RC 22590 (W0210-083) October 14, 2002
Computer Science

IBM Research Report

BPM^2: A Grid-Based Architectural Framework for Business Process Meta Management

Jun-Jang Jeng, Henry Chang, Jen-Yao Chung
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598

BPM^2: A Grid-Based Architectural Framework for Business Process Meta Management

Jun-Jang Jeng, Henry Chang and Jen-Yao Chung

IBM T.J. Watson Research Center
Yorktown Heights, New York
{jjjeng,hychang,jychung}@us.ibm.com

Abstract

Business process management tools and techniques are used in a large number of commercial institutions for automating flows of products and services within corporate and without. It is not uncommon that business processes are involved with versatile collaboration, complicated computation and geographical distribution of data. Hence, it is necessary to combine different technologies for implementing Business Process Management systems. We argue that the traditional BPM is not sufficient for dynamic and adaptive business environment; and a new paradigm on BPM needs to exist – the one we proposed in this paper is called Business Process Meta Management (BPMM). On the other hand, computational Grids are emerging as an infrastructure for next generation computing, enabling distributed resource management, and large-scale computational problems in science, engineering and commerce. In this paper, we introduce a framework for BPMM based upon our proposed BP Grid services that is built on top of Grid services so as to provide uniform, robust, scalable and flexible access to highly diversified business processes and business systems. The proposed architectural framework adopts the Grid services and defines a set of additional components to implement the services of distributed BPMM services on Grid-connected software and hardware platforms.

1 Introduction

A *process* is a specific ordering of activities across time and space, with commencement, a termination, and clearly defined inputs and outputs: an organization for actions. A *business process* refers to a process in which work is organized, coordinated, and focused to produce a valuable product or service. Business processes comprise both internal and external business entities and drive their collaboration to accomplish shared business goals by enabling highly fluid process networks [1]. Business process management technologies are being adopted by more and more technologies to better the efficiency and effectiveness of their business processes within and without. However, managing business process management applications that are executed in the Business Process Management (BPM) systems creates special challenges to an IT organization, and when the applications are critical to business operations and used by almost every role involved the business process, the focus on management issues such as availability, performance and security for that solution grows rapidly. This requires the BPM systems to become more proactive and manage the expectations to the provided management services as well as take the appropriate measures to actively monitor and control the behavior of business processes and critical resources. The gaining popularity of automated business process applications on BPM systems has brought on new demands for how BPM system administrators to manage and maintain the components residing in the systems. Managing business process applications requires knowledge of both business domains and the platforms where they are being executed. The increasing system complexity of BPM infrastructure is reaching a level beyond human ability to manage and secure. This increasing complexity with a shortage of skilled I/T technical staff points towards an inevitable need to *automate* many of the managerial functions associated with BPM platforms today.

This paper proposes a new approach called business process meta-management (BPMM) to the above problems that have surfaced in the BPM domain. BPMM is a higher level concept of BPM. It takes both business processes and BPM platforms as the first class citizens so they can be created, composed, analyzed, virtualized and managed eventually. While the traditional BPM systems are aimed for managing business processes, BPMM systems also manage BPM systems. BPMM systems allow business process applications to exploit and assemble collections of business process resources on an as-needed basis without regard to real business process management platforms. Various BPM systems have been implemented in industry very few of BPMM systems have been implemented so far. This paper is concerned with the architectural models for business process meta-management: that is, with the problems of abstracting business process resources and defining business process QoS, and with business process integration and execution, and other activities to prepare a business process resource for use. We do not address other issues that are conventionally associated with scheduling and allocation in the areas of resource management or the management of other resources such as systems, networks, and storages.

What Are the Problems?

The BPMM environment introduces six challenging architectural problems: business processes autonomy, heterogeneous business process systems, business management policy extensibility, versatile business process organizations, commitment-governed management, and self-adjusting management.

1. The problem of business processes autonomy refers to the fact that business processes resources are typically owned and performed by different companies/organizations, in different administrative domains. Therefore, it is impractical to expect acceptable access policy, QoS policies, security mechanism, use policy, and the like.
2. The problem of heterogeneous business process systems refers to the fact that different organizations may have their own business process systems such business process integration (BPI) platforms, supply chain management (SCM) systems and customer relationship management (CRM) systems, and enterprise

resource planning (ERP) systems. Even two business process systems are used in the organization, different configurations and data models often lead to significant distinction in functionality.

3. The problem of business policy extensibility are related to the above two problems which imply a wide range of BPMM domains, and refers to the fact that a BPMM solution must support the dynamic changes of new domain-specific business management policies without requiring the changes to the implementation modules in the BPMM systems.
4. The problem of versatile business process organizations arises because the nature of business process management applications, most of which inherit the straits of enterprise application integration (EAI) applications. In actuality, many a BPM system is touted for being an inter-enterprise integration platform in one way or another. The collaboration among the business process organizations participating in the same BPMM solution becomes a very challenging issue.
5. The problem of commitment-governed management arises because the BPMM systems in fact regulate the interactions between two autonomous organizations under a contractual commitment between them, which needs to be formalized and enforced. The situations become more complicated when the behaviors of BPM systems need to be regulated as well. For example, an SCM participating organization may prefer a different kind of demand forecast algorithms for some products in an unpredictable market.
6. The problem of self-adjusting management refers to the fact that the survivability of an enterprise is largely based upon the speed of adjusting itself to the environmental change. Similarly, BPMM systems must be adaptive in order to meet the discontinuous change of managed business processes and corresponding management obligations. [3] For example, an e-commerce web hosting environment should be able to *learn* from the historical log data to understand the pattern of logging from customers to pre-allocate more compute cycles to the peak time.

BPMM Examples

An application of BPMM is the management of SCM solutions. SCM is the series of activities that an organization uses to deliver products, services or a combination of both to their customers. Contrast to traditional SCM systems, modern SCM systems relied heavily on the technologies of business process integration. An SCM system is actually a BPM application managing changing business process structure due to the modification of the requirements of products and services, changing business process policies due to the changes of corporate strategies and tactics, and highly diversified business process systems and resources. SCM applications falls into the BPMM domain since an SCM system may comprise multiple BPM applications, which consequently manages various business processes and system resources. SCN is certainly involved with different business process organizations such as suppliers, manufacturers and consumers. A strong need of defining management commitments and enforcing them is present in the communities of SCM systems. Therefore, almost all of the aforementioned problems related to BPMM also belong to SCM systems.

Another notable example of BPMM is the domain of Product Lifecycle Management (PLM), which is a new paradigm for enterprise-level business process management that extends product content knowledge into other enterprise processes by coupling business process management (BPM) technologies with applications focused on product development and manufacturing. Before PLM, applications such as computer-aided design (CAD), computer-aided manufacturing (CAM), and computer-aided engineering (CAE) were somewhat self-governed and encapsulated from the enterprise mainstream. For example, design and manufacturing engineers could benefit from the abundant data associated with a material model of a product, but others in the enterprise did not have convenient access to this data. Thus, the benefits of product knowledge associated with a material mode of the product were not shared with the rest of the enterprise processes outside of product development. The move towards PLM is driven by the new wave of consumerism enabled by the Internet. Customers are beginning to expect to buy products from Web-based storefronts that offer products that can be customized to suit personal needs and desires. Business consumers hold the same expectation as well. Consequently, manufacturers are moving from make-to-stock and assemble-to-order styles of manufacturing to mass customization and personalization. PLM shares the same problems with BPMM with particular emphasis on the seamless and efficient collaborations among participating organizations.

Why Grid?

The early development Grid technologies was motivated by the problems of creating scientific resource sharing applications, e.g., collaborative visualization of large scientific data sets, and increasing functionality and availability by coupling coupling scientific instruments and remote computer and archives. [2] Grid has proven itself a viable infrastructure for distributed resource sharing platforms for scientific computing domain. Grid promises to offer solutions to the construction of reliable, scalable, and distributed systems, all of which are very important characteristics of BPMM systems. To survive today's business environment, an enterprise needs to consider leveraging business processes and system resources in an area beyond the boundary of itself. Moreover, enterprises must integrate distributed BPM systems, applications and data with QoS guarantee, addressing the issues of traceability, navigation, security, and content distribution inside the enterprise as well as with external enterprises. Traditionally, the QoS was associated with computing systems such as mainframes, networks, and storage systems. In the domain of BPMM, the measurement and enforcement of QoS must be supported not only at the system level but also at the level of business processes. For example, an SCM system must provide consistent lead time for some products, despite the health of underlying BPM systems. Thus, it requires flexible business-process-level resource allocation in accordance with management commitments and workload demands. Yet the current paradigm for delivering QoS to business process applications via the vertical integration of BPM platform-specific components and services does not work in today's loosely coupled environment. A key feature of BPMM systems is the support of cross-enterprise cross-platforms business-to-business collaboration such as SCM and PLM. The relationships among the participating enterprises of a BPMM system are, in effect, virtual organizations. Thus, BPMM systems signify the source of request for the style of BPMM covering distributed business process management and distributed system management, which is very close to the focus of Grid technologies in general.

In this paper, we describe an architectural framework for BPMM, named BPM^2 that we have developed to address the six problems of BPMM. In BPM^2, we address the problems of business process autonomy and heterogeneous BPM systems via the introduction of business process managers to provide well-defined interfaces for local BPM systems and business processes. Naturally, following the legend of Grid Computing, the participating parties of BPMM are modeled as virtual organization with well-defined protocols with both one another and BPM^2. To support business management policy extensibility and commitment-governed management, we define an extensible management commitment specification that supports the formalization of management commitments of both managed BPM systems and BPM^2 itself, and we introduce business process broker to conduct the mappings between business-process-level management commitments and system-level management commitments, e.g. service-level agreements, which are complied by all governed components. To realize the adaptive of BPMM systems, we develop the configuration managers to perform *meta-management* upon BPMM components, e.g, changing the management commitments, altering business processes by consulting intelligent analytics components, discovering the usage patterns and the like. Another services called business process analyzers are used to analyze the behaviors of managed business processes and systems in order to improve the effectiveness of BPMM systems.

The rest of this paper is structured as follows. In the next section, we review current business process management systems. In the subsequent sections, we first outline the conceptual framework of BPMM and then describe our architectural framework BPM^2 with each major function explained in detail: the management commitment specification, business process resource managers, business process probes, business process managers, business process brokers, business process evaluators, business process evaluators, and business process explorers. We summarize the paper and discuss the future work in the last section.

2 The Conceptual Framework of BPM^2

In general, BPM^2 can be categorized as a system that is continually interacting with its managed substrate, and that is capable of autonomous actions in this substrate in order to meet its management commitments. A platform for BPM^2 takes inputs from the managed substrate, and produces actions that affect it. As such, BPM^2 interact directly or indirectly with the situated entities in the managed substrates. Examples of situated entities include business processes, business organizations, managed resources, and business systems, where

a) *Business processes* are the first-class citizens that can be observed, measured, analyzed and managed. In the example of PLM, managed business processes can be supply-chain management, customer relationship management, enterprise resource planning and so on.

b) *Business organizations* refer to the participating parties of the business processes and BPM systems which are managed by BPM^2. Business organizations can come from many roles: enterprises, business analysts, BPM system administrators, BPM integrators and the business executives who are interested in know the status of business processes.

c) *Business Policies* refer to the management contracts established between BPM^2 and virtual organizations. An example is that the maximum cycle time of some supply chain management process shall not be greater than 48 hours. If the agreement was violated, certain amount of penalty will be exerted.

d) *Business Systems* comprise of manageable entities that are situated in the environment. Resource's manageability defines information that is useful for managing a resource and details the aspects of the resource including the instrumentation which allows BPM^2 to interact with it. There have been many standards of defining manageability at various levels, e.g., SNMP [4], CIM [5] and M12 [6]. Through instrumentation, a resource is turned into *managed* resource because its state can be perceived, aggregated, analyzed and modified through the standard interfaces provided by the instrumentation layer that is located between BPM^2 and its environment.

Horizontal Decomposition

The functionality of BPM^2 can be decomposed either horizontally or vertically. Figure 1 shows that BPM^2 is decomposed horizontally into three pillars: *perception*, *evaluation*, and *actuation*.

1. Perception pillar receives the data and events from the BPMM substrate.
2. Evaluation pillar processes the perceived information.
3. Actuation pillar renders management directives to the managed entities within the substrate.

Conceptually, the Evaluation pillar can be further decomposed into three sub-functions: *measure*, *transition*, and *adapt*; and four local data stores: *percepts*, *metric stores*, *control states*, and *commitment store*. The percepts store contains the perceived values from the BPMM substrate. The function *measure* computes the metrics according to the values of percepts and stores them into the *metrics store*. The control states represent the current situation of BPM^2 as a whole. Control states are different from the states perceived from the substrate. In general, control states capture the status of BPM^2 as a whole, and the environment states capture the status of the managed entities within the BPMM substrate. The function *transition* changes the control state of BPM^2 according to the current control state, the local commitments, and the resultant metrics. The function *adapt* changes the local management commitments according to certain the existing control state and the business policies. The commitments store contains the local management commitments that the BPM^2 enforces on the managed entities of the substrates, business organizations, business systems and business processes.

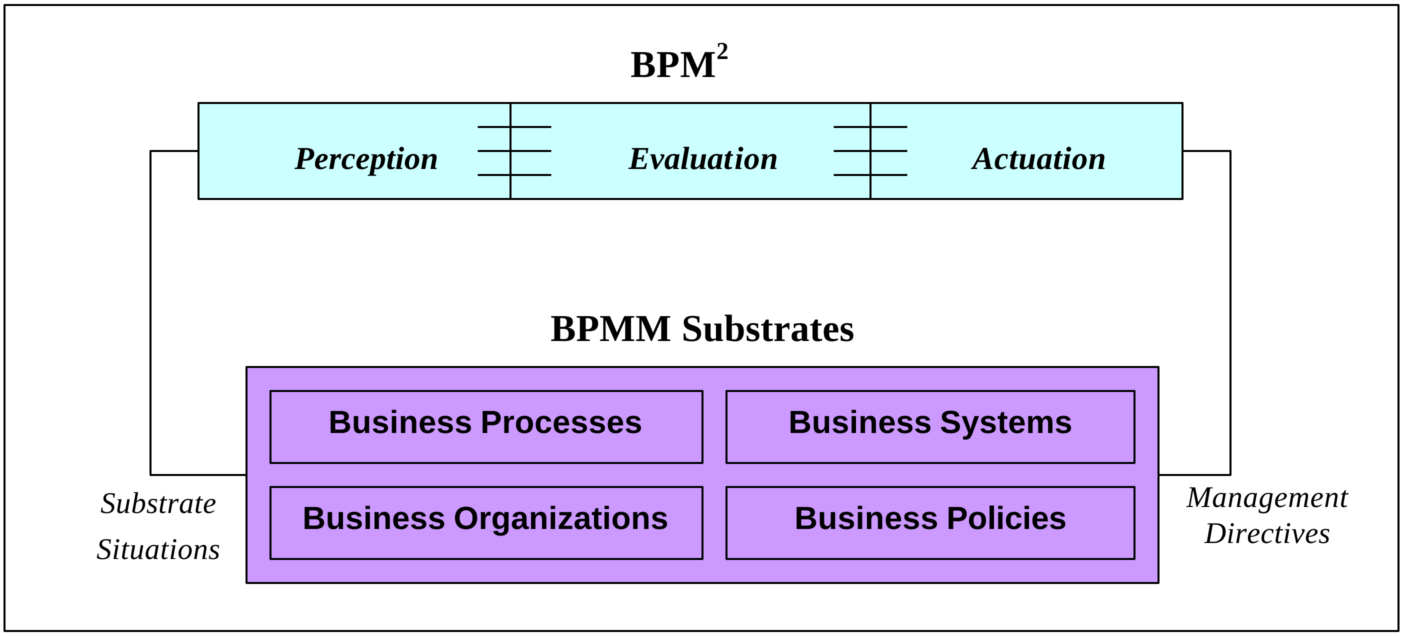

Figure 1: Horizontal Decomposition of BPMM.

Vertical Decomposition

BPM^2 can be also decomposed vertically into three layers: *Reactive* Management, *Deliberate* Management and *Reflective* Management. Figure 2 illustrates the vertical decomposition, where solid lines represent the flows of management directives, and the dotted lines represent the flows of management events.

1. Reactive management layer responds to the management events quickly and directly through scripted business process models. A notable example of the reactive management is deterministic workflow management where workflow models are defined at the build time and executed at the run time. Another example is the alarm system that will notify the system administrator if some managed resource is suffering from severe performance problems, and demanding immediate attention.

2. Deliberate management layer performs managerial tasks that require more reasoning and more complicated computation. It is not uncommon that BPM^2 needs to provide decision support capability so more intelligent management directives can be derived towards managed resources. An example of such managerial tasks is the business processes with the ability of sense-and-respond [3][7]. Another example of deliberate management is mapping QoS metrics from IT-level into business process level and vice versa. An event such as “disk failure” may mean little out of business context. However, it may imply a loss of gigantic capital for a business organization if it has causal relationship with critical business process performance such as financial trading. The mapping rules in the layer of deliberate management should capture this relationship to prevent business loss.

3. Reflective management layer enables BPM^2 to maintain information about itself and use this information to remain extensible and adaptable [9]. Reflexive management layer performs meta-management directives unto the lower management layers and managed entities. A meta-management directive is a higher sphere of control such as adapting the management commitments, modifying measurement and analysis

algorithms in the deliberative management layer, or changing the alarm rules in the reactive management layers. As such, BPM^2 can have detailed knowledge of the managed resources, current status of managed business processes and business systems, the ultimate capacity in the inventory, performance expectation, and all connections to other systems to manage itself. Therefore, through reflective management mechanism, BPM^2 achieve the goals of both 2^{nd} order management and autonomic computing [8].

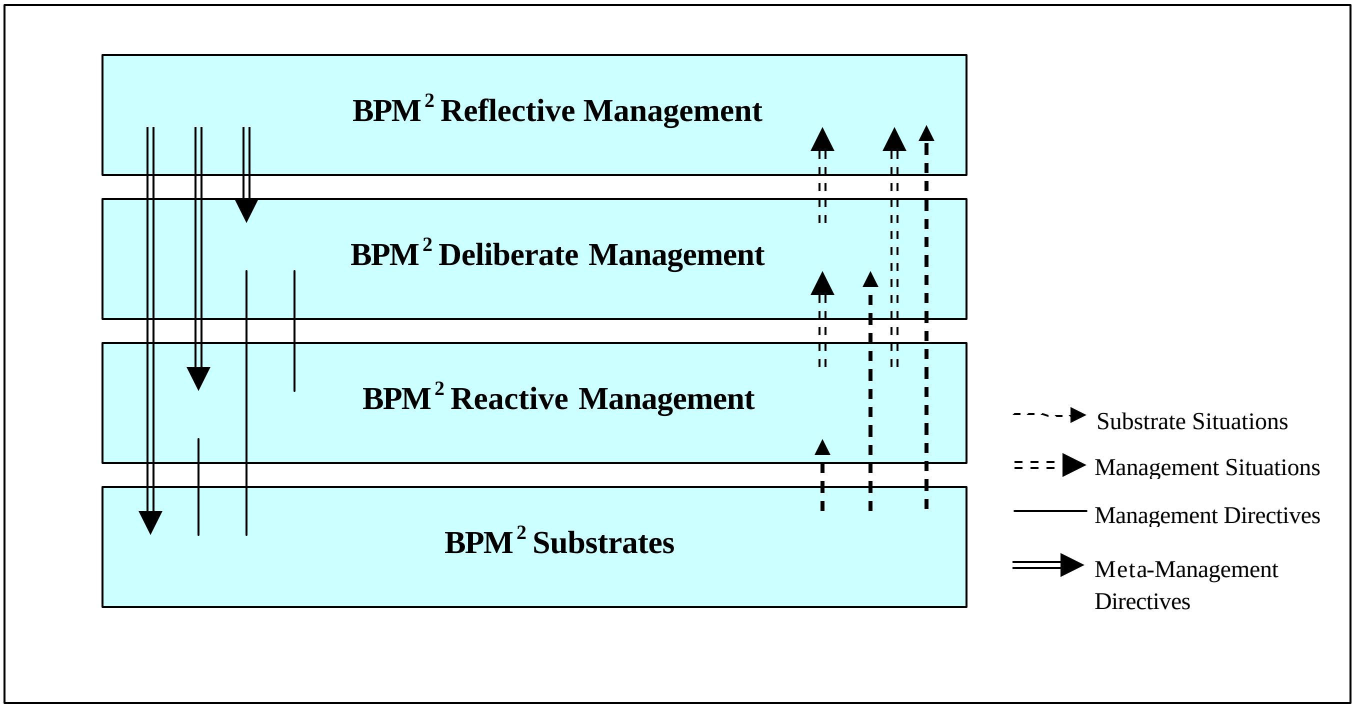

Figure 2: Vertical Decomposition of BPMM.

Mesh Decomposition

Horizontal and vertical decompositions can be combined to form *mesh decomposition* as shown in Figure 3. Mesh model is used as a formal modeling tool for cognitive architecture [10]. We found that Mesh model is very suitable to illustrate different architectural aspects of management spaces for BPM^2. Figure 3 defines nine regions, ($S_{i,j}$ 1= i,j = 3), called *management spaces* (or *m-spaces*) and their potential interactions. Only legitimate flows, both management directives and management events, are allowed to be transmitted between m-spaces. The entities in the BPMM substrates emit management events and receive management directives. Evaluation is completely an internal processing inside BPM^2 without interaction with its substrate. The meta-management directives are rendered only through the actuation pillar. Within BPM^2, management events can be generated or transformed between layers upwardly and management directives can be rendered downwardly.

Figure 3 shows two typical management scenarios based upon the decomposed m-spaces:

1. Flow A is a common management scenario: management events are perceived by the reactive management layer, the evaluation is performed by the deliberate management layer, and the actuation is activated through the reactive management layer. This management scenario is called O model of management flow, which is the most common scenario among all.
2. Flow B is a meta-management scenario where management events are delivered all the way to the reflective layer, evaluated, eventually some meta-management directives are delivered through the actuation pillar in the reflective management layer. An example of such scenario is data mining of the business data residing in the substrate. The reactive management layer extract the data, the deliberate management layer transforms and

loads the data, and the reflective management layer analyzes the data and makes/suggests business decisions according.

We refer to members of a management space as *management agents* or *m-agents* shortly, by which we mean autonomous actors interacting with each other, and with the substrate. Such an m-agent can be a software component, with its own state and thread of control, or it might be a proxy for human users interacting with BPM^2 with some interface. Mesh decomposition model actually define a couple of m-agent classes, each of which embodies certain rules of engagement that its situated m-agents must abide. For example, the m-agents of m-space $S_{2,3}$ only receive management events from m-space $S_{2,2}$ and must comply the management commitments imposed by the m-agents in m-space $S_{1,3}$. Consequently, the BPM^2 conceptual framework entails regulated m-agents, management events, managerial functions, and management directives. The following section will address how the rules of management can be defined and enforced via BPM^2.

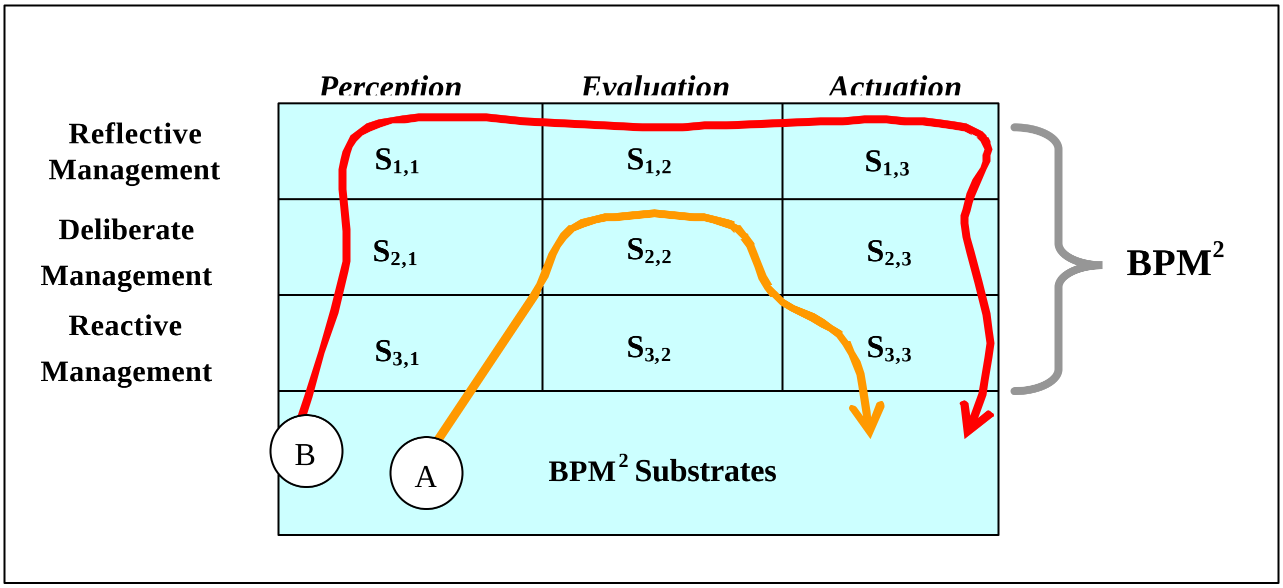

Figure 3: Mesh Decomposition of BPMM.

3 The BPM^2 Architecture

The overall architecture of BPM^2 is shown in Figure 4. Business organizations (companies, business analysis offices, product design teams in PLM, and business executives) are modeled as *virtual organizations* (VOs), each of which may own some business process resources and system resources that are available for sharing across networks [12]. Business systems such as SCM, CRM, ERP and general BPM systems are modeled as Business Process (BP) Grid Services, which are network-enabled entities that provide Grid specific capability through exchange of messages. This Grid Service centric view separates the concerns of definitions (Grid Services) and implementations (real business systems themselves). The Open Grid Service Architecture (OGSA) [19] supports the creation, maintenance, and application of the Grid service collections that VO maintains.

BP Grid Services are an extension of Grid Service Specification [13] with the focus on defining standard interfaces for business systems and interested resources. We will account the main features of BP Grid Services in this paper. BP Grid Services simplify virtualization through encapsulation of diverse implementation behind a common interface. Virtualization of business organizations enables consistent access to business systems and managed resources across heterogeneous business systems. Virtualization also enables mappings of multiple logical business process instances on the same physical business processes that are embodied in business systems.

Furthermore, virtualization helps us compose services from various business processes and business systems without regard how these services are implemented and configured in their respective business systems. Business process grid services can be regarded as an adapter to be installed at each virtual organization as a wrapper of its real services so they can be accessible on the Internet and participate in an Internet-based business process operations and workflows.

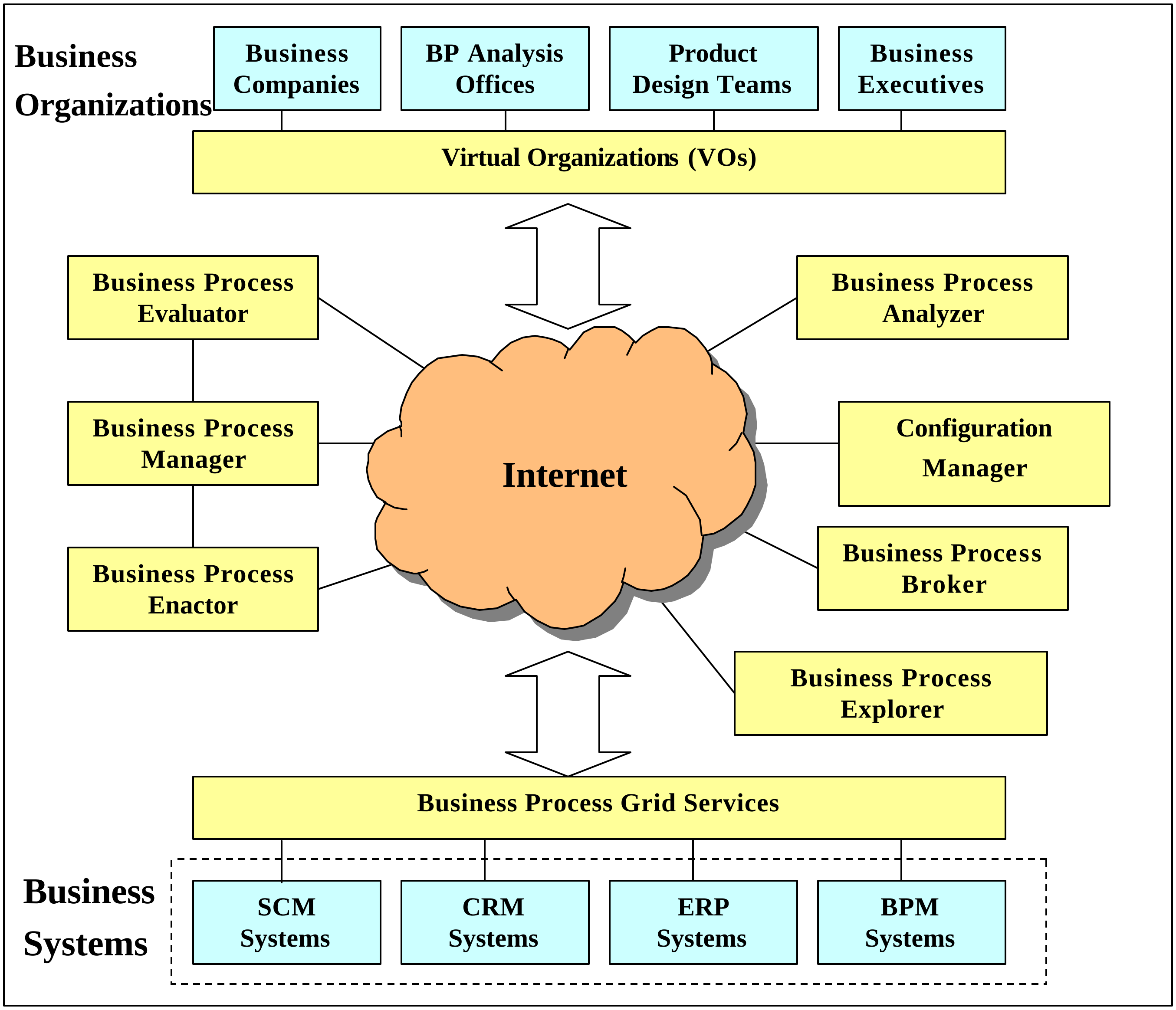

Figure 4: The BPM^2 Architecture.

The developers of business systems might implement *Business System Adapters* that wrap the underlying system-specific services of VOs as BP Grid services. This module takes existing business system implementations, to create BP Grid services. The business system adapter thus hides the heterogeneity of business process implementation and presents a uniform view of these system-dependent services as BP Grid services. The adapters are the components actually performing the service invocation on the underlying business systems. The business system adapter deals with functions such as translate standard BP Grid service queries to system-specific invocation methods. In specific term, business system adapter parses SOAP request and determine the business system services to be invoked. Using mapping information, suitable objects can be dynamically created by the adapter and some method invocation is performed. It builds the parameter list that is required by the underlying business system services and invokes the services. The business system adapters deal with the issues like creation of proxy objects for underlying business system services, management of proxy objects, translation of business process queries from SOAP to native business system calls, and translation of service results back to the SOAP format. We envision the adapters will be available as COTS as the BP Grid services become popular.

The *Business Process Manager* is a Grid Service composed of two sub-services: Business Process Evaluator and Business Process Enactor, the former perceives business process situations and computes business metrics according to pre-defined management commitments; the latter reasons about the next activity to be executed in the business process and schedules the enactment of business processes accordingly. The scheduling of executing activities may come from scripted business processes such as workflow model specification (reactive management layer), or non-deterministic business process rules (deliberate management layer). The choice of BP enactors is recorded in the management commitments.

The *Business Process Brokers* are responsible for taking high level business process queries and transforms them into more concrete queries at lower levels such as business systems and operating systems. Business process brokers receive business process requests from virtual organizations, translate the queries into more concrete BP Grid service queries. Then, the queries are delivered to appropriate business process managers for business process enactment. The *Business Process Explorers* are used for both discovering business process managers and locating available business process resources that meet the management commitments. Transformation effected by business process brokers creates a specification in which the locations of business process resources can be fully specified. Such a request can be sent to *Business Process Evaluators* to evaluate the business process conditions and enact business process execution at multiple business systems. Business process managers break a business process request into its constituent elements and pass each component to the appropriate business process enactor. Each *Business Process Enactor* is responsible for taking the business process query and translating it into business process operations in the local business systems. A business operation consists of activities, services, and the parameters that are used for invocation. The BP Grid Services provides the status and data about the availability and capability of the managed business processes and business system (in the substrates). The information is used to locate business processes and business systems with specific characteristics, to identify the BP resource managers associated with this business process, to determine the properties of interested business processes and business systems, to decide and modify the access privilege of service clients, to obtain business metrics from a selected range of business processes, and for many other purposes as high-level business process queries are translated into requests to specific BP managers.

Business Process Analyzer provides the services performing behavioral analysis on the business processes and business systems that are managed within the BPM^2 environment. A notable example is the services of knowledge discovery and data mining [20]. Business process analyzer is itself a Grid Service that can be discovered and invoked through SOAP based messages. For the domain of business process management, it is both critical and challenging to understand the behaviors of the systems and users from the viewpoints of business processes. This understanding cannot be achieved by logging and tracing only. More useful information needs to be extracted and discovered from large amount of business data. Business process analyzer is a Grid service built on the top of BP Grid services that provide dependable, consistent, and pervasive access to business process resources and to high-end computational resources as well. Business process analyzer analyzes and predicts the behaviors of managed business processes and provides suggested changes of management commitments and configuration accordingly. Considering the process of supply chain management, the business process analyzers may trigger demand forecast Grid services to predict the demand of some products in next 30 days, which consequently triggers another set of optimization Grid services to provide recommended build plans for the product. Therefore, the behavior of the business process has been changed due to the actions of business process analyzers and the result of invoked analysis Grid services, i.e., demand forecast and optimization.

Similar to the role of business process analyzer, the *Configuration Manager* is kind of Grid services for the purpose of meta-management with the capability of self-reflection and self-adjustment. In general, the configuration manager is the "executioner" of meta-management actions, e.g., replacing build plan with a new one in the above example. At the start-up time of BP Grid services, they are initialized and configured by the configuration manager via the exposed Grid operations. At the run time, the configuration of the system is enforced by the configuration manager based on the associated management commitments. In this paper, we will focus on describe the following three types Grid services and one management paradigm used in BPM^2:

a) BP Grid Services;

b) Commitment-Governed Management;

c) Business Process Brokers; and

d) Business Process Managers.

4 BP Grid Services

One of the foremost goals of BP Grid services is to enable them to be accessed in an access paradigm which is independent of execution environments such as BPM systems. BP Grid service is a higher level abstraction than conventional BP management systems and WebServices. BP Grid services based on both Grid computing and WebServices provide a set of standard interfaces for vendors to develop open platforms for business process management. On the other hand, BP Grid services by themselves are WebServices, they are provided by the business process service provider and are utilized as needed. The data exchange is by default using SOAP as the format. Since the data is well presented in BP Grid services, similar to WebServices, they can be easily searched, discovered and composed during business process discovery. BPM^2 provides the suitable interfaces to register the BP Grid services into UDDI-like persistent business process and service registries. Similar to Grid service, BP Grid services also distinguish business process definition, business process binding and business process implementation. A business process can support multiple implementations on multiple business process management systems, facilitating seamless overlay to native platform facilities. The business process implementation can dispatch operations and queries to lower-level services such as activities and workflows.

BP Grid services virtualize business processes, managed resources and business systems. Description of BP Grid Services is very important as it helps the services to be discovered, composed and invoked remotely in a homogenous fashion. A BP Grid Service description has to provide the required information on the bindings and invocation of the Grid service. In our implementation, we create an extension of Grid Service Specification [13] to describe the BP Grid Service interfaces. A Grid service is a WSDL-defined service that conforms to a set of conventions relating to its interface definitions and behaviors.

A Grid service document uses the following elements in the definition of Grid services:

i. InterfaceNaming: naming conventions and immutability of portType, serviceType and serviceImplementation names.

ii. CompatibilityAssertion (extends definitions): mechanism to associate equivalent interface elements and implementations.

iii. serviceType (extends definitions): named aggregation of portType elements forming an interface definitions.

iv. serviceImplementation (extends services): mechanism to assert implementation semantics with a serviceType.

v. Grid Service Handle (GSH): conventional use of URL to act as unique identifier of a Grid service instance.

vi. Grid Service Reference (GSR): mechanism to convey capabilities of a service to a client, can be a WSDL document.

vii. handleMap (portType definition) OGSA map service to map GSH to GSR.

BPGridService is an extension of GridService. Every BP Grid service must support BPGridSerivce interfaces; in addition, we define some operations for applications to interact with BPGridServices. A BPGridService consists of the following data elements:

i. process (extends service): generic interface of business processes.

ii. processType (extends definitions): named aggregation of activityType.

iii. activity (extends service): generic interface of activities.

iv. activityType (extends definitions): named aggregation of serviceType.

v. processImplementation (extends process): mechanism to assert implementation semantics with a processType.

vi. activityImplementation (extends activity): mechanism to assert implementation semantics with a activityType.

vii. BPGridServiceHandle (BPGSH): conventional use of URL to act as unique identifier of a BPGrid service instance.

viii. BPGridServiceReference (BPGSR): mechanism to convey capabilities of a service to a client, can be a WSDL document.

ix. BPHandleMap (portType definition) OGSA map service to map BPGSH to BPGSR.

BPGridServices consists of the following definitions of operations and messages, which are manadatory.

i. findBusinessProcessData: an interface to query the data of a BPGridService. Examples of business process data include owning VOs, business process definitions (such as activities), platforms where the process is running upon, governing management commitments, resource requirements etc.

ii. queryBusinessProcessStatus: used to get the current status of execution of the business processes, typically return values include status like "expected to complete in 30 min" or "have been idle for 2 hours", and so on.

iii. queryByBusinessProcessDataName: an interface to query a BPGridService and result in all service data elements with the specified name.

iv. notify: as a unidirectional notification message to a BPGridService.

v. subscribe: as a mechanism of subscribing the business metrics generated by the BPGridSerivce.

vi. migrate: as a call of migrating the BPGridService from current business system to another.

vii. commit: commits the current execution of BPGridService.

viii. abort: aborts the current execution of the BPGridService.

ix. passivate: pauses the execution of the BPGridService.

x. resuscitate: revives the execution of the passivated BPGridService.

xi. addProbe: adds business process probes to the BPGridService. BP probes can emit useful information of business processes to interested VO participants and Business Activity Monitors.

xii. removeProbe: removes business process probes from the BPGridService.

BPGridServices can maintain internal state for their lifetime, which is very important for a long-running business process. At run time, BPGridServices are often instantiated as a new transient service instance dynamically to handle the interactions and collaborations with the state of specific business processes. When the state is no longer required, the BPGridService can be destroyed. For example, in a supply chain management environment, performing an optimization algorithm might involve creating BP Grid services at the intermediate points to extracting historical data from remote databases and to manage end-to-end data flows according to pre-defined QoS constraints in the management commitments. While BPGridService is prescriptive on the matter of basic business process behavior, it does not address on what a business process does and how it is implemented in the execution environment. In the BPM^2 context, the container of BPGridServices have the primary responsibility for ensuring the business processes adhere to BP Grid service semantics.

5 Commitment-Governed Management

BPM^2 characterizes two perspectives of BPMM: policy and mechanism. Management commitments specify the aspect of management policies. A management commitment commits the behavior of BPM^2 to both management contracts and service level agreements among virtual organizations. Experiences manifest that service level agreements alone do not provide the full spectrum of the useful management policies required in BPM^2. In general, a service level agreement merely defines the *external* management contract among participating parties, and provides limited information of the *internal* management contracts.

In general, a management commitment is a synthesized form of internal and external management contracts. Management commitments are enforced the business process brokers when a business process query is granted. It can be concluded that the behavior of BPM^2 is governed by the management commitment, therefore, the management style of BPM^2 is called Commitment-Governed Management (CGM). The value of BPM^2 implementation really depends on how well the managerial behavior of BPM^2 can be governed such that it can comply with given management commitments.

The change of business context usually implies the change of management commitments. Hence, in a dynamic business process environment, static management commitments are insufficient. Management commitments are designed to have run time representation and can be dynamically updated, of which both features enable BPM^2 to create, manipulate, and propagate the information of management commitments. The need of such features can be understood in the context of supply chain management processes.

a) From a *buy-and-supply* perspective, the functions for optimization analysis enable better forecasts and flexible risk management, therefore greatly reducing the risks. The result of optimization changes the management commitments with improved partner management and risk sharing, both of which further simplify buy-and-supply activities, and allow companies to focus on collaboration with their partners to gain support with minimal risks.
b) From a *make* perspective, the optimization results in the change of management commitment of the functions for flexibility and better forecasts, both of which enable dynamic risk management and partner value analysis, thus increasing flexibility and decreasing response times.
c) From a *sell-and-support* perspective, the changed management commitments imply the functions for planning analysis, which enable enterprises to optimally plan product offerings and sales, optimize channels strategies, as well as making quick, informed decisions about changing market demand. The management commitments about dynamic pricing capabilities enable companies to flexibly set the price to optimize profits.
d) Finally, the management commitments for intelligent resource planning support the functions for the allocation of resources to different enterprise activities, considering the risk and return of the activity and its strategic importance to the business.

The change of management commitments makes the functions for SCM more adaptive to the business process environment. To support the above functions, the business process managers must comprise the following functionality: agents

i. Perception grid services that sense, monitor, and organize critical business information from the environment (cp. Figure 1);
ii. Analysis grid services that analyze perceived information and conduct certain evaluation tasks;
iii. Filter grid services that filter perceived enterprise data to enable management directives to disturbances, which are events, in the environment;
iv. Response grid services that analyze global value chain relationships and information, and derive the optimal strategy for the best SCM performance;
v. Predictive grid services that provide predictive modeling capabilities, e.g., demand forecast (part of the reflective management layer);
vi. Learning grid services that are capable of learning by comparing previously predicted trends with recorded data and information in order to improve future response performance (also part of the reflective management layer).

Regarding CGM and management commitments, the following design principles are recognized:

i. Management commitments should be explicitly defined by Management Commitment Specification (MCS) [14].
ii. Management commitments must be enforced by BPM^2.
iii. Management commitments comprise the management policies (monitoring, evaluation, prediction, data aggregation and actuation).
iv. Management commitments specify the management contracts (service level agreements) and policies of BPM^2 – both internal and external.
v. Management commitments should define management commitments at different levels of abstractions.
vi. Management commitments and CGM implementation should be developed independently, hence for business process autonomy.
vii. Management commitments can be deployed and enforced incrementally.

The major abstraction mechanism of MCS is defined as follows:

i. *dimension* defines the dimensions that can be used to characterize a particular management commitment perspective. Each dimension contains a set of metrics that can be monitored and measured during process execution. A metric has a domain of values that may be ordered. There are three kinds of domains: set domains, enumerated domains, and numeric domains.
ii. *context* defines the condition of the business process context that will is to enforce management commitments. For example, in SCM, four primary business processes of Plan, Source, Make, and Deliver can be defined into the context to indicate which organization this management commitment is aimed for.
iii. *process* defines the type of the business process this management commitment will be used. In Supply Chain Operation Reference Model (SCOR), three basic process types are defined: *planning*, *execution* and *enabled* [15].
iv. *perceive* defines a set of business and IT events that are relevant to the evaluation of this management commitment.
v. *evaluate* contains a set of commitment predicates. The disjunction of all the predicates represents a specific management commitment that business process grid services will observe, obey, and enforce.
vi. *actuate* defines the actions that must be taken when a management commitment is violated.

Considering the *Planning* business processes of SCM domain align business process resources to meet expected demand requirements (embodied business process queries). On the other hand, the *Execution* business processes are triggered by planned or actual demand that changes the status of targeted products. Both business processes include scheduling and sequencing, transforming materials and services, and moving product. The *Enabled* business processes prepare, maintain, and manage information or relationships upon which planning and execution processes rely. Business process brokers map business process-level events (actions) to physical events (actions) at the concrete system level, and vice versa. Business process evaluators are used to evaluate the metrics of business processes at the triggering point, which can be driven by either events or time. An example of management commitment is given as follows.

```
type Reliability = dimension {
  DeliveryQuantityPerformance: type/numeric,unit/month;
  SupplierOnTimeDeliveryPerformance: type/numeric, unit/sec;
  ProductReceivedWithoutDeliveryIssuesOrErrors: type/numeric;
};
type Flexibility = dimension {
  SourceCycleTime: type/numeric unit/sec;
};
type Cost = dimension {
  InventoryCarryingCost: type/numeric unit/dollar;
};
type commitment SCMReceiveProductCommitment
     concerns Reliability, Flexibility, Cost;

commitment ReceiveProductCommitment  instantiates
SCMReceiveProductCommitment {
context: ReceiveProductBusinessProcess;
process: Execution;
perceive: E1 ProductReceivedEvent
perceive: E2 TimerEvent
evaluate: C1
      DeliveryQuantityPerformance < 100;
evaluate: C2
      SupplierOnTimeDeliveryPerformance < 50;
evaluate: C3    ProductReceivedWithoutDeliveryIssuesOrErrors
               < 50;
evaluate: C4 SourceCycleTime>10;
evaluate C5 InventoryCarryingCost > 2000;
actuate: C1 or C2 or C3 -> invoke ReliabilityAnalyzer;
actuate: C4 -> invoke PerformanceOptimizer;
actuate: C5 -> invoke InventoryOptimizer;
actuate: any -> notify SolutionOwner
actuate: any ->log BusinessEvents;
};
```

The above specification includes three dimensions: Reliability, Flexibility and Cost. The Reliability dimension defines three kinds of metrics. The first metric represents the delivery performance of the interested products. It also indicates the type of this dimension and the unit of measurement. The second metric indicates the "Supplier On Time Delivery Performance." The 3rd metric indicates the counts of "Product Received Without Delivery Issues Or Errors." The 2nd management dimension Flexibility contains one metric: "Source Cycle Time" with unit of measurement as second. ." The 3rd dimension Cost contains one metric: "Inventory Carrying Cost" with unit of measurement as dollar. The commitment type SCMReceiveProductCommitment includes the dimensions Reliability, Flexibility, and Cost by using the keyword *concerns*. We finally define a commitment ReceiveProductCommitment that *instantiates* the commitment type SCMReceiveProductCommitment. It is indicated that the *context* where this commitment will be situated is the business process called ReceiveProductBusinessProcess. Its process type is Execution. Two business process events must be monitored: ProductReceivedEvent and TimerEvent. The *commit* phrases C1, C2, C3, C4 and C5 describe the violation predicates that need to be evaluated and enforced by BPM^2.

The *actuate* phrases indicate what actions need to be taken when some management commitments are violated. For example, if the violation predicate of C4 is true (i.e., source cycle time > 10) then the performance optimizer will be triggered in order to analyze this problem and possibly provide remedy. The last *actuate* phrase indicates that any commitment violation should trigger both logging and sending alarms to the solution owners.

Commitment-Governed Instrumentation

Figure 5 illustrates the commitment-governed instrumentation based upon the management commitments, where

1. BP Probes detect management events from the business processes and business systems (substrates).
2. BP Evaluators measure the business metrics of business processes based the data captured by the BP probes.
3. BP Enactors render management directives to the business processes and business systems situated in the substrate according to the management commitments.

The proxies of BP probes, BP evaluators, and BP enactors are bound to the business process managers through the instrumentation process that is triggered and controlled by the configuration manager. What instrumentation achieves is to establish the referential relationships among the grid services. For example, the business process brokers were connected to different business process managers based upon the management commitments. the Deliberate Management layer in the vertical composition, the BP brokers are also responsible for the following mappings that are needed to enforce management commitments properly.

1. Semantic mappings of business process events between different levels. For example, the monitored event ProductReceivedEvent can be mapped to underlying database events indicating the change of the product status. As the status is changed, a database event will be sent to sensors that subsequently generate the event ProductReceivedEvent accordingly, and notify the commitment coordinator.
2. Semantic mapping of actions between different levels of abstractions. For example, the commitment violation C5 will trigger the actuation “invoke InventoryOptimizer,” which can be translated into an ordinary method call to an EJB named InventoryOptimizerBean for performing optimization of the inventory by consulting the events, historical data, and business context.
3. Semantic mappings of evaluation mechanisms between different levels of abstractions. A BP evaluator may be mapped to a set of concrete analysis algorithms located in SCM business systems.

Since management commitments are decoupled from the business process execution models of the underlying business systems. BPM^2 can be integrated with third-party instrumentation subsystems to enforce management commitments. This paper does not cover in detail on lower-level, e.g. network management, instrumentation because much work has been done in this area [16][17][18]. Commitment-governed management is attributed to the reflective management layer of the vertical decomposition in the BPM^2 conceptual framework (Section 2).

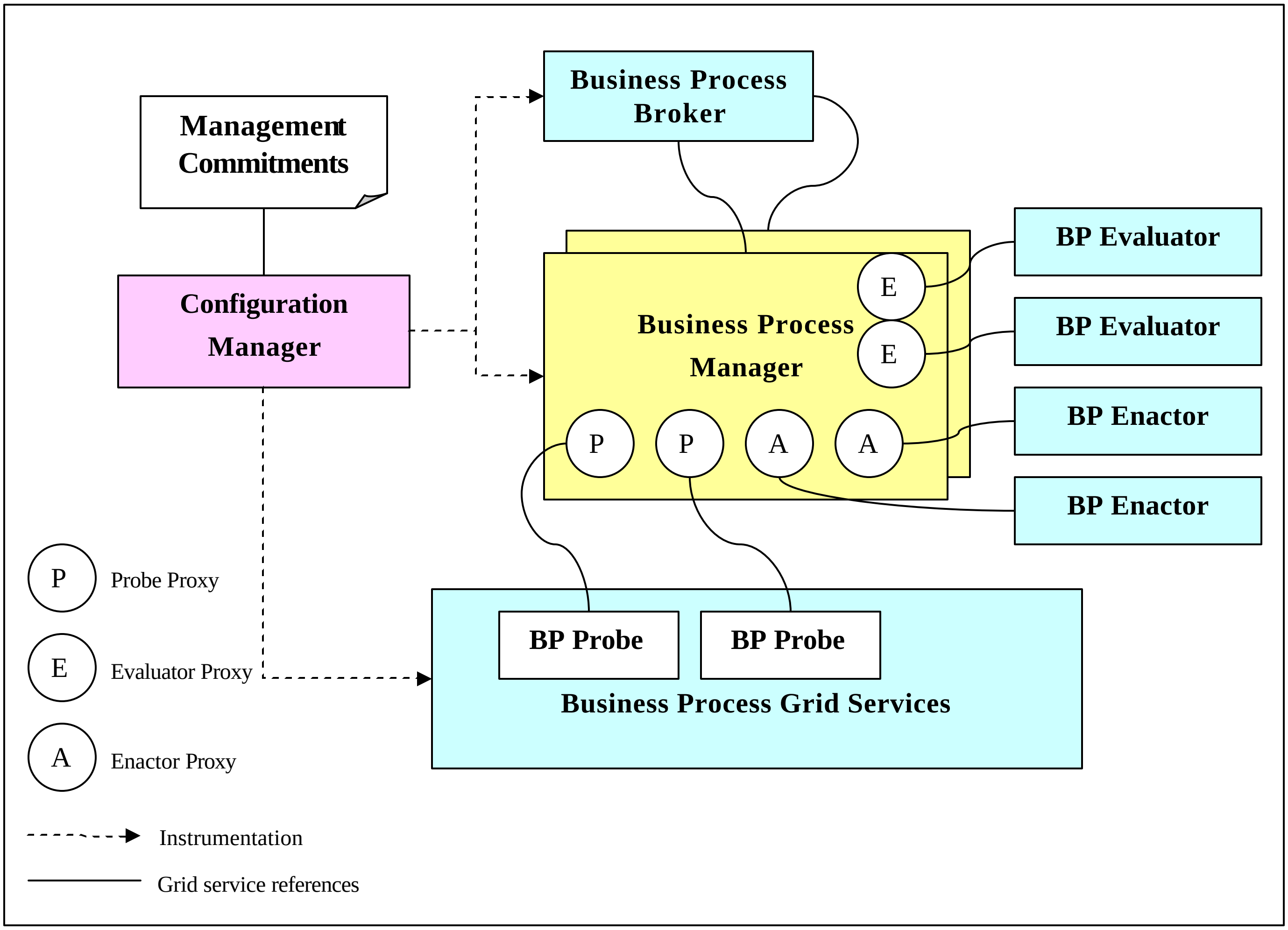

Figure 5: Commitment-Governed Instrumentation.

6 Business Process Brokers

Figure 6 shows the relationship between the business process broker with other entities. The *Business Process Provider* is the owner of a service, i.e., virtual organization in the Grid term. The *Business Process Consumer* is the client of the services provided by business processes. The *Business Process Broker* maintains a local node of UDDI-like *Business Process Repository* containing the business process descriptions from the business process providers.

The operations provided by the business process broker include:

i. Publish/Un-publish. The business process providers publish their business processes to the business process broker(s) in the form of BP Grid services. Published business processes can also be un-published from the business process registry.

ii. Query. The business process consumer sends business process queries to the business process broker to request the services of business processes. The queries can be in different form than BP Grid services. The business process broker then translates this query into concrete queries and sends them to the repository to conduct searching.

iii. Select. When a set of BP Grid services is found to be a match, they will be selected by the broker based on search constraints given by the consumer. The search constraints come with many forms, e.g., performance requirements, the availability of various types and resources, the owners of the business processes, and the like. The issues of matchability between business process queries and BP Grid services will be covered in our later papers.

iv. Notify/Bind. The business process broker use its knowledge about business processes and their provider to provide the possible binding information for the business process consumer.

In our implementation, the business process queries are written in a business level SQL-like language. The broker *specializes* [21] the business process queries to BPGridService-compliant queries before they are delivered to the repository for searching. Queries can be passed to multiple brokers, effectively to composing the concrete queries figured out by different brokers, until eventually the query is specialized to the point that it can be used to search against the business processes stored in the repository.

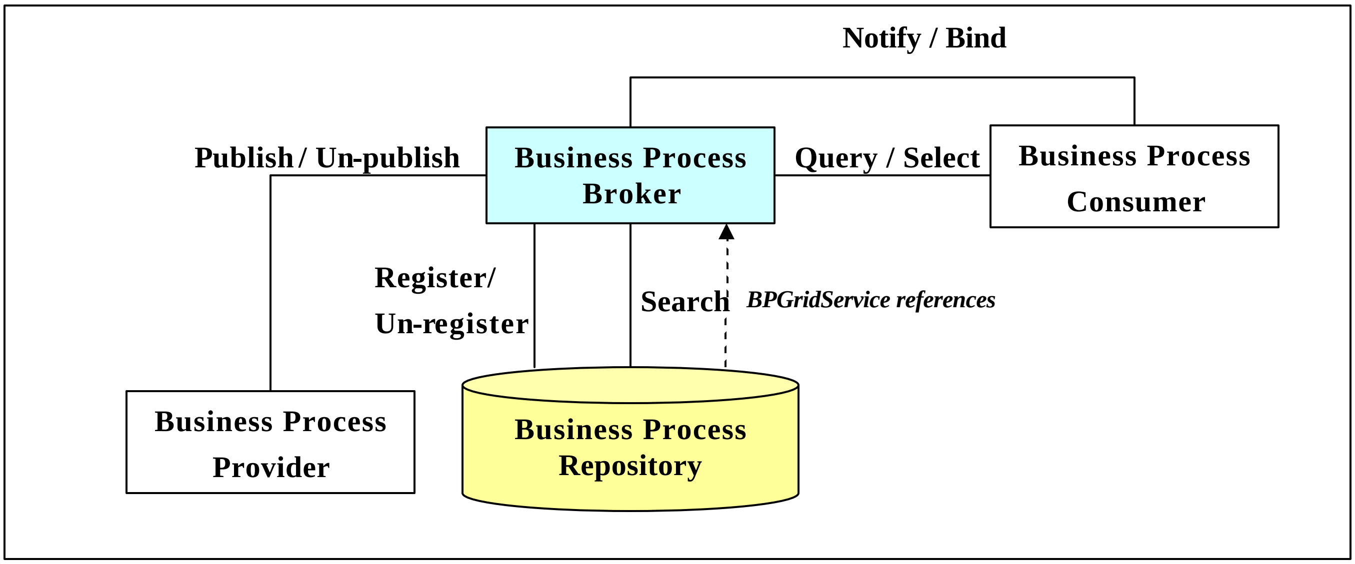

Figure 6: The BP Broker and other Components.

7 Business Process Managers

Figure 7 shows a Business Process Manager and other Grid services. The business process manager consists of three parts:

1. Business process probes that emit necessary data to its owning business process manager.
2. Business process evaluators that, based on the criteria given in the management commitments, evaluate the *conditions* of those managed business processes.
3. Business process enactors that is responsible for the execution of business process based upon the queries and given management commitments.

All are modeled as BP Grid services.

The main responsibility of the business process managers is to enforce the management commitments at different levels of business process operations: instrumentation, evaluation and enactment. The architecture of enforcing management commitments is structured according to the Event-Condition-Action pattern [22][23]. According to MCS and the architecture of BP managers, business process related events are perceived by the BP Probes, business conditions are reasoned and inferred by the BP Evaluators, and management actions are enacted through

BP Enactors. As mentioned, the probes, evaluators and enactors are bound to Business Process Manager by the Configuration Manager through the instrumentation process given in previous section. BP Probes obtain raw events from managed business processes and convert them into BPGridService-compliant events. On the other hand, BP enactors collect raw data and statistics from the business context and collect necessary metrics from the system using some business process analyzer; and meaningful business conditions are derived by the BP Evaluators for the business process manager. The BP Enactors receive management directives (actions) from the business process managers, convert them to BPGridService-compliant calls, and make an invocation accordingly.

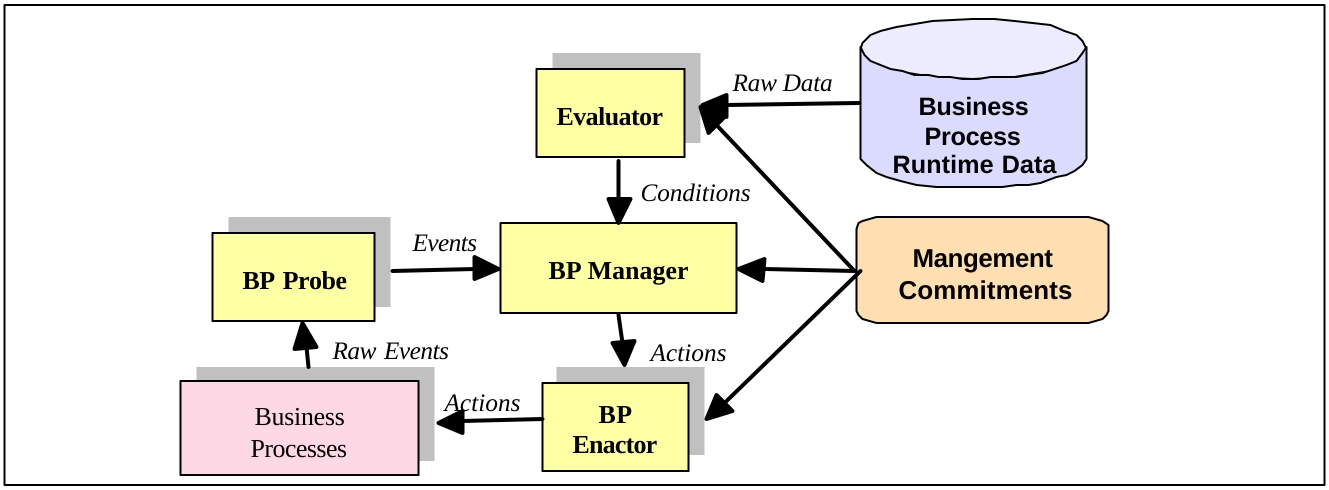

Figure 6: The BP Manager and other Components.

Many management events can be generated from the environment called raw events. Raw events are perceived by BP Probes and sent to interested business process managers to process them. Before raw events reach BP managers, some transformation often needs to be performed since an event at certain level of abstraction may mean nothing at different level. For example, an event of router failure implies high impact on network performance and demands immediate attention from network administrator. But, this event could mean nothing to the domain of supply chain management that is concerned with the quality of services at the level of business process. However, low network performance very likely implicates degraded quality of services for the execution of business process.

To make the BP manager who is in charge of SCM business process be aware of the potential impact, a mechanism of semantic event mapping needs to be in order. We create an architectural pattern called *semantic event transformation*, for this very purpose. We define an event mapping entity called Event Transformer (ET) at the semantic level. Each ET receives (raw) events from the probes or from other ETs, processes the events on the basis of embedded data, the management commitments, and the transformation rules, and sends the output to either subscribed BP managers or other ETs which may be at different semantic levels. Events are usually propagated from lower-level ETs , e.g., network domain, to higher-level ETs, e.g. SCM or PLM business process domains. Event transformation rules are derived from the given management commitments specified in MCS and are dynamically imported into ETs by the Configuration Manager.

Event transformation is particularly critical for the activities at the reactive management layer where substantial part of the processing is triggered by events. Event transformation resolves the problem of bridging the gap that exists between “events,” which are reported by various channels, and the “reactive conditions,” which are the cases to which the system should react. These conditions are composition of events or other conditions (e.g. “when at least four events of the same type occurred”) or content filtering on events (e.g. “only events that relates to inventory”) or both (“when at least four alarms generated for the router failures have been processes on the platform where customer care business process is running in a week”). Hence, a condition can be regarded as a metamorphosed form of an event or a group of events, which embodies the semantic information that is meaningful to specific level of abstraction.

The event transformation used in ETs follows ECA pattern as described earlier and takes two-phased approach of filtering: composition filtering (event) and content filtering (condition part on the result of the first phase). In our solution, phase 1 combines the composition filtering with content filtering capabilities. This approach enables to construct more efficient reactive management relative to those which are being developed by current tools. The two-phase approach may be inefficient when the number of detected sonditions is much smaller relative to the number of the combinations that are produced in phase 1. Furthermore, the number of combinations produced in phase 1 can be exponential. The ability to combine composition and content filtering is a unique property, and it improves the performance in the general case, and enables the detection of conditions that are not practically feasible in other solutions, in extreme cases. Figure 8 presents the architectural pattern for event transformation in BPSM, where managed resources act as event sources, and m-agents act as event sinks. Event transformers (ETs) form an ET semantic net that contains nodes (ETs) and directional edges (event flows). ETs generate conditions based on rules, policies and commitments that are configured by the Configuration Manager at both build time and run-time. Business conditions can be either consumed by BP managers or transmitted to another ET for further processing.

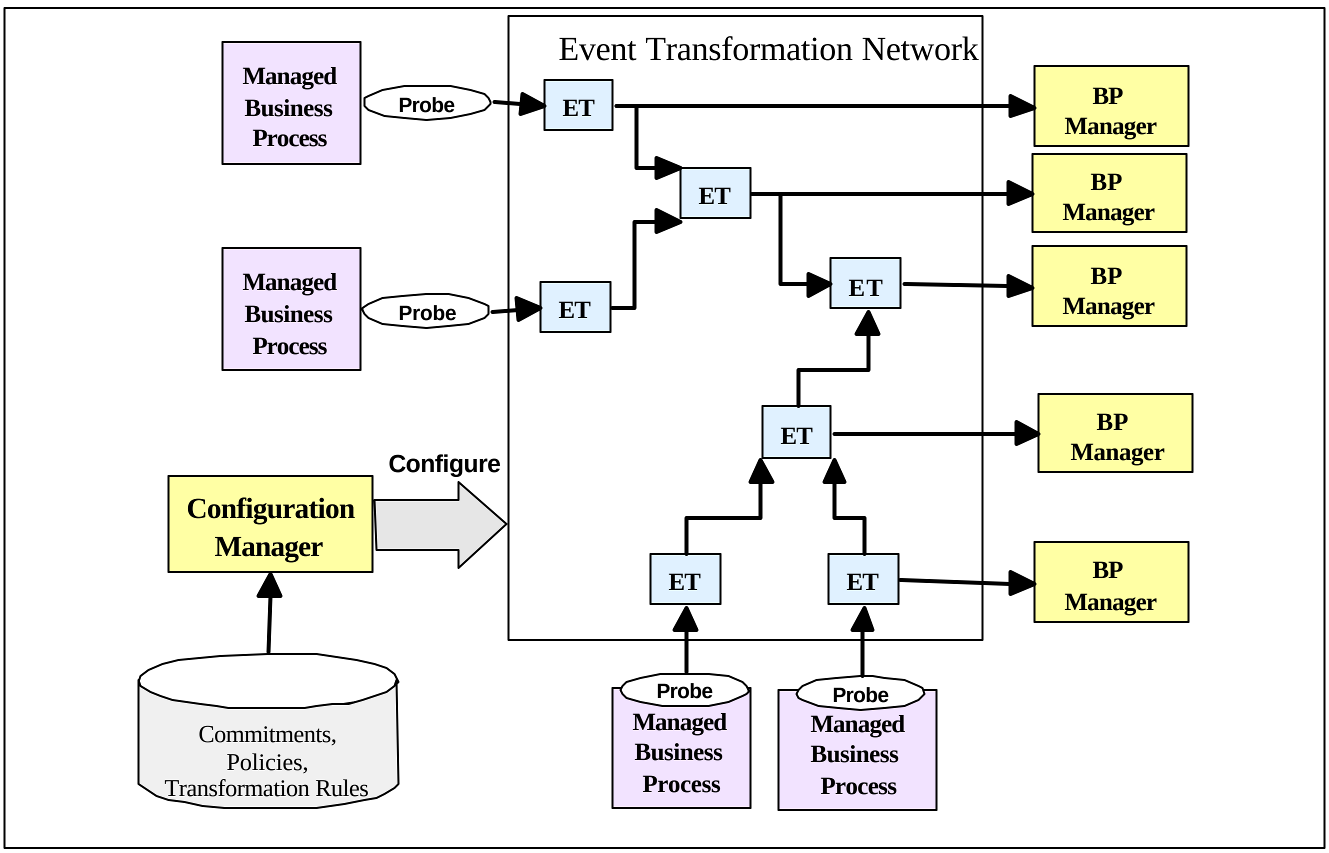

Figure 4: Event Transformation Network.

8 Related Work

Minsky and Ungureanu [25] described a mechanism call law-governed interaction (LGI), which is designed to satisfy three principles: (1) coordination policy needs to be coordinated; (2) the enforcement needs to be decentralized; and (3) coordination policies need to be formulated. BPSM satisfies all of the LGI principles. LGI uses decentralized controllers co-located with agents. LGI does not address the issues such as Grid computing, event transformation and meta-management.

BPM^2 uses ECA to enforce the compliance of management commitments. ECA rules are also used in the area of active database that is strong at defining high-level business events but weak at lower-level IT events. In fact, the event composition operators used in active databases do not allow the correlation and transformation of events whose forms can only be determined at run time. With active database language, dynamic correlation can only be guessed by the designer at the build time. On the other hand, BPSM allows events can be perceived and transformed to desired forms according to policies specified in the management commitments. The performance of event transformation can be modulated through ET semantic net. In general, active databases do not address the issues of performance and scalability [26][27].

There are other efforts on system management domain, notably, Tivoli Management Environment (TME), and Microsoft Web-Based Management (WBEM) schema and protocols. Unlike BPSM, these are not targeted on business process solution management but on system and network management toolkits. Over last years, Sun Microsystems has attempted to make Java an enabling technology for distributed systems management. The main products include JMX, Java Dynamic Management Kit,cand J2EE Management Specification [28][29]. Compared with BPSM, JMX and JDMK are more complementary than different. While JMX addresses the instrumentation of managed resources using MBean (Managed Bean), BPSM is more concerned regarding how various m-agents can be coordinated and how managers can leverage the architecture to fulfill management requirements. Propagating and transforming events are not addressed in JMX.

Helal et al. [30] proposed a framework called Internet Enterprise with three major components (1) an e-service framework; (2) a hierarchical brokering community based on UDDI, and (3) a dynamic workflow engine that uses e-services as entities, to create and enact workflow models. BPM^2 is similar to their work in the sense of modeling reusable entities as Web services (called e-services), nevertheless, it is the end of similarity. BPM^2 is aimed for providing a Grid-based business process-focused meta management framework and their work is more geared for creating workflows based on the composition of e-services.

9 Conclusion

In this paper, we have presented our vision and framework of BPM^2, which is a Grid-based Business Process Meta Management. Business processes and business systems are modeled as BP Grid services. The components of have BPM^2 been described, and the management paradigm anchored by management commitments has also been presented.

The primary focus of our future work in this area will be on the development of more sophisticated BP Broker, BP Manager, BP Explorer and BP Analyzer within our architecture, and on extension of our commitment-governed management to encompass other business process other than SCM and PLM. We are also interested in the question of how local commitment information can encoded so as to facilitate automatic negotiation of global management commitments by business processes, business systems, users, and enterprises.

REFERENCES

[1] Arkin, A., Business Process Modeling Language (BPML), Working Draft 0.4, http://BPMI.org, March 8th, 2002.

[2] Foster, I., and Kesselman C., eds., *The Grid: Blueprint for a New Computing Infrastructure*, Morgan Kaufmann, San Francisco, 1999.

[3] Haeckel, S.H., and Slywotzky, A.J. *Adaptive Enterprise: Creating and Leading Sense-and-Respond Organizations*, Harvard Business School Publisher, August, 1999.

[4] "SNMP Version 3 (snmpv3)," The Internet Engineering Task Force (IETF), http://www.ietf.org/html.charters/snmpv3-charter.html.

[5] Bumpus W. et al. Common *Information Model: Implementing the Object Model for Enterprise Management*, John Wiley & Sons, Dec. 1999.

[6] Management and Manageability: Manageability (M12) Model Core Specification, Version 1.3, Feb. 2002, IBM Internal Report.

[7] Anupindi, R. Chopra, S. Deshmukh, S.D. Mieghem, J.A.V. and Zemel, E. *Managing Business Process Flows*, Prentice-Hall, 1999.

[8] Autonomic Computing: IBM's Perspective on the State of Information Technology, IBM Research External Web Site, http://www.research.ibm.com/autonomic/.

[9] Buchmann, F., Meunier, R., Rohnerr, H., Sommerlad, P., and Stal, M., "A System of Patterns: Pattern-Oriented Software Architecture," New York: Wiley, 1996.

[10] Sloman, A., Varieties of Affect and the CogAff Architecture Schema, in *Proceedings of AISB'01 Symposium on Adaptive Agents and Multi-agent Systems*, University of York, United Kingdom, March 21-24, 2001.

[11] Lin, G. et al., The New Frontier: Sense and Respond System for Global Value Chain Optimization, To be published in *OR/MS Today*, May 2002.

[12] Foster, I., Kesselman, C., Nick, J.M., and Tuecke, S. , The Physiology of the Grid, http://www.globus.org/research/ogsa.pdf.

[13] Foster, I., Kesselman, C., Nick, J.M., Graham, S., Czajkowski, K., and Tuecke, S., Grid Service Specification, http://www.globus.org/research/papers/gsspec.pdf..

[14] Li, H., Jeng, J., Chang, H., Management Commitment Specification for Business Process Solution Management, IBM Internal Report, May, 2002; also submitted to *IBM System Journal*.

[15] Supply-Chain Operations Reference Model, Supply Chain Council, http://www.supply-chain.org, Version 3.1, March, 2000.

[16] Lutfiyya, H.L., Bauer, M.A., Marshall, A.D., Stokes, D.K., Fault Management in Distributed Systems: A Policy-Driven Approach, *Journal of Network and Systems Management*, Vol.8, No.4, 2000, Pages 499-525.

[17] Wies, R., Policies in Network and Systems Management – Formal Definition and architecture, *Journal of Network and Systems Managements*, Vol. 2, No. 1, pp. 63-83, 1994.

[18] Sloman, M. and Twidle, K., Domains: A framework for structuring management policy, In Morris Sloman, ed., *Network and Distributed Systems Management*, pp. 433-453, Addison-Wesley, 1994.

[19] Foster, I, Kesselman, C., and Tuecke, S., "The Anatomy of the Grid: Enabling Scalable Virtual Organizations", *International High-Performance Computing Applications*, vol. 15, no. 3, pp. 200-222; http://www.globus.org/research/papers/ogsa.pdf.

[20] Cannataro, M., Talia, D., and Trunfio, P., "Knowledge Grid: High Performance Knowledge Services on the Grid", *Proceedings of Grid 2001 – 2nd International Workshop on Grid Computing*, pp. 38-50, Denver, CO, Nov., 2001.

[21] Czajkowski, K., et al., "A Resource Management Architecture for Metacomputing Systems", In *4th Workshop on Job Scheduling Strategies for Parallel Processing*, Spring-Verlag, 1998, pp. 62-82.

[22] Tsalgatidou, A., Karakostas, V., Loucopoulos, P., Rule-Based Requirements Specification and Validation, in: Steinholtz, A. Solvberg, L. Bergman (Eds), *Proceedings of the Second Nordic Conference on Advanced*

Information System Engineering, Berlin et al.: Springer 1990. pp. 251-263.

[23] Hanson, E.N., Widom, J., An Overview of Production Rules in Database Systems, in: *Knowledge Engineering Review*, Vol. 8, No. 2, 1993, pp.121-143.

[24] J2EE Management Specification 1.0, JSR-077, Final Draft, http://java.sun.com/j2ee/tools/management/

[25] Minsky, N.H., and Ungureanu V., "Law-Governed Interaction: A Coordination and Control Mechanism for Heterogenous Distributed Systems," ACM Transaction on Software Engineering and Methodology, Vol. 9, No. 3, July, 2000, Pages 273-305.

[26] Casati, F., Ceri, S., Pernici, B., Pozzi, G. Specification of the Rule Language and Active Engine of Foro, Vol. 1, *WIDE Technical Report 3008-6*, 1997.

[27] Krishnamurthy, B., and Rosenblum, D., Yeast: A General Purpose Event-Action System, *IEEE Transaction on Software* Engineering, Vol. 21, No. 10, pp. 845-857, October, 1995.

[28] Sun Microsystems Inc. Java Management Extensions Instrumentation and Agent Specification, July, 2000.

[29] Sun Microsystems Inc. Java Management Tool Kit, White Paper, April, 2000.

[30] Helal, S. et al, The Internet Enterprise, In *Proceedings of the 2002 Symposium on Application and the Internet (SAINT2002)*.