REALIZING THE VISION FOR WEB SERVICES: STRATEGIES FOR DEALING WITH IMPERFECT STANDARDS

Sanjay Gosain
4301, Van Munching Hall, Robert H. Smith School of Business,
University of Maryland, College Park, MD 20742, USA
Phone: (301) 405 3224
Fax: (301) 405 8655
Email: sgosain@rhsmith.umd.edu

ABSTRACT

Organizations considering the adoption of the web services framework for their Information Technology (IT) applications are confronted with a period of technological ferment, as standards for supporting non-trivial business process functionality are not yet in place. Evolving standardization poses challenges in the form of inter-temporal dependencies, as organizations' conformance to the standards that emerge in the future is contingent on their current design choices that need to be made ex-ante without complete information. At the same time, there are significant early-mover benefits to be gained by executing an IT strategy using web services as a cornerstone. This paper draws upon coordination theory to develop a conceptual framework outlining three approaches for organizations to deal with changing standardization regimes: (a) The dependencies across components, conforming to different standardization regimes, are continually bridged through intermediary services (e.g. using a protocol adapter that translates to an emergent standard), (b) The dependencies across components are minimized through loose coupling so that standardization regime changes for any component have a minimal impact on other components (e.g. encapsulating the functionality susceptible to design change into a module with abstract interfaces), and (c) The impacted components are rapidly reconfigurable as and when standardization regime changes (e.g., by building in "extension" features into applications). The risk for organizations investing in web services can be further managed by mechanisms such as organization's attention to signals from the periphery, undertaking low-risk experiments to learn in different areas, and bricolage-like improvisations of their legacy components at hand.

Keywords: Imperfect standards, web services, coordination theory.

INTRODUCTION

A Web Service (WS) is a software system identified by a Universal Resource Identifier (URI), whose public interfaces and bindings are defined and described using the eXtended Markup Language (XML). Its definition can be discovered by other software systems. These systems may then interact with the Web service in a manner prescribed by its definition, using XML based messages conveyed by Internet protocols (W3C 2002). Web services depend upon a number of standards governing web-based interactions that essentially enable applications to find and connect with other applications that provide a desired functionality. The vision for web services is for scaleable "snap-together" system-to-system communication. While there is great potential for web services, the standards that they represent are still imperfect and will be likely to mature over time. Although many vendors have agreed on a core foundation for HTTP and

XML-based Web services protocols and interfaces (e.g., SOAP, UDDI and WSDL[1]), this core set is only sufficient in trivial situations (Gartner 2002).

This paper presents a conceptual framework to suggest how enterprises may deal with the lack of a complete set of standards for building business applications that use web-services protocols. This paper is organized as follows. It starts by providing a brief introduction to the web services framework and its core standards. It then discusses some of the critical characteristics of technological standards such as path dependence, network effects, and tipping which provide insights into managing in an era of technological ferment. We then review the current state of standardization and conclude that the standards are still not complete and there is a potential threat of fragmentation for some high level standards. Next, a framework is developed to provide a typology of approaches to manage in a world of imperfect standards. This is followed up with details of approaches using this typology. Finally, we point to directions to further work.

UNDERSTANDING TECHNOLOGICAL STANDARDS

Technological standards are codified specifications that prescribe rules of engagement among components of a system (Garud & Kumaraswamy 1995). Technological standards have distinct characteristics that need to be understood in order to manage in their absence:

The structuration paradox – standards enable and constrain at the same time: Giddens' structuration theory has been widely applied to suggest a duality for information technology (IT) --it enables and constrains organizational practices at the same time and structure emerges as a result of repeated enactment (Orlikowski 1992). Technological standards have similar characteristics --they *constrain* the practices of organizations by inscribing rules that force adopters into specific trajectories of technology use. On the other hand, they also *enable* enterprises by equipping them with functional capabilities that arise from interoperability. Standards can be enabling, as they create an orderly framework within which economies of scale can be realized and technological progress can materialize, but they also create rigidities as they amplify the costs of any change that would require modification of the standard.
Adoption of technological standards generates network effects: Network effects refer to the utility that a user derives from consumption of the good increasing with the number of other agents consuming the good (Katz and Shapiro 1985). Standards create synchronization value for their adopters because of the additional value derived from being able to interact with other adopting organizations. A key characteristic of such product markets is that often only one technical standard is likely to prevail and the markets are prone to tipping --as one product gets ahead, it becomes progressively more attractive to the other adopters.

Interdependencies occur among standardization decisions across different layers/components: A natural progression in the layered development of information technology means that new standards come into place enabled by an underlying layer that gains maturity. For instance, the Web Services Choreography Working Group is chartered under the auspices of W3C to design a language to compose and describe the relationships between Web services. This standard is predicated on the existence of lower-level standards. Due to this hierarchical layering, the vertical industry standards have been the slowest to arrive. This also implies the existence of interdependencies in standardization decisions across layers as adherence to a specification in one layer may force the organization into a constrained choice for a different layer.

[1] The appendix provides a brief introduction to these protocols

Path dependence affects emergence of dominant designs: Path dependence refers to a process of economic allocation such that the outcome of the process does not depend only on a-priori determinants such as technology and factor endowments but on the specific history of the process. In essence, chance historical events shape future outcomes. Processes of technological change are known to be path dependent under three conditions (a) Technological interrelatedness of system components (b) Quasi-irreversability of investments, and (c) Positive externalities of increasing returns to scale (David 1985). Path dependence has been shown to arise when increasing returns to adoption are realized dynamically on the supply side through learning effects or on the demand side due to positive externalities (Arthur 1994). Given that technological standards for application connectivity meet these criteria, the emergence of dominant designs due to path dependence is likely. Thus, organizations may be locked into technological trajectories based on small accidental events that are magnified due to complex interactions among different organizations. However, Leibowitz & Margolis (1995) suggest that allocation processes may not be path dependent if organizations have foresight into the implications of their choices, or are able to coordinate individual organization choices through market interventions or communications, or internalize the network externalities.

Lastly, changing or evolving standardization implies inter-temporal dependencies: An organization embedded in a technological domain characterized by evolving standardization faces inter-temporal dependencies that need to be managed. Their current technical choices could be at variance with the dominant designs that emerge in the future and restrict the level of compatibility with new standards.

This discussion suggests that technological standards have important implications for how organizations make their technology choices. Next we look at the specific complexity of standardization in the web services space.

WHY IMPERFECT STANDARDS?

Markets for information goods may fail to coordinate on standards as few firms have the market power to set de-facto standards and most must accommodate the choices made by firms across the world. Standard setting is also prone to failure as interested stakeholders may be part of rapid changing and highly decentralized networks (Bailetti & Callahan 1995). Markets forces thus often fail to achieve standardization under oligopolistic competition in industries with network effects (Farrell and Saloner 1986a), and may even settle on a standard that is inferior in terms of overall social welfare.

Mobilizing a collective around common standards is inherently difficult as standards constitute a duality – they simultaneously constrain and enable organizational practices. New technologies will need to overcome the inertia in supplanting standards that enable existing technologies to work, while the new standard will also constrain their practices in the future (Garud, Jain & Kotha 2001). Vendors with their own versions of technical designs and associated investments may not acquiesce to giving up their market control in the interests of creating essentially a public good. IBM and Microsoft, for example, have been at odds with each other over the standards for web services choreography, leading to the threat of fragmentation (LaMonica 2003).

A rapid rate of technical progress leaves formal standardization efforts slow to catch up, if the standards are formulated by relatively slow moving and deliberate standard-setting bodies. In the case of web services, the underlying technologies are relatively new and still evolving –-with some apprehension that the technology evolution is still trying to catch up to the marketing hype (Thompson 2000). There is also a need to create consensus across multiple stakeholders among different organizations that are impacted by the standards. Industry analysts, for example, consider public UDDI registries a lost cause because they are "complicated and only generally defined" (Chappell 2002). Also, a number of different standard-setting bodies (W3C, IETF, OASIS, WS-I etc.) are involved with developing standards for different areas that impact the design, development and deployment of web services. This variety has been due to the different functional scopes, different time frames, and different approaches to protection of intellectual capital, among others. While standard-setting organizations help to mediate between different interests and technologies to impose a standard (Chiesa 2002), they are often slow to react to change and, in the absence of concrete implementations, may lack specification clarity. It may also be that the process of conceiving and delineating standards is inherently easier than the process of equipping current software to interpret them.

An alternate standard-setting approach is to let the marketplace decide in favor of a dominant design such as when there is no explicit technical basis for a choice. In this case as well, there is likely to be an interim time frame when competing and incompatible designs complicate the emergence of one dominant design. Technological change is not only driven by technical logic but dominant designs result from complex processes influenced by economic, social, political and organizational factors – "from a negotiated logic enlivened by actors with interests in competing regimes" (Tushman and Rosenkopf 1992). Given that web services have been extensively deployed by only a small number of early-adopter companies, a complete and dominant design is yet to emerge.

Lastly, there is significant heterogeneity across organizational contexts so that issues of semantic differences and contextual specificity may be very difficult to overcome, even over time, though XML namespaces and XML schemas provide some basis for structured extensibility.

The problem with imperfect standards is particularly an issue with web services because of a distributed deployment environment with no central authority, limited possibility of "out-of-band" communication, and increasing complexity in transaction-heavy environments. Next, we examine the current state of standards for web services applications.

THE CURRENT STATE OF WEB SERVICES STANDARDIZATION

Figure 1 illustrates the different layers in the web services application stack needed for enterprise applications (Figure 1).

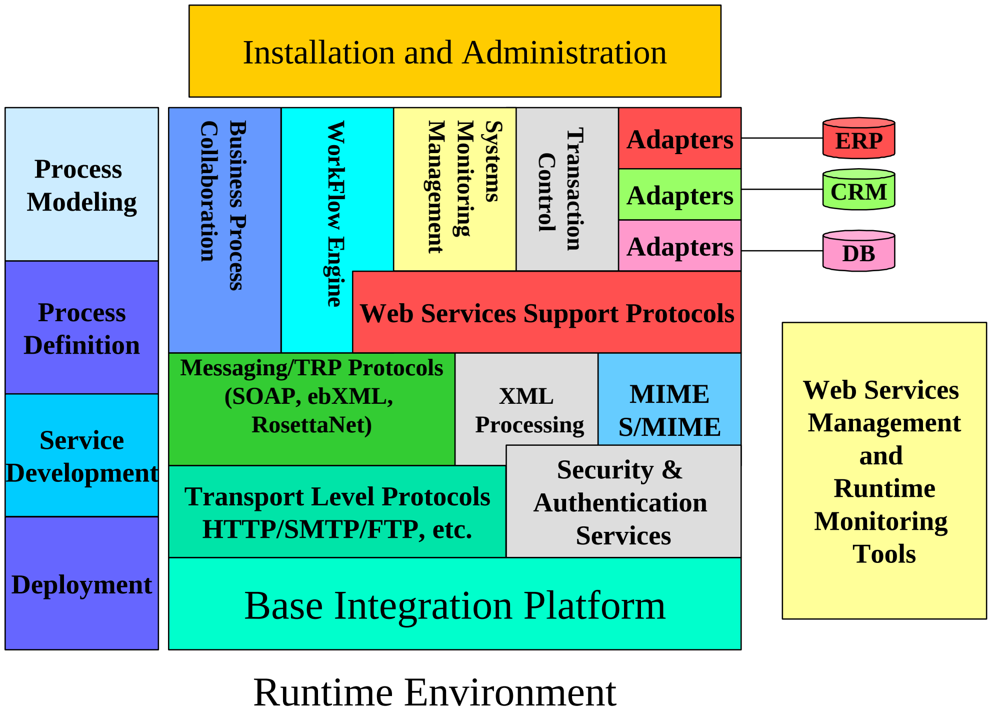

Figure 1. The Web Services Application Stack

Table 1 outlines some of the emerging standards in these different areas. It is apparent that there is a lack of consensus on a dominant standard for these different areas.

Technology	Description	Standard
1. Reliable Messaging	The process of building guaranteed data delivery into applications.	WS-R, WS-RM. Rosetta-Net and ebXML specifications.
2.Identity Authentication	The process of determining if someone or something is what it is declared to be.	WS-Security, PKI, Kerberos, Passport.
3. Authorization	The process of granting someone or something permission to do or have something.	SAML, XACML, LDAP, Active Directory.
4. Transport Encryption	The process of obscuring the connection between two end-points by applying specific security protocols at the transport layer e.g. HTTP/S.	SSL, S/MIME, TLS, VPN.
5. Message Encryption	The process of applying a cryptographic algorithm to obscure all or part of a message, e.g. a header or body element, an attached document, or the entire message.	XML-SIG, X.509, Symmetric keys.
6. Web Services Choreography or Orchestration	The process of specifying how Web services interact to form business transactions.	WSCL, WSCI, W3C BPML, ebXML BPSS, WSFL, Rosetta Net PIP, XAML, XLANG, BPEL4WS.
7. Provisioning	The process of defining communication between provisioning systems, users and resources.	SPML.

Table 1. Web Services Standards

Next, we propose a framework to guide organizational decision making in such a context, where the complete set of standards is not yet in place.

A FRAMEWORK TO DEAL WITH IMPERFECT STANDARDS

Domain with Imperfect Standards

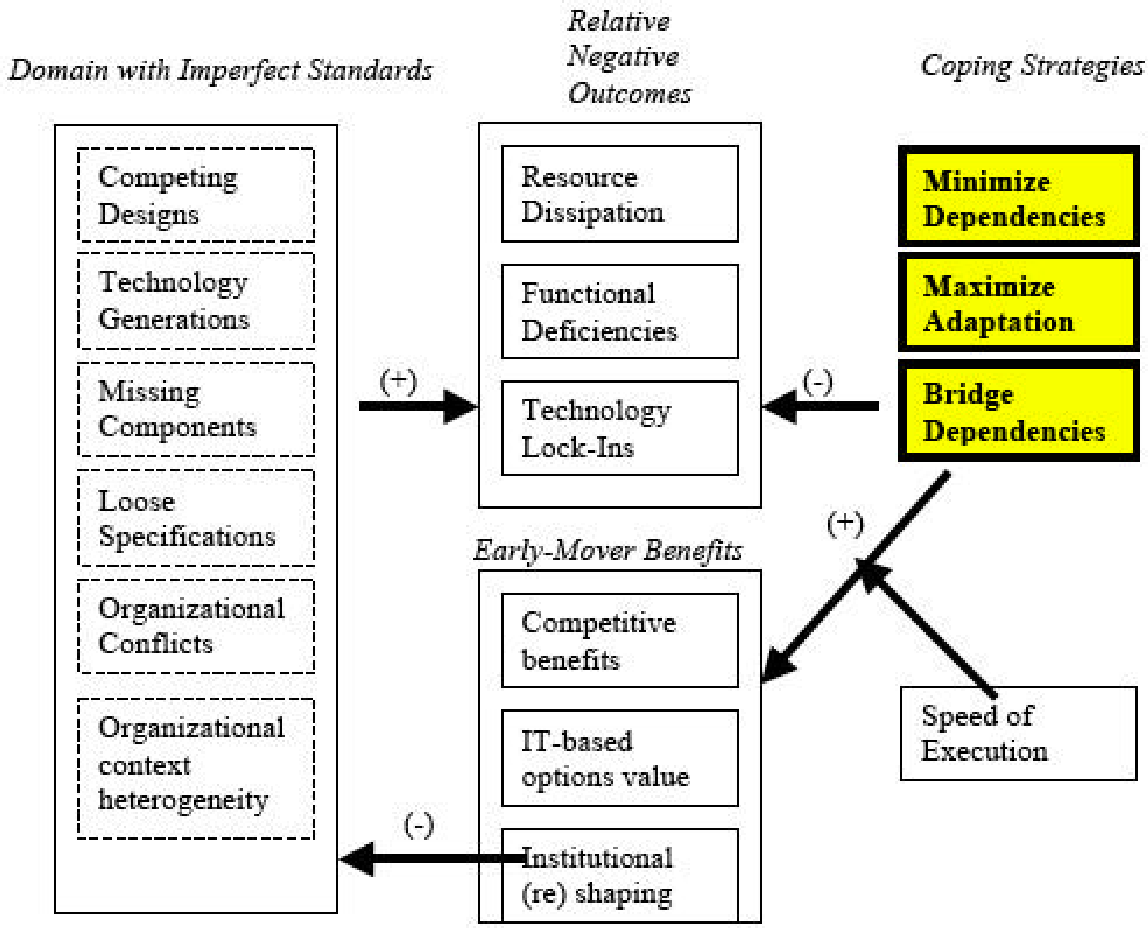

Figure 2. The Framework

The framework (Figure 2) illustrates why a technological domain can have imperfect standards and how that leads to negative outcomes for an organization. In essence, the web services applications stack can be seen as reflecting an organization's adoption of different high-technology products. The level of compatibility or the availability of complementary products (e.g., network externality considerations) form critical barriers for high-technology adoption (McCade et al. 2002). The lack of such compatibility can cause the organization to suffer from resource dissipation, functional deficiencies and lock-in effects. In case of web services there are a number of areas where clear standards are yet to emerge such as reliable messaging, security, identity authentication, distributed transactions, business process workflow, billing, payment, provisioning, monitoring and reporting. These contribute to the *relative* negative outcomes, compared to a base case where such standards exist.

The coping strategies allow these negative outcomes to be ameliorated to some extent. They also provide valuable benefits to the early movers, in terms of IT-based capabilities that have competitive implications, the generation of real options that have strategic value and increase an organization's flexibility, and, perhaps most importantly, allow the organization to influence the shaping of the standards.

IMPERFECT STANDARDS AND TECHNOLOGICAL FERMENT

Technological discontinuities trigger periods of technological ferment –a period in which there is experimentation and competition in a product class until a dominant design emerges as a synthesis of proven concepts (Tushman & Anderson 1986). During the era of ferment there is relatively little agreement on rules of engagement and criteria on which performance should be measured, and several technological trajectories may exist, each with its own distinct institutional environment. These institutional environments may further legitimize certain norms and operational concerns and can cause enterprises to be tied to specific boundaries of competences. Ultimately, rivals compete on architectural standards, which eventually congeal into a dominant design (Anderson & Tushman 1990). The figure illustrates such a period of ferment showing how rival technological trajectories and institutional environments may cause lack of convergence on technical standards.

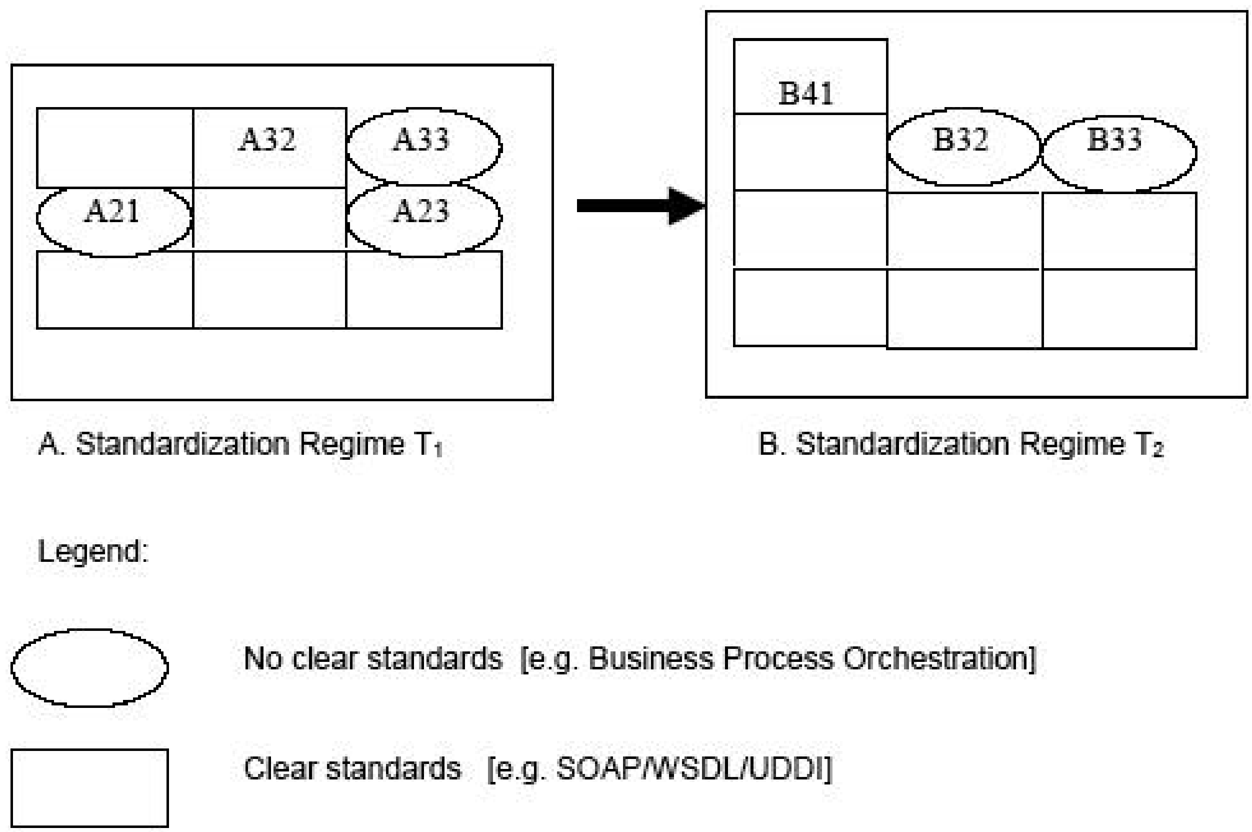

Figure 3. Evolution of Standardization

The figure illustrates how imperfect standardization regimes evolve through stages where a complete set of standards is not available for building web-service application components. At T_1, A21, A33 and A23 are layers in the architecture where there are no clear standards. By T_2, a clear consensus has emerged for the A21 and A23 so that they can now be built to standard specifications. However, a standard is still not available for B33, while a conflicting set of specifications has emerged for B32. Also, a new layer of standardization is enabled at T_2 and B41 becomes a new specification where there was no capability available earlier.

This stylistic representation illustrates the situation that organizations are confronted with. There is a greater possibility of fragmented standards at the higher layers while core specifications have emerged at the lower levels (Gartner 2002).

Organizations need to be able to upgrade design for layers where there are no current standards but where standards may emerge (A21), be cognizant of the layers where standards are unlikely to emerge (A33), be aware of the possibility of altogether new layers (A41) and also attend to potential mismatches that could result when competing designs emerge for an earlier standard (A32).

COORDINATION THEORY AND IMPERFECT STANDARDS

A standard may be conceptualized as a coordination mechanism around non-proprietary sknowledge that organizes and directs technological change (Reddy 1987). In the information technology arena standards enable coordination between technical systems. In an imperfect standardization regime, this mechanism breaks down. Coordination theory provides us with three main ways of achieving coordination (Table 1). In the absence of standards, these will provide us with alternate mechanisms to seek coordination across different technical systems.

We adopt a coordination theory perspective to understand how interoperability may be achieved among possibly mismatched application components. Coordination is the process of managing dependencies between activities (Malone & Crowston 1994). For web services applications there are dependencies between application components that need to interoperate and these may be designed to conform to different standardization regimes. Based on coordination theory (Table 2), we propose that the overall application can better deal with imperfect and *evolving* standards if:

(a) The dependencies across these components are continually bridged through intermediary layers as and when standardization regime changes.
(b) The dependencies across these components are minimized through loose coupling so that standardization regime changes in any layer have a minimal impact on other layers.
(c) The impacted components are rapidly reconfigured when standardization regime changes.

The residual risk from standardization regime changes not handled strategically through these approaches may be minimized through other organizational risk management approaches.

	Bridge Dependencies	**Minimize/Uncouple Dependencies**	**Maximize Adaptation**
March & Simon (1958)		Coordination by plan	Coordination by feedback
Lawrence & Lorsch (1967)	Integrative Subsystems, Specialists in integration, Integrating teams		
Van de Ven et al. (1976)		*Impersonal mode* - preestablished plans, schedules, formalized rules, policies and procedures, and standardized information and communication systems	*Personal mode* and *group mode* (mutual task adjustments through vertical or horizontal channels of communication)
Galbraith (1973)		Rules, programs & procedures, common understanding of targets & goals	Creation of slack resources, self-contained task groups, information systems, lateral linkages
Daft & Lengel (1986)	Integrators	Plans, formal information systems, rules & regulations	Direct contact, group meetings
Nadler & Tushman (1989)	Liaison role, integrator role, matrix structure		
Mintzberg (1989)	Direct supervision	Standardization of work processes, outputs, skills & knowledge, norms	Mutual Adjustment
Crowston (1991)		Interdependencies between tasks, interdependencies between task and object, interdependencies between objects	
Gittell (2002)	Boundary spanners	Routines	Relational Coordination

Table 2. Coordination typologies mapped to three pathways

STRATEGIES TO BRIDGE DEPENDENCIES

Use of Protocol Converters (Gateways/Bridges/Proxies)

This strategy relies on the existence of gateways that take information available via one protocol and make it available via another. The availability of converters, translators, emulators, and other ‘gateway technologies’ that achieve compatibility ex post, serve to reduce the social costs of failure to standardize *ex ante* (David and Greenstein 1990). The cost of providing converters would be influenced by the variety of distinct technical systems that need to be made interoperable, as well as by the number of dimensions in which alternative technological designs diverge. In Electronic Data Interchange (EDI) applications, such protocol converters are often used to translate incoming messages to the formats of internal business applications. In case of Internet applications, proxy services may be used to tunnel through firewalls or to provide conventional HTTP interfaces on unconventional transports. Intelligent adapters may also encapsulate business rules and error handling routines to enable them to provide robust protocol conversion.

Glue Services

This strategy relies on services that provide the resources to convert to components that adhere to different protocols. For example, namespaces and resource registries that provide interface characteristics may make it easier to adjust to different protocols, or context translation intermediaries may help to overcome semantic heterogeneity issues.

As an example, web services lack standards for semantics and thereby the information exchanged among services may have inconsistent meanings. A context mediation framework that provides a domain model to define rich types, and context definitions to define different interpretations of types in each web service (e.g. MIT COIN project - Madnick 1999; Bressan et al. 2001) would help to avoid potential semantic conflicts. The W3C is also pursuing the "semantic web" initiatives on similar lines.

Wrappers

This approach involves creating a layer of code that provides a web-service interface that meets the needs of a specific standardization regime. As an example, a major retail chain manages its gift card usage and balance on an IBM 390 mainframe in a DB2 database. An application server platform on a Unix server is used to build Enterprise Java Bean (EJB) wrappers for several services such as gift card balance, redemption and authorization. The EJBs talk through an adapter to the mainframe application and are exposed as web services.

STRATEGIES FOR MINIMIZING DEPENDENCIES

Modular Design

Modularity refers to the extent to which system components may be produced separately and used interchangeably in different configurations without compromising system integrity (Garud & Kotha 1994). The power of modular design arises because they allow flexible responses to change --evolution favors systems consisting of stable sub-assemblies that may be discarded or modified in response to environmental change (Simon 1962). Modularity allows for environmental disturbance to be confined to specific modules and for component innovations to be easily integrated with the overall system (Orton & Weick 1990).

In contrast to integrated architectures, modular architectures provide many entry points for firms to design complementary and compatible components. Firms benefit from the economies of substitution in modular systems as the cost of designing a higher performance system through a partial retention of existing components is lower than a complete redesign (Garud & Kumaraswamy 1996).

The modularity principle suggests that proprietary extensions to fill in the gaps in the features of web services should be implemented in a modular manner with clearly defined interfaces. Organizations should be wary of developing hybrids of applications with interdependent pieces that are standards-compliant and proprietary. Another implication of the modularity principle is to design web services without "chatty" protocols. As an example, services should not expose interfaces that make small updates to data elements, but should update a transformed element in a single service call (Burner 2003). The modularity principle also suggests that higher layers in the application stack (e.g. those focused on business processes) be also expressed in modular terms so that they can be easily recombined and reused, and limit the locus of changes that may be needed as standards evolve.

Abstracting from the implementation level

Interfaces refer to linkages shared among components of a given product architecture. Interface specifications provide the protocol for fundamental interactions across components. As the figure shows, an abstract interface can be specified for minimizing dependencies with external components, and different concrete implementations can be used to vary the actual implementation details.

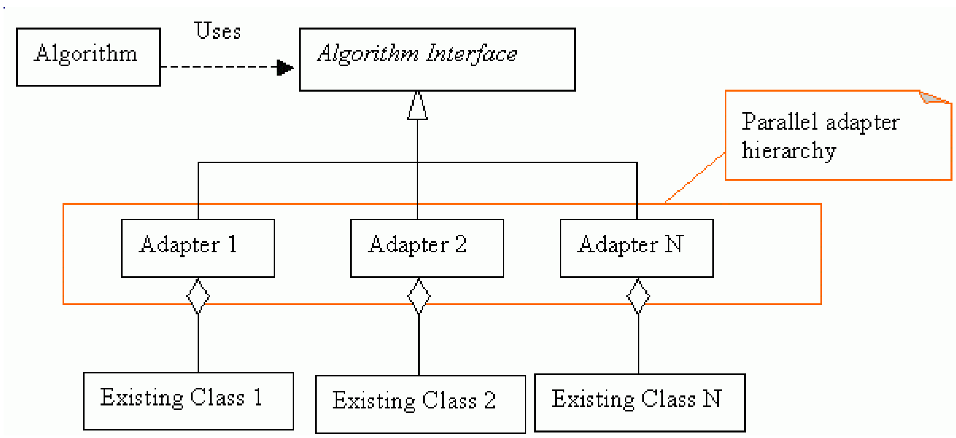

Figure 4. Abstracting Implementation Details.

[Source: http://www.shindich.com/sources/patterns/implied.html]

In the case of web services, through the abstraction idea, many different network endpoints can be specified to implement the same service contract and a client can select the best endpoint at run-time. Another important design implication is that organizations should "separate out" functions devoted to different areas – thus the business logic should be specified in abstract terms that are independent of the specific technology platforms.

Buffering through products

Commercial software products may provide products that wrap around a standard that may be susceptible to change or difficult to use. For instance, the XML DOM (Document Object Model) provides application programming interfaces (APIs) that developers use to manipulate XML documents. However, these are cumbersome and difficult to use. Software vendors provide Java toolkits that are simpler and more comprehensive. The task of dealing with XML and DOM specifications then falls to the toolkit vendor, thus buffering the organization from change. Vendors are then expected to provide clear migration paths as the level of standardization evolves.

Most major web services platforms provide services such as software developer kits that wrap soap functionality in a class library as well as toolkits that generate WSDL interfaces, web service containers and client proxies from existing applications. In addition these platforms may also support the ability to manage, deploy, maintain, monitor and test web service end-points.

As standards evolve, it is expected that these would get encapsulated into specific services offered by such platforms. As an example, following on the release of the BPEL4WS Web services orchestration standard to describe business processes, toolsets have been developed to enable the modeling and documentation of business processes using a rich graphical modeling environment.

Despite the clear advantages of using such products, there may remain interoperability issues that need to be addressed. For instance, there are a number of commercial tools that provide SOAP toolkits in the form of libraries to prepare XML-based SOAP envelopes, send the envelopes over a transport protocol such as HTTP or SMTP, and process incoming SOAP envelopes (Hong 2001). There have been found to be interoperability issues among these toolkits because they may implement only part of the SOAP or XML specification or implement ambiguous parts of the specification differently or not send optional information that is expected by the other toolkit (e.g., type information for encoded parameters).

STRATEGIES TO MAXIMIZE ADAPTATION

Intelligent controllers

This strategy relies on a hierarchical nature of control whereby layers at a higher level in the hierarchy are able to coordinate across differing protocols at the lower level. Thus changes in standards can be more easily responded to by augmenting the capability of this super-layer.

Web-services based applications adapt themselves to changing interfaces, by downloading the latest WSDL interfaces and adapting its interface appropriately. But this is not always possible. In case of more drastic changes, they may employ a Service Broker architecture that scans incoming requests and informs the older interface invocations of incompatibility, by giving an appropriate response. Third-party services can also dynamically scan WSDL schemas looking for changes in their structures. (Irani 2001).

Another approach would be to build in management points (figure) that use rules to provide protocol translation and migration to cope with a change in standardization regime.

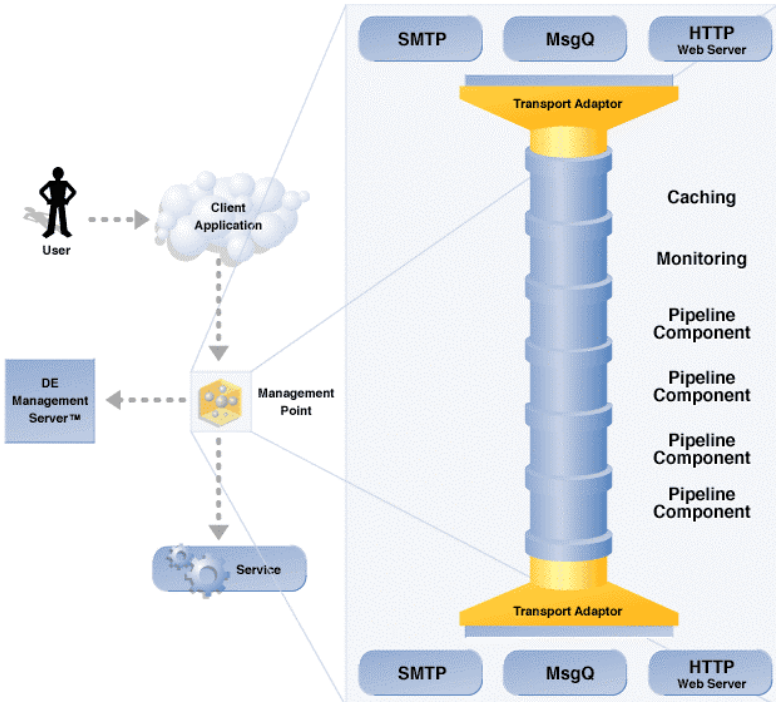

Figure 5. Intelligent Control Example. Source: Digital Evolution.

Architectural conformance

Product architecture refers to the arrangement of the elements of a product into its physical building blocks, including the mapping from functional to physical elements and the specification of interfaces among interacting physical components (Ulrich 1995). Through conformance to the underlying architecture of a system, firms can guarantee enough compatibility for customers and suppliers to upgrade from one product generation to the next. Innovations and improvements may then be embodied in the components.

One of the ways of instituting architectural conformance is through the use of design patterns. Patterns provide structure by specifying the high-level logical components and their interactions needed to implement business functions. As an example, before standards such as WSDL, UDDI and SOAP came along, companies like Google had been exposing their site information through XML-over-HTTP. This prepared the organization to then move on using the standards when they became available, since there was a fundamental conformance to the overall loosely-coupled architecture.

Specific design patterns can also be used to deal with versioning issues that arise when standardization domains change. Burner (2003) suggests a "web service façade" pattern where version numbers are encoded into a URI and when new versions of a service are published they do so in a namespace created by incrementing the version number. This protects clients written to an old contract trying to call a new service.

Upgradability of components

Organizations need to consider the upgradability and extensibility of components designed using existing standards that may be subject to change. As an example, web services applications that operate with the remote procedure call (RPC) style tend to be fragile as development tools simply wrap application objects that may not have been meant to be exposed as public APIs. On the other hand, XML-document-type web services are more robust and allow for evolution. An application may only work with a portion of the document and ignore others allowing the document and the system to evolve independently (Rourke 2003).

Given that XML schemas are core to the deployment of web services, it is important to ensure that general-purpose schemas be developed, anticipating the need for extensibility. A useful approach can be based on the use of "extensions" that modify a base data type in an XML schema definition. This will allow applications to extend data with custom properties.

Organizational Capabilities

In order to respond quickly to technology standards once they are instituted, organizations need to understand the current state of technology, the gaps in standards, and architectural break-points where standards are likely to emerge across time. They need to prepare their own organization as well as partners to quickly take advantage of standards as they gain maturity and, if possible, even shape them. XML schema definitions are particularly key to web services and an important step is to research and keep track of the work on schema definitions for the organization's industry. Consumer expectations are also critical in shaping the adoption of products with network effects (Gandal 2002) –hence the organization should also carefully monitor shifts in consumer perceptions on web-services design.

An important caveat needs to be highlighted as organizations monitor the technological changes. An obvious strategy for technology providers in network markets that are experiencing intense supply-side competition is to pre-announce their products. This is based on the expectations of self-fulfilling prophecies –that the illusions held out for consumers will be rationally acted upon resulting in their concretization. Incumbents may also make strategic announcements about forthcoming releases to cloud the prevailing incentives that favor switching (Farrel & Saloner 1986b). Thus organizations should be careful of "vaporware" claims in making their web-services decisions.

Organizations should be mindful of the layered nature of web services applications. Conformance at lower layers may particularly help to clarify the scope of incompatibilities at the higher layers. For instance, XML as a standard eliminates incompatibilities in the parsing of data that is exchanged across applications –-a developer does not have to rewrite a parser for each application that accepts data from other applications. At the same time, it does not overcome the problem of semantic heterogeneity. However, it does make the problems of semantic differences much clearer at a higher level as they are no longer obfuscated by syntactic issues.

Organizations also need to be actively engaged with the progress made by standard-setting bodies. Non-market relationships, such as those embodied in industry standard-setting bodies influence the process of technology commercialization (Aram et al. 1992) and organizations need to be sensitive about the direction and shaping of such inter-institutional efforts. In the adoption space, organizations need to be aware of the extent of adoption for competing designs in different technology layers and be alert to likely "tipping points" that would be followed by bandwagon adoption.

Organizations need to be particularly engaged in schema definitions and business process specification activities in their industry and be able to incorporate common specifications in their web-services applications.

Organizations need to conceive their IT architecture as a service-oriented architecture (SOA) --a method of designing and building loosely coupled software solutions that expose business functions as programmatically accessible software services for use by other applications through published and discoverable interfaces (Adams et al. 2002). An SOA also separates application functionality into specialized tiers dealing with presentation, business logic and persistent storage. A business can use an SOA to compose and organize applications from a collection of distributed services; thus they can use it to construct new applications and alter existing ones by reusing their own assets as well as the business functions of their partners. Web services represent one implementation of a service-oriented architecture, but not all SOA applications can be considered Web services. Organizations that have already prepared for the SOA would be able to leverage web services standards as they are unveiled.

POINTERS TO FURTHER WORK

The framework presented in this paper has important implications for managers designing their IT applications as well as researchers interested in IT infrastructure issues. Due to the word limit specified for the workshop, this paper does not present the implications that follow from the framework presented earlier. The insets provide a brief outline of the direction of theorizing that builds upon the ideas presented in the paper. This will be more fully developed in the complete paper.

Inset 1. Legacy Components & Change: Strategy as *Bricolage*

Bricolage (origin: French, 'doing odd jobs') refers to the process of using the materials on hand to create a response to a task (Levi-Strauss 1966). The bricoleur engages in continual reconstruction from the same materials. By being open to and trying out new uses of an object, a bricoleur develops a rich understanding of the object and is able to improvise in its use (Weick 1993). The bricoleur conducts practical experimentation followed by thoughtful modifications, is mindful of when to change or when to persist with existing configurations, and is cognizant of the fact that she is placing bets that have probabilistic outcomes (Garud & Karnoe 2001). The bricolage process links the observations that result from probing the real world to the cognition that conceptualizes how changes will affect the emergent order.

In a world of technological ferment, the bricolage strategy suggests:

(a) Conceptualize technology artifacts in multiple ways and be open to their use in different ways as technology and standards evolve.

(b) Loosely couple experiments/actions and emerging outcomes in the real world.

(c) Be “mindful” of resources, embedded in earlier institutional regimes, that can be reused

Inset 2. Attending to Signals from the Periphery

The lack of well-defined standards in the web services space may be conceptualized as due to emerging technology (high level of uncertainty and embryonic industry structure). Day & Schoemaker (2000) suggest that emerging technologies signal their arrival long before they grow into full-fledged commercial successes. However, managers often are unable to filter out the noise that obfuscates the signals that inform about the future potential and direction of these technologies. An important reason for this lies in the mental models employed by managers. These mental models that result from a history with current technologies may be situated in past experience and reinforced by ongoing commitments. On the other hand the signals from new technology are only going to be sensed if the managerial mindset is attuned to looking past the disappointing results, limited functionality, and modest initial applications to anticipate the future possibilities. The weak signals to be captured usually come from the periphery, where new competitors are making inroads, unfamiliar customers are participating in early applications, and unfamiliar technology or business paradigms are used. However, the periphery is very noisy, with many possible emerging technologies that might be relevant. In order to cultivate a mindset that will be attuned to capturing these weak signals at the periphery, an organization requires:

- Openness to a diversity of viewpoints within and across organizational units,
- Willingness to challenge deep-seated assumptions of entrenched mental models while facilitating the forgetting of outmoded approaches, and
- Continuous experimentation in an organizational climate that encourages and rewards "well-intentioned' failure.

APPENDIX
The Building Blocks of Web Services

Essentially, web services provide programmatic access to business functionality using the following Internet protocols:

XML (eXtensible Markup language): ***The lingua franca of Web services***
A universal standard for identifying content and specifying the structure of data in documents. Makes it easier to exchange data among applications and to validate data elements. XML Schema is a format for describing XML data types and allows applications exchanging XML documents to parse and validate data elements. XML standards are widely accepted support by vendors in terms of tools that allow XML documents to be manipulated.

SOAP (Simple Object Access Protocol): ***Cleaning up the exchange***
A set of rules that facilitate XML exchange between applications. Along with WSDL, SOAP performs message transport functions. SOAP describes envelope and message formats, and has a basic request/response handshake protocol.

WSDL (Web services Description Language): ***Accessing the service***
Provides a common framework for describing tasks performed by a web service and a way for service providers to describe the basic format for requests made to their systems regardless of the underlying protocol (such as Simple Object Access Protocol or XML) or encoding (such as Multipurpose Internet Messaging Extensions). With WSDL, it is possible to automate the generation of proxies for Web services in a truly language- and platform-independent way.

UDDI (Universal Description, Discovery, and Integration): ***The Yellow Pages***
A set of specifications for creating XML based directories of web services offerings. Users and applications may discover and locate web services through UDDI repositories. UDDI is an independent consortium of vendors, founded by Microsoft, IBM, and Ariba, for the purpose of developing an Internet standard for Web service description, registration and discovery.

ACKNOWLEDGEMENTS

Financial support for this study was provided by the Center for Telecommunications Management (CTM) at the University of Southern California. I also gratefully acknowledge the valuable suggestions of Prof. Omar A. El Sawy (CTM) that shaped this study, as well as the research assistance of Salil Sawhney and the inputs shared by Eric Pulier and Alistair Farquharson (Digital Evolution).

REFERENCES

Adams, H., Gisolfi, D., Snell, J., and Varadan, R. (2002). Service oriented architecture, Web services, and IBM patterns for e-business, November. [Online]. Available: http://www-106.ibm.com/developerworks/webservices/library/ws-best2/

Aram, J. D., Lynn, L. H., Reddy, N. M. (1992). Institutional Relationships and Technology Commercialization: Limitations of Market-Based Policy. *Research Policy*, 21(5), pp. 409-422.

Arthur, W. B. (1994) *Increasing Returns and Path Dependence in the Economy*, Ann Arbor: University of Michigan Press.

Anderson, P., and Tushman, M.L. (1990). Technological discontinuities and dominant designs: a cyclical model of technological change, *Administrative Science Quarterly*, 35, pp. 604-633.

Bailetti, A.J., Callahan, J.R. (1995). Managing Consistency between Product Development and Public Standards Evolution, *Research Policy*, 24(6), pp. 913-931.

Bressan, S., Goh, C., Levina, S., Madnick, S., Shah, A., and Siegel, M. (2000). Context knowledge representation and reasoning in the Context Interchange System, *Applied Intelligence*, 13 (2), pp. 165-179.

Burber, M. (2003). The Deliberate Revolution --Transforming Integration With XML Web Services? *ACM Queue*. Available: http://www.acmqueue.org/issue/burner1.cfm?client_no=NEW.

Chappell, D. (2002). Opening keynote presentation for the June 2002 *XML-Web Services One conference*, San Jose, CA.

Chiesa, V., Manzini, R., and Toletti, G. (2002). Standard-setting processes: Evidence from two case studies, *R & D Management*, 32 (5), pp. 431-450.

Crowston, K. (1991). Towards a Coordination Cookbook: Recipes for Multi-Agent Action. Unpublished doctoral dissertation, MIT Sloan School of Management.

David, P.A. (1985). Clio and the Economics of QWERTY, *American Economic Review*, 75, pp. 332-337.

David, P., and Greenstein, S. (1990). The Economics of compatibility standards: An introduction to recent research, *Economics of innovation and new technology*, 1, pp. 3-41.

Day, G.S., and Schoemaker, P.J.H. (2000). Avoiding the pitfalls of emerging technologies, *California Management Review*, 42(2), pp. 8-33.

Farrell, J., and Saloner, G. (1986a). Standardization and Variety, *Economic Letters*, 20, pp. 71-74.

Farrel, J., and Saloner, G. (1986b). Installed base and compatibility: Innovation, product pre-announcements and predation. *American Economic Review*, 76, pp. 940-955.

Galbraith, J. (1973). *Designing complex organizations*, Addison-Wesley, Reading, MA.

Gandal, N. (2002). Compatibility, standardization, and network effects: Some policy implications, *Oxford Review of Economic Policy*, 18 (1), pp. 80-91.

Garud, R and Karnoe, P. (2001) "Path creation as a process of mindful deviation", in Garud, R and Karnoe, P. (eds.) *Path dependence and Creation*, Lawrence Earlbaum Associates, pp. 1-38.

Garud, R., and Kumaraswamy, A. (1994). "Coupling the technical and institutional faces of Janus in network industries", in R. Scott and S. Christensen (Eds.) *Advances in the Institutional Analysis of Organizations: International and Longitudinal Studies*, Thousand Oaks, CA: Sage Publications, pp. 226-242.

Garud, R. and Kumaraswamy, A. (1995). Technological and organizational designs for realizing economies of substitution. *Strategic Management Journal*, 16, 93-109.

Gartner (2002). Will web services standards ever happen. *Gartner Report* by M. Pezzini,.

Gittell, J.H. (2002). Coordinating mechanisms in care provider groups: Relational coordination as a mediator and input uncertainty as a moderator of performance effects. *Management Science*, 48 (11), pp. 1408-1426.

Hagel III, J. (2002). Edging into Web Services. *The Mckinsey Quarterly*, 4.

Hong, T. Advancing SOAP Interoperability. IBM Developer Works. [Online] Available: http://www-106.ibm.com/developerworks/webservices/library/ws-asio/

Accessed March 17, 2003.

Irani, R. (2001). Versioning of Web Services - Solving the Problem of Maintenance. [Online]. Available: http://www.webservicesarchitect.com/content/articles/irani04.asp

Katz, M., and Shapiro, C. (1985). Network Externalities, Competition, and Compatibility, *American Economic Review*, 75, pp. 424-440.

LaMonica, M. (2003). Microsoft leaves standards group. ZDNet USK News. [Online]. Available: http://news.zdnet.co.uk/story/0,,t269-s2132448,00.html.

Accessed March 31, 2003.

Lawrence, P., and Lorsch, J. (1967). Organization and environment: Managing differentiation and integration, Division of Research, Harvard Business School, Boston.

Leibowitz, S.J., and Margolis, S.E. (1995). Path dependence, lock-in and history. *Journal of Law, Economics and Organization*, 11 (1), pp. 204-226.

Levi-Strauss, C. (1966). *The savage mind (La pensee suavage)*. London, UK: Weidenfeld and Niolson.

Madnick, S. (1999). Metadata Jones and the Tower of Babel: The challenge of large-scale heterogeneity, *Proc. IEEE Meta-Data Conf.*, April 1999.

Malone, T.W., and Crowston, K. (1994). Towards an Interdisciplinary Theory of Coordination, *Computing Surveys*, 26 (1), 1994.

March, J. G., and Simon, H.A. (1958). *Organizations*, John Wiley & Sons, New York.

McCade, S.R., Olivia, T.A., and Pirsch, J.A. (2002). The organizational adoption of high-technology products "for use": Effects of size, preferences, and radicalness of impact, *Industrial Marketing Management*, 31(5), pp. 441-456.

McGrath, R.G. (1997). A Real Options Logic for Initiating Technology Positioning Investments, *Academy of Management Review*, 22, pp. 974-996.

Mintzberg, H. (1989). *Mintzberg on management*, The Free Press, New York.

Orlikowski, W. J. (1992). The duality of technology: Rethinking the Concept of Technology in Organizations. *Organization Science*, 3 (3), pp. 398-427.

O'Rourke, C. (2003). Waiting for web services, *Oracle Magazine*, March/April, p. 61.

Reddy, N.M. 1987. Voluntary product standards: Linking technical criteria to marketing decisions, *IEEE Transactions on Engineering Management*, 4, pp. 236-243.

Sanchez, R. and J. T. Mahoney. 1996. Modularity, flexibility, and knowledge management in product and organization design. *Strategic Management Journal*, 17, 63-76.

Simon, H.A. (1962). The architecture of complexity. *Proceedings of the American Philosophical Society*, 106, pp.467-482.

Thompson, J.D. (1967). *Organizations in action: Social science basis of administrative theory*, McGraw-Hill, New York.

Thompson, H.S. (2000). Web Services and the semantic web – Separating hype from reality. University of Edinburgh presentation.

Tushman, M. L., and Anderson, P. (1986). Technological Discontinuities and Organizational Environments, *Administrative Science Quarterly*, 31(3), pp. 439-465.

Ulrich, K. (1995). The role of product architecture in the manufacturing firm. *Research Policy*, 24(3), pp. 419-440.

Van de Ven, A.H., Delbecq, A.L., Koenig, R. (1976). Determinants of Coordination Modes within Organizations, *American Sociological Review*, 41, 1976, pp. 322-338.

Weick, K. (1993). Organization redesign as improvisation. In *Organization change and redesign*. G.P. Huber & W.H. Glick (eds.) Cary, NC: Oxford University Press, pp. 347-379.

Wouters, M.J.F., Sharman, G.J., and Wortmann, H.C. (1999). Reconstructing the sales and fulfillment cycle to create supply chain differentiation, *International Journal of Logistics Management*, 10 (2): 83-98.

W3C (2002). Web Services Architecture. W3C Working Draft 14 November 2002. [Online]. Available: http://www.w3.org/TR/ws-arch/

Author biography

Sanjay Gosain is assistant professor of information systems at the Robert H. Smith School of Business at the University of Maryland, College Park. He received his PhD in information systems from the University of Southern California. His current research interests are in the area of IT infrastructure design, particularly for inter-enterprise applications. He was part of the research team working with the RosettaNet consortium on issues related to common electronic interface specifications for the IT industry.