Ten-Step Survival Guide for the Emerging Business Web[1]

Aad van Moorsel
Hewlett-Packard Laboratories,
Palo Alto, California, USA
aad@hpl.hp.com

Webservices technology is converging, and today we are at least able to define what we mean if we use the term webservice (SOAP, XML, WSDL). Given the maturing technology, it is opportune to get concrete about the future of webservices-based technologies. An area that traditionally has been assumed to become a major beneficiary of webservices technology is that of business-to-business interactions. In this paper we try to get to the core issues we face in creating this emerging 'business web,' these dynamic, digital business ecosystems.

For the reader's entertainment, we do this in the form of a 10-step survival guide, each step being a technology 'invariant,' that is, a statement about the future business web that we expect to remain true for considerable time to come. Our hope is that this will provide you with enough insides to find your way among all the hype in the emerging business web, or at least allow you to survive a variety of water cooler conversations in the years to come. In addition, while going through the 10 steps we uncover the principles of the architecture that will support the future business web.[2]

Our 10 invariants are the following:

1. IT's all about business, stupid
2. Let's talk, but no deep conversations, please
3. Standards drive the industry—interoperability drives standards
4. Webservices: the final layer in the Internet stack
5. A planetary business web emerges: contracts, contracts, contracts
6. The coming of semantic disasters
7. The core technology issue: multi-party conversations
8. Management goes incognito
9. The business web will be as impaired as the society that creates it
10. The business web will happen!!

---

1 Opinions and intuitions expressed in this invited keynote address at CaiSE's workshop on Web Services, E-Business and the Semantic Web, are the author's and do not necessarily reflect Hewlett-Packard Company's position.

2 Throughout the text we use ■ to end paragraphs in which we introduce pieces of the business web architecture, see also Fig 1 through 6.

1. IT's All About Business, Stupid

In [1], Garbani puts it very succinctly: "IT, in any enterprise, exists solely for the purpose of supporting the business processes." That is, information technology has no intrinsic purpose, but becomes relevant only through what it is used for—in this paper we discuss technology used to automate tasks in a business, be it a Fortune 500 company, a small business or a service provider.

In the technology picture of the business web we develop in this paper, the core abstraction is therefore that of 'business,' instead of service, resource, etc. The business processes of a business drive the IT requirements, and information services, resources and web pages only exist in relation to a business, never in isolation. The business web, then, automates various aspects of interacting between businesses, as illustrated by Fig. 1. This figure depicts a variety of businesses, not only customers and providers, but also IT service providers and management service providers. ■

The Internet has generated three ways of doing business that bring IT closer to business activities than ever before: B2C, B2B and service providers [2]. As a consequence, IT managers are more and more faced with making investment decisions that influence a business' bottom line in directly demonstrable ways [3]. To quote Lou Gerstner from IBM: "It's gotten to the point where it's almost impossible to distinguish between the business strategy and the IT strategy of any successful enterprise. Approximately half of the investments that customers make in IT are now driven by line-of-business managers, not chief information officers." This will force technologies to be developed that manage systems based on "quality of business" [2] instead of quality of services considerations. That is, we will need to execute on business-goal driven management [2] instead of unguided, and potentially endless, (self-) management. The realization that *IT is all about business* is thus more critical than ever in driving technology and technologists. Major kudos await the engineers, scientists and companies that are able to bridge the gap between IT and business.

2. Let's Talks, but No Deep Conversations, Please

Enterprises are eager to adopt electronic execution of business activities because they want predictable, reliable and speedy execution of their business processes, and want to outsource functions that are not core to their business. At the same time, they want to retain control over their assets and activities, and be secured against malicious attacks on their electronic business. As a consequence, enterprises need coupling, but they want the coupling to be as loose as possible.

From a technology perspective, what, then, could be better than relying on proven and pervasive technology (Internet, HTTP), and add to that some richness in the document contents and exchange protocols through the introduction of SOAP, XML and WSDL [4]? No new holes in the firewall, predictable and reasonable robustness, easy integration in legacy systems, and straightforward extensibility for the future. This is exactly what webservices establish. As a consequence, the distributed computing technology is so unsophisticated it makes a computer scientist's hard bleed. This 'simplic-

'simplicity'[3] of the technology, though, is key to its widespread acceptance. Attempts to fight this have failed, not only in B2B computing; see for instance the variety of advanced technologies, such as ATM, CORBA or atomic broadcast, that did not break through because they were too complex to gain acceptance.

The state of the art in B2B computing can best be compared with having conversations using letters (or e-mails) instead of through direct verbal conversation. A business sends an XML 'letter' to its partner, who parses it and executes on what it reads, following prescribed rules. If necessary, the receiver replies by sending an XML letter back, etc. The bottom line is that few messages go over the wire, and that the interaction pattern is straightforward. The process of sending letters is awfully tedious for humans, but computers have no particular objections against parsing long documents. Besides, although software implementers of such business interactions have a laborious job to do, the work is relatively straightforward and needs relatively little training.

All those elements together explain why webservices are becoming pervasive as a technology. They play by the golden rule behind any business web technology: *businesses want to talk, but don't want deep conversations, please*.

3. Standards Drive the Industry—Interoperability Drives Standards

It has been said that standards made the Internet [5], and there is hardly any area in which standards dictate progress so heavily as in webservices (W3C and OASIS are but two examples of standardization bodies [4]). If you want to make money with a service on the Internet, you have to adhere to webservices standards, otherwise no potential partners can connect with you. That implies that providers will use implementation platforms that are webservices compliant, and it implies they will purchase solution offerings from those that know how to embed their solution in webservices standards. Obviously, to be perceived as a leader and to be able to be quick to market with implementations, it is beneficial for technology companies to lead standardization efforts.

To put this in context, let us develop our model of the business web one step beyond businesses and business processes. Each activity in a business process is executed as a service, which is accessed based on webservices standards. This model is recursive, in that each service may be implemented through a business process, which activities are executed through accessing a service, etc. [6]. So, the business abstraction we started from materializes in information technology through an 'everything is a service' paradigm. ■

[3] We hesitate to try to define in detail the intuitive notion of 'simplicity,' but to allow scientific scrutiny we attempt: technology is 'simple' if it executes some task well, but in a 'minimal' way, by ignoring in the design sophisticated system/execution properties and certain future usage patterns.

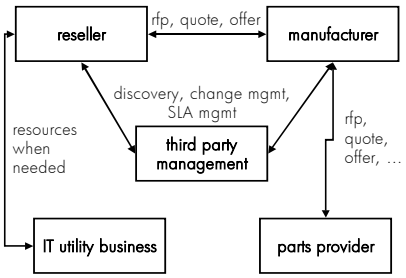

Fig. 1 Business as core abstraction. The boxes are examples of businesses, the arcs illustrate examples of interactions that are (partly) automated through the business web.

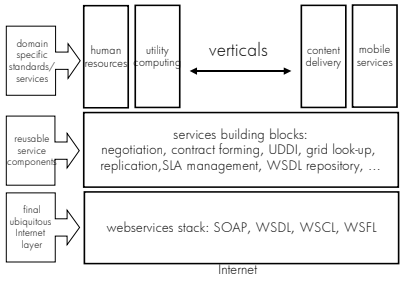

Fig. 2 Webservices: the last ubiquitous layer in the Internet stack. Service building blocks and domain-specific ontologies will be developed on top of the webservices layer.

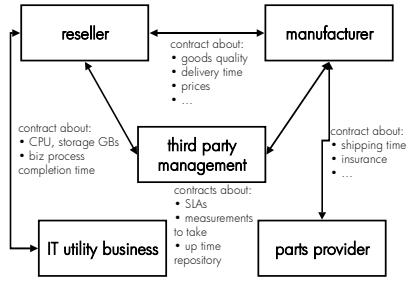

Fig. 3 Contracts, of varying level of preciseness, must be formed and managed between partners.

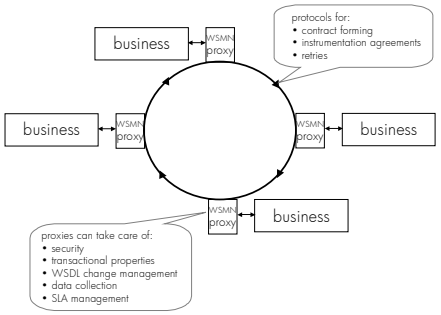

Fig. 4 Webservices management networks: communicating proxies for security, transactionality, manageability, SLA management.

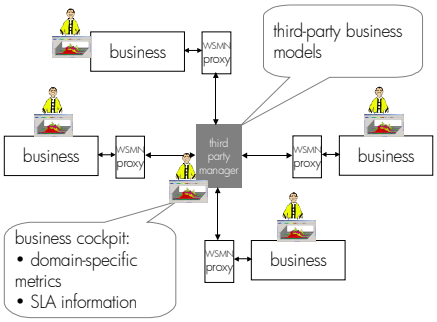

Fig. 5 Business metrics and new business models

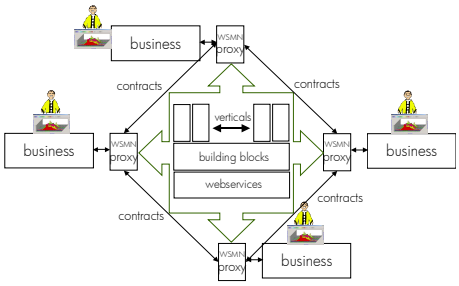

Fig. 6 In summary, the emerging architecture for the business web: businesses, services, the webservices stack, contracts, webservices management networks and business cockpit.

Webservices rely on two aspects: (1) protocols to communicate and (2) services that act. The fact that XML, SOAP, WSDL dictate progress, however, has an implication: although we look at all things as a service, and although one may argue that much of the interesting technology challenges reside in the services, current webservice technology is actually still not about the services, but about interoperability. As a consequence, superior e-service technology (e-speak [7] comes to mind) does not create impact if its superiority is in the endpoints, since the market is set through the definition of interoperability standards.

The third guiding principle behind webservice technology therefore is that *standards drive the industry, and that interoperability drives standards*. As a consequence, to judge the relative importance of standard efforts, it is sufficient to identify if a standard will gain acceptance as an interoperability technology. Unfortunately, classifying standards is not an easy task given the hype typically surrounding standard proposals, but it is useful to distinguish between 'interoperability-first' standards and 'service-first' standards. The latter focus on end points as well as process and design, the former on the enabling of interactions. Examples of interoperability-first standards are SOAP, WSDL, WSCL, WSFL [4]. Examples of services-first standards are Semantic Web [8], agent technologies [9], UML [10]. Although both classes of standards serve their purpose, the interoperability standards dictate progress in business web technology.

4. Webservices: the Final Layer in the Internet Stack

Webservices standards (XML, SOAP, WSDL) are becoming almost as pervasive as the major Internet protocols (IP, TCP, HTTP). Based on various press releases, webservices technology is (or will be) used in all possible domains: mobile computing, scientific applications, B2B, content delivery, etc. It therefore makes sense to regard the webservices layer as the latest horizontal layer in the Internet stack.

A potential candidate layer on top of webservices is a layer that deals with semantics of XML documents. Webservices work through parsing XML, which is only meaningful if (1) standards exist that specify the XML contents, and (2) the semantics of those terms is understood by application builders. However, we do not think that a powerful generic approach to semantics issues results in simple enough technology to gain widespread acceptance. The rather involved semantic web technologies provide a level of adaptability to future usage patterns that is not necessary, and lags the required simplicity—domain-specific solutions are more likely to emerge.

We argue, therefore, that the webservices layer is not only the latest, but also the *last layer in the Internet stack*. Standards on top of the webservices layer will either be reusable services with specific functionalities (e.g., authentication, management, advertising, look-up) or domain-specific standards (so called verticals). As depicted in Fig. 2, the resulting technology stack that enables the business web will thus consist of three main parts: (1) the webservices layer of the Internet stack, (2) core, reusable service building blocks (authentication, etc.), and (3) domain-specific XML ontolo-

gies for a rich diversity of verticals. Various verticals are being addressed already, such as human resources, tax services, mathematical services and utility computing. ■

5. A Planetary Business Web Emerges: Contracts, Contracts, Contracts

To make the business web work, the core organizational principle in the business web will be that of *contract*, in whatever incarnation. We foresee that all relationships will be governed through contracts, and as a consequence, the world will be covered with and coupled together through a dense web of contracts, see Fig. 3. Contracts will contain agreements about values of metrics, expressed in business level terms (what kind of service is offered, who pays, what bank is used, etc.), as well as IT-related metrics (the time it takes to complete a service, its availability, etc.). Such IT contracts typically go by the name of service level agreements (SLAs) [11]. ■

There are various reasons why contracts will be digitized more and more in the future. Already, enterprises may very well have more partners than employees, and maintain even more contracts, of various types. The sheer maintenance of the contract pool requires digitalization, such that change management can be done cost effectively [12]. More interestingly, automation can assist in all other phases of the contract life cycle: formation, negotiation, execution, compliance, assurance and optimization. To achieve automation, one needs a formal representation of the contract. Then tools can be developed to help checking fulfillment, to instrument the business process for the right data, and to negotiate and optimize contract conditions. Highly digitized and automated contract agreement will thus become more and more pervasive [13].

The biggest challenge to the widespread acceptance of contracts is if they are 'simple' enough (see Section 2). However, contracts come in many shapes and forms, from informal expectations (e.g., posted on a web site) to fully digital formal specifications. No matter how contracts are specified, the parameters of a contract are of major influence on the return a business may expect from the associated business partnerships, and form the end objective against which to manage the business web.

6. The Coming of Semantic Disasters

Infrastructure failures are getting harder to deal with. In general, increasingly networked computing introduces a mixture of coupled and autonomous behavior that is vulnerable to such issues as failure propagation (as we have learned from the telecom industry [14]), chaotic usage patterns [15] or oscillating reconfiguring [16]. In the business web this gets compounded by the fact that business level interactions have direct semantic connotations. This magnifies the impact traditional infrastructure failures will have on the business, and opens up new failure scenarios at the semantic level.

As an example, recently, for close to an hour, United Airlines by mistake sold tickets on-line for as little as five dollars. In this case, a programming bug led to a seman-

tic failure, and directly implicated the business. Similarly, security breaches at the semantic level are disastrous, since it allows semantically meaningful malicious modifications of transactions. This kind of scenarios is the stuff crime and disaster movies are made off (Entrapment, Swordfish, you name it). Hence, we have to prepare for new, largely unpredictable failures at the semantic level—these failures may not even be that frequent, but can have grave consequences.

It may appear that we have hit on a paradox: how is it possible that simplicity (so typical for all Internet technology) leads to such hard to control complexity, and even chaos? The explanation for this can be found in the simplicity/complexity spiral. For technologies to gain acceptance, simplicity is indeed required; the ensuing widespread acceptance, however, introduces usage patterns that exhibit extreme complexity, either through the sheer numbers, or because of unexpected behaviors, or even because of malicious behaviors. To deal with the complexity introduced by these usage patterns, one introduces automated management solutions, which will again exhibit simplicity to gain acceptance. In turn, this will allow for rapid deployment, increased acceptance, which leads to more and new complexity, and so on, and so fort.

Lawson identifies the HW/SW spiral [17], similar in spirit to the simplicity/complexity spiral we identify here. However, the SW/HW spiral omits the real crucial element, which is the unpredictable and enormously voluminous usage patterns. If we add increased requirements for dynamism in this mix (these follow the same pattern), one can see how this simplicity/complexity connection may spiral out of control. The hope is that ultimately, the simplicity/complexity spiral finds its resolution in fully self-managed systems.

It is up to the mathematicians to work on control algorithms to make self-management succeed, and resolve the simplicity/complexity spiral. Alternatively, the spiral may spin out of control (because we can not find ways to deal with the new usage patterns)—that would be a dream scenario for system engineers, since it would imply that we have to create a new technological world order, for instance with new streaming or rich media networks, novel collaboration systems or new networks for personalized mobile services.

7. The Core Technology Issue: Multi-Party Conversations

From a distributed computing perspective, what makes the business web interesting, challenging and different? We already touched on issues such as scale, business-level metrics, partnerships and self-management. In the business web, the overarching distributed computing theme, however, is that of *multi-party conversations*. Conversations are the flow of interactions necessary to execute a task (series of activities in a business process), for instance specified through a workflow, possibly across multiple parties [4].

There are various problems to be resolved regarding multi-party conversations. To start, we need to encode complicated manual interactions in computing programs. To achieve this we have to formalize the execution steps and identify the right interaction protocols, automating the patterns humans would otherwise have used. In webservices

land we then encode this into WSDL, XLANG, etc. Examples of the functionalities we must try to automate are price negotiation, partner selection, contract compliance checking, etc. If the resulting interactions involve multiple parties, we have to work out how services build up their common business process [18], without violating guideline 2, which says that mechanisms must be simple and very loosely coupled.

In addition, we require manageability and reliability properties for those business interactions. Since we deal with transactions across multiple parties, this becomes particularly challenging. Solutions need to deal with distributed, conversation-dependent state, and need to scale to conversations with any number of parties. Currently, we have techniques available that enable end-to-end performance measurement for conversation segments [4], and we have developed protocols to achieve end-to-end exactly-once semantics for conversations with any number of participants [19]. To achieve exactly-once semantics, we had to deal with the fact that state that is relevant to the conversations is distributed across parties, which makes that failures cannot be recovered from through straightforward redundancies (as at the network layer).

Typically, solutions such as [19] to the above problems in multi-party conversations will result in building blocks in the services building block layer of our architecture (see Fig. 2). Any type of service can then rely on these building blocks to address various functional, manageability and reliability issues of their business transactions.

To deal with the quality properties of multi-party conversations, we need some type of overlay to monitor and assure contracts, SLAs, reliability, security, etc. As a consequence, management of services will be done through what we call Webservices Management Networks (WSMNs), see Fig. 4. WSMNs connect proxies that sit between the business and the outside world and take care of transaction monitoring and assurance, or provide the necessary reliability and security to business web transactions. Such an architecture is being used by various start-up companies (Flamengo, Commendo, Talking Block) to achieve security, transactionality and versioning control of WSDL. At HP we are using the architecture as the basis for service management, in particular SLA and contract management. ■

8. Management Goes Incognito

System management is about instrumentation, observation, adaptation and control of a system. However, adaptation and control have typically been considered as an afterthought. If control is executed, the mechanisms to control are themselves part of what provides the functionality to a system in the first place. As an example, consider a load balancer, which is based on a ‘dumb’ dispatcher, but can be made into an adaptive system that controls customer service levels by adding some intelligence. Hence, these mechanisms must be designed in. This comes full circle because of the reverse phenomenon: the dynamism of future systems requires adaptation, and thus force adaptation and control algorithms to be implemented. Think in this case of for example dynamic assignment of resources or trading partners. So, dynamic mechanisms require management algorithms, and management algorithms rely on dynamic mechanisms.

With dynamism and management becoming intertwined, management becomes inseparable from core functionality. The dichotomy between core and management functionality will thus become less and less meaningful. In other words, management as we know it disappears—*management goes incognito*; management will be called self-management (an oxymoron in an of itself, but a perfect illustration that management can no longer be distinguished from core functionality).

System management will also 'go incognito' in another dimension, that of level of metrics, or layer in the Internet stack. Traditional system management at various levels in the stack will slowly be displaced by managing for overarching business goals. To drive home the argument that IT management, and certainly business web management, is all about business metrics, we depict in Fig. 5 the 'business cockpit.' Associated with each business will be a cockpit that displays metrics at the business level. Ultimately, a business metric should be in terms of money, by identifying and specifying explicit relationships between IT metrics and monetary cost and benefit [20], but also intermediate abstractions are of practical importance [21]. ■

Monetary metrics are often considered hard to establish, although there are opportunities to execute market mechanisms to establish the value of system characteristics [16]. However, in reality all optimization exercises ultimately boil down to choosing the best option with respect to a single objective function. Instead of ending up arbitrarily choosing a single objective function, one may as well realize the necessity to find a single objective function, and invest in identifying monetary consequences of actions.

9. The Business Web Will Be as Impaired as the Society that Creates It

After all this technology talk, let's take a step back and reflect on the forces that are creating the business web.

Since its incarnation, people have considered mimicking human behavior to be the highest possible achievement for a computer. Such thinking has dominated artificial intelligence, but also IT in general—the main task of computers is to take over a person's job through automation. New formulations around self-management are not much different; they typically refer to the human body (or its autonomic nervous system [17]) as the perfect system, which is then considered worthy of mimicking. Sometimes, this leads to excesses such as the statement that computers need down time like humans need sleep (heard in an autonomic computing summit).

For the business web things are slightly different—the business web mimics capitalist society, rather than humans, by automating business interactions, partner selection, contract fulfillment, etc. Moreover, as we have formulated in guideline one, the forces that dominate progress at the technology front are the forces of business, typically motivated by monetary concerns. It therefore follows the rules of capitalist society, as far as acceptance and evolution is concerned.

As IT and artificial intelligence got stuck with the paradigm of mimicking humans, and self-management is getting obsessed with resembling the human body, the busi-

ness web is getting stuck with the paradigm of mimicking capitalist society. In all three cases, this has lead to false expectations about the abilities of computers. Moreover, it may very well be possible that by starting from mimicking humans and society, we neglect opportunities to let computers do things better than us. Where is the technology that will go beyond mimicking human and social constructs? Or is it a philosophical truth that human society cannot produce man-made machines different from (or better than) itself? If that is so, it seems a foregone conclusion that *the business web will be as impaired as the society that is creating it*.

10. The Business Web Will Happen!!

Having cast a somewhat dark scenario for the future (semantic failures, simplicity/complexity spiral, issues of management, the sins of capitalist society), there is room for an enthusiastic ending. *The business web will happen*, and it is happening today. The fact that it goes beyond technology per se is exactly what makes it exciting. The future is to the researchers and engineers who are able to think beyond middleware, storage systems, e-services into sensing the society at large, and its need for business IT solutions.

To be sure, we have communicated many reservations in this paper about how advanced the technologies will be that eventually will make the business web. We believe that technologies such as dynamic discovery and semantic web will not take off to the fullest, because the technology is too complex to gain acceptance and addresses issues that can be worked around in more static fashion. Instead we argue that domain-specific ontologies and service building blocks will emerge on top of the webservices layer, and that the webservices layer will be the last pervasive layer of the Internet stack.

Fig. 6 summarizes the emerging business web architecture we developed in this paper. On top of the Internet stack and the webservices layer, the business web establishes partnerships that will be governed through explicit or implicit contracts. Management of partnerships will be done through webservice management networks, which connect proxies that sit between the business and the outside world. WSMNs will provide for various quality guarantees, among which security, reliability and manageability. On top of the WSMNs, business cockpits will be placed, which monitor the business web using business metrics. ■

11. Acknowledgements

This paper contains thoughts, intuitions and opinions that undoubtedly have fed from many different sources. In particular, I thank Giuliano Di Vitantonio and Vijay Machiraju, for ongoing research discussions around creation and management of digital business ecosystems. Others may recognize their thinking in this write-up, and since I presume that list is too long to write down, I thank all of them together. Furthermore, I thank Fabio Casati for his comments on a draft of this paper.

12. References

1. J-P. Garbani, *Implementing Service Level Management*, from 'nextslm.org,' 2002.
2. A. van Moorsel, *Metrics for the Internet Age: Quality of Experience and Quality of Business*, Fifth International Workshop on Performability Modeling of Computer and Communication Systems, Arbeitsberichte des Instituts für Informatik, Universität Erlangen-Nürnberg, Germany, Band 34, Nr. 13, pp 26—31, Sep. 2001; also HP Labs Technical Report HPL-2001-179, Jul. 2001.
3. J. Wrenn, *The IT Train That Could*, Jan 15, 2002 Issue of CIO Magazine, Jan. 2002.
4. A. Sahai, S. Graupner and W. Kim, *The Unfolding of the Web Services Paradigm*, to be published in "Internet Encyclopedia", J. Wiley, also HP Labs Technical Report HPL-2002-130, May 2002.
5. C. Cargill, *Why Are We Doing This?*, IEEE Computer, pp. 116—117, Oct. 2001.
6. V. Machiraju, J. Rolia, A. van Moorsel, *Quality of Business Driven Service Composition and Utility Computing*, HP Labs Technical Report HPL-2002-66, March 2002.
7. A. Karp, R. Gupta, G. Rozas, A. Banerji, *The Client Utility Architecture: The Precursor to E-Speak*, HP Labs Technical Report HPL-2001-136, 2001.
8. T. Berners-Lee, J. Hendler, O. Lassila, *The Semantic Web*, Scientific American, May 2001.
9. S. Poslad, P. Buckle, R. Hadingham, *Open Source, Standards and Scaleable Agencies*, Autonomous Agents 2000 Workshop on Infrastructure for Scalable Multi-agent Systems, Spain, 2000.
10. M. Fowler, K. Scott, *UML Distilled: A Brief Guide to the Standard Object Modeling Language*, Addison-Wesley, 1999.
11. R. Sturm, W. Morris, M. Jander, *Foundations of Service Level Management*, Sams, 2000.
12. A. Kyte, *Contract Life-Cycle Management: A $20 Billion Market*, Gartner Note SPA-15-7700, Apr. 2002.
13. A. Sahai, V. Machiraju, M. Sayal, A. van Moorsel, F. Casati, L-J. Jin, *Automated SLA Monitoring for Web Services*, in preparation, available from authors, Jun. 2002.
14. F. Schneider, S. Bellovin, A. Inouye, *Critical Infrastructures You Can Trust: Where Telecommunications Fits*, 26th Annual Telecommunications Policy Research Conference, Oct. 1998.
15. H. Leland, M. Taqqu, W. Willinger, D. Wilson, *On the Self-Similar Nature of Ethernet Traffic*, Proc. SIGCOM93, San Francisco, California, pp. 183-193, 1993.
16. T. Hogg, B. Huberman, *Dynamics of Large Autonomous Computational Systems*, Santa Fe Workshop on Collective Cognition, 2002.
17. H. Lawson, *Rebirth of the Computer Industry*, Communications of the ACM, pp. 25—29, Vol. 45, No. 6, Jun. 2002.
18. U. Dayal, M. Hsu, R. Ladin, *Business Process Coordination: State of the Art, Trends, and Open Issues*, 27th International Conference on Very Large DataBases, pp. 3—13, Italy, Sep. 2001.
19. S. Frølund, R. Guerraoui, *X-Ability: A Theory of Replication*, Distributed Computing, Dec. 2001.
20. *Crosscurrent*, *The Magazine for Financial Service Executives*, Issue 8, (pp. 14—23 in particular), Winter 2001-2002.
21. F. Casati, *Web ServiceScope: A Platform for Defining, Measuring and Analyzing Quality Metrics on Web Services and Business Processes*, HP Labs Technical Report HPL-2002-174, 2002.