ORGANIZATIONAL STRUCTURE AND WEB SERVICES: A REALITY CHECK

Kevin Wilkinson

ABSTRACT

An argument is mounted that to be a virtual organization is to share a network of overlapping but discontinuous resemblances. The characteristics most often used to classify organizations as virtual are discussed. Accounts are shown to vary on issues such as stability and longevity, but agility is the characteristic most easily associated with virtual organizations because ICT is often used in an innovative way to quickly gain a competitive advantage.

Recent developments in ICT are examined that support collaborative arrangements. Web services, Grid computing, and the Semantic Web are shown to be converging in ways that should make the benefits of virtual organizations readily available. A realistic assessment is made of the opportunities and threats posed by these Web-based technologies. The general conclusion is that organizations wishing to gain dominance by exploiting the advantages on offer should begin the adoption process early despite the emergent status of some important features.

THE RATIONALE OF VIRTUAL ORGANIZATIONS

A New Way Of Doing Business

Prompted by recent developments in information and communication technology (ICT), new types of organizational structure are being proposed that promise to change the way business is conducted. We examine the technological developments in the second section. For now, we simply characterize them by the slogan “just-in-time computing”, which alludes to the fact that, using the emerging technology of Web services, application software can be cobbled together on the wing allowing organizations to quickly establish strategic alliances despite radical differences in their technology platforms.

Known generally as the boundaryless organization, these more flexible structures are redefining the criteria for success. Speed of response to changing conditions now ranks above sheer size. Where clarity of job definition was once considered vital, flexibility coupled with a willingness to cope with ambiguous and multi-faceted roles is now sought. Specialization is still required, but there is an increasing emphasis on the ability to integrate diverse activities. Similarly with control. Rather than focussing exclusively on controlling the work of others, often by instituting a repressive regime of approvals and double-checks, enlightened managers now encourage innovation and creativity (Ashkenas, Ulrich et al., 2002: 4-7).

Kevin Wilkinson is a Senior Lecturer in Information Systems at the Wellington campus of Massey University in New Zealand. This paper was written during a visit to the Peninsula School of Information Technology of Monash University in Melbourne, Australia, where he shares an ongoing collaborative research interest in Web services. His other research interests include database theory and ontology modelling.

The adjective "boundaryless" is not to be taken literally. Instead, it indicates a softening of organizational borders to form a "dynamic ecosystem" that allows strategic alliances to be formed with trading partners such as suppliers, distributors, agents, dealers, brokers, and retailers (Zhang, Chen et al., 2005: 50). Collaboration may even extend to competitors, a phenomenon known by the awkward neologism "co-opetition" (Kasper-Fuehrer & Ashkanasy, 2003: 42). Marketing may come full circle when customers are included in the design of a product or service they later purchase (Kasper-Fuehrer & Ashkanasy, 2003: 42).

A developing ICT is but one of the changes in the business environment that are prompting the creation of virtual enterprises through the formation of alliances and partnerships. Others include the reduction of trade restrictions and the globalization of supply and consumption markets (Walters, 2005: 231). Even though the new technology enables alliances based on rapid responses, it does not guarantee successful outcomes. Some of the problems and benefits of formal alliances are listed and discussed in Walters, 2005: 231. The problems, such as asymmetric benefits among the partners, protection of sensitive information, or an inability to make tough decisions, are inherent dangers of any alliance no matter how it is enabled. The benefits too, which include being able to reduce or control risk through having a negotiated business environment or achieving growth by combining the complementary resources of each partner, are also mostly true of alliances in general. Here, our focus is on those features, problems, and benefits of alliances that arise directly from the enabling technology itself. The question we ask can be framed as: given that a sound business case has been made for the formation of a strategic alliance with a number of external organizations or technologically remote internal business units, what special opportunities and threats can be expected because of its dependence on "just-in-time" computing?

Before attempting to answer this question, however, we need to consider whether the use of an emerging ICT is *implied* by the definition of a virtual organization (VO) or whether it is merely one of the available options.

A Family Resemblance Definition Of Virtual Organizations

As Roger Sor has observed, the only thing that people agree on who offer a definition of a VO is that there is no agreement about how that concept should be defined (Sor, 1999: 825). Intuitively, for something to be a virtual *x*, it has to behave or present itself *as though* it is an *x* without really being one. Virtual memory, for example, is an area of hard disk used to extend the size of random access memory. It is addressed by the operating system as though it is part of the RAM even though, physically, it is part of the permanent store. Beyond that, there does not seem to be much consensus on what constitutes virtuality.

Rather than add yet another analytic definition to the list, I propose instead a family resemblance argument (as employed by Ludwig Wittgenstein) that to be a VO is to share a network of overlapping but discontinuous resemblances, just as members of the same family resemble each other in some ways but not others. No particular feature is necessary, but if an organization possesses enough of them it qualifies as virtual. What results is not a precise definition, but the flexibility implicit in the term "enough" allows a cluster of features to serve as a useful category while at the same time avoiding unproductive semantic border disputes. Some of the characteristics that VOs are said to share are discussed in the following sections.

Temporary existence

A VO is sometimes said to be a temporary and *ad hoc* alliance between a number of organizations. The partners in the alliance have an on-going bricks-and-mortar existence but the alliance itself is relatively short-lived and is supported by selected parts of the infrastructure that the partners normally use, made interoperable by some enabling technology. There seems to be an assumption of innovation at work here such that a VO will be formed to make a quick and flexible response to a rapidly-changing situation and that, because the partners are different kinds of organizations with different strategic goals, the alliance will be inherently unstable in the long term. However, if the partners share common goals there seems to be little reason for limiting the life expectancy of a VO.

In fact, many Grid computing applications, which involve the formation of VOs, come from the fields of science and engineering and are expected to be long-lived. Examples include distributed aircraft engine diagnostics; the NEESgrid project of the National Science Foundation's Network for Earthquake Engineering Simulation; the Virtual Telescope, which gives astrophysicists access to and computation over terabytes of data collected from many different detectors, satellites, and observatories around the world (Foster & Kesselman, 2004 Part III).

What can be said is that, *if* a temporary alliance needs to be formed, Web services offer a quick and convenient way of providing the communication infrastructure. Just-in-time computing suits short-term alliances, even if temporary existence is not a necessary characteristic of a VO.

Innovation and flexibility

By supporting the creation of VOs, just-in-time computing provides rapid and flexible responses to new opportunities and challenges and thus promotes innovation. Instead of constructing a solution from the ground up, a novel configuration of already existing partial solutions can be assembled. Because the architecture already handles diversity, adding new partners is relatively easy. We can set aside the question of longevity: an innovative solution is no less innovative for being long-lived.

It seems reasonable to ask whether a VO might be usefully created to provide a more flexible way of carrying out an existing procedure rather than to exploit a new opportunity. I think the answer is: Yes, but entering into an alliance involves relinquishing a certain amount of control, so there will probably be a trade-off in which flexibility is gained but control is lost. There are also other costs and risks to consider. Partners may have conflicting strategic objectives and, even if an acceptable compromise is reached, it usually requires time to achieve and an on-going effort to maintain, so VOs are probably best suited to innovation and to exploiting opportunities that would otherwise be lost rather than to producing more flexible versions of old solutions.

Mission overlap

It is sometimes built into the definition of a VO that the missions of the partners overlap to some extent. Intuitively, they must, but to what extent? The overlap could be total when, for example organizations that collect astronomical data form an alliance so that astrophysicists can access it across organizational boundaries. In such cases, the alliance allows each partner to produce the results it needs without having to copy terabytes of data from one location to another. Having identical or highly similar missions does not diminish the opportunities presented by collaboration.

If the partners are commercial organizations with similar missions they must be actual or potential competitors and the motivation for collaboration may be an externally imposed one. Imagine, for example, a number of oil companies that are obliged to provide pricing or profit information to satisfy statutory obligations. Instead of setting up a separate regulatory body to whom they report their statistics, it might be possible to create a VO that performs the same function more cost-effectively.

Competitors may wish to collaborate even when the motivation is not externally imposed. For example, when a type of cost is very high, as research and development costs are in the pharmaceutical industry, collaboration with competitors can be seen as an acceptable strategy. Web services could be used to share the results of clinical trials, thus shortening the time it takes for new drugs to be approved.

When partners have different missions, the motivation to form a collaborative alliance derives from a business-to-business relationship between them, which invariably involves the supply of goods or services from one to the other. Organizations that have complex products, such as in the aerospace industry, or huge numbers of suppliers, such as industries that retail packaged goods, can make many savings from an automated exchange of supplier information.

When the scale of business is large, there can be a fine line between collaboration with an external company and the integration of different systems within the same corporate entity, such as the design and manufacturing arms of an aerospace manufacturer. The greater the managerial or technological distance between business units, the more nearly the integration of them resembles collaboration. Web services can be used to bridge gaps within a corporation as well as between collaborating partners.

So we can conclude: what matters for the success of an alliance is not whether the degree of mission overlap is high or low but whether each of the partners is sufficiently motivated to contribute an area of competence to it.

Geographical spread

As we shall discover below, it is part of our definition of Web services that they be accessed across a distributed network, so it follows that a VO built on the basis of a Web services architecture has partners that are spread geographically. However, the motivation for forming a VO derives more from the last part of the definition — that the exchange of data is facilitated by vendor, platform, and language-neutral protocols — rather than simply by the ability to overcome distance. It is by overcoming *technological* distance that Web services will promote innovation and provide flexibility.

Concurrent operation mediated by information technology

Having an interaction between partners mediated by information technology is only to be expected of a VO, given the prevalence of information and communication technology today. Once we add the rider that Web services are involved, information technology is implicated necessarily. The interesting part of the feature is the stipulation that the partners must operate concurrently. This requirement does seem to bias the definition of a VO towards the use of information technology. I tend to agree with Sor, 1999 that information technology should be seen as an enabler of virtuality and is therefore a secondary rather than a defining characteristic of it. However, in this family resemblance definition, the distinction between primary and secondary characteristics is lost as none is required in all cases. This allows concurrent operation to be a part of the family resemblance between VOs and yet, because it is not necessary, Sor's example is accommodated that the Housing Construction Industry in a Western Australian metropolitan area was operating in a virtual mode before information technology was so pervasive.

Customer-based identity

Another criterion that is sometimes applied to virtuality is that, to count as virtual, an organization must be seen as a single entity by those who use its services even though, in reality, those services are supplied by a variety of different organizations. An element of deception seems to be implied. However, being "seen as" something need not involve ignorance or deceit. Having an interface that *functions as* a single organization is fully compatible with having users who know the true provenance of the services they obtain. This is most obvious in the case of scientific collaborations where the users *are* the service providers. What matters is not whether the users are tricked into thinking the VO is real but whether the network of organizations it represents allows them to receive services they would not otherwise enjoy.

Competing Typologies

Authors who have a list of essential features in mind that VOs must conform to will distinguish them from other cooperative organizational forms. If a VO is necessarily temporary in its focus, it will have to be distinguished from a strategic alliance, syndicate, joint venture, or franchising operation, all of which are permanent. If it must be managed in a decentralized manner, it cannot be a conglomerate. If it is essentially collaborative, it will not be a franchising operation, which is a cooperative venture but not a collaborative one, the distinction being that collaborating partners actively assist each other whereas those who cooperate merely arrange their operations to avoid mutual competition (Kasper-Fuehrer & Ashkanasy, 2003: 52).

Other authors define VOs more liberally. For example, Campbell's typology, discussed in Walters, 2005: 233, has four categories: internal, stable, dynamic, and agile. The characteristics and typical activities of these types are summarized in Table 1.

Table 1: Campbell's typology of virtual organizations

VO Type	Character	Typical Activities
Internal	An autonomous business unit within a large organization	Low risk, such as extending a product range in response to the demands of specific customers
Stable	A conventional organization with a small network of established partners, such as suppliers, to whom it outsources non-core activities	Medium risk, such as a product differentiation exercise
Dynamic	An organization that focuses on its core activities and engages in cooperative ventures with others	Higher risk, such as a product or market development exercise
Agile	A temporary network of organizations formed rapidly	High risk, such as exploiting a short life-cycle opportunity

With such a typology, it is clearly possible for VOs to be composed of internal or external entities and to be designed for long or short term existences. Other theorists agree that VOs may be intra or interorganizational but insist that interorganizational ones are inherently temporary (Kasper-Fuehrer & Ashkanasy, 2003: 42), which is incompatible with the stable type in Campbell's classification.

Dynamic VOs tend to be short term, but their main attribute is flexibility, which allows partners to join or leave as projects begin, evolve, and end.

Agility is the property most easily associated with a VO because we usually assume that ICT will be used in an innovative way to quickly gain a competitive advantage. The requisite assets, expertise, and relationships must be assembled, and an organization may have to make special demands on its customers, partners, and internal business processes in order to achieve this before the opportunity disappears (Huang & Hu, 2004: 62).

These variations in the definition of a VO lend credence to the family resemblance account proposed here because it allows us to judge in a conveniently flexible way whether a collaborative arrangement should count as a VO or not. Armed with it, we now consider the features of recent developments in ICT that have the potential to support and promote such arrangements.

CONVERGING TECHNOLOGIES BASED ON WEB SERVICES

Web Services

In a service-oriented architecture, software services are exchanged between agents who are loosely coupled. Web services are a special case of this in which the interface between a service provider and a consumer is based on Internet protocols and messaging is achieved using XML. Web services are defined as "…loosely coupled, self-describing services that are accessed programmatically across a distributed network, and exchange data using vendor, platform, and language-neutral protocols." (Marks & Werrell, 2003: 27)

Services are sometimes distinguished from components (Marks & Werrell, 2003: 167-8, Singh, Brydon et al., 2004: 4-5). A service is a coarser-grained concept than a component and operates at the level of a business transaction, whereas components provide the logic for a transaction step. Having said this, services may also be composed of other services, which makes the lower-level services seem more like components. However, as we discuss below, components only work on systems that adopt a component-oriented architecture, whereas services can be shared by systems based on different platforms.

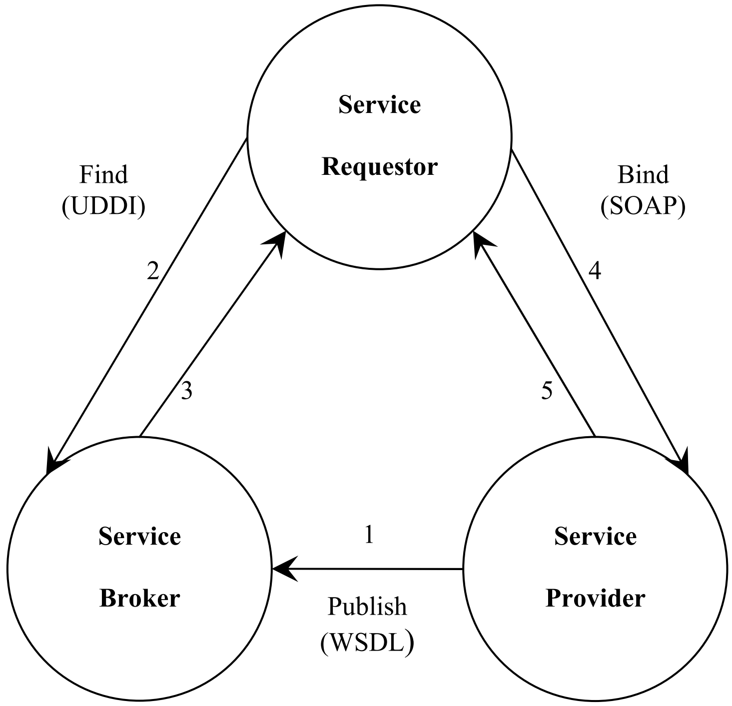
Web services extend the principle of loose coupling to a new level. As a principle of application programming design, loose coupling ensures that when one procedure calls on the services of another it need know only the serving procedure's name, port, and what types of parameters its interface expects. The implementation details remain hidden and can change without producing pesky side effects. In Web services, the principle is extended by a series of abstractions (Marks & Werrell, 2003: 33, 170). Above the well-established enabling protocols of network transport (TCP/IP, HTTP, and others) and the meta language XML that allows data structures to be defined, there is a layer of evolving standards that currently consists of

SOAP — Simple Object Access Protocol
WSDL — Web Services Description Language
UDDI — Universal Description, Discovery, and Integration

SOAP is a protocol that allows us to invoke services on remote systems and exchange data between processes. WSDL allows us to describe services and their interfaces. UDDI is used to create public and private registries of services so that users can discover and bind with them (Huang & Hu, 2004: 58). A UDDI registry is often analysed in terms of white, yellow, and green pages (Singh, Brydon et al., 2004: 36). The white and yellow pages identify businesses and categorize their services respectively. Green pages provide interface, location, and other technical information that allows services to be executed. (Zhang, 2004-2005: 85)

As far as abstraction is concerned, SOAP hides the language of implementation. Services can be accessed and bound without having to worry about whether they were developed using Microsoft's .NET platform or Java 2 Enterprise Edition (J2EE). WSDL allows services to be published in such a way that their parameters and network addresses are hidden. Only the name of the service description must be found manually. UDDI allows a consumer to find a service dynamically by using the automatic search facility of a registry of services. Obviously, the registry itself has to be known together with the name or type of service which is required but, once these three additional levels of progressive abstraction are in place, services can be published and used in an automated fashion as illustrated by Figure 1.

Figure 1: The evolving standards of Web services (after Marks & Werrell, 2003 Figure 2.9)



1: The provider publishes a service with a service broker

2: A requestor uses a service broker to find published services

3: The broker provides the requestor with service providers

4: The requestor sends a service request to the provider

5: The provider fulfils the service request

As Marks and Werrell point out, this model represents the way Web services are *intended* to operate, with repositories of services being made publicly available. Currently, however, actual implementations are likely to involve a private repository, or none at all, and services that are for private consumption behind a corporate firewall (Marks & Werrell, 2003: 41).

In addition to this layer of evolving but well-defined standards, there is a layer of emerging standards intended to enable business processes to interact with Web services. This layer will provide vital capabilities in the areas of transaction control, service management, and security (Marks & Werrell, 2003: 48-54). Recently, a workflow language called Business Process Execution Language for Web Services (BPEL4WS) has been proposed with the backing of IBM, Microsoft, and BEA Systems that will allow business processes to be constructed out of ensembles of existing services.

Finally, services are vendor, platform, and language-neutral. Once developed, they are interoperable but not portable (Marks & Werrell, 2003: 174-8). I can use the results of a service you have published even though my technology platform differs from yours, but I cannot easily port the service to my machine and run it.

This account suits data-oriented services where what is passed from one process to another is raw data or data resulting from computation. However, there is an emerging standard called Web Services for Remote Portlets (WSRP), which deals with presentation-oriented services. These services define how data is to be presented and how the user can interact with it. At the moment, portlets must be developed for each specific presentation, but WSRP will allow presentation and interaction details to be included on a portal in a generic way (Polgar, Polgar et al., 2005: 266).

We now take a closer look at this high level of abstraction by which the functionality of legacy systems can be made available to any computer attached to the Web, including wireless and hand-held devices (Zhang, 2004-2005: 89-90).

Other application integration approaches

Traditionally, computer systems that handle business processes interact with each other (if they interact at all) only within organizational boundaries. Web services are an attempt to overcome this limitation. There have been other recent attempts but, unlike Web services, they work only within the scope of a particular technology platform, operating system, or programming language.

The Component Object Model (COM) protocol allows objects in different applications to interact, but only if they reside on the same system. The Distributed Component Object Model (DCOM) allows objects on different systems to interact as long as their systems are part of the same network. Web services, however, allow objects on different networks to communicate across the Internet (Murtaza & Shah, 2004: 44).

COM-based applications work only on a Windows platform. More flexible is Remote Method Invocation (RMI), a Java-based technology used by Sun's Java J2EE, which allows an object running on one machine to invoke the methods of an object running on another. However, although the machines may run on different platforms and use different operating systems or middleware, they must both be Java virtual machines (Singh, Brydon et al., 2004: 395), which means they require special software to interpret the byte-code of the executable Java programs whenever they are run.

The Common Object Request Broker Architecture (CORBA) supports multi-language and multi-platform applications, but only when both ends of an interaction use the Object Request Broker library (Zhang, 2004-2005: 83). Systems like CORBA and J2EE may be a good way to *implement* Web services but, because they have a large footprint, not having to use them to *run* services is a strong benefit (Hündling & Weske, 2003: 109).

The benefits of Web services

Web services are based on well-established standards such as HTTP and XML and all vendors agree on SOAP as the messaging format. Consequently, they cost less to develop and are more flexible than traditional distributed computing software (Huang & Hu, 2004: 59). Table 2 analyses the benefits and explains them in more detail.

Table 2: The benefits of Web services (after Murtaza & Shah, 2004: 45)

Benefit	Comment
New partner discovery and interoperability	The "yellow pages" of service registries allow new business partnerships to be formed spontaneously
Universal accessibility	Services are distributed and decentralized and can be accessed from almost any device connected to the Internet
Efficient application sharing	Web services can act as a wrapper for the functionality of internal applications, including legacy systems, exposing it efficiently to prime customers and business partners
Universally acceptable specifications	With universally agreed protocols and standards for structured data exchange, messaging, and service description and discovery, applications on different platforms can be integrated without the need for proprietary technologies
Corporate network compatibility	The transport mechanisms used by Web services are not affected by firewalls and proxy servers

The scenarios in which these benefits are most obvious include those in which an organization has a large and dispersed client base, wishes to expose the functionality of its legacy systems, needs to integrate heterogeneous technologies (perhaps as the result of a merger), or is planning to develop extranet applications (Murtaza & Shah, 2004: 45-6).

Business partners have traditionally communicated by using electronic data interchange (EDI). As Murtaza and Shah observe, EDI usually involves too much cost and effort for small-to-medium enterprises, so Web services offer an affordable alternative (Murtaza & Shah, 2004: 46). This also affects large organizations when, for example, real-time collaboration with suppliers is required as part of an agile manufacturing operation. After all, large organizations can have suppliers of any size (Murtaza & Shah, 2004: 50).

There is a trend towards adopting Web services technology because, in an increasingly competitive global economy, it becomes a strategic necessity to be able to seamlessly connect a geographically distributed IT infrastructure. Up-beat accounts of the technology predict that it will not only change the way business applications are developed and deployed, but also the way business itself is conducted (Huang & Hu, 2004: 59).

Other Web Service Developments

Grid computing

The technology of Web services and the idea of a virtual organization are explicitly combined in an emerging paradigm known as Grid computing. The term "the Grid" has been in use since the mid 1990s (Foster, Kesselman et al., 2001: 200) and is based on an analogy with utilities such as electricity or the telephone. When we plug a device into the electricity grid, we obtain a resource, electricity, on demand without having to generate it ourselves or even to worry about where it was generated or by whom. The idea that computing services could one day be available in this fashion was suggested as early as 1969 (Foster, 2002), but the idea began to come to fruition in the late 1980s with the seminal work done by researchers such as Ian Foster and Carl Kesselman.

The role of the Grid is to enable the creation of virtual organizations, which Foster, Kesselman, and Tuecke define as

"…flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions, and resources." (Foster, Kesselman et al., 2001: 200)

Foster has since proposed a checklist of the essential features of Grid systems, suggesting that they should

- Coordinate resources that are not subject to centralized control
- Use standard, open, general-purpose protocols and interfaces
- Deliver qualities of service so that the utility of the combined system is significantly greater than the sum of the parts (Foster, 2002)

Although the standards of Grid computing have been developing rapidly, they are now stabilizing as Grid services technology becomes an extension of Web services (Holliday, Wilkinson et al., 2005: 207). Grid computing provides resource sharing, which is often a matter of optimizing the computational load of cycle-hungry calculations among a network of processors, but a software service is a type of resource and Foster's checklist seems valid for Web services and Grid computing alike.

The World Wide Web began with scientific applications and has been taken up by the business community. The Grid too was created to serve science and is also expected to be adopted for e-business as a result of the development of standards for the construction of Grid systems such as the Globus Toolkit, which is currently evolving into the Open Grid Services Architecture, an extensible set of services that virtual organizations can call upon and that combines features of both Grid computing and Web services (Foster, Kesselman et al., 2002: 38).

Despite the fact that the technologies of Web services (WS) and Grid computing (GC) are converging, they can be distinguished in a number of ways.

- WS do not necessarily result in the formation of a virtual organization (services may be purchased as commodities rather than exchanged collaboratively), whereas GC involves cooperation between trusted partners as evidenced by an exchange of certificates
- WS do not necessarily involve parallel computing (an entire task may be outsourced to a service provider), whereas in GC a task typically involves so much data or calculation that it must be solved using multiple processors, the results being combined at the end
- WS and GC both use the request/response protocol of HTTP but, whereas WS tasks are normally stateless, GC maintains an internal state for each task by means of the Web Services Resource Framework (WSRF), which defines conventions for managing state so that applications can reliably share changing information (Looselycoupled, 2005)
- GC has its own set of distinctive tools, including a Grid file transfer protocol and resource management capabilities across multiple nodes

In general, because of its focus on sharing resources across organizational boundaries, Grid computing complements rather than competes with existing distributed computing technologies (Foster, Kesselman et al., 2001).

Adding semantics to Web services

The Web is currently the subject of major efforts to provide it with a semantic rather than a merely syntactic interface (Berners-Lee, 2003, Berners-Lee, Hendler et al., 2001, Davies, Fensel et al., 2003, Heflin & Hendler, 2001). Once this vision of a Semantic Web is realized, the Web will be accessible by meaning instead of merely by location so that, rather than searching by keywords and manually selecting among the sites returned by the search engine, users will be able to pose queries as they do in a database setting, in terms meaningful to their domain of activity, and receive answers formatted to their convenience.

At the heart of the Semantic Web are ontologies, which provide a machine-readable semantics. Because it is unrealistic to require all users of the Web to subscribe to the same conceptual scheme, multiple ontologies are required to reflect the terminological habits of local groups. Ontologies can be linked hierarchically, which allows local conceptual schemes to be aligned with more general concepts by means of automated reasoning techniques, which in turn allow wide-ranging queries to be addressed (McGuinness, 2003, Klein, Broekstra et al., 2003, Maedche, Motik et al., 2003).

Work is also underway to produce the Grid-equivalent of the Semantic Web called, unsurprisingly, the Semantic Grid. Such a facility, seen as an application of Semantic Web technology rather than a replacement of it, would bring the advantages of a machine-readable semantics to the Grid, increasing its flexibility and power by allowing local vocabularies to be used when defining, requesting, and discovering services (Goble & De Roure, 2002, Goble, De Roure et al., 2004, Brooke, Garwood et al., 2003, Moreau, Miles et al., 2003, Pouchard, Cinquini et al., 2003).

The same point can be extended to Web services in general, where there is a need for domain-specific standards to facilitate the selection, composition, and invocation of services, including the real-time checking of non-functional attributes to filter out services that are unsuitable (for reasons of quality, trust, or security) even though they may provide the desired functionality (Zhang, Chen et al., 2005: 62, Howard & Kerschberg, 2004: 442). The practical challenge is that

"…Web services are very likely developed by different organizations, which may use different semantic models to describe services. It is necessary that service consumers and providers use a shared ontology to denote concepts while describing services so that the description of services can be understood appropriately." (Zhang, Chen et al., 2005: 54)

This challenge is being addressed by seeking to apply Semantic Web techniques to Web services so that consumers can be connected with the services that best suit their needs even when they have no prior knowledge of the services that exist or who supplies them.

This vision of converging technologies, each extending and enhancing the other, has two faces. So far, the account has been generally optimistic. We end by tempering this tone, not with pessimism, but with a realistic assessment of the opportunities and threats that are involved for organizations that seek the kinds of flexibility and convenience offered by just-in-time computing.

WEB SERVICES: A REALITY CHECK

On-Going Development

The development of Web services technology is very much an on-going affair. As we saw earlier, the basic transport mechanism and standards for the definition of structure are agreed and stable. Standards for defining, discovering, and binding with services are agreed and evolving, but there are many important aspects of business processing such as security, reliability, service composition, transaction management, workflow, and system performance for which no agreement exists or whose standards are still emerging (Marks & Werrell, 2003: 33, 170, Murtaza & Shah, 2004: 51).

Unfortunately, the existing XML-based standards require programmers to hard-code service requests with information such as the name of the provider, service, or port. Matchmaking between advertisements and requests is conducted by keyword search and is an all-or-nothing affair. This means that requests are unable to adjust automatically when a better provider appears or when the existing provider develops problems (Paolucci, Sycara et al., 2003). It would be preferable if matchmaking were conducted using semantic markup so that the best match between the available and required qualities of service could be established automatically. Work to this effect is in progress (see for example Akkiraju, Goodwin et al., 2003 and Trastour, Bartolini et al., 2001).

Current Web service techniques are further limited by the fact that each operation tends to be independent so that, if none matches a consumer's request, the matchmaking attempt fails. This would be alleviated if existing services could be combined into composite solutions, and this too is an active research area (Zhang, Chen et al., 2005). Of course, such a feature will introduce new difficulties: once services are composed, the failure of any one of the component services could result in the failure of the whole.

In some important areas there is a dearth or a surfeit of standards. As yet, there is no agreed standard for an ontology language for Web services. OWL-S is being developed from a combination of DAML-S (DARPA Agent Markup Language for Services) and OIL (Ontology Inference Layer), but OWL exists in versions of differing strengths and OWL-S makes use of constructs in the full-strength version that introduce unwanted complexities (Elenius, 2004 Section 3). Other shortcomings attributed to OWL-S include conceptual ambiguity, narrow scope, loose design, the coupling of function and description, and static process declaration (Howard & Kerschberg, 2004: 444).

In the area of managing the properties of business transactions in a Web services environment, there are competing standards (Business Transaction Protocol and WS-Coordination) that differ significantly but are geared towards similar types of work (Hündling & Weske, 2003: 117).

Workflow management is another area in which standard solutions are being sought. A workflow management system controls the execution of business processes. Creating a workflow involves creating job instances out of selected resources, placing them at specified sites, and identifying any dependencies between the orderings in which the job instances are executed (Holliday, Wilkinson et al., 2005: 207).

We have already mentioned BPEL4WS, an XML-based process definition language that allows companies to describe processes for Web services environments. The approach has the backing of some major players, which is fortunate because the features it must include are demanding. A business activity may involve many atomic transactions. In a WS environment, this leads to protocols even more complex than for distributed database transactions. In a business activity, the response may take a long time, so locking the resources is not feasible (Hündling & Weske, 2003: 116). With our interest in virtual organizations, we should also note that workflows involving multiple organizations are still in their infancy (Hündling & Weske, 2003: 113). The state of the art of workflow management systems can be summed up by this comment on the attempt to automate supply-chain processes.

"Supply-chain processes are very complex and highly dynamic. Attempts have been made to enable dynamic supply-chain reconfiguration with agent technology and dynamic workflow technology. A large body of work (e.g. ARIS, BPEL4WS, Business Process Modeling Language (BPML)) exists that has not yet found its way into industry and real applications." (Howard & Kerschberg, 2004: 444)

Another cluster of concerns centre around the issues of security and trust. We noted earlier that the transport mechanisms used by Web services are not affected by firewalls and proxy servers. While this is an advantage when the messages are benign, it is a security concern for SOAP and UDDI when they are not. The following quotes illustrate the nature of the concerns.

"In the case of Web services, most firewalls do not know how to examine SOAP messages. These messages will pass right through your firewalls without the firewall checking for attacks." (Egan, 2005: 211)

"Public UDDI registries are too generic and lack control over the quality of Web services and the trustworthiness of service providers." (Zhang, Chen et al., 2005: 61)

"…major issues like security and authentication for real world applications require additional work in research and development, which is currently underway." (Hündling & Weske, 2003: 111)

Grid computing is better off in this regard. The Globus toolkit implements a standard called the Grid Security Infrastructure (GSI) in which mutual authentication is performed through certificates and the Secure Socket Layer (Holliday, Wilkinson et al., 2005: 208).

As we noted earlier, Web services are currently data-oriented, but there is a need for the WSRP standard to be implemented so that user interface details can be defined as part of a service rather than needing to be defined anew for every application of that service in a portal.

Finally, we should note that, although a major selling-point of Web services is that they can make the functionality of existing software available across organizational boundaries, development overheads are still incurred. Large teams of application programmers may not be required, but a few much more expensive and scarce people are: systems architects with the depth of knowledge to deal with problems that range over a complex mix of platforms, tiers, products, standards and languages.

Early Versus Late Adoption

We have seen that Web services, by employing a service-oriented architecture, promote the formation of virtual organizations, allowing cooperative relationships to be developed between partners in an agile manner, even if each uses very different mixes of hardware and software vendors, communication protocols, and programming languages. These relationships are maintained by the automated exchange of sensitive information, resulting in business-to-business transactions that were not available, or not practicable, in the past.

Traditional enterprise-wide systems tend to take large amounts of time and money to develop and to make a large footprint once they are operational in the sense that major commitments have to be made to standardize the technology throughout the contributing business units. When independent organizations wish to merge or align their information and communications technology, solutions usually require the minor players to adopt the technology of the dominant one. With Web services, by contrast, each player is free to retain its existing core technologies but implements additional protocols for publishing and discovering shared software services. The solutions grow incrementally rather than being implemented with a "big bang" and leave a smaller footprint.

The advantages of this approach mirror the virtues of virtual organizations themselves, allowing innovative alliances to be developed quickly and flexibly, overcoming geographical and technological distance, and making even temporary collaborations feasible regardless of the extent of mission overlap between the partners and whether they are natural allies or competitors.

As with any major technological development, there are reasons for playing a waiting game and being a late rather than an early adopter. As we saw above, core parts of the Web services technology are still evolving and other vital parts are still emerging. Those already nervous about the changing nature of the technology could easily find the very idea of shared software services alienating. Business processes that in the past have been directly controlled now seem to be controllable only by trying to influence an organization that has no real existence! As well as this threat of a loss of control, there are other threats associated with managing a virtual organization, such as threats to identity and to esteem (Wiesenfeld, Raghuram et al., 1999), but these are more likely to arise in situations where the manager works *only* in a virtual organization rather than in situations where the virtual organization is an *extension* of the work of its real partners.

Even if one accepts the volatility of the technology and the virtuality of the controlling decision maker, there is still the time-consuming practical hurdle to overcome of reaching and maintaining agreement among the stakeholders.

Against these negative arguments, however, there are positive rejoinders that are at least as convincing. Waiting for technology to stabilize is to wait for a moment that may never come and, even if a plateau is reached as was the case with database technology when the relational model was introduced in the 1970s, being slow to adopt a significant advance is to give away the chance to gain dominance in a particular domain. Assuming that Web services technology delivers the promises it has made, the first group of partners within an industry to exploit its potential and form a virtual organization will have the chance to make a break on the rest of the field.

Organizations that remain inward-looking may suffer an even worse fate: that of being left behind completely rather than merely failing to gain dominance. It is well-known that, once a majority adopt a new technology, such as having an on-line Web presence, every player needs one in order to stay in the race even if, considered in isolation, the technology seems to offer the player very little of what it wants. Once the competition has adjusted to the rules of a new paradigm, in this case seeing software as an externally negotiated and shared service rather than as a proprietary product, there may be little option but to join them.

Finally, of course, the argument that negotiation takes time is not an argument against doing it but, rather, a case for starting as soon as possible. Only in this way will the wave of change be caught before it slips past and carries the competition into the distance.

In conclusion we can say that, although a conservative approach to new technology is often wise, organizations wishing to gain dominance by exploiting the advantages offered by Web services should begin the adoption process early rather than late. Because the process is incremental rather than a "big bang" the risk of adoption can be monitored and the benefits gained progressively.

Future work in this area consists of turning the emerging and evolving technologies of Web services into a stable unified whole so that there are no longer incomplete or competing standards. Assessing the state of the art will remain an on-going activity until there is a received methodology for publishing and discovering Web services including composition, transaction management, workflow, and other features required to guarantee the secure and flexible exposure of software functionality. Well-written case studies describing applications of this methodology will greatly assist the spread of the technology, especially if the level of detail is sufficient to inform rather than merely advertise but not so great as to obscure the general strategy.

REFERENCES

Akkiraju, R., Goodwin, R., Doshi, P., & Roeder, S. 2003. A Method for Semantically Enhancing the Service Discovery Capabilities of UDDI. Proceedings of IJCAI-03 Workshop on Information Integration on the Web (IIWeb-03): 87-92.

Ashkenas, R., Ulrich, D., Jick, T., & Kerr, S. 2002. *The Boundaryless Organization: Breaking the Chains of Organizational Culture* (2 edn). San Francisco, CA: Jossey-Bass.

Berners-Lee, T. 2003. Forward. In D. Fensel, J. Hendler, H. Lieberman, & W. Wahlster (Eds), *Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential* : xi-xxiii. Cambridge, MA and London, England: The MIT Press.

Berners-Lee, T., Hendler, J., & Lassila, O. 2001. The Semantic Web. *Scientific American,* 284 (5): 34-43.

Brooke, J., Garwood, K., & Goble, C. 2003. Interoperability of Grid Resource Descriptions: A Semantic Approach. *Proceedings of the Global Grid Forum Semantic Grid Research Group 9th Global Grid Forum*: 45-51.

Davies, J., Fensel, D., & Harmelen, F. 2003. *Towards the Semantic Web: Ontology-Driven Knowledge Management*. Chichester, England: Wiley.

Egan, M. 2005. The Executive Guide to Information Security: Threats, Challenges, and Solutions. Indianapolis: Addison-Wesley.

Elenius. 2004. *Modeling Services With Protégé.* Web site: URL http://protege.stanford.edu/conference/2004/posters/Elenius.pdf.

Foster, I. 2002. *What Is the Grid? A Three Point Checklist.* Web site: URL http://www-fp.mcs.anl.gov/~foster/Articles/WhatIsTheGrid.pdf.

Foster, I., & Kesselman, C. 2004. *The Grid: Blueprint for a New Computing Infrastructure* (2 edn). San Francisco, CA: Morgan Kaufman Publishers.

Foster, I., Kesselman, C., Nick, J. M., & Tuecke, S. 2002. Grid Services for Distributed System Integration. *Computer,* 35 (6): 37-46.

Foster, I., Kesselman, C., & Tuecke, S. 2001. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *International J. Supercomputer Applications,* 15 (3): 200-222.

Goble, C., & De Roure, D. 2002. The Grid: An Application of the Semantic Web. *SIGMOD Record,* 31 (4): 65-70.

Goble, C. A., De Roure, D., Shadbolt, N. R., & Fernandes, A. A. A. 2004. Enhancing Services and Applications With Knowledge and Semantics. In I. Foster, & C. Kesselman (Eds), *The Grid: Blueprint for a New Computing Infrastructure* 2 edn., 431-461. San Francisco, CA: Morgan Kaufmann Publishers.

Heflin, J., & Hendler, J. 2001. A Portrait of the Semantic Web in Action. *IEEE Intelligent Systems,* 16 (2): 54-59.

Holliday, M. A., Wilkinson, B., House, J., Daoud, S., & Ferner, C. 2005. A Geographically-Distributed, Assignment-Structured Undergraduate Grid Computing Course. *Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education*: 206-210.

Howard, R., & Kerschberg, L. 2004. A Framework for Dynamic Semantic Web Services Management. *International Journal of Cooperative Information Systems,* 13 (4): 441-485.

Huang, D. C., & Hu, Q. 2004. Integrating Web Services With Competitive Strategies: The Balanced Scorecard Approach. *Communications of the Association of Information Systems,* 13: 57-80.

Hündling, J., & Weske, M. 2003. Web Services: Foundation and Composition. *Electronic Markets,* 13 (2): 108-119.

Kasper-Fuehrer, E. C., & Ashkanasy, N. M. 2003. The Interorganizational Virtual Organization: Defining a Weberian Ideal. *Int. Studies of Mgt. & Org.,* 33 (4): 34-64.

Klein, M., Broekstra, J., Fensel, D., van Harmelen, F., & Horrocks, I. 2003. Ontologies and Schema Languages of the Web. In D. Fensel, J. Hendler, H. Lieberman, & W. Wahlster (Eds), *Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential* : 95-139. Cambridge, MA and London, England: The MIT Press.

Looselycoupled. 2005. *Grid Computing Glossary.* Web site: URL http://looselycoupled.com/glossary/WSRF.

Maedche, A., Motik, B., & Stojanovic, L. 2003. Managing Multiple and Distributed Ontologies on the Semantic Web. *The VLDB Journal,* 12: 286-302.

Marks, E. A., & Werrell, M. J. 2003. *Executive's Guide to Web Services*. Hoboken, NJ: John Wiley & Sons.

McGuinness, D. L. 2003. Ontologies Come of Age. In D. Fensel, J. Hendler, H. Lieberman, & W. Wahlster (Eds), Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential : 171-194. Cambridge, MA and London, England: The MIT Press.

Moreau, L., Miles, S., Papay, J., Decker, K., & Payne, T. 2003. Publishing Semantic Descriptions of Services. *Proceedings of the Global Grid Forum Semantic Grid Research Group 9th Global Grid Forum*: 39-44.

Murtaza, M. B., & Shah, J. R. 2004. Managing Information for Effective Business Partner Relationships. *Information Systems Management,* (Spring): 43-52.

Paolucci, M., Sycara, K., Nishimura, T., & Srinivasan, N. 2003. Towards Semantic Web Services. *WWW 2003 Workshop on E-Services and the Semantic Web (ESSW'03)*.

Polgar, J., Polgar, T., & Bram, R. 2005. *Building and Managing Enterprise-Wide Portals*. (Forthcoming): Idea Group Publishing.

Pouchard, L., Cinquini, L., Drach, B., Middleton, D., Bernholdt, D., Chanchio, K. et al. 2003. An Ontology for Scientific Information in a Grid Environment: The Earth System Grid. *3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'03)*: 626-632.

Singh, I., Brydon, S., Murray, G., Ramachandran, V., Violleau, T., & Stearns, B. 2004. *Designing Web Services With the J2EE 1.4 Platform: JAX-RPC, SOAP, and XML Technologies*. Boston: Addison-Wesley.

Sor, R. 1999. Virtual Organisations - Not Such a New Idea: A Case Study of the Housing Construction Industry in Western Australia. Proceedings of the 10th Australasian Conference on Information Systems: 825-836.

Trastour, D., Bartolini, C., & Gonzalez-Castillo, J. 2001. A Semantic Web Approach to Service Description for Matchmaking of Services. *Proceedings of the International Semantic Web Working Symposium (SWWS)*: 447-462.

Walters, D. 2005. Performance Planning and Control in Virtual Business Structures. *Production Planning & Control,* 16 (2): 226-239.

Wiesenfeld, B. M., Raghuram, S., & Garud, R. 1999. Managers in a Virtual Context: The Experience of Self-Threat and Its Effects on Virtual Work Organizations. In C. L. Cooper, & D. M. Rousseau (Eds), *The Virtual Organization* : 31-44. Chichester, England: John Wiley & Sons.

Zhang, D. 2004-2005. Web Services Composition for Process Management in E-Business. *Journal of Computer Information Systems,* 45 (2): 83-91.

Zhang, D., Chen, M., & Zhou, L. 2005. Dynamic and Personalized Web Services Composition in E-Business. Information Systems Management, 22 (3): 50-65.

ACKNOWLEDGEMENTS

This paper was completed during my time as Visiting Researcher at the Peninsula School of Information Technology, Monash University, Melbourne. I would like to thank the staff there, especially Jana Polgar, for the advice they provided on aspects of Network Computing.