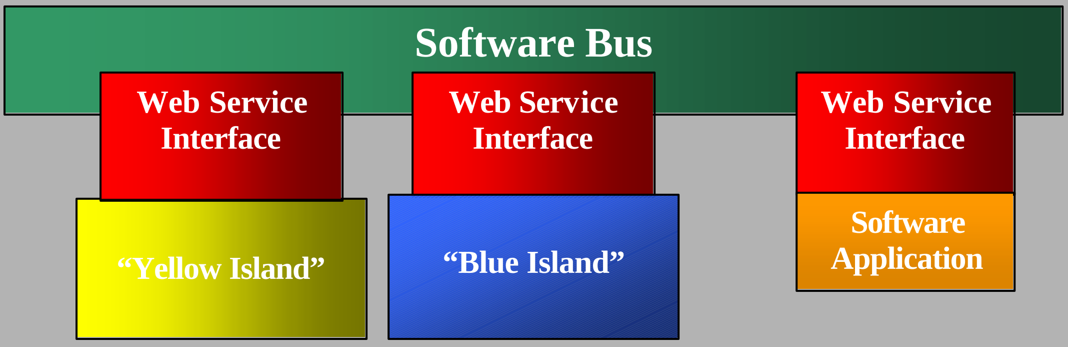

***Figure 5**: Service Oriented Architecture using Web Services*

So are Web Services going to solve problems in a new way? No - not at all. In fact the problems that Web Services will typically be used to solve are currently being solved by technologies that are much more mature and are more functionally complete. Technologies such as **J2EE** and **CORBA** have been widely and successfully deployed to solve such problems and will continue to do so in the near term. Therefore, Web Services can be used as a new tool for integration, albeit currently an immature tool, but in reality it can't be used to solve business problems in a new way.

5.3 Future Direction

- There are aspects that are supporting the growth of Web Services.

- There is **key industry player support**, such as from the likes of Microsoft, IBM and HP. Microsoft are fully endorsing Web Services with their push of **.NET**

- However there is **not universal support** and it is unclear if Sun Microsystems are being *allowed into* the key consortium on an equal footing (as of the time of this writing). However, Sun's technologies have moved in a direction that supports Web Services so they have not become isolated.

- Analysts expect that the **standards will evolve** to maturity as all the big industry players are giving it their backing and financial support. For example, standards that are not stateful will become so.

- **SOAP** is seen by some technologists to be *better* at connecting endpoints (ie: building software applications) than RPC, RMI and DCOM.

- However, much of the Web Services standards are **lacking in full functionality** and although there are parts of Security being implemented with XML Signature, XML Encryption and WS-Security, there is still a long way to go.

- Interoperability of currently available products is a problem (ie: tools from vendors do not work together!)

Web Services will **continue to develop** and make progress. Although there is sufficient industry momentum behind Web Services to allow it to become a dominant technology, this will be a several if not many years down the line.

5.4 Business Exploitation

Currently we are at the early stages of development of Web Services. Using the classic model of *new product take-up*, we are just moving into the **early adopters** stage (see Figure 6).

At most this will interest 10% of potential businesses. Furthermore, the initial problems that are being tackled are large-scale integration problems for large businesses with distinct *islands* of data, applications and information. These large businesses only constitute about 1% of all companies. Thus we are looking at a solution which is likely to be targeted for use by a mere **0.1% of all companies**, not exactly a technology that is going to sweep across all industry.

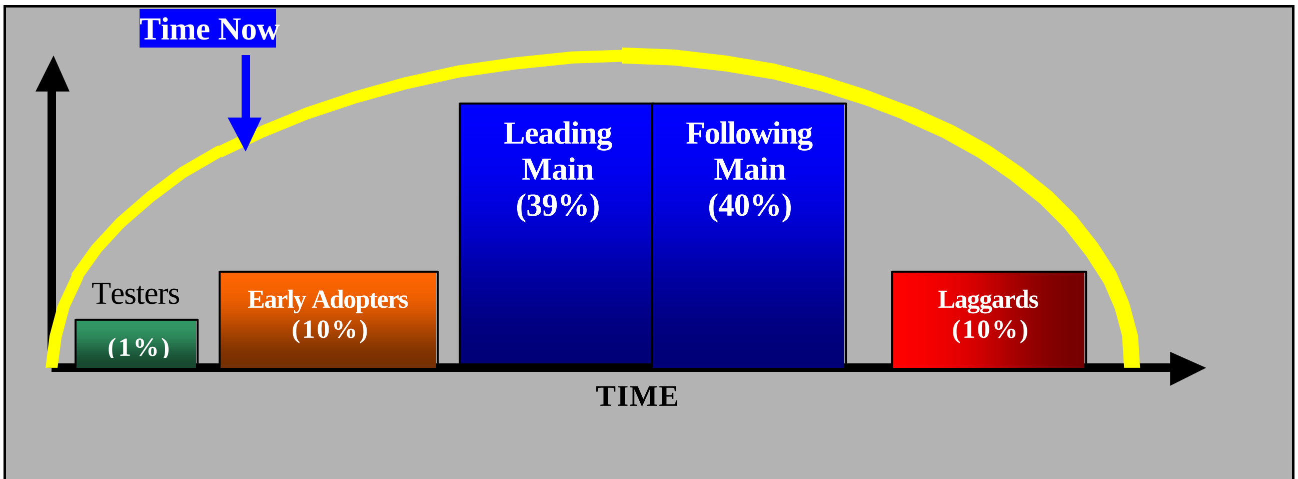

***Figure 6**: Adoption profile of Web Services* © Copyright 2002, Softonomy Ltd.

However, this picture can be somewhat misleading. In practice, businesses start "testing" and working with new technologies as they become available. Suitable **pilot projects** are selected to try out any new technology. In some cases, these projects aren't successful when compared with other implementations and in some situations can even be cancelled. But in many situations, the new technologies are tested with success and the pilot project is the first of many that will use the new technology for deployment.

This is where **Softonomy** see some **take-up of Web Services** technologies in the short term. Businesses, whether large or small, early adopters or from the main groups, will attempt to use a Web Services-based application to solve a particular business problem. The business problem will be small enough to have a low impact should things go wrong, but will be functionally wide to test Web Services technologies as a solution basis. The sooner the "standards" become official or de facto standards, the earlier businesses can start building Web Services-based software solutions.

5. Web Services for EAI

This section outlines in more detail the pro's and con's associated with **Web Services** for **EAI (enterprise application integration)** and compares it to the technologies that are filling this role today.

5.1 Overview

There is no right way or wrong way to do EAI / Integration. Currently, EAI is achieved with **middleware**, **application adapters** and **custom software**. It is a big business globally and is estimated to consume about 25% of annual IT budgets. EAI is also complicated and although several technologies have taken the lead in adoption when implementing EAI solutions (eg: Java for the language), the technological challenge remains daunting. Each vendor has its own method of bundling data and exposing application interfaces and EAI architects struggle with selecting middleware layers that can talk to all applications. In large environments, several middleware layers may have to be selected.

IBM and Microsoft have an aggressive approach to add Web Services to their software products and they are pushing these developments into their tools that can be used for EAI. IBM also will want to push their consulting services for implementing such EAI solutions. It is believed that with Web Services, more businesses will attempt EAI solutions. However, **Java is the *lingua franca*** and the current "glue" for EAI and many vendors' products are solidly based on Java. So, to some extent the entrance of Web Services and the new vendor offerings are challenging the traditional EAI vendors.

Another challenge with Web Services is that Clients are expecting their EAI costs to be reduced. This won't happen in the short-term as applications need to be adapted to work in a Web Services way. Also, current EAI solutions include security, data integrity and guaranteed delivery and this functionality is not included as yet in the Web Services standards. In most sites, the **current EAI technologies are working** and there will be inertia to dramatic changes.

However, new projects at any current EAI site can start some preparation for Web Services. XML can be used for data exchange between applications. XML vocabularies can be defined and potential Web Services can be identified.

The hype surrounding Web Services needs to be tamed with the business realities of EAI. Web Services will not wipe out the current EAI solutions and both sets of tools will evolve side by side over the coming years. For both, EAI is an expensive and to some extent a continuing process - this is a challenge to both "sides".

5.2 Microsoft's Approach: .NET

Microsoft have adopted a Web Services-based EAI strategy within their **.NET** framework. They are promising ease of integration using Web Services as the technology basis. They are enticing current Java sites to convert to .NET, with their **Visual Studio** integrated development environment and are including a new Java-like language called **C#** (c-sharp) and a run-time layer called **CLR** that supports many languages.

However, both Java and current Windows developers face a stiff challenge in porting their current applications to .NET. Re-building connections using .NET to external applications is a non-trivial task. IT departments will have to ask themselves why should they re-do this work that is already done, not to mention the costs to the business.

A key aspect of the Microsoft slant on Web Services is that it will allow Client's internal developers to build the EAI solutions themselves, as it is simpler. However, although Web Services sets a standard (incomplete!) way for doing things and for tying applications and data together this is not enough to reduce costs nor necessarily enough to make things simpler. There is no empirical evidence thus far that a Web Services approach to EAI will be any simpler or cheaper than the current methods.

Until the Web Services standards mature, there will be difficulties when implementing solutions with .NET that must integrate with non-.NET Web Services. However, many software application developers are likely to add .NET Web Services to their applications. It will be these developers and Clients that will force the Java and the .NET camps to come together on the standards so that there can be interoperable Web Services. This is after all, what a standardised EAI platform is all about.

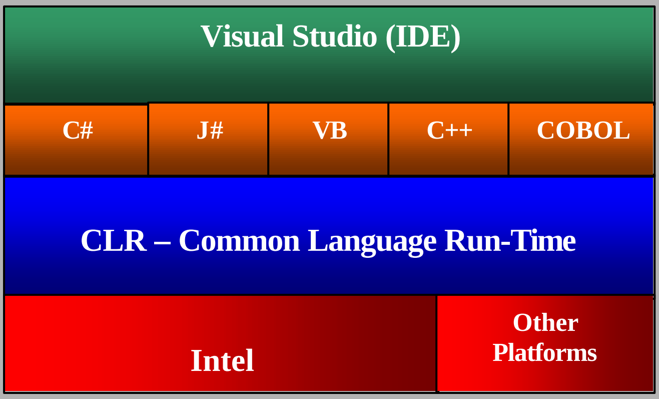

***Figure 6**: Microsoft's .NET*

5.2 Sun Microsystems's Approach: J2EE

Sun has brought us **J2EE (Java2 Enterprise Edition)** as a way to integrate enterprise applications as well as linking them to the web. The J2EE family with **Java**, **JSP's**, **Servlets**, **EJB's**, **J2EE** services and **JCA** adapters is well suited to building EAI solutions. Sun have added to the J2EE set by including support for Web Services, with its support for XML, SOAP, UDDI and ebXML.

However, the future of Java as the bedrock for Web Services is fraught with daunting issues. The Java standards authorities have not announced a track of developments for enhancing J2EE as a Web Services solution. J2EE can be used without Web Services to build EAI applications and is currently popularly used to do so today. However, many of the early adopters of Web Services, being J2EE sites are using Sun's tools as a foray into the Web Services world. There are of course issues, such as lack of tied-down security, but this is to be expected with Web Services at this stage to some extent.

J2EE also has a lot of support from IBM and other companies, such as Oracle and BEA. It has in-built support for n-tier, scalable, distributed computing and includes support for transactions and messaging. J2EE is compelling in that it is solving the real-world problems today. Additionally, support has been added for the key areas of Web Services, such as **XML** (JAX, JAXP, JAXB), **SOAP** (JAXM) and **UDDI** (JAXR).

Third-party tools for building Web Services solutions through Java are also available from companies such as Iona, Borland and Cape Clear. J2EE will be seen by many as a platform of choice for developing Web Services applications as it can expose interfaces in existing Java components requiring neither re-architecture nor substantial modification to code. As such, J2EE has the early advantage over .NET in terms of Web Services and has the foothold in the current EAI space.

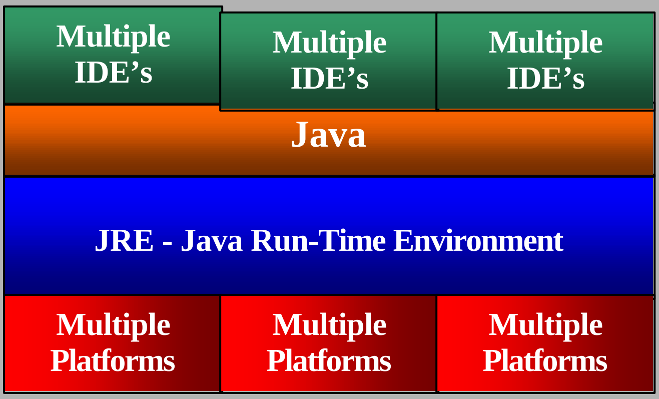

***Figure 6**: Sun's J2EE*

6. EAI

This section outlines **EAI (enterprise application integration)** and its relationship with other technologies and with **components**.

6.1 Layers of Integration

The key to developing an EAI solution architecture is recognizing that there are multiple layers of integration, each with its own requirements. Successful implementations of scalable, reliable and cost-effective solutions depend on the standards and methodologies defined for each layer. Integration typically must take place:

- **within an application**
- **between applications** (intra enterprise)
- **between enterprises** (inter enterprise)
- **direct with customers**

By covering these four areas, you'll have an architecture that's a complete covering of the main integration requirements.

6.2 Component-Based Development

In most situations, **component-based development (CBD)** is the right method to use when building such applications. This allows for well-defined, proven reusable components and can significantly reduce the cost of building solutions. Microsoft with its **COM**, COM+, DCOM component object model is well suited for such purposes. **CORBA** (Common Object Request Broker Architecture) as provided via several products is also a good fit, as is **Enterprise JavaBeans (EJB)**. Components written to one of these technologies can be easily made to work and interoperate. However, mixing these technologies will necessitate yet another layer of integration.

6.2 Application-to-Application Integration

For application-to-application integration solutions running on the same system, COM can be used. However, running applications on the same system is not a typical requirement. Usually, applications run on separate systems and on different environments. Both CORBA and EJB can tackle this situation better, but not perfectly. Although CORBA is a standard the various ORB's don't work very well with each other and can be difficult to implement. EJB isn't designed to work with either COM or CORBA.

Most application-to-application integration is not as simple as having one component communicate with another solely via EJB, COM or CORBA. Messaging middleware products from IBM, Microsoft and BEA are used to handle the complexities of exchanging information within a distributed environment. They use the concept of a guaranteed message delivery to handle asynchronous communications. A message broker is a solid solution. It provides the means to bridge the *islands* with routing, content translation, flow control and brokering.

Architects are building interface engines so that applications need no longer communicate with applications directly, but instead communicate with an interface engine using multi-use wrappers and I/O exchange API's. This is referred to as a **loosely coupled** solution.

6.3 Enterprise-to-Enterprise Integration

When sharing information between enterprises, integration problems become more difficult. Each enterprise has its own set of technologies and standards. Loosely coupled components can communicate using XML, such as Microsoft's BizTalk server. COM can be used to invoke any interface defined within an interface engine. Using Java to write servlets and EJBs that communicate with back-end systems is also straightforward. A loosely coupled architecture (see Figure 7) has the following flow:

- App1 talks to its interface engine, IE1, via its wrapper, W1
- IE1 talks to its web gateway, WG1
- WG1 talks to web gateway WG2
- WG2 talks to the interface engine, IE2
- IE2 talks to App2 via its wrapper, W2

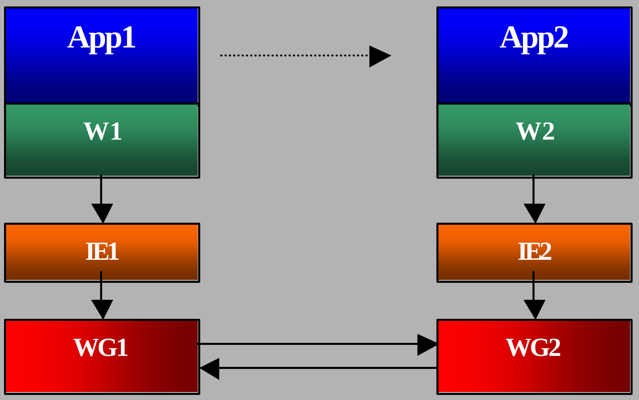

***Figure 7**: Loosely Coupled Integration Architecture* © Copyright 2002, Softonomy Ltd.

As applications change or are replaced the only need is to change the wrappers and interface engine definitions. Applications, whether local or externally owned, are not tightly bound to each other. The interface engine and wrappers provide an insulation layer that ensures application integrity and significantly reduces the costs.

CORBA, COM and EJB are good for building new component-based applications but they're not particularly suited for all layers of integration. Middleware products provide a transport mechanism for a loose coupling, but can be difficult to use and lack the typical features of a message broker. Web gateways are a means of delivering information but they lack facilities to integrate with the back-end. EAI is not easy!

7. Conclusion

Web Services are an emerging set of technologies. Currently, they are generating more hype than building actual software solutions for businesses. But that is not to say that applications cannot be built today with whats available.

One way to build applications using Web Services is when collecting an array of information from a disparate set of data sources. However, such a set of informational sources will by definition have a disparate set of data “owners”. This is likely to lead to regular data source changes or degradation in data.

The main use for Web Services is in integrating datasets from various applications within and towards other enterprises (such as EAI solutions). As such, **Web Services can be used as a type of middleware**, a term which is not very descriptive as it covers a range of differing technologies.

To implement an integrated application using Web Services, it will be necessary to change each affected application, which in a system with many applications is likely to be a non-trivial task. Of course, integration is a worthwhile cause, and Web Services is one way to do it, but it is only one of several ways to go about it. And as it requires arguably more changes to the data sources themselves, it is likely that it will take a while to catch on. **Web Services can be used for small projects and pilot projects today**.

Adherence to Web Services-based solutions will grow as the “standards” grow into real standards! This will allow application development teams to build applications in a consistent manner and data source providers to build adapters for their systems so that they can offer their data and information in a consistent and portable way. **Web Services are not yet “prime-time”** but they are a promising set of technologies that need to be watched.

Further Information

For further information on **Softonomy** and other white papers, please refer to our website located at www.softonomy.com

If you have any questions, please send an email to: info@softonomy.com

Alternatively, contact us as follows:

Tel:	+353 (0)86 821 0917
Fax:	+353 (0)1 284 6381

We look forward to hearing from you.

The key features of the **Software Solutions** provided by **Softonomy** are that:

1) we **build software**, first and foremost
2) we **work closely with Clients**, delivering optimum solutions for the business
3) we offer a **return on investment**
4) we build **tailored** solutions, iteratively
5) we provide support, maintenance, and **a path for enhancements**
6) we provide a **business-led** approach to software and technology
7) we **re-use** pre-built proven software **components** where possible
8) we are **practical about technologies**
9) we provide **skilled software development resources**

Softonomy focus on software development to bring our Clients cost effectiveness and efficiency, technology flexibility, effective software solutions and to provide a future solution path.