BUSINESS PROCESS POLICY OBJECT

Cross Reference to Related Applications

This application claims priority to U.S. provisional application entitled "System and Method for Treating Business Practice Policies as Common Objects," Serial No. 60/303,424, filed July 6, 2001.

Technical Field

The methods, systems, and computer readable media described herein relate generally to object oriented programming and more particularly to employing objects in business process policy computing.

Background

Conventionally, logic implementing business process policies has been implemented in stand-alone computer applications that have had difficulties interfacing with different computer components and technologies in a business environment. The difficulties arise, at least in part, because the conventional applications are designed to run on one computer component and/or with one technology.

Summary

The following presents a simplified summary of methods, systems, and computer readable media associated with business process policy objects. This summary is not an extensive overview and is not intended to identify key or critical elements of the methods, systems, and/or media or to delineate the scope of the methods, systems, and media. It conceptually identifies the methods, systems, and media in a simplified form as a prelude to the more detailed description that is presented later.

This application concerns methods, systems and computer readable media for employing and accessing objects that model and implement business process policy logic. Employing objects in business process policy computing facilitates managing and optimizing business processes and performance. Furthermore, business process policy objects simplify mitigating problems associated with integrating business process policy logic across

computer components, technologies, and boundaries. Thus, business process policy object users can focus on using technology to increase business value by executing business strategy rather than focusing on technology integration issues. The methods, systems, and computer readable media described herein relate to a dynamic, data driven environment in which business process policies are treated as common objects in an object based computing environment. Employing business process policy objects facilitates reusability, portability, simplified messaging, and simplified integration with heterogeneous technologies and applications.

Thus, one aspect of this application concerns a method for implementing business process policy logic in an object. The method includes providing data items associated with modeling a business logic, providing methods associated with modeling the business logic, and providing an interface to the business logic that facilitates accessing the data items and/or methods through object oriented messaging. The method also includes instantiating an instance of an object that stores the data items, stores the methods, and implements the interface, receiving a message into the instance through the interface, and selectively performing business logic in the instance of the object based, at least in part, on the received message, where the business logic is performed by at least one of, invoking one or more of the methods and accessing one or more of the data items.

Another aspect of the application concerns a computer implemented method for automatically applying business process policy logic implemented in a business process policy object. The method includes identifying a business process policy object to which a message will be sent, and sending a first message to the business practice policy object.

Yet another aspect of the application concerns a system for automatically performing business process policies. The system includes a business process policy object that models and implements a business logic associated with a business practice, and a business manager that performs business management by selectively interacting with the business process policy object.

Certain illustrative aspects of the methods, systems, and computer readable media are described herein in connection with the following description and the annexed drawings. These aspects are indicative, however, of but a few of the various ways in which the principles of the methods, systems, and media may be employed and thus the examples are intended to include such aspects and equivalents. Other advantages and novel features may

become apparent from the following detailed description when considered in conjunction with the drawings.

Brief Description of the Drawings

Figure 1 illustrates an example computing environment on which an example business process policy object can reside.

Figure 2 illustrates an example business process policy object facilitating spanning a variety of heterogeneous computing environments.

Figure 3 illustrates example business process policy objects employed in facilitating contextual visualization in an environment that includes event management and business management.

Figure 4 illustrates a collection of example business process policy objects supporting event processing for an enterprise distributed across a global computer network.

Figure 5 illustrates example business process policy objects interacting with heterogeneous applications through heterogeneous bindings and a common environment.

Figure 6 illustrates an example data and control flow through a system employing a business process policy object in conjunction with an event processor to selectively perform business logic for an enterprise that includes computer components that generate business events.

Figure 7 is a flow chart of an example method for implementing business process policy logic in an object.

Figure 8 is a flow chart of a portion of an example method for modeling business process policy logic.

Figure 9 illustrates an example of an object.

Figure 10 illustrates an example of a business process policy object.

Figure 11 illustrates a collection of example business process policy objects interacting with a variety of applications and performing a variety of functions.

Figure 12 illustrates an example business process policy object receiving messages and/or events from a variety of environments and facilitating various data visualizations.

Figure 13 is a flow chart of an example method for automatically applying business process policy logic implemented in a business process policy object.

Detailed Description

Example methods, systems, and computer media are now described with reference to the drawings, where like reference numerals are used to refer to like elements throughout. In the following description, for purposes of explanation, numerous specific details are set forth in order to facilitate thoroughly understanding the methods, systems, and computer readable media. It may be evident, however, that the methods, systems, and computer readable media can be practiced without these specific details. In other instances, well-known structures and devices are shown in block diagram form in order to simplify description.

One example method described herein facilitates implementing business process policy logic in objects by providing modeling, instantiation, messaging, and selective performance of business logic based on the messaging. Once a business process policy has been modeled and implemented in an object, then the object can be exposed, via its interface, to business applications to facilitate processes like enterprise management, workflow management, and so on. In one example, the objects can be exposed via a common environment. Thus, the objects can be accessed by heterogeneous computer components by interface protocols including, but not limited to, ODBC/XML, JAVA/EJB, C/C++, COM, and C#/.NET.

Figure 1 illustrates a computer 100 that includes a processor 102, a memory 104, a disk 106, input/output ports 110, and a network interface 112 operably connected by a bus 108. Executable components of the systems described herein may be located on a computer like computer 100. Similarly, computer executable methods described herein may be performed on a computer like computer 100. Furthermore, a business process policy object may reside on a computer like computer 100. It is to be appreciated that other computers may also be employed with the systems and methods described herein.

The processor 102 can be a variety of various processors including dual microprocessor and other multi-processor architectures. The memory 104 can include volatile memory and/or non-volatile memory. The non-volatile memory can include, but is not limited to, read only memory (ROM), programmable read only memory (PROM), electrically programmable read only memory (EPROM), electrically erasable programmable read only memory (EEPROM), and the like. Volatile memory can include, for example, random access memory (RAM), synchronous RAM (SRAM), dynamic RAM (DRAM), synchronous DRAM (SDRAM), double data rate SDRAM (DDR SDRAM), and direct RAM bus RAM (DRRAM). The disk 106 can include, but is not limited to, devices like a magnetic

disk drive, a floppy disk drive, a tape drive, a Zip drive, a flash memory card, and/or a memory stick. Furthermore, the disk 106 can include optical drives like a compact disk ROM (CD-ROM), a CD recordable drive (CD-R drive), a CD rewriteable drive (CD-RW drive) and/or a digital versatile ROM drive (DVD ROM). The memory 104 can store processes 114 and/or data 116, for example. The disk 106 and/or memory 104 can store an operating system that controls and allocates resources of the computer 100.

The bus 108 can be a single internal bus interconnect architecture and/or other bus architectures. The bus 108 can be of a variety of types including, but not limited to, a memory bus or memory controller, a peripheral bus or external bus, and/or a local bus. The local bus can be of varieties including, but not limited to, an industrial standard architecture (ISA) bus, a microchannel architecture (MSA) bus, an extended ISA (EISA) bus, a peripheral component interconnect (PCI) bus, a universal serial (USB) bus, and a small computer systems interface (SCSI) bus.

The computer 100 interacts with input/output devices 118 via input/output ports 110. The input/output devices 118 can include, but are not limited to, a keyboard, a microphone, a pointing and selection device, cameras, video cards, displays, and the like. The input/output ports 110 can include but are not limited to, serial ports, parallel ports, and USB ports.

The computer 100 can operate in a network environment and thus is connected to a network 120 by a network interface 112. Through the network 120, the computer 100 may be logically connected to a remote computer 122. The network 120 includes, but is not limited to, local area networks (LAN), wide area networks (WAN), and other networks. The network interface 112 can connect to local area network technologies including, but not limited to, fiber distributed data interface (FDDI), copper distributed data interface (CDDI), ethernet/IEEE 802.3, token ring/IEEE 802.5, and the like. Similarly, the network interface 112 can connect to wide area network technologies including, but not limited to, point to point links, and circuit switching networks like integrated services digital networks (ISDN), packet switching networks, and digital subscriber lines (DSL).

Turning now to Figure 2, a business process policy object 200 is illustrated facilitating spanning a variety of heterogeneous computing platforms (e.g., 210, 220, 230, 240). Conventionally, business process policy logic was coded in stand-alone applications, not objects. Thus, the benefits of object oriented analysis, design, and programming were not available to the applications. For example, benefits like reusability, portability, data hiding, encapsulation, interface based messaging, inheritance, and polymorphism were not available.

Therefore, integrating these standalone applications and/or spanning multiple heterogeneous computing platforms was difficult. Taking advantage of the benefits of object oriented analysis, design, and programming facilitates embedding intelligence in objects that mitigate problems associated with spanning various computing platforms (e.g., hardware, software, operating system). Similarly, business process policy objects facilitate interacting with diverse business entities (e.g., information management technology, enterprise management technology, rules based intelligent technology, predictive technologies). Thus, in one example, business process policy objects simplify interactions between applications like workflow, financial, supply chain, and human resources. Rather than interacting with unique stand-alone applications, each of which may require unique codings, bindings, messaging, and so on, business process policy objects facilitate a standard object oriented messaging interface.

In Figure 2, the example business process policy object 200 is illustrated facilitating interactions between computing platforms 210, 220, 230, and 240. Each of these computing platforms may include a variety of hardware, software, and operating systems in unique heterogeneous combinations. Software includes but is not limited to, one or more computer readable and/or executable instructions that cause a computer or other electronic device to perform functions, actions and/or behave in a desired manner. The instructions may be embodied in various forms such as routines, algorithms, modules, methods, threads, and/or programs. Software may also be implemented in a variety of executable and/or loadable forms including, but not limited to, a stand-alone program, a function call (local and/or remote), a servelet, an applet, instructions stored in a memory, part of an operating system or browser, and the like. It is to be appreciated that the computer readable and/or executable instructions can be located in one computer component and/or distributed between two or more communicating, co-operating, and/or parallel processing computer components and thus can be loaded and/or executed in serial, parallel, massively parallel, and other manners.

By implementing the object oriented messaging interface, the business process policy object 200 simplifies interactions between the computing platforms since a computing platform need only implement the interface to the business process policy object 200 and need not implement an interface to each of the computing platforms with which it desires to communicate. This mitigates integration and interoperability difficulties associated with conventional systems.

Referring now to Figure 3, an example business process policy object 300 employed in facilitating contextual visualization of data in an environment that includes event management and business management is illustrated. One or more business process policy objects 300 are illustrated interacting with a graphical user interface 310. The graphical user interface 310 facilitates contextual visualization of data received from, for example, an event manager 320 and/or a business manager 330. The event manager 320 and/or business manager 330 can receive data from a variety of sources including, but not limited to, business applications 340, computer components interfacing through the Internet 342, data processing applications 344, and other sources 346. A computer component refers to a computer-related entity, either hardware, firmware, software, a combination thereof, or software in execution. For example, a computer component can be, but is not limited to being, a process running on a processor, a processor, an object, an executable, a thread of execution, a program and a computer. One or more computer components can reside within a process and/or thread of execution and a computer component can be localized on one computer and/or distributed between two or more computers.

Conventionally, a collection of sources, applications, and managers may have individually implemented interfaces to the graphical user interface 310. By employing the business process policy object(s) 300, the sources need only implement the interface to the object 300. While a single graphical user interface 310 is illustrated, it is to be appreciated that the object 300 may interact with one or more graphical user interfaces 310. Thus, by implementing the interface to the object 300, the sources receive the benefit of simplifying interaction with a variety of graphical user interfaces 310 that facilitate contextual visualization.

The object 300 can be a part of a system that automatically performs business process policies. The system can include, for example, one or more business process policy objects 300 that model and implement business logic associated with a business practice and a business manager 330 that performs the business management by selectively interacting with one or more business practice policy objects 300. Furthermore, the business process policy object 300 can be part of a system that includes an event manager 320 that provides to and/or receives messages from the business process policy object 300.

One source (e.g., other 346) from which a business process policy object 300 can receive events, messages, and the like is a software robot (BOT). A business process policy object 300 can receive information from a BOT and then selectively perform business logic

that answers questions like, identify when an event occurs (e.g., bankruptcy, merger, disclosure, acquisition), identify why an event has occurred (e.g., XYZ revenues decline), identify how things are related (e.g., XYZ and ABC revenues are directly related because XYZ acquired ABC six months ago). A BOT is an entity that works like a mini web browser that can access and/or generate information. A BOT can, for example, crawl the Internet and produce an index of uniform resource locators (URL), correlate data (e.g., find prices on similar items), perform pattern matching, and so on. By way of illustration, a business process policy object 300 can model and implement business logic for optimizing on-line spot commodity purchases. Thus, one or more BOTs may be released into the Internet to substantially continuously locate, analyze, and report on spot commodity prices. A business process policy object 300 can receive the spot commodity data from the BOTs and perform the appropriate business logic based on the BOT data. Business process policy objects 300 provide advantages for implementing such logic and performing such processing by implementing object oriented messaging, facilitating cross platform implementation and integration, and simplifying re-use and extensibility of logic, for example.

Business process policy objects 300 can receive information from BOTs of various levels of sophistication. For example, a business process policy object 300 can receive information from a spider BOT that crawls the Internet, indexes the Internet, and provides pre-discovered, pre-indexed data that can be data mined. Similarly, a business process policy object 300 can receive information from an intelligent BOT that performs higher level functions like correlation, pattern recognition, and fuzzy logic processing. In one example, a business process policy object 300 can receive information from a BOT that generates business events for input to a business process policy object 300. BOTs may produce events including, but not limited to, reference events, change events, threshold events, task completion events, and task failure events. A reference event, which may be a discrete event, can provide information like a date when a company files a financial disclosure, or a notification that a company has filed its financial disclosure. A change event can be employed to relate prior intelligence that has not yet been related to other events. For example, a change event may provide information concerning when a product price page changes, or when a company stock price changes. A threshold event facilitates simple levels of correlation between current knowledge and prior knowledge. For example, a threshold event may provide information concerning when a company's stock price has gone up or down 10% over the previous price. A task completion event relates to business process

intelligence and thus may provide information concerning when an on-going task has completed (e.g., informing a business process policy object that a financial disclosure data download has completed).

Events can be programmatically related and aggregated in a business process policy object 300 to facilitate comprehensive command and control by applying business intelligence. Furthermore, business process policy objects 300 can subscribe to BOTs for a flow of certain types of data and events for certain entities. Thus, business process policy objects 300 can be configured to listen for certain events in an enterprise management system.

A business process policy object 300 can store values associated with the business practice that it models. The values can be stored in data items. The data items can include, but are not limited to, variables, arrays, lists, and so on. Similarly, a business process policy object 300 can store computer executable instructions that implement decisions associated with modeling a business practice in one or more methods. To facilitate accessing the methods and/or data items, a business process policy object 300 can selectively expose methods through an interface.

Turning now to Figure 4, a collection 400 of example business process policy objects (e.g., 402, 404, 406, 408) is illustrated interacting with a variety of sources and destinations through a global computer network 420 and an event processor 410. In one example, the sources and destinations may be arranged into an enterprise, and the event processor 410 may be an enterprise management system. In Figure 4, the collection 400 of business process policy objects is illustrated communicating with entities including a local business computer 430, a customer computer 440, a supplier computer 450, a news server 460, an email server 470, a voice mail server 480, and a remote business computer 490. It is to be appreciated that business process policy objects can communicate with other entities. The communication occurs through a global computer network 420 (e.g., Internet). The collection 400 of business process policy objects may perform services like command and control services, event management services, workflow management services, and visual management services. For example, the collection 400 of business process policy objects may perform workflow management services like facilitating defining a workflow amongst the sources and destinations, checking the status of work in progress, starting workflows, and/or stopping workflows.

Figure 5 illustrates example business process policy objects (e.g., 510, 520, 530) interacting with a collection of heterogeneous applications through heterogeneous bindings and a common environment 500. For example, a first business process policy object 510 is interacting with a business rules expert 512 (e.g., CleverPath Aion Business Rules Expert provided by Computer Associates International, Inc.) and the common environment 500. Similarly, a second business process policy object 520 is interacting with an intelligent technology 522 (e.g., Neugents neural network technologies provided by Computer Associates International, Inc.) and the common environment 500 while a third business process policy object 530 is interacting with other applications 532 and the common environment 500. In one example, the common environment 500 interacts with a variety of applications through a variety of bindings. For example, the common environment 500 may interact with an application 540 through an ODBC/XML binding 542. Similarly, the common environment 500 can interact with a JAVA/EJB application 550 through JAVA/EJB binding 552. Likewise, the common environment 500 can interact with a C/C++ application 560 via a C/C++ binding 562, a COM application 570 via a COM binding 572, and a C#/.NET application 580 via a C#/.NET 582 protocol. Thus, by modeling and implementing business logic in business process policy objects, a variety of heterogeneous applications can interface through the common environment 500 to a variety of applications through a variety of bindings.

Figure 6 illustrates an example data and control flow through a system employing a business process policy object 600 in conjunction with an event processor 610 to selectively perform business logic for computer components in an enterprise 630 that generates business events 620. The business process policy object 600 is illustrated establishing a secondary, two-way communication 640 with the enterprise 630 to facilitate gathering additional information associated with a business event 620 and/or processing performed by an event processor 610.

In view of the exemplary systems shown and described herein, methodologies that are implemented will be better appreciated with reference to the flow diagrams of Figures 7, 8 and 13. While for purposes of simplicity of explanation, the illustrated methodologies are shown and described as a series of blocks, it is to be appreciated that the methodologies are not limited by the order of the blocks, as some blocks can occur in different orders and/or concurrently with other blocks from that shown and described. Moreover, less than all the

illustrated blocks may be required to implement an example methodology. Furthermore, additional and/or alternative methodologies can employ additional, not illustrated blocks.

In the flow diagrams, rectangular blocks denote "processing blocks" that may be implemented, for example, in software. Similarly, the diamond shaped blocks denote "decision blocks" or "flow control blocks" that may also be implemented, for example, in software. Alternatively, and/or additionally, the processing and decision blocks can be implemented in functionally equivalent circuits like a digital signal processor (DSP), an application specific integrated circuit (ASIC), and the like.

A flow diagram does not depict syntax for any particular programming language, methodology, or style (e.g., procedural, object-oriented). Rather, a flow diagram illustrates functional information one skilled in the art may employ to program software, design circuits, and so on. It is to be appreciated that in some examples, program elements like temporary variables, routine loops, and so on are not shown.

In one example, methodologies can be implemented as computer executable instructions and/or operations and the instructions and/or operations can be stored on computer readable media including, but not limited to, an application specific integrated circuit (ASIC), a compact disc (CD), a digital versatile disk (DVD), a random access memory (RAM), a read only memory (ROM), a programmable read only memory (PROM), an electronically erasable programmable read only memory (EEPROM), a disk, a carrier wave, and a memory stick.

Referring now to Figure 7, a method 700 for implementing business process policy logic in an object is illustrated. At 710, business logic is modeled. Modeling can include analyzing a business practice to determine its logical and/or physical inputs, processes, and outputs. Inputs and outputs may be stored in fields in an object, while the processes may be implemented in computer executable instructions that are organized into methods. Since the method 700 concerns business process policy logic as captured in an object, modeling the business logic can also include defining an interface to the data and/or methods. An interface facilitates interacting with the object through object oriented messaging, rather than through conventional procedure calls. Thus, modeling a business logic includes providing data items associated with a business process policy logic, providing methods associated with the business process policy logic, and providing an interface to the business logic that facilitates accessing the data items and/or methods through object oriented messaging.

A business process policy object models and implements intelligence (e.g., decision making) applied in understanding, analyzing, and/or responding to business events. A business event can be, for example, a message between an information source and an intended target (e.g., workflow manager, enterprise monitor) that describes a significant business activity. Business events can also be modeled in objects, and can be communicated to business process policy objects via, for example, object oriented messaging. Business events can be associated with business logic processing and can be involved, for example, in reporting a state, in noting a status change, in providing data, and so on.

By way of illustration, a business process policy object can facilitate automatically performing actions like responding to an inventory change, predicting sales activity, generating appropriate discounts, and scheduling deliveries. These and similar actions can be triggered by business events and/or messages, for example. The types of business events for which a business process policy object can implement logic include, but are not limited to, reference events, change events, threshold events, task completion events, and task failure events. Furthermore, a business process policy object can model and implement logic that deals with individual events and/or aggregations of events.

At 720, an instance of an object is instantiated. Thus, memory for the data items, instructions for the methods, and resources for the interface can be allocated and addressed. Once the object is instantiated, the business logic that it models is available to be employed in an object oriented environment. Thus, at 730, a message can be received through an interface. The message will be received by the instance created at 720. The message can be, for example, a business event. An example schema of a business event facilitates identifying the description of and details concerning the business event being reported. The event may be related to object happenings, business processes, and/or intelligence (e.g., facts) gathered, for example. One example XML schema, for a business event processed by a business process policy object is:

```
< COMPANY A>
    <EVENT NAME> . . . </EVENT NAME>
    <NAME SPACE> . . . </NAME SPACE>
    <OBJECT NAME>
        <CLASS NAME> . . . </CLASS NAME>
            <OBJECT ID>
                <NAME_VALUE>
```

```
                    <NAME> . . . </NAME>
                    <TYPE> . . . </TYPE>
                    <VALUE> . . . </VALUE>
                </NAME_VALUE>
                <NAME_VALUE>
                    <NAME> . . . </NAME>
                    <TYPE> . . . </TYPE>
                    <VALUE> . . . </VALUE>
                </NAME_VALUE>
            </OBJECT ID>
        </OBJECT NAME>
        <MSGTEXT> . . . </MSGTEXT>
            <PROPERTY>
                <NAME> . . . </NAME>
                <TYPE> . . . </TYPE>
                <OLD VALUE> . . . </OLD VALUE>
                <NEW VALUE> . . . </NEW VALUE>
            </PROPERTY>
    </COMPANY A>
```

Based, at least in part on the message of 730, business logic can be selectively performed at 740. The business logic processing can include, but is not limited to, invoking one or more of the methods, accessing one or more of the data items, and/or updating one or more of the data items. Therefore, in method 700, various business logic choices (e.g., 750, 752, . . . 754) are illustrated. These blocks of business logic indicate that multiple, substantially simultaneous paths may be taken based, at least in part, on the message received at 730. The business logic paths can individually and/or collectively access the data items and/or methods. The business logic that is modeled and implemented in a business process policy object can be expressed, for example, in IF/THEN rules, rule capturing tables, and neural networks. By way of illustration, simple inventory logic may be modeled and implemented by the following IF/THEN rule.

```
IF inventory_item < X THEN
    order more inventory_item
END IF
```

Similarly, simple discount logic may be modeled and implemented by the following IF/THEN rule.

```
IF order value > relevant limit THEN
        generate 10% discount
ELSE
        generate standard discount
END IF
```

Simple IF/THEN rules can be aggregated to construct sophisticated logic that models and implements logic that handles situations like, a new account being opened, a balance on an account exceeding a threshold, a new purchase order arriving, and so on.

While the business logic in a business process policy object can respond to actual business events, business process policy objects can also interact with predictive alerts generated, for example, by neural agents that predict the occurrence of events (e.g., manufacturing problems). For example, a business process policy object may identify sales orders and deliveries that could be compromised by a manufacturing problem and produce data that facilitates visualizing the problems and thus responding to such problems.

At 760, a determination is made concerning whether there is another message received for the object. If the determination at 760 is yes, then processing returns to 730, otherwise, processing can conclude.

Turning now to Figure 8, block 710 is illustrated in greater detail. At 712, data items that facilitate modeling business logic are provided. Data items can include, but are not limited to, variables, arrays, tables, lists, stacks, queues, records, and so on. It is to be appreciated that a variety of data items can be employed in modeling business logic. At 714, methods for capturing the processes of a business logic are provided. Methods can be implemented in computer executable instructions. It is to be appreciated that methods can employ one or more processes, threads, and the like and that the methods can access and/or update one or more of the data items. At 716, an interface that facilitates object oriented based messaging access to the data items of 712 and the methods of 714 is provided. While one interface is described in connection with 716, it is to be appreciated that a business process policy object may provide and expose more than one interface. Exposing multiple interfaces facilitates making the logic modeling provided by the business process policy object available in a variety of contexts.

Turning now to Figure 9, an example business process policy object 900 is illustrated. The object 900 can receive a message 910 and/or an event 912 through its interface 920. The interface 920 facilitates gaining regulated access to methods 940, which in turn provides programmer/designer controlled access to data 930, IF/THEN modeling rules 932, rule encoding tables 934, and neural networks 936, for example. The data items 930 can include, but are not limited to, object class identifiers, object identifiers, message identifiers, a message schema, a message format, a textual description of an event, a current value for a discrete business data point, a range for a discrete business data point, a desired value for a discrete business data point, an entity identifier that identifies the entity from which the message 910 and/or event 912 was received, and a Uniform Resource Locator (URL) that identifies the entity from which the message 910 and/or event 912 was received.

In an object oriented environment, rather than applications directly accessing the data 930 and/or methods 940, the interface 920 accepts the message 910 and/or event 912, which facilitates features like data hiding and abstraction, for example. However, the object 900 is not so limited. Additionally, and/or alternatively, the object 900 may expose certain data and/or methods 950 to an object oriented infrastructure 960. This type of access facilitates, for example, internal communication between members of a class hierarchy (e.g., parent, child). The data and/or methods 950 that are accessed by the object oriented infrastructure 960 are typically not exposed to a general consumer of the object 900.

One example business process policy object 900 captures information like key performance indicators (KPI) and business data associated with key performance indicators. Having received the KPI data, the business process policy object 900 can then take actions like management by exception, event correlation, prediction, generating notifications, and so on. For example, a business process policy object 900 can be messaged concerning a business event and thus select an action based on the business process policy object 900 state and the data gathered concerning the business event. By way of illustration, a business process policy object 900 can receive data that XYZ has reported above expected revenues and may, therefore, take actions resulting in, for example, scheduling sales visits to XYZ based on the assumption that XYZ will have more money to spend. The business logic assumption was modeled and implemented in the business process policy object 900. Similarly, if XYZ reports lower than expected revenues, different business logic (e.g., adjusting credit terms) can be triggered. In some cases, the business process policy object 900 may need more data to analyze and respond to the event and/or messages. Thus, the

business process policy object 900 can set up a secondary, two way communication with an information source to facilitate obtaining additional information concerning the triggering message and/or event.

Turning now to Figure 10, an example object 1000 that includes business process policy logic 1010 and a business process policy object interface 1020 is illustrated. Objects (e.g., 1030, 1032, 1034, . . . 1036) interact with the object 1000 through exposed interface methods (e.g., 1022, 1024, . . . 1026). While three exposed methods are illustrated in Figure 10, it is to be appreciated that a greater and/or lesser number of methods may be exposed by the business process policy object interface 1020. Similarly, while four objects are illustrated in Figure 10, it is to be appreciated that a greater and/or lesser number of objects may interact with the business process policy object 1000. Exposed methods can include, but are not limited to methods that facilitate, obtaining a message in XML format, obtaining the class of a received object, obtaining a unique identifier for a received object, retrieving a textual description of a received event, retrieving the name of a property for which an event was generated; retrieving the value of a property for which an event was generated, retrieving a unique identifier that identifies the provider of an event, retrieving a unique identifier that identifies an entity to which the event is related, and retrieving a URL of an entity associated with an event.

Methods exposed by the business process policy object interface 1020 facilitate accessing the business process policy logic 1010 in a manner desired by the object 1000 designer and/or programmer. The logic 1010 (and thus one or more methods that implement the logic) can perform actions like, evaluating one or more IF/THEN rules, evaluating one or more rule encoding table results, and interpreting an output from one or more neural networks. Thus, the objects (e.g., 1030, . . . 1036) interact with the object 1000 through the exposed methods (e.g., 1022, . . . 1026), which selectively triggers the performance of business process policy logic 1010 as implemented in, for example, IF/THEN rules, rule encoding tables, and neural networks.

Figure 11 illustrates a collection 1100 of example business process policy objects (e.g., 1102, . . . 1104) interacting with a variety of applications 1130 and performing a variety of functions 1110. Thus, as illustrated in Figure 11, the collection 1100 of business process policy objects models and implements business logic associated with functions including, but not limited to, manufacturing processes 1132, finance processes 1134, human resources processing 1136, enterprise management functions 1140, enterprise monitoring functions

1142, enterprise resource planning functions 1146, supply chain management functions 1148, electronic commerce 1150, and key performance indicator processing 1152. The processes 1130 can interact with the collection 1100 through object oriented messaging and/or event passing, for example. The activities 1110 that the collection 1100 can perform for the processes 1130 include, but are not limited to, performing rules based management 1112, generating new events or notifications 1114, generating data suitable for use in visualization objects 1116, generating data suitable for use in application level definitions 1118, performing management by exception 1120, facilitating event correlation 1122, and facilitating event prediction 1124.

By way of illustration, a manufacturing process 1132 may generate an event that is forwarded to the collection 1100. One or more of the business process policy objects (e.g., 1102, . . . 1104) can then perform activities based on the event. For example, a business process policy object may apply rules based management 1112 to the events supplied by the manufacturing process 1132 to determine, for example, that current manufacturing levels are below expected levels. Thus, the business process policy object may perform an action to generate a new event 1114 that serves to notify other objects, processes, and the like, of the shortfall in manufacturing process 1132. Furthermore, to simplify understanding the problem in the manufacturing process 1132, the business process policy object can generate data that will facilitate viewing the problem graphically and pass the data to a visualization object 1116. Additionally, the business process policy object or the collection 1100 may correlate the manufacturing downfall with other events (e.g., blizzard in the midwest impacting suppliers who support just in time delivery). Thus, event correlation 1122 may occur which in turn leads to a prediction activity 1124 that predicts which retail outlets will suffer a shortfall based on the initial problem in manufacturing 1132. If the shortfalls fall outside of certain exceptional ranges, then management by exception 1120 may be triggered, which may, for example, re-distribute the output from the manufacturing process 1132.

Turning now to Figure 12, an example business process policy object 1200 is illustrated interacting with a number of visualization processes and a number of event generating and/or receiving technologies. One application of a business process policy object 1200 is to facilitate visualizing events, status, changes, process flows, workflows, correlations, and predicted situations for a business and/or enterprise. Data transmitted to a contextual visualization interface facilitates, for example, line of business visualization 1210, application level visualization 1212, and role based visualization 1214. By way of

illustration, a first business process policy object can provide events, data, and/or messages that interact with a tool employed by a chief financial officer (e.g., accounting, return on investment), while a second business process policy object provides events, data, and/or messages that interact with a tool employed by an inventory control manager (e.g., warehousing, production, delivery). In this way, various roles within an organization are supported by business process policy objects.

The business process policy object 1200 is illustrated interacting with a line of business visualization process 1210. A business can be viewed from different perspectives at different points in time. Thus, the business process policy object 1200 provides data that facilitates viewing the business along certain lines. For example, the manufacturing aspects of a business may be visualized at a first time, while the delivery aspects and the inventory aspects may be viewed at a different time.

The business process policy object 1200 also provides data that facilitates application level visualization 1212. By way of illustration, a first application (e.g., accounts payable), may receive data from the business process policy object 1200 and a second application (e.g., accounts receivable) may also receive information from the business process policy object 1200. Thus, the object facilitates visualizing various applications.

The business process policy object 1200 also provides data that facilitates role based visualization 1214. For example, a CEO may receive a first type, amount, and character of data from the business process policy object 1200 to facilitate performing the CEO role while a shop floor employee may receive a different type, quantity, and character of data from the business process policy object 1200 to facilitate performing the shop floor role. Thus, visual displays can be configured for the role of the viewer. While line of business visualization 1210, application level visualization 1212, and role based visualization 1214 are illustrated in Figure 12, it is to be appreciated that the business process policy object 1200 can provide data suitable for display on a graphical user interface that facilitates other types of contextual visualization including, but not limited to, enterprise management and workflow management.

The business process policy object 1200 can also interact with a variety of event generating and/or receiving technologies. For example, the business process policy object 1200 can interact with information management technology 1202 and thus receive messages from and/or send messages to the information management technology 1202. The business process policy object 1200 can also interact with an enterprise management technology 1204.

Thus, the business process policy object 1200 can receive messages from and/or send messages to the enterprise management technology 1204. One example enterprise management technology 1204 is Unicenter TNG provided by Computer Associates International, Inc. Similarly, the business process policy object 1200 can interact with a rules based intelligent technology 1206 and a predictive technology 1208 receiving messages from and sending messages to such technologies. Thus, the business process policy object 1200 facilitates interactions between the technologies and also facilitates a variety of visualizations associated with such technologies.

By way of illustration, a predictive technology 1208 may generate an event that is forwarded to the business process policy object 1200. This event may concern a determination that a disk crash is imminent. The business process policy object 1200 may therefore generate data for an application level visualization 1212 (e.g., computer network management) and for a role based visualization 1214 (e.g., systems administrator). The business process policy object 1200 may also generate an event informing an enterprise management technology 1204 that the crash prediction has been visually distributed to the application level visualization 1212 and the role based visualization 1214, which can facilitate reducing duplicate notifications.

While the example event described above was a predictive event (e.g., imminent disk crash), it is to be appreciated that the business process policy object 1200 can process other types of events including, but not limited to, reference events, change events, threshold events, and task completion events. A reference event, which may be a discrete event, can provide information like a date when a company files a financial disclosure, or a notification that a company has filed its financial disclosure. A change event can be an event that is related to prior intelligence but which has not yet been related to other events. For example, a change event may provide information concerning when a product price page changes, or when a company stock price changes. A threshold event facilitates simple levels of correlation between current knowledge and prior knowledge. For example, a threshold event may provide information concerning when a company's stock price has gone up or down 10% over the previous price. A task completion event relates to business process intelligence and thus may provide information concerning when an on-going task has completed (e.g., informing a business process policy object that a financial disclosure data download has completed). Furthermore, while the business process policy object 1200 generated a new event, it is to be appreciated that the business logic selectively performed by the business

process policy object 1200 can include, but is not limited to, generating a business event, generating an enterprise event, generating a notification, and generating a log entry.

Turning now to Figure 13, an example method 1300 for automatically applying business process policy logic implemented in a business process policy object is illustrated. At 1310, a business process policy object is identified. For example, an application, object, process, and/or thread that desires to communicate with a business process policy object could acquire a unique identifier for a business process policy object to which it will send a message. Additionally and/or alternatively, a type of business process policy object with which the thread, process, application, object, etc. desired to interact could be identified. By way of illustration, an object that handles task failure events (e.g., failed download) may be identified. The sender may not require that the event be sent to a specific object but rather to one or more instances of a class of objects that are tasked with handling the business logic associated with the event.

At 1320, a message is sent to the business process policy object of 1310. The message may concern items like a threshold event, a reference event, a change event, a task completion event, and a task failure event, for example. In response to the message of 1320, the receiving business process policy object may generate an event that is forwarded to other business process policy objects and/or returned to the sender of the initiating message. The receiver may generate, for example, an event, a message, and/or an enterprise event. Thus, at 1330, method 1300 may receive an event and/or message.

At 1340, a determination is made concerning whether the method 1300 will send another message. If the determination at 1340 is yes, then processing returns to 1320, otherwise processing can conclude. The determination at 1340 can be made, at least in part, on whether the message and/or event received at 1330 indicates that an additional message should be sent. For example, the business process policy object to which the message was sent at 1320 may desire to establish a secondary, two-way communication to facilitate obtaining additional information associated with processing the message. Thus, the method 1300 may cycle through the blocks 1320, 1330, and 1340 one or more times.

It is to be appreciated that no message may be received at 1330. Thus, time out logic may be employed in the method 1300 whereby the method 1300 will wait for a message and/or event for a predetermined, configurable period of time before determining whether to send another message to the business process policy object identified at 1310. It is to be further appreciated that although a business process policy object identification is illustrated

at 1310, that the method 1300 may send messages to one or more business process policy objects, serially and/or substantially in parallel.

The systems, methods, and objects described herein may be stored, for example, on a computer readable media. Media can include, but are not limited to, an application specific integrated circuit (ASIC), a compact disc (CD), a digital versatile disk (DVD), a random access memory (RAM), a read only memory (ROM), a programmable read only memory (PROM), a disk, a carrier wave, a memory stick, and the like. Thus, an example computer readable medium can store computer executable instructions for automatically performing business process policy logic in an object. The instructions can initialize an object (e.g., initialize data, initialize methods, publish an interface), facilitate receiving a message at the initialized object and selectively execute the business process policy logic by calling one or more the methods. Similarly, an example computer readable medium can store computer executable instructions for automatically accessing logic stored in a business process policy object. The instructions can identify a business process policy object to send a message to, send a message to that business process policy object, and facilitate receiving messages from that business process policy object and/or other business process policy objects.

What has been described above includes several examples. It is, of course, not possible to describe every conceivable combination of components or methodologies for purposes of describing the systems, methods, and computer readable media associated with business process policy objects. However, one of ordinary skill in the art may recognize that further combinations and permutations are possible. Accordingly, this application is intended to embrace such alterations, modifications, and variations that fall within the scope of the appended claims. Furthermore, to the extent that the term "includes" is employed in the detailed description or the claims, such term is intended to be inclusive in a manner similar to the term "comprising" as that term is interpreted when employed as a transitional word in a claim.

CLAIMS

What is claimed is:

1. A method for implementing business process policy logic in an object, comprising:

providing one or more data items associated with modeling the business process policy logic;

providing one or more methods associated with modeling the business logic;

providing an interface to the business logic that facilitates accessing the data items and/or methods through object oriented messaging;

instantiating an instance of an object that stores the data items, stores the methods, and implements the interface;

receiving a message into the instance through the interface; and

selectively performing business logic in the instance based, at least in part, on the received message, where the business logic is performed by at least one of, invoking one or more of the methods and accessing one or more of the data items.

2. The method of claim 1, where the business logic is modeled, at least in part, by at least one of, one or more IF/THEN rules, one or more rule encoding tables, and one or more neural networks.

3. The method of claim 1, where the business logic concerns at least one of, enterprise monitoring, enterprise management, enterprise resource planning, supply chain management, electronic commerce, key performance indicator processing, manufacturing, finance, and human resources.

4. The method of claim 1, where the business logic concerns at least one, of line of business visualization, application level visualization, and role based visualization.

5. The method of claim 1, where the one or more data items store information associated with at least one of, an object class identifier, an object identifier, a message identifier, a message schema, a message format, a textual description of an event, a current value for a

discrete business data point, a range for a discrete business data point, a desired value for a discrete business data point, an entity identifier, and a URL.

6. The method of claim 1, where the one or more methods perform at least one of, management by exception, event correlation, prediction, generating notifications, and rules based management.

7. The method of claim 1, where the message is received from at least one of, an information management technology source, an enterprise management source, a rules based intelligent technology source, and a predictive technology source.

8. The method of claim 1, where the message concerns at least one of, a threshold event, a reference event, a change event, a task completion event, and a task failure event.

9. The method of claim 1, where selectively performing business logic comprises generating at least one of, a business event, a notification, and a log entry.

10. The method of claim 1, comprising establishing a secondary, two-way communication with an information source to facilitate obtaining additional information associated with the message.

11. The method of claim 1, comprising updating one or more of the data items.

12. The method of claim 1, where the one or more methods perform at least one of, evaluating one or more IF/THEN rules, evaluating one or more rule encoding table results, and interpreting an output from one or more neural networks.

13. The method of claim 1, where the instance of the object produces data suitable for display on a graphical user interface that facilitates at least one of, visualizing enterprise management, and workflow management.

14. The method of claim 1, where the interface exposes one or more of the methods that perform at least one of, obtaining a message in XML format, obtaining the class of a received

object, obtaining a unique identifier for a received object, retrieving a textual description of a received event, retrieving a name of a property for which an event was generated, retrieving a value of a property for which an event was generated, retrieving a unique identifier for a provider of an event, retrieving a unique identifier for an entity to which an event is related, and retrieving a URL of an entity associated with an event.

15. A computer implemented method for automatically applying business process policy logic implemented in a business process policy object, comprising:

identifying a business process policy object to which a message will be sent; and

sending a first message to the business practice policy object.

16. The method of claim 15, comprising, receiving at least one of, a return message, and an enterprise event from the business process policy object to which the message was sent.

17. The method of claim 15, where the first message concerns at least one of, a threshold event, a reference event, a change event, a task completion event, and a task failure event.

18. The method of claim 16, comprising selectively sending a second message to the business process policy object based, at least in part, on the return message and/or enterprise event received from the business process policy object to which the first message was sent.

19. A system for automatically performing business process policies, comprising:

a business process policy object that models and implements a business logic associated with a business practice; and

a business manager that performs business management by selectively interacting with the business process policy object.

20. The system of claim 19, the business process policy object comprising:

one or more data items that store values associated with modeling a business practice;

one or more methods that implement decisions associated with modeling a business practice; and

an interface that exposes one or more of the methods.

21. The system of claim 20, comprising an event manager that provides a message to the business process policy object.

22. The system of claim 21, where the message concerns at least one of, a threshold event, a reference event, a change event, a task completion event, and a task failure event.

23. The system of claim 19, where the business process policy object communicates with one or more computer components performing at least one of, information management, enterprise management, rules based intelligence management, and predictive management.

24. The system of claim 19, where the business process policy object performs at least one of, command and control services, event management services, workflow management services, and visual management services

25. The system of claim 19, where the business process policy object communicates with one or more computer components through a common environment that in turn communicates with one or more computer components through at least one of, ODBC/XML communication, JAVA/EJB binding, C/C++ binding, COM, and C#/.NET protocols.

26. A computer readable medium storing computer executable instructions operable to perform a computer implemented method for automatically performing business process policy logic in an object, the method comprising:

initializing an object that models business logic associated with the business policy, where initializing comprises:

initializing one or more data items;

initializing one or methods; and

publishing an interface to the object;

receiving a message into the initialized object; and

selectively executing a business process policy by invoking one or more of the methods.

27. A computer readable medium storing computer executable instructions operable to perform a computer implemented method for automatically accessing business process policy logic implemented in a business process policy object, the method comprising:

identifying a business process policy object to which a message will be sent;

sending a message to the object; and

receiving a return message from the object.

28. A computer readable medium storing computer executable components of a system for automatically performing a business process policy, the system comprising:

a business process policy object that models business logic associated with a business practice; and

a business manager that performs business by communicating with the business process policy.

29. A computer readable memory storing a computer executable object, the object comprising:

one or more fields that store one or more values that model a business practice;

one or more methods that implement logic associated with the business practice; and

an interface that exposes one or more of the methods to facilitate receiving messages.

30. A system for treating business process policies as objects, comprising:

means for modeling a business practice;

means for storing data values associated with modeled business;

means for performing logic associated with the business logic; and

means for receiving messages that selectively trigger the implementation of the business logic.