Technologies for Business-Driven IT Management

Vijay Machiraju, Claudio Bartolini, Fabio Casati
HP Laboratories Palo Alto
HPL-2004-101
June 7, 2004*

E-mail: firstname.lastname@hp.com

Business-driven IT Management, Service Oriented Architectures, Web-Services, Agent Technology, Utility Data Centers, business processes

This article describes a vision for an adaptive IT infrastructure driven by business goals. We show the requirements for such an infrastructure, and then discuss the technologies and the architecture that enable this vision. We structure the problem into layers at different levels of abstraction, from the business process level down to the hardware level, and we show how management technologies can statically and dynamically improve business operations within each layer and across layers, with the ultimate goal of having all the layers of a company's IT infrastructure synchronized and working together to meet high-level business goals, adapting to ever-changing conditions. We believe that Web services and Service-Oriented Architectures (SOAs) will be a key part of both the IT infrastructure and of its management system. There are two ways in which Web services contribute to making business-driven management possible: First, they abstract the heterogeneity in IT infrastructure through a set of well-defined interfaces. In fact, as shown later in this document, the main benefit of Web services is that of standardizing the way functionality of various IT elements is exposed (e.g., through the use of common protocols and data models), thereby removing much of the heterogeneity present in current IT infrastructures. This makes it easier to develop generic management components that can interoperate with a number of IT elements.

* Internal Accession Date Only Approved for External Publication

TECHNOLOGIES FOR BUSINESS-DRIVEN IT MANAGEMENT

Vijay Machiraju, Claudio Bartolini, Fabio Casati
Hewlett-Packard
1501 Page mill road
Palo Alto, CA, 94304
firstname.lastname@hp.com

1. INTRODUCTION

In recent years, the industry is witnessing several significant trends that are changing the way companies think about their IT infrastructure and about how it should serve their business. The first is the increased cost of managing the IT infrastructure. In fact, while the cost for purchasing hardware is constantly going down, the labor cost for managing hardware and software of ever increasing complexity is not. Indeed, the cost or management has recently overtaken the cost for the hardware, and is predicted to become three times as expensive as the HW cost over the next few years. This makes it clear that companies are now hungry for techniques that help them to dramatically reduce the IT management cost, especially at a time where cost control is considered to be of paramount importance.

Another trend that is significant to the topics discussed in this chapter is the quest for a greater alignment of the IT infrastructure with the business needs. In the past, the opportunity for such an alignment was not there, as the IT was only supporting a small part of the business operations, the remainder being carried out manually to a large extent. Nowadays, however, the level of automation in the companies' business processes is rapidly increasing. As more and more business operations are supported by IT, the quality and efficiency with which IT operations are performed has a growing impact on how the business is performed, and the achievement of business objectives depends more and more on how the IT environment can support the business.

Finally, another trend that is sweeping across the industry is the desire for an *adaptive* (or autonomic) IT infrastructure. Indeed, many people perceive the current business climate as being much more dynamic than it used to be only a decade ago, both in terms of business partnerships and in terms of customers' behavior. Coping with continuous change in the business climate implies the need for an IT infrastructure that dynamically adapts to such

changes, and is able to deliver with adequate performance and quality levels under different and changing conditions.

Recognizing these needs and the enormous business opportunities that come with them, many large scale IT vendors have put in place programs that are targeted at enabling such a dynamic, business-driven IT infrastructure. The most significant efforts to date are IBM's autonomic computing[18] program and HP's adaptive enterprise[15] program.

In this chapter we provide a vision for an adaptive IT infrastructure driven by business goals. We show the requirements for such an infrastructure, and then discuss the technologies and the architecture that enable this vision. We structure the problem into layers at different levels of abstractions, from the business process level down to the hardware level, and we show how management technologies can statically and dynamically improve business operations within each layer and across layers, with the ultimate goal of having all the layers of a company's IT infrastructure synchronized and work together to meet high-level business goals, adapting to ever-changing conditions.

As this chapter will show, we believe that Web services and Service-Oriented Architectures (SOAs) will be a key part of both the IT infrastructure and of its management system. There are two ways in which Web services contribute to making business-driven management possible: First, they abstract the heterogeneity in IT infrastructure through a set of well-defined interfaces. In fact, as shown later in this document, the main benefit of Web services is that of standardizing the way functionality of various IT elements is exposed (e.g., through the use of common protocols and data models), thereby removing much of the heterogeneity present in current IT infrastructures. This makes it easier to develop generic management components that can interoperate with a number of IT elements.

The second way in which Web services enable business-oriented management is by making the management system itself more homogeneous and integrated. In the previous paragraph we focused on Web services used to expose functionality of IT infrastructure elements. However, the same benefits of Web services can be leveraged by the management infrastructure, which is in itself a complex system composed of multiple parts, often provided by different vendors. Using Web services, it is possible to create a homogeneous layer that hides the differences provided by the various management agents deployed on the IT systems and applications and that provides a uniform view to a management platform, thereby making it easier to monitor and control the underlying system in a holistic way.

Although we believe that IT infrastructures as envisioned in this chapter will not be ready for a few years, many of the technological pieces are there

today, and we hope that the discussions in this chapter will help clarify how these pieces fit together and therefore lay the conceptual foundations necessary to realize such a vision.

The remainder of the chapter is structured as follows: we begin in Section 2 by presenting IT infrastructures as they are today. This will help us show the limitations of current technology in supporting the adaptive enterprise concept. Next, in Section 3, we present business scenarios that motivate the need for a business-oriented utility infrastructure. Section 4 discusses our vision for such an infrastructure and presents the technologies that can contribute to its realization. Finally, Section 5 makes some concluding remarks.

2. STATE OF THE ART IN SERVICE DELIVERY

This section presents a reference architecture of how companies view their IT infrastructure as multiple layers, and how they execute their business processes and deliver their services. Later in the chapter we will show why this architecture is, by itself, inadequate to support the adaptive, business-driven enterprise vision painted in here. The reference architecture is composed of essentially three layers, shown in Figure 1 and described in the following.

2.1 Resource layer

The lowest layer of the IT stack consists of the physical resources, and specifically servers, mainframes, networks, storage, and in general the hardware infrastructure. This layer can be structured and organized in different ways, depending on the needs of software applications (the next layer), on the anticipated workload of those applications, and on their performance requirements.

Traditionally, a set of hardware resources is designed to be dedicated to each software application or to a group of software applications (e.g., databases, middleware, custom code, etc) that together support a delivered service Once the design decision was made, changes at the resource layer were rather infrequent. As we will see later in the chapter, having a dynamic resource layer, where resources can be assigned to services almost literally on the fly to meet changing capacity needs is key to making the enterprise adaptive. If the resources are allocated once and for all, it is impossible to respond to business-driven adjustments and optimizations of IT infrastructure.

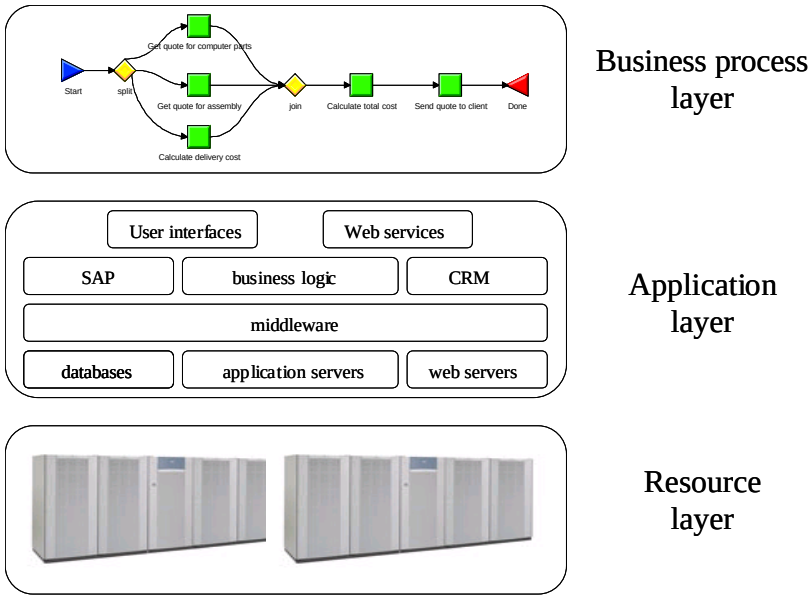

Figure -1. Layers of a typical service delivery infrastructure

Moreover, each department or business unit in charge of delivering the service was actually purchasing and managing the necessary hardware resources. This is also a problem in terms of cost of IT management, as each department is faced with the challenging task of managing complex IT systems, often resulting in less than optimal resource allocation with unnecessary duplication of hardware. Indeed, in recent years companies are aggressively trying to consolidate and centralize the location and management of the resources (either by physically co-locating them into a small number of large-scale data centers within the company or by outsourcing them altogether to IT firms such as HP, IBM, or EDS).

2.2 Application layer

This layer provides the software applications required for implementing the business processes (next layer above) in a company. It includes such applications as Database Management Systems (DBMSs), message brokers, application servers, enterprise software such as SAP and, in general, all middleware applications. It also includes support for security, document management, directory management, and other features commonly needed by many applications. We also consider part of the application layer any

custom business logic that has to be implemented on the top of middleware infrastructure in order to support the business processes.

Most of these applications are used for implementing the day-to-day business processes in a company (e.g., financial accounting process) and are exposed to humans through user interfaces. Other applications are wrapped into “services” that can be invoked programmatically from other applications. The latter allow clients to be shielded from the actual implementation of the underlying application through a well-defined application programmer’s interface (API). Recently, technologies such as Web services are taking this approach a step further by standardizing the protocols necessary for such application-to-application interactions to happen over the Internet in a platform and programming language agnostic manner. This enables applications that are potentially running in multiple companies to interact with each other (e.g., an order processing application in one company can interact directly a supplier’s catalog application).

Having a dedicated application infrastructure for each business process- as it is most often the case in today’s IT world - goes against the vision of the adaptive business-driven enterprise. In particular, it means higher costs in terms of software licenses and of management. Instead, companies would benefit from an IT environment in which the hardware and software infrastructure layers are to a large extent centrally managed, and where both of them can be deployed (installed and configured) in an adaptive fashion (quickly and automatically), to be able to meet requirements in response to dynamic changes.

2.3 Business process layer

The topmost layer in the IT infrastructure is composed of *business processes*, also called *composite services* in this context. These are complex services that are implemented by composing and invoking other services (we use the term service to denote an application that is invoked programmatically, as opposed to application used by humans through a GUI or a browser).

As an example, Figure 2 shows a simple composite service that allows customers to order products. From the users' perspective, a composite service is just like any other service: it has its own, defined, published interface (in this example, represented by one request/reply operation), and can be accessed in the same way any service can be accessed. What is then the point of considering composite services as belonging to a separate layer, as opposed to "yet another" application? There are several reasons for this, some business-oriented and other technology-oriented.

From a technology perspective, once composition techniques and tools are available, composition itself can be offered as a service. This means that users could leverage a composition service provided by the IT infrastructure to define new (composite) services on the fly and then use them. From a business perspective, this could mean that the user could not only create the composite service for "personal" use, but also act as the provider of this newly created service and offer it to other users, allowing a potentially unlimited number of players may join the IT infrastructure ecosystem. This is in contrast with what happens today, and constitutes a new business model for data center operators and IT companies, that can use the same infrastructure to serve the IT needs of different customers, thereby facilitating business process outsourcing.

Another important aspect is that business processes are what enterprises care about. The goal for companies is to improve the quality and efficiency of their processes, as perceived by customers and by the providers themselves. Hence, the process is the place where the relevant business metrics are defined. Lower-level metrics, such as the response time of this or that application, are important only if they affect the value of business metrics. Improving on lower-level metrics is never the goal, rather they should be treated as symptoms and control points to understand and improve business process metrics.

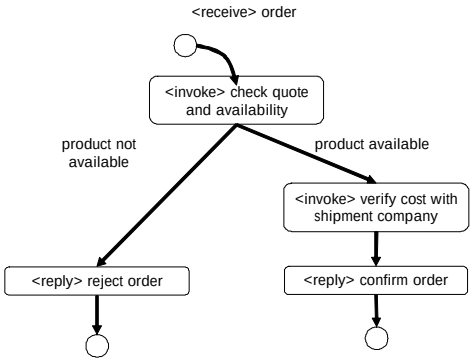

Figure -2. A sample composite service

In summary, the big challenges in this layer for realizing the adaptive enterprise vision are the ability to quickly create a new process from existing applications and services, the ability to translate the requirements on the business process into requirements on the layers below, and the ability to dynamically propagate the changes from one layer to the next.

3. BUSINESS-DRIVEN MANAGEMENT OF IT INFRASTRUCTURE

In this section we present a typical business-to-business order management scenario to be used as motivation for a sophisticated management infrastructure that spans through all the layers presented in section 2. Indeed, it is essential that these layers are coordinated and work together towards a common goal if we want to achieve the very ambitious goals of dramatically reducing the cost of IT management, of making the IT infrastructure adaptive to changes in the environment, and of aligning the IT environment with the business goals.

3.1 Order management scenario

The scenario we present models an enterprise that sells microprocessors to their business partners. We described in terms of the three layers that we identified in the previous section.

3.1.1 Business process

A sample order management process (composite service) is shown in Figure 3. On arrival of a new order, and after the new order has been validated (checked for completeness and consistency), the provider carries out a credit check and the order would be subject to block until the requester's credit is vetted. Once the credit check phase has been carried out, scheduling takes place. The order is divided up into order lines (line items), which represent units of shipment. Depending on the inventory state on the provider side, some of the order lines are committed and the schedule of their delivery takes place. Some of the order lines may not be committed because of temporary inventory shortage. For each order line, an acknowledgement signal is sent back to the counterpart. At this point a delivery note is created and the fulfillment stage begins.

3.1.2 Applications

The order is transmitted through EDI (Electronic Data Interchange[31]), which is the de facto standard for exchanging business documents (such as a purchase order, invoice, shipping schedule, inventory inquiry, and claim submission) in a number of industries. The acknowledgment of commitment of order lines is also sent back to the requesting party through EDI. Most of the functionality needed for implementing this process can be supported by SAP application modules. SAP R/3 is used for order entry, incompletion

hold, and for extracting relevant information from the order lines to create delivery notes. Delivery notes are created through a legacy application.

Functionalities such as order validation, credit check, availability check, procurement, and fulfillment are also supported through SAP either through user interfaces (manually executed steps) or through Web service interfaces. In addition, the order management process requires middleware for process execution (such as service composition engines or workflow engines), Web services support software (such as HTTP servers and SOAP routers), and databases to store and log execution data.

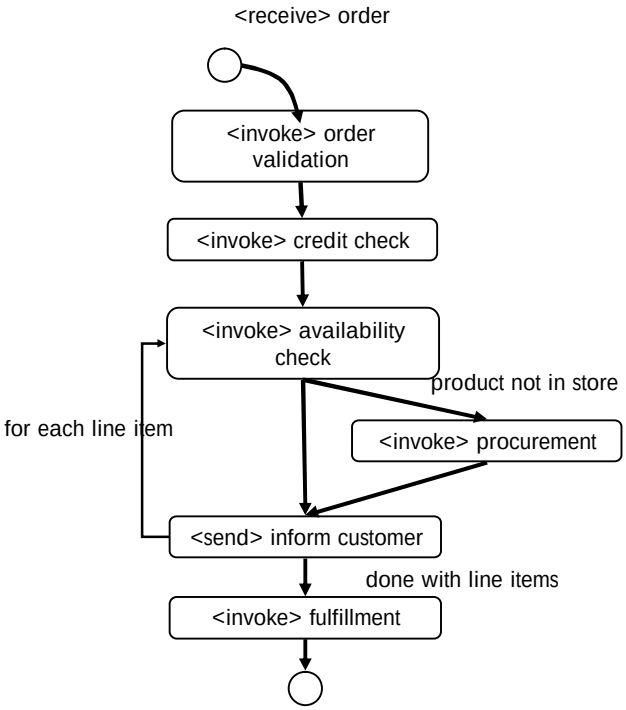

Figure -3. Order management process

3.1.3 Resources

A large number of resources are needed to support the order management process. Depending on the number of orders generated per day, one or more front-end servers will be in charge of receiving users' requests. These servers will typically run HTTP servers. Other server machines and storage devices are needed for running application servers and the workflow engines,

database servers, and the other applications. The overall number can range from a few units to, potentially, even hundreds of devices, as is the case for online stores such as Amazon.

3.2 Reference architecture for management

It is clear from the discussion so far that the only thing common across all IT systems is their complexity. This complexity arises from the size and scale of IT infrastructure in terms of number of resources, number of applications, number of processes, number of roles and players, number of configuration parameters, number of measurements to handle, number or corrective actions that can be taken, and the web of dependencies between all of the above. On the top, business alignment and adaptability require changes in the IT infrastructure in much shorter time scales than is done today, including in particular the coordinated changes to these parameters to improve business goals.

The only way to deal with this complexity in a dynamic environment is to rely on automation. It is impossible to manually manage this complexity, both because of the number of factors involved in making decisions and because these decisions must sometimes be taken on the fly and enacted right away, and not offline. Hence, many of the operational tasks that are typically carried out by human operators need to be streamlined and automated. Automation is needed in all phases of the IT lifecycle – from provisioning to monitoring, decision making, and controlling – and in all layers of the stack. Besides enabling better management, automation also enables new opportunities and, in particular, allows the same data center to support multiple customers by switching data center configurations and the applications/processes executed on it based on the business needs of the different service providers supported by the data center itself, as well as the data center operator.

Figure 4 shows a reference architecture for such a management system. It highlights the various functionalities that should be provided in the different stages of management lifecycle and across the layers of the IT stack. In the rest of this section, we will describe this reference architecture which we will use in the next section to position the various technologies that are being researched and developed today in the industry and academia.

3.2.1 Provisioning

The first set of management capabilities has to do with *provisioning of IT systems*. Provisioning deals with creation and deployment of business processes right from identification of business-level requirements to the act

of implementing, allocating, integrating, and configuring application and hardware resources in order to realize that business process. In section 2, we have discussed the need for better sharing of resources and the need to shift resources from one application to another on as-needed basis. In other words, we need automated provisioning capabilities that are driven by business policies and that use resources optimally.

The provisioning phase also includes activities such as Service Level Agreement (SLA) and relative metric definition and negotiation if multiple parties are involved in implementing the business process (for e.g., data center operator, application service provider, business service provider, business process owner, and customers). All of these parties are typically linked by contractual relationships. For example, a database application service provider expects a certain availability of compute and storage resources from the data center. The business service provider expects a certain database transaction throughput from the application service provider, and the business process owner frames his/her SLA in even higher-level terms (e.g., number of purchase orders processed). It is essential that the management infrastructure allow service providers to model the SLAs they have stipulated. In addition to SLAs, service providers may want to define metrics that correspond to properties of interest to them, for example to evaluate the effectiveness with which a service is provided. For example, a frequently used metric involves the definition of the cost it takes to deliver a service to a customer.

3.2.2 Monitoring

The second set of capabilities needed by the management system is *automated monitoring.* This includes discovery of IT infrastructure elements, their dependencies, and how they affect each other and the overall business processes that they are part of. It also includes proper instrumentation of these elements, collection of the necessary data, and their aggregation to measure the metrics and SLAs that are of interest. While allowing the computation of higher-level metrics and SLAs, the technologies for monitoring should also facilitate drill-down into detailed measurements for problem resolution and root-cause analysis.

The introduction of multiple players once again adds to the complexity of monitoring. When each of the players owns a portion of the IT infrastructure and collects data pertinent only to that portion, it becomes harder to compute aggregate metrics, to agree on the status of SLAs, or to assign blame in case of SLA violations. For example, consider the failure of a server owned and operated by a data center provider. If the applications hosted on that server are owned by another service provider, then the failure may be attributed to a

hardware failure (data center provider's responsibility) or to an application error (service provider's responsibility).

3.2.3 Decision-making and control

The third set of capabilities required in the management infrastructure is *automated decision-making and closed-loop control*. Being able to define and monitor metrics and SLA is only the first step towards the adaptive enterprise. To enable proactive management, it must be possible for users to *analyze* why metrics have unsatisfactory values (e.g., in which situations is an SLA not met). Understanding this can lead to a redesign of the business process, of the services, or to adjusting the resource needs. This can be typically done in conjunction with optimization techniques. While analysis and optimization enable off-line changes, *prediction* is at the hearth of the adaptive enterprise vision, since it enables dynamic change. In fact, the possibility of predicting the likelihood of a certain exceptional situation (such as an SLA violation) may lead to its prevention.

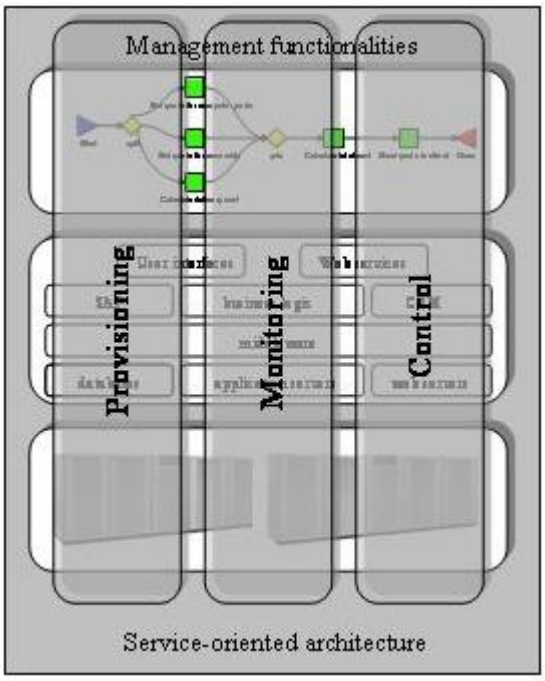

Figure -4. Reference architecture for management

Predicting an exception only solves one side of the adaptive management problem. The other side requires techniques for managing the exception. Just like all aspects of the adaptive business-driven IT infrastructure presented here, exception handling also has to be driven by business goals. Indeed, the problem here consists in identifying what action to take, based on the predicted or detected problem, based on the perceived business damage that occurs if the problem is not corrected, and based on the cost of implementing the solution.

Beyond the generic ability to detect, predict, and fix SLA violations, decision-making and control takes different forms when applied to different layers. In the resource layer, we need the ability to adjust resources allocated to applications in a dynamic and optimal manner. In the application layer, we need the ability to tune application parameters to meet business requirements. Finally, in the business process layer, we need the ability to re-design business processes if that is indeed the cause of a problem.

3.2.4 Service-oriented architecture for management

Finally, there are certain fundamental architectural principles that must be adhered to in creating this complex management infrastructure. There is clearly a danger of creating a large number of ad hoc management components that are tightly coupled in assembling such a management system. For example, one common mistake is to create a different management component for every different kind of resource, application, and business process. This, for instance, manifests in the form of several different languages for expressing SLAs, and different ways of interacting with resources and applications (based on their kind). Given the large number of possible kinds of elements in the IT stack, this will lead to a fragile and inflexible management system that is far less adaptive than the IT infrastructure it is used to manage.

What we need is a structured approach to creating the management infrastructure based on standardized interfaces and service-oriented architectures. This allows management components across layers and functionalities to interoperate with each other easily, as all manageable components will adhere to the same framework and communicate through standardized protocols. This adds another dimension of requirements to the management infrastructure and is shown as a back plane in Figure 4.

4. TECHNOLOGIES FOR BUSINESS-DRIVEN IT MANAGEMENT

In this section, for each of the three management functionalities domains identified in the previous section – provisioning, monitoring and control – we present a collection of technologies that are being developed in the academia and in the industry and fit them in the map that we began sketching in the previous section.

4.1 Technologies for provisioning

4.1.1 Virtualization

As discussed earlier, resources and applications are traditionally allocated in a dedicated manner. The amount of resources to be allocated is decided either based on peak demands (the result of which is poor average utilization) or based on average demands (the result of which is inability to cope with fluctuations in demand). In either case, dedicated resources imply high equipment/license costs. In many cases, application demands cannot be predicted in advance and having dedicated infrastructure decreases the flexibility of shifting resources from a low-priority application to a high-priority one. *Utility computing* presents a paradigm where shared infrastructure can be provided on demand to multiple applications.

Virtualization is the primary enabler of utility computing. It is a set of transformation processes that make one set of "underlying" resources with some capabilities look like another set of "virtualized" resources with different capabilities. For example, a server with a certain number of CPUs and physical memory can be virtualized into several virtual machines (VMs), with each of them having a subset of the CPUs and physical memory. Virtualization ensures the isolation between the VMs and provides the illusion that each of the VMs is a complete server.

Virtualization, as described here, is not a new technology. Virtual memory in machines provides application processes with the impression of using larger memory than is actually available. Virtual disks of RAID arrays provide the impression of faster and more reliable disks than physical disk devices. Network virtualization allows to fully decouple sub-networks used by different applications from one another by providing individual IP address spaces.

Although virtualization by itself is not a new technology, its use in a commercial data center environment to provide more agility to enterprise IT is new. Recent developments in virtualization technologies are providing better security isolation (completely hiding one virtualized entity from

another) and performance isolation (minimizing the performance impact one entity has on another), which are essential in a data center. Server virtualization technologies such as VMWare[32] and Xen[4], network virtualization through VLANs, and storage virtualization through Storage Area Networks (SANs) are being applied to achieve the goals of flexibly sharing physical resources and adjusting those shares on the fly. Similar notions of virtualization are also being thought of at the application layer[30].

The programmability and isolation provided by virtualization technologies such as the ones described above are critical for utility computing – in order to allocate resources on demand and in order to shift resources from one application to another dynamically.

4.1.2 Utility data centers and Grids

To address the problems associated with high costs of owning and operating IT infrastructures, companies are trying to consolidate their physical and application resources into data centers. Such consolidation typically involves:

- Co-location of resources (as opposed to distribution of resources across all departments in a company),
- Creation of structured and manageable physical topologies (as opposed to ad hoc networks that evolve over time in a decentralized environment),
- Use of a single or a small number of vendors for all the physical and application resources (as opposed to largely heterogeneous resources from different vendors and with different versions that builds up in an uncontrolled environment),
- Co-location of human expertise for providing support and services on the infrastructure (as opposed to multiple sets of consultants that are hired by each department), and
- Sharing of physical and application resources through virtualization as explained above.

If the data center also supports utility computing by allowing resources to be shared and adjusted dynamically, then it is called a Utility Data Center. HP's Utility Data Center (UDC)[17] is the first among such data centers to offer a fully virtualized server, network, and storage environment. Applications can programmatically request a securely isolated 'farm' to be carved out of the UDC resources. A farm in UDC is a set of servers, connections, and storage connected in a particular topology (e.g., 3-tier topology, cluster topology, etc). The UDC selects the best unused resources that satisfy the application's requirements, and configures them to create the

illusion of the farm. It does so by configuring the VLANs on network switches and the SANs on storage switches.

In order to adapt to application's changing workloads, HP's UDC also supports dynamic flexing of resources. An application can request that additional resources be added to the farm or that existing resources be removed without affecting running applications. This capability increases the ability of IT infrastructure to react to business changes such as increase in customers, creation of a new service, or the rolling out of a new promotion.

There are other efforts that are aimed at providing similar programmatic access to resources when needed. Popular among these are the *Computational Grids,* which are large clusters of thousands of servers distributed across universities, super computer centers, and research organizations around the world. These Grids are commonly used by companies, government organizations, and academia for large scientific computations such as weather forecasting, simulations, and bio-medical applications. More recently, the concept of Data Grids[11] has also been gaining popularity, where the focus is on having a large pool of data stores as opposed to compute servers. The core technology behind these Grids is a set of protocols for discovering, requesting, using, and releasing resources, and job scheduling. Examples of these Grids include West Grid[33] and NASA Information Power Grid[22].

4.1.3 Resource allocation

While virtualization provides the underlying mechanism for resource sharing, the intelligence for how many resources to allocate for each application and the choice of application mix to place on a particular resource is resident in resource management systems.

There are several factors to be considered for proper resource allocation. First, the constraints imposed by the application or process demanding those resources must be satisfied. Examples of such constraints include the time periods when the resources are required, the quality of service guarantees to be obeyed, and the number and type of resources required. Second, the affect of allocating an application on existing applications must be considered. Certain applications exhibit peak workloads at the same time. Co-allocating such applications on to the same resource may result in performance problems. Third, priorities and business policies must be taken into account while making trade-offs. For instance, if there is not enough capacity, then resources should be moved from a lower-priority application to a higher-priority one.

Many technologies are being developed to simplify and automate resource allocation. Market-based allocation schemes such as resource

auctions are being used to automate fair allocation of resources based on business priorities. As the name suggests, market-based resource allocation is based on designing a market for allocating the available IT resources to the entities responsible for running the applications and the business process on top of them. These entities bid in this market to make sure that the portion of the IT infrastructure that is made available to them meets their business objectives. Naturally the metaphor of the market works at its best when there are multiple parties involved, as in some of the scenarios described above, but it's equally applicable when the agents bidding in the market all cooperate and are bidding to represent the only party that has interest in the market. Although market-based resource allocation has not yet reached the level of maturity for successful industrial deployment, system based on this paradigm have been investigated in the academic literature, see for example Wolski et al.[34] where market-based resource allocation is investigated in a Grid setting.

Mathematical optimization techniques[36] and workload characterization techniques[27] are also being used for optimal or near-optimal placement of applications onto resources. These techniques rely on a deep understanding of the physical infrastructure, application workload characterization, and the impact of those workloads on the resource requirements. Simulation and correlation techniques[13] are useful in inferring how resource requirements change with changing workloads.

Another aspect of utility computing is to simplify installation and configuration of applications once resources are allocated to them. Workflow and scripting technologies for orchestrating and managing various configuration activities and languages for specifying such configurations and workflows[16] are being researched in this regard.

4.2 Technologies for Monitoring

4.2.1 Data integration

Once a business process is operational, there are a number of instrumentation sources that are activated in order to collect various kinds of data from the applications and resources that implement that process. This is required to validate whether the objectives of the process are being met, whether there are any improvements that need to be made in the future, or simplify to detect unforeseen conditions and bugs.

Given that a number of business processes, applications, and resources make up any given IT infrastructure, the amount of data collected from these sources becomes quickly unmanageable. But, an even more significant problem is the ability to integrate data from all the data sources. This arises

because of the lack of agreement on structure or semantics of the provided data. For instance, servers in the physical resource layer produce data about CPU utilizations, memory utilizations, and failures in hardware. Applications such as databases produce instrumentation data such as number of queries serviced per second, performance problems, and SQL errors. Business processes, on the other hand, can be instrumented to provide data about number of orders processed (considering an order processing process, for instance) or the number of orders that did not go through. Often there are no relationships between these individual pieces of data to enable correlation.

Even within the physical resource layer, not all servers provide their data (e.g., CPU utilizations) in the same format. This changes from vendor to vendor and largely depends on the kind of instrumentation used. For instance, some may report CPU utilizations averaged over 5 minute intervals, while others do that for 10 minute intervals. Similarly, the structure and types of events reported by every element in the IT stack are different.

These problems make it very hard to build automated tools that can understand, analyze, and predict problems in IT infrastructure. There are some efforts aimed at standardizing the semantics behind the data collected from various sources. Examples of these standards include the Common Information Model (CIM)[12], Application Response Measurement (ARM)[29] and, more at the business level, SCOR measures[28]. However, these standards are confined to a certain aspect of the IT infrastructure, and are not designed to work together. This is part of the reason why there is today no end to end management solution.

4.2.2 Metric definition and monitoring

Most modern applications provide a way for users or other programs to access monitoring information. For example, operating systems offer access to a variety of statistics and real time information about processor and memory usage, network management applications provide information on network load, while process management applications provide such information as the number of processes executed and the average duration of each process.

All this is very useful, and has greatly improved the manageability of such applications. However, it is not sufficient to realize the vision set forth in this chapter. There are two aspects that are missing in this picture:

1. Predefined metrics are not sufficient to model business goals, which is what we ultimately want to optimize. For example, a process management software may offer metrics such as the average duration of

each step, but users may be more interested in modeling an SLA on top of a process, that may for example require that orders of a certain product and below a certain volume be delivered within 5 days. This is something that is not provided out of the box, but that can in principle be measured since all the information is known and logged by the process management system. Hence, analysts must be provided with a way for defining and monitoring metrics

2. Each application has its own format for storing metric data, and its own protocol for communicating this information. This makes it extremely difficult for a management system to collect and correlate information at the different layers of the stack and use this information in a coordinated fashion to determine causes of unsatisfactory metric values and determine the root causes. This problem is analogous to the data integration one mentioned above. To achieve the vision set forth in this chapter, a uniform metric model and a uniform protocol to access metric values is needed.

Although these are fairly new requirements, there are already some proposals that try to fill these needs. The first issue is being addressed by several proposals that aim at providing easy-to-use metric models, so that analysts can specify new metrics in addition to the built-in ones[8, 20]. The underlying principle of all these approaches is that a certain set of logical "measurement devices" are provided to the user, which can be customized and combined in several ways for defining metrics and how they should be computed . For example, a "measurement device" could provide the time distance between two steps A and B in a process. This can be used to model an SLA on the order management process that says that the time interval between the order receipt (step A in this case) and the confirmation sent to the customer (step B) should be kept below a certain threshold. Other examples of such logical devices are those returning the value of a process variable or the duration of a step. In this way, the metric definition can be made rather simple as it only requires instantiating the logical devices. It is up to the application provider to implement these devices. For example, the process engine will implement the (parametric) business logic that returns the time distance between two steps, and offer this device for use within metric definition.

The second issue is addressed by Web service technology, which is a strong candidate for masking heterogeneity and for providing homogeneous access to heterogeneous systems (in this case, to heterogeneous application monitoring systems). We do not comment further on the use on Web services since we will discuss this issue later in the chapter.

4.3 Technologies for decision making and control

Monitoring is useful per se, but its greatest value is when it is combined with analysis and optimization. The goal here is to take the outcome of the monitoring system and use it to identify problems (again, expressed in terms of business metrics) and make changes to the system to remove such problems. The challenge, again, lies in how to do this automatically and without requiring ad hoc solutions for each metric or each system.

4.3.1 Business process analysis and optimization

Analysis and optimization of business processes has been traditionally based on simple techniques that show simple statistics of process execution data in tabular or chart form. As people started to realize the importance of process analysis, tools became more sophisticated and were extended with warehousing mechanisms and OLAP analysis. While applying OLAP to process data is certainly helpful, it is insufficient to provide explanations of why certain metrics have certain values (e.g., unsatisfactory values) and to predict future values, which are the two things users care about. To perform this kind of "intelligent" analysis, data mining technologies are being put to use.

Data mining techniques are being applied to understand patterns that lead to poor quality workflow or service executions, and as a result, to identify areas for improvements. For instance, the problem of analyzing why a metric has a certain value can be mapped to a *classification* problem. Classification applications take as input a labeled training data set (typically in the form of a relational table) in which each row (tuple) describes an object (e.g., a *customer* in a customer management application, or a process execution in our case) and the *class* to which this object to be classified belongs (e.g., “profitable”, “neutral”, or “unprofitable” customer). The classifier then produces a set of *classification rules*, i.e., mappings from a condition on the objects' attributes to a class, with the meaning that objects whose attributes satisfy the condition belong to the specified class. Therefore, classification rules identify the characteristics of the objects in each class, in terms of values of the objects' attributes[5].

This approach provides many benefits: first, it allows intelligent analysis and prediction of metrics at the business process level. Second, if application and resource information are included among the factors considered by the classifier, then it is possible to assess the impact that resource and application characteristics have on business metrics. For example, we can see that whenever a certain low-power resource executes a process, then a certain SLA. This information can be used to drive resource and application

requirements down from business metric. Third, this approach is metric- and process-independent, and is therefore generally applicable.

4.3.2 Application tuning

Service level objectives as well as results coming from process analysis (either performed through data mining, or through more traditional means such as "simple" metric definition and management technology) can be used to better tune applications and size systems.

Most applications, especially server applications, include a number of parameters that need to be set appropriately for the application to operate in the "best" possible way from the perspective of meeting business goals, and can work in many different configurations. For example, databases can be supported by a single machine, or can run on parallel computers. Furthermore, a DBMS can be set in many different ways based on the requirements of the service using it. For example, administrators could select different logging and data archival mechanisms, based on the data recovery needs.

There are essentially two ways in which one can do application tuning and sizing: a priori and a posteriori. The a priori approach is based on known mappings between service level objectives and application parameters. For example, well-known benchmarks can be used (or extrapolated) to determine how to configure a database or how many servers should be used to run the database. This mapping is done manually today, but there are approaches that are aimed at automating this.

The a posteriori approach is a form of closed-loop management, where service execution metrics are evaluated and used to change the application characteristics. Control theory approaches[26], or data mining techniques like those described in the earlier section are being used to dynamically adjust web server, application server, and database parameters. This being said, achieving automation in application tuning is still a challenging problem as most of the existing techniques are very specific to the application they manage and are not yet effective under all workloads. We expect this to be one of the most interesting and tough research topics over the next few years.

4.4 Technologies for service-oriented architectures

4.4.1 Web services

The term *Web services* is used to refer to applications that are accessible programmatically using Web technologies[1]. Web services technology comprises of languages and infrastructure that enables users to:

- Describe the interface, behavior, and possibly other non-functional aspects of services. Examples of such description languages are WSDL[10], enabling the description of the service interface, and BPEL[3] and WSCI[2], that allow the description of the order in which the different service methods should be invoked to achieve a certain goal.
- Publish service description information on private or public registries, to enable static service discovery or dynamic binding. Examples of service registries are provided in UDDI[24] and ebXML[23].
- Achieve interoperability among services, by providing and supporting standardized interaction protocols. Examples of such protocols are SOAP[14], supporting basic interaction by prescribing how information should be packaged into XML documents, WS-ReliableMessaging[6], introducing a protocol for achieving reliable information exchange, or WS-Transactions[7], defining how a set of interactions can be given transactional semantics.

The reader will notice that, from what is described above, Web services do not appear to be fundamentally different from services in conventional middleware, such as CORBA or COM objects. Indeed, description languages, registries, and interaction protocols were also available and successfully used in earlier middleware platforms, before the advent of Web services. However, Web services and the related technologies have been designed to operate in a B2B context. This fact implies important differences in how the interaction takes place: firsts, there is no obvious place where to put the middleware. In intra-organizational interaction, many middleware components (such as the TP monitor or the message broker) were conceptually centralized and managed. In B2B interactions there is no centrally managed component, and the interaction must occur in a fully distributed fashion.

In turn, this means that standardizations become essential: while with a central middleware it is the middleware itself that imposes certain description languages or interaction protocols, in B2B the different parties (that will likely use different middleware platforms) will need to agree on common interchange formats and interaction protocols. Unless such formats and protocols are standardized, it is very costly, not to say unfeasible, for a company to maintain interactions with several business partners. Finally,

decentralization also means that B2B protocols, besides being standardized, need to be designed to work in a distributed fashion and in the absence of a central coordinator. In those cases where a central coordinator is required, then protocols are needed for the only purpose of defining who the protocol coordinator is.

4.4.2 Web services for management

The reason we talk about Web services in this chapter is that they are very relevant to the management of utility infrastructure. In fact, the discussion above has emphasized that Web services reduce heterogeneity. Thanks to standardization, they make it possible to access very disparate applications and resources by using the same set of protocols and data interchange formats. From a management perspective, this means that in the near future both applications and management agents will provide a Web service interface to allow their management. This considerably simplifies the development of an overall management platform, which can use the same technology to gather monitoring information and control heterogeneous set of software and hardware resources.

In the service-oriented paradigm, there are certain patterns of interaction that are essential to the successful operation of the environment. These interactions establish and ensure that services adhere to acceptable guidelines of behavior when they interact. Without these guarantees, the service-oriented approach becomes chaotic and incomprehensible leading to unreliable performance, and unsubstantiated finger pointing when failures occur. In general, the following four interactions as essential:

1. Negotiation is the process by which services interact to set-up guidelines prior to other interactions. Negotiation permits a wide set of parameters to be established, and it permits a multi-phase process of offers and counter offers that result in the services working together to form an agreement that governs further interactions.
2. Monitoring occurs during the operation of the services. Measurements are taken and aggregated, and can be compared to agreed upon guidelines or other requirements specified by administrators.
3. Control refers to the ability to make changes to a service's characteristics. Often, the goal is to bring it back in-line with agreed upon parameters if they should fall out of bounds during operation.
4. Renegotiation is a re-running of the negotiation process after service interactions have begun. It can be used, for example, when no control operations are possible, and a service or administrator determines that the existing agreement can no longer be satisfied, so an updated one is needed.

One concerted effort that is aimed at applying Web services to standardize these kinds of interaction patterns is WS-Resource Framework (WSRF)[25]. For example, WS-Agreement[35], which is a part of WSRF, defines a new approach for interactions between a service provider and its consumers. In most present systems, the conditions under which a service is provided are either left completely open (usually resulting in "best effort"), are specified out-of-band with the service (for example, a contractual engagement specifying a consumer as having "gold-level" service), or are handled as invocations on the service itself (in the utility model, this is equivalent to requesting resources without prior assurance that they will be available). In contrast, WS-Agreement makes the statement and negotiation of these "terms of service" explicit. Making these explicit has the benefit of setting expectations for both the service consumer and provider, and can provide specific metrics that can be monitored during operation. Another similar effort - attempting to standardize interactions for monitoring resources and applications - is ongoing in OASIS, and is called Web services management framework (WSMF)[9].

4.4.3 Agent technology

Software agents are defined as software artifacts that act autonomously to undertake tasks on behalf of users[19]. An agent-based approach to software development allows an analyst – or more in general the agent's stakeholders - to declaratively specify the agent's goals instead of issuing explicit instructions, leaving to the agent the decisions on how to best accomplish such goals.

The academic literature is rich with applications of both autonomous agents and multi agent systems to management problems at all levels, ranging from intrusion detection at the resource level to transforming infrastructure data into real-time business intelligence, for managing at the higher level of the stack. Market-based resource allocation that we have covered in one previous section can itself be seen as an example of a multi-agent system to solve the dynamic resource reallocation problem. In some notable cases, agent-based solutions to management problems have been adopted in the industry as well. HP Openview features agent-based solutions to storage area network configuration and monitoring and security management among others, and so do other leading management software vendors such as IBM, BMC, and Micromuse.

In addition, because of the declarative nature of the goal directed approach, agent-based solutions are well suited to tackle the problem of service composition that we exposed when presenting the business process layer of our conceptual architecture. See Maamar et al.[21] for an example of

an agent-based architecture to interleave web services composition and execution.

4.5 Realizing the management infrastructure

This section discusses the different ways of realizing the adaptive management infrastructure. We study the alternatives along different dimensions. In many cases, each of the alternatives presents a valid solution and differs from the other alternatives only in the approach taken to reach the end goal.

4.5.1 Off-line versus online

The management system may operate in an off-line or on-line fashion, and different technologies may be better suited for one or the other mode of operation. In off-line management, the goal is to structure or correct the system either before starting to provide the service or periodically during down times. Online management refers instead to changes and fine-tuning of the IT infrastructure to respond to a predicted or detected exceptional situation or in general to improve the quality of each individual service being delivered to a customer. Online management pushes the notion of adaptive enterprise to its limit, as the IT infrastructure and the allocation of resources to services and business processes is changed on the fly, to meet the need of each service and process execution. As closed-loop control techniques mature and as the level of trust that operators have in these increases, on-line techniques will be used more and more.

4.5.2 A priori versus a posteriori

Technologies for optimal resource allocation and process and service optimization can be characterized based on the kind of information and technique they use for determining how to improve the process. A priori approaches start from a model of the processes, services, and resources, as well as of the estimated load on the system Based on the model, these approaches derive the requirements necessary to satisfy the desired business goals, either through mathematical analysis or through simulation. For this reason, they are also called *model-based* approaches.

A posteriori approaches treat the system as a black box, and try to infer its behavior by looking at execution data. Specifically, these approaches try to correlate observable parameters of the system (at all levels, ranging from business data exchanged with customers to resource utilization data) with business-relevant metrics, and based on this they perform sensitivity analysis

(identify how variability in a parameter can affect the metric). The most classical examples of a posteriori approaches are those based on statistical analysis and on data mining.

There is a tension between these two options since a very good model that takes into account a number of anticipated scenarios decreases the need for a sophisticated a posteriori system. On the other hand, a very powerful a posteriori technique can do fine despite a poor initial model, since the control-loop anyway adjusts and adapts to changes. In general, a management infrastructure needs both sets of technologies – the former to be used in the case of lack of availability of execution data and the latter to validate the former.

4.5.3 Local versus global management

Another dimension of the management problem is related to the layers to which the technology can be applied and to whether the management platform performs the control and optimizations separately on the different layers or considers all layers in a holistic fashion.

In general, today the technologies that we discussed are applied (if at all) on only one layer. However, we believe that the vision of the adaptive enterprise requires both the application of these technologies to all layers and the use of the results to collectively manage all the layers in order to optimize business goals.

5. CONCLUSION

This chapter has presented a framework and a vision that enable the automated management of an enterprise's IT infrastructure so that it is aligned with business goals and is able to adapt to changes in the business and IT environment to keep the IT at the service of the business. We have motivated why there is an increasing need for such an alignment, why current IT infrastructures are far away from achieving this ambitious objective, what are the challenges that need to be addressed to get there, and which are the technologies that can help companies meet these challenges.

In particular, we believe that the recipe for adaptive business-driven management is based on the basic architectural principle of service orientation and on few key technological ingredients such as resource virtualization, standardized metric definition and integration, and optimization driven by data mining. These technologies, along with the other presented in the chapter, constitute the basic tools that are needed to build a management system that spans across all the layers of the IT infrastructure

and that is able to translate business goals into requirements at the IT (resource) level. Many IT vendors are already going in the direction of integrating these technologies to achieve adaptive management, but current solutions are still far away from the vision painted in this chapter: they either focus on specific layers of the IT stacks (or even specific applications) or they cover the whole stack, but only through ad hoc solutions and long deployment efforts.

The reader will have observed that we placed little emphasis on techniques geared towards automatically modifying a business process. The research on automated process modification is still in the very early stages, and while users may be comfortable with an automated system managing minor application configuration details or load balancing strategies, altering a business process on the fly is something that few IT managers would allow. However, we believe that as management technology mature, and as process models are extended with "hooks" that enable a controlled form of closed-loop management, automated process evolution will also become a reality.

6. REFERENCES

1. G. Alonso, F. Casati, H. Kuno, V. Machiraju. *Web Services. Concepts and Applications.* Springer Verlag. 2004
2. A. Arkin et al.,Web Service Choreography Interface (WSCI) 1.0. W3C Note. August 2002
3. T. Andrews et al. Business Process Execution Language for Web Services. May 2003.
4. P. Barham et al. Xen and the art of virtualization. In *Proc. SOSP 2003*. Bolton Landing, New York, U.S.A. Oct 19-22, 2003.
5. M. Berry. *Mastering Data Mining.* Wiley. 2002.
6. R. Bilorousets et al. Web Services Reliable Messaging Protocol. March 2004.
7. F. Cabrera et al. Web Services Transaction (WS-Transaction). August 2002
8. F. Casati, E. Shan, U. Dayal, M.C. Shan. Business-oriented management of Web services. *ACM Communications*, Oct. 2003.
9. N. Catania et al. Web Services Management Framework (WSMF) Version 2.0. July 2003.
10. R Chinnici et al. Web Services Description Language (WSDL) Version 2.0. November 2003.
11. Data Grid. http://www.eu-datagrid.org.
12. DMTF – Distributed Management Task Force. Common Information Model. http://www.dmtf.org/standards/cim.
13. D. Grigori, F. Casati, M. Castellanos, U. Dayal, M. C. Shan, M. Sayal. Business Process Intelligence. *Computers in Industry*. To appear.
14. M. Gudgin et al. SOAP Version 1.2. W3C Recommendation. June 2003. http://www.w3.org/TR/soap
15. HP. Adaptive Enterprise program. http://www.hp.com/go/adaptive
16. HP. SmartFrog. Smart Framework for Object Groups. http://www.smartfrog.org.
17. HP Utility Data Center. http://www.hp.com/go/udc.
18. IBM. Autonomic Computing program. http://www.research.ibm.com/autonomic/

19. N. Jennings, K. Sycara, and M. Wooldridge. A Roadmap of Agent Research and Development. Autonomous Agents and Multi-Agent Systems, *1(1):7{38*, 1998.
20. A. Keller, H. Ludwig. The WSLA Framework: Specifying and Monitoring Service Level Agreements for Web Services. IBM Research Report rc22456. May 2002
21. Z. Maamar, Q. Z. Sheng and B. Benatallah. Interleaving Web Services Composition and Exectution using Software Agents and Delegation. *In Proc. Workshop on Web Services and Agent-Based Engineering.* July 2003
22. NASA Information Power Grid. http://www.nas.nasa.gov/About/IPG/ipg.html.
23. Oasis Consortium. ebXML: enabling a global electronic market. Information available at http://www.ebxml.org/
24. Oasis Consortium. UDDI Universal Description, Discovery and Integration http://www.uddi.org/
25. Oasis Consortium WS-Resource Framework. http://www.globus.org/wsrf/.
26. G. Pacifici et al. Performance Management for Cluster Based Web Services. IBM Technical Report, 2003
27. J. Rolia et al. Resource Access Management for a Resource Utility for Commercial Applications, In *Proc. Integrated Management (IM)*, 2003.
28. Supply Chain Council. Information available at http://www.supply-chain.org/
29. The Open Group. Application Response Measurement. Version 2.0. http://www.opengroup.org/management/arm.htm.
30. The Oracle Corporation. Grid services with Oracle 10G. http://otn.oracle.com/tech/grid/index.html.
31. UN/EDIFACT. EDI: Electronic Data Interchange. Information available at http://www.unece.org/trade/untdid/
32. VMWare. An EMC Company. http://www.vmware.com.
33. West Grid. http://www.westgrid.ca/home.html.
34. R. Wolski, J. S. Plank, J. Brevik and T. Bryan. Analyzing Market-Based Resource Allocation Strategies for the Computational Grid *In The International Journal of High Performance Computing Applications*, Sage Science Press, Volume 15, number 3, Fall, 2001, pages 258-281
35. WS-Agreement. http://www.fz-juelich.de/zam/RD/coop/ggf/graap/graap-wg.html.
36. X. Zhu et al. Optimal resource assignment in Internet data centers, In *Proc. Mascots*, Aug. 2001.

URLs were last checked for correctness at the time of publication.