US 20130339922A1

(54) **SYSTEMS AND METHODS FOR EXECUTING BUSINESS PROCESSES OVER A NETWORK**

(71) Applicant: **Aptean, Inc.**, Atlanta, GA (US)

(72) Inventor: **Craig Sproule**, Santa Cruz, CA (US)

(73) Assignee: **Aptean, Inc.**, Atlanta, GA (US)

(21) Appl. No.: **13/953,299**

(22) Filed: **Jul. 29, 2013**

Related U.S. Application Data

(63) Continuation of application No. 12/605,610, filed on Oct. 26, 2009, now abandoned, which is a continuation of application No. 10/453,236, filed on Jun. 2, 2003, now Pat. No. 7,610,575.
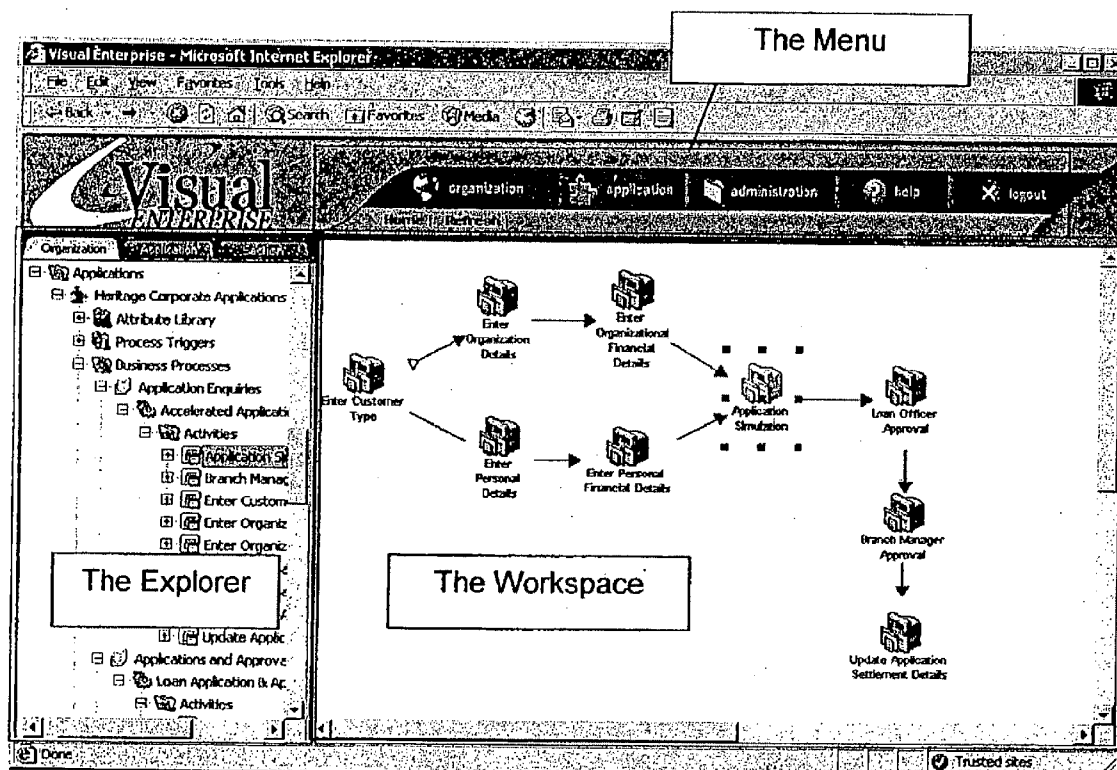
(60) Provisional application No. 60/438,833, filed on Jan. 8, 2003.

Publication Classification

(51) **Int. Cl.**
G06Q 10/06 (2006.01)

(52) **U.S. Cl.**
CPC *G06Q 10/063* (2013.01)
USPC 717/103

(57) **ABSTRACT**

Systems and methods for executing business processes over a network. In an exemplary embodiment of a method of executing business processes via a computer network, the method comprises the steps of creating and installing software resources available over a network, browsing a database of software resources using a process model builder to identify software resource definitions, loading the software resource definitions identified by the process model builder to create a business process model comprising the software resources from the database of software resources, mapping inputs and outputs of the database of software resources to allow the business process model to be executed, saving the business process model on a storage medium accessible by the network, and initiating a collaborative business process by loading the business process model into a process interpreter and executing the software resources defined within the business process model.
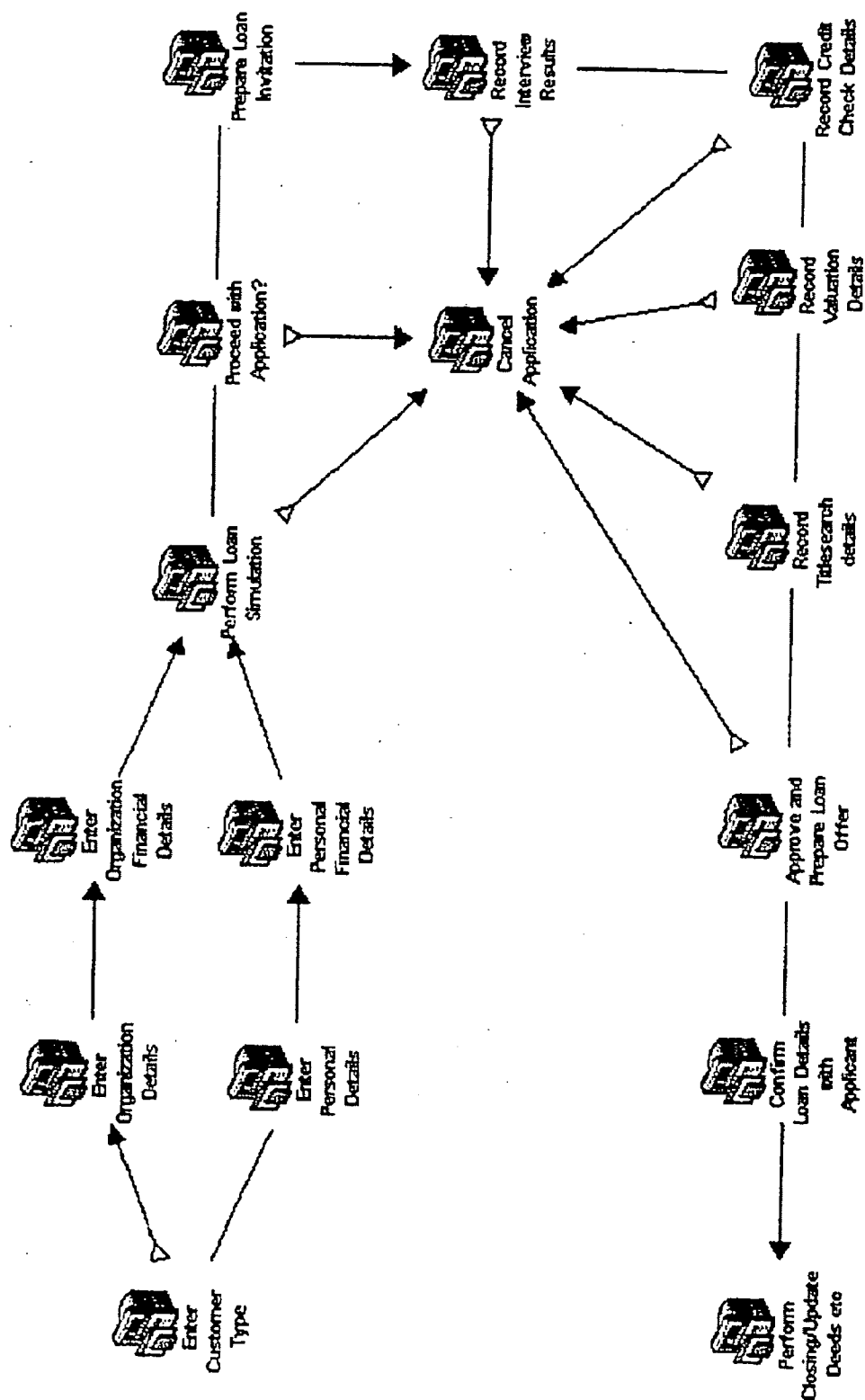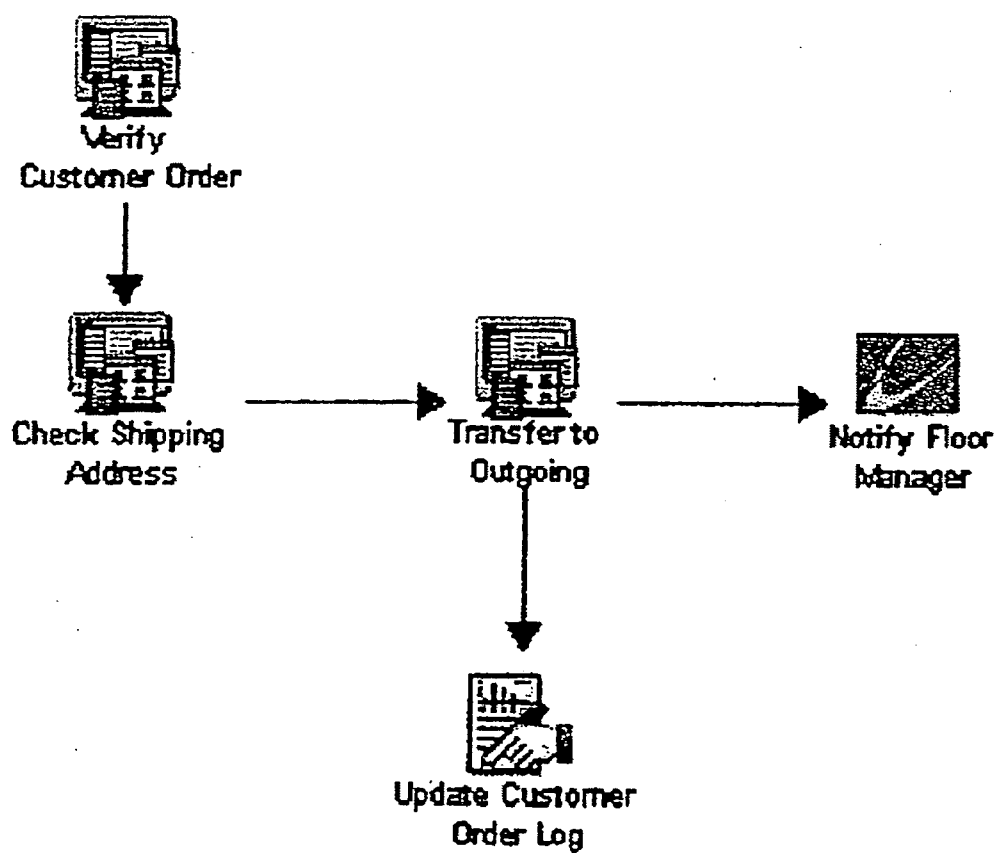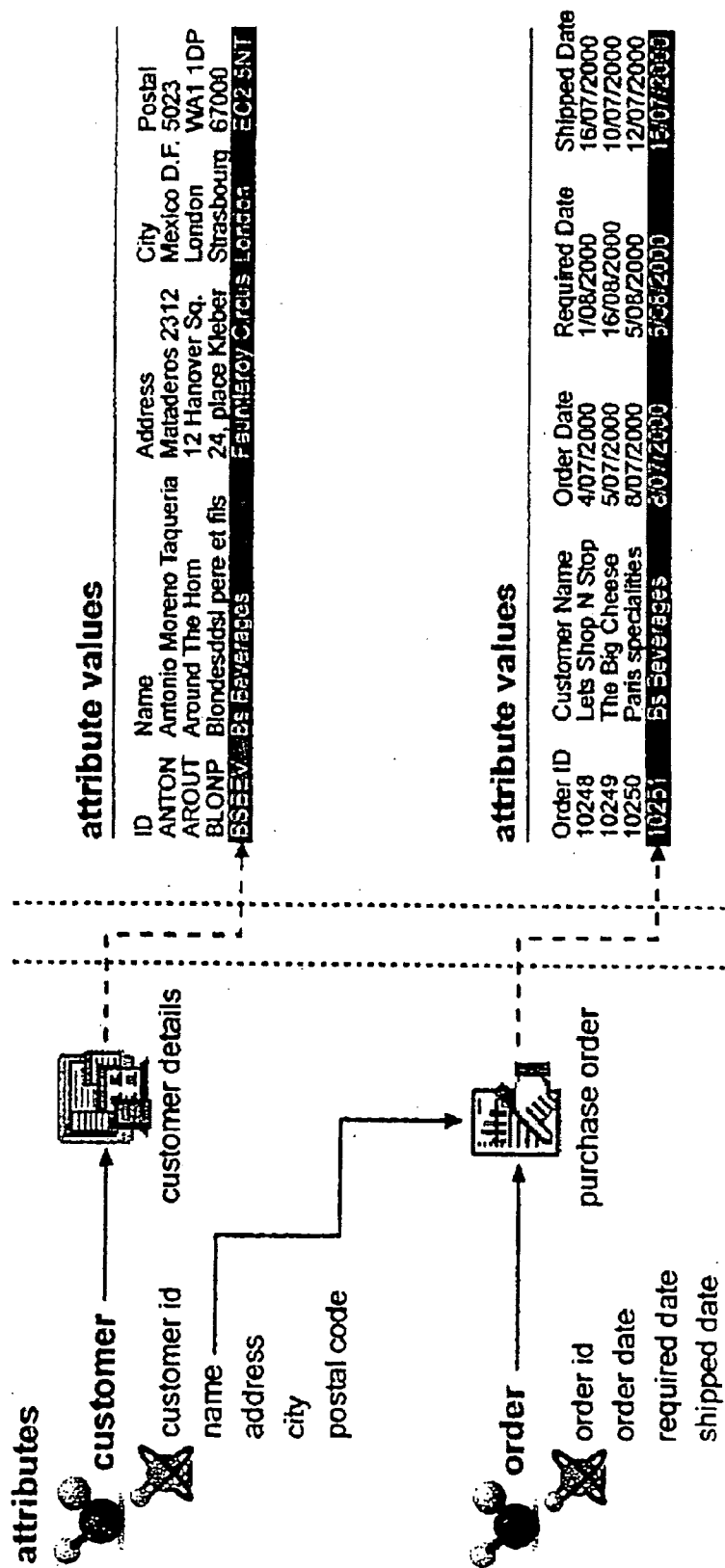
Figure 1

Figure 2

attributes

customer
customer id
name
address
city
postal code

customer details

order
order id
order date
required date
shipped date

purchase order

attribute values

ID	Name	Address	City	Postal
ANTON	Antonio Moreno Taqueria	Mataderos 2312	Mexico D.F.	5023
AROUT	Around The Horn	12 Hanover Sq.	London	WA1 1DP
BLONP	Blondesddsl pere et fils	24, place Kleber	Strasbourg	67000
BSBEV	Bs Beverages	Fauntleroy Circus	London	EC2 5NT

attribute values

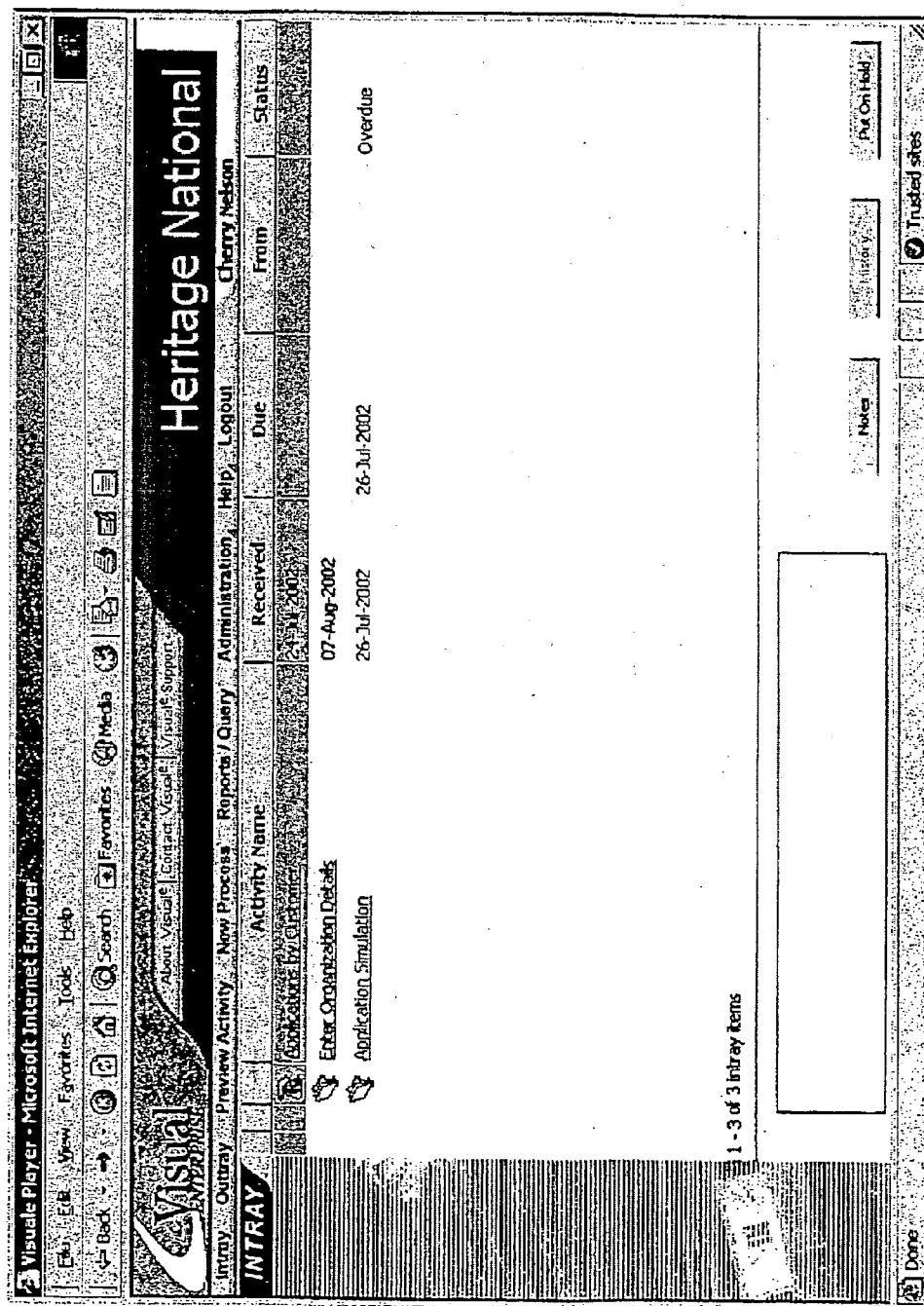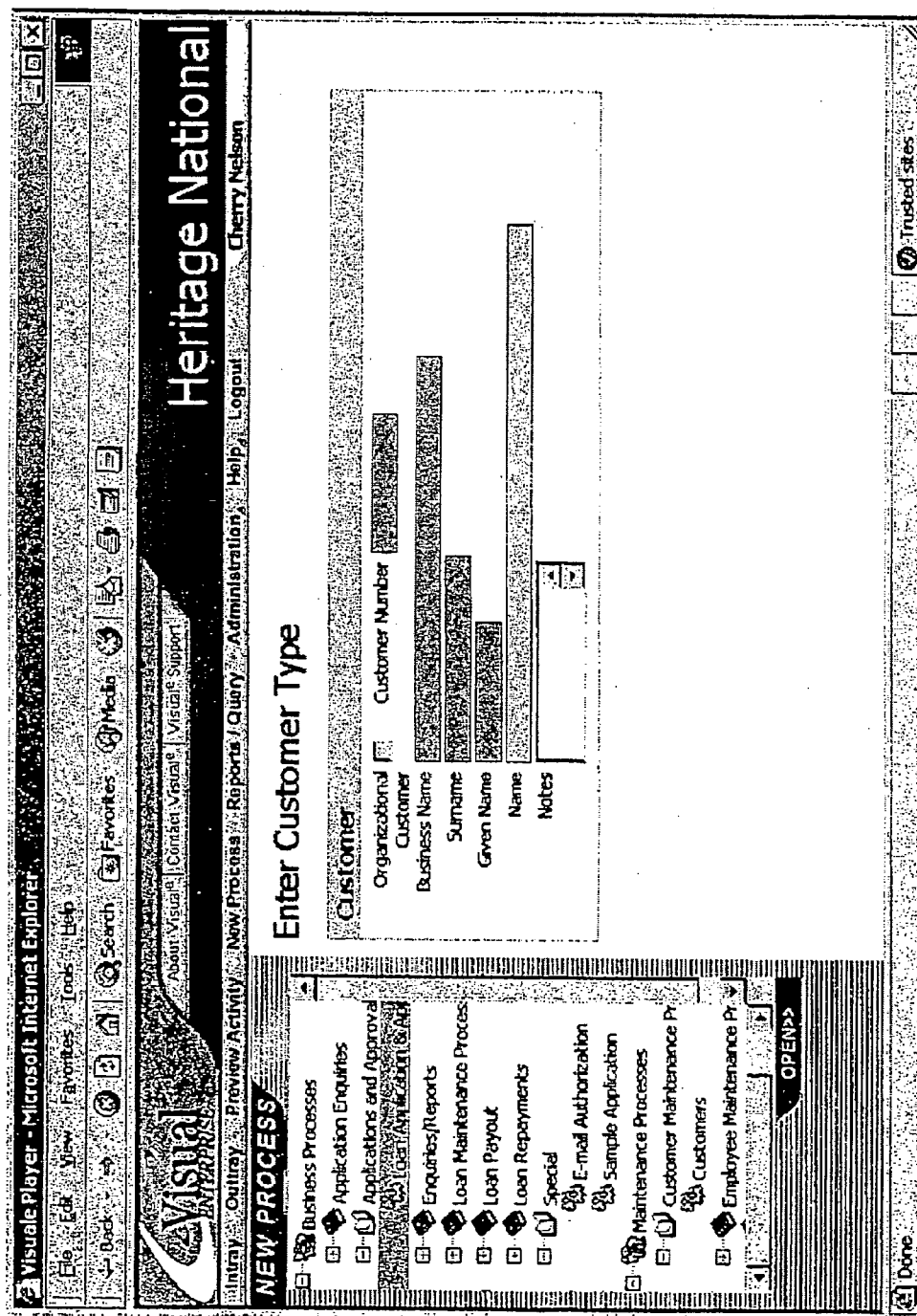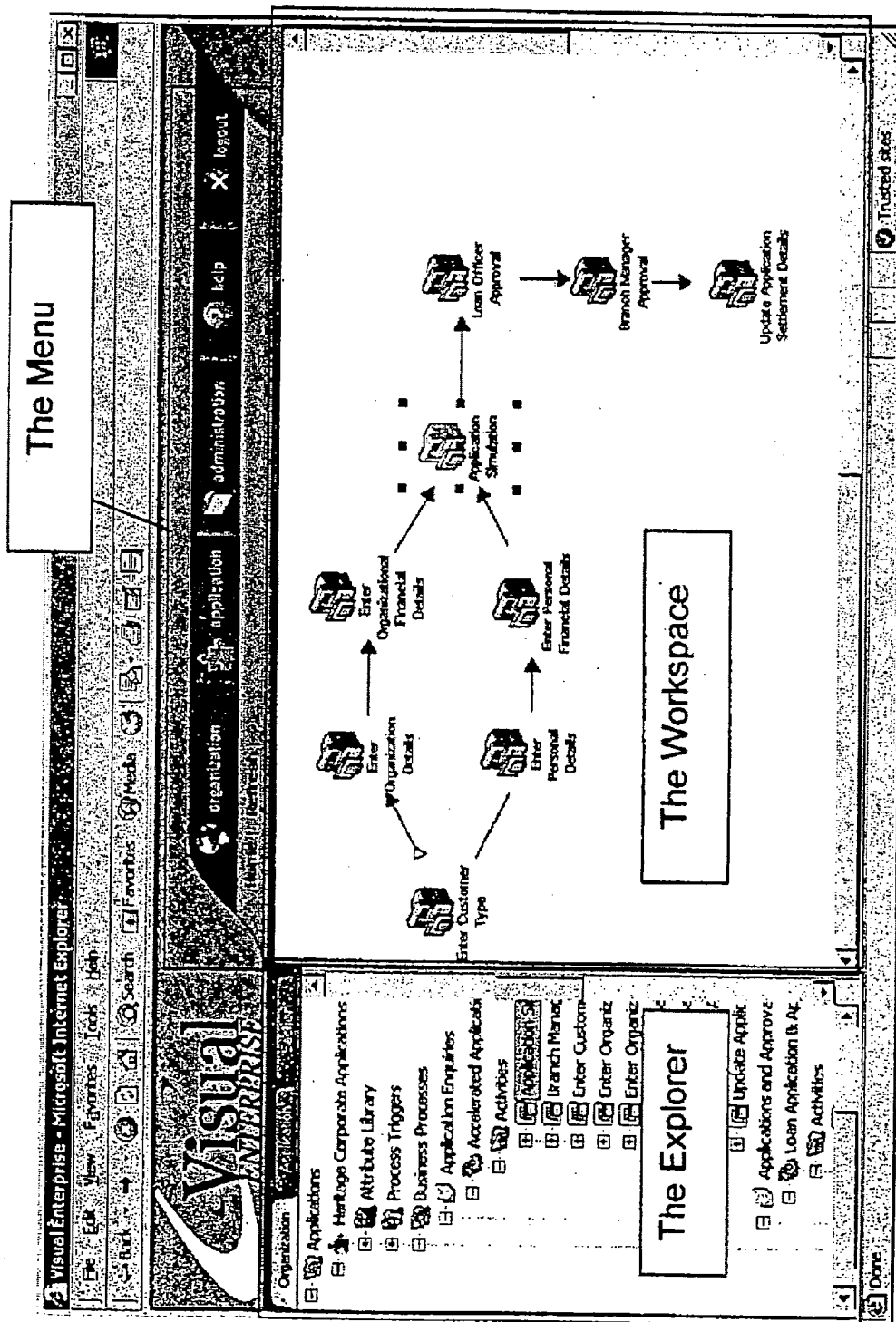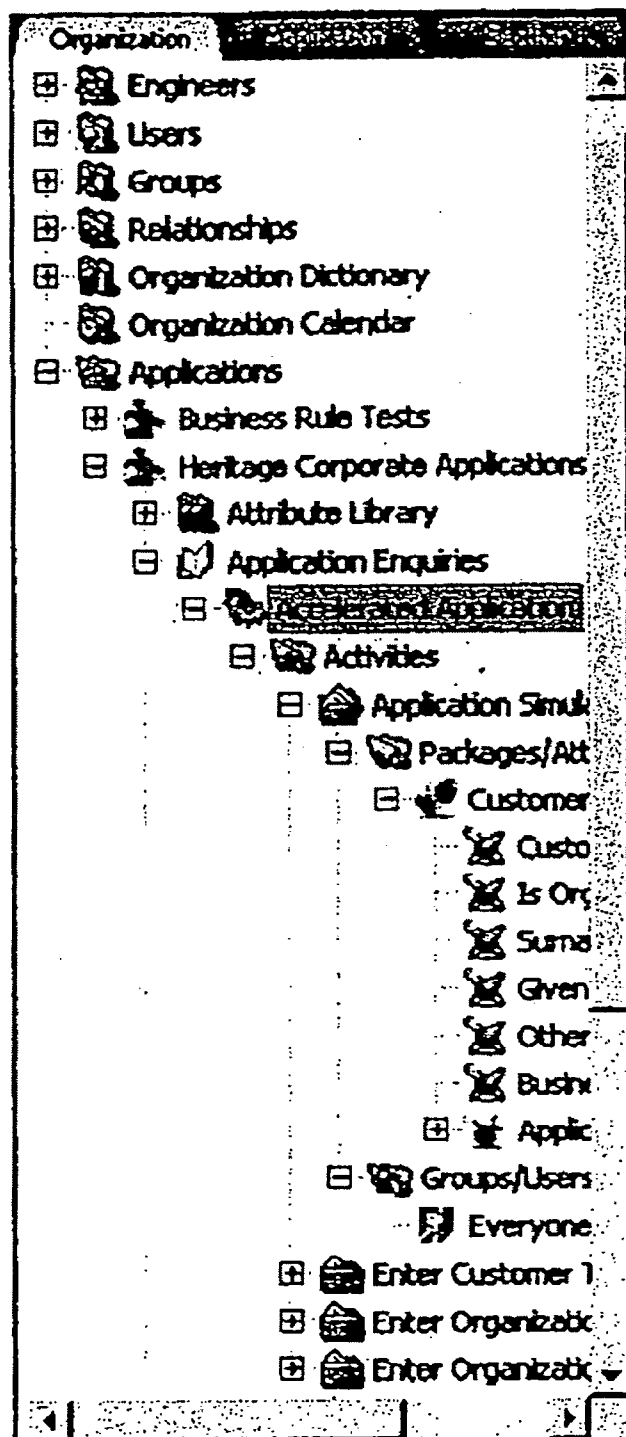| Order ID	Customer Name	Order Date	Required Date	Shipped Date
10248	Lets Shop N Stop	4/07/2000	1/08/2000	16/07/2000
10249	The Big Cheese	5/07/2000	16/08/2000	10/07/2000
10250	Paris specialities	8/07/2000	5/08/2000	12/07/2000
10251	Bs Beverages	8/07/2000	5/08/2000	15/07/2000

Figure 3

Figure 4

Figure 5

Figure 6

Figure 7

Figure 8

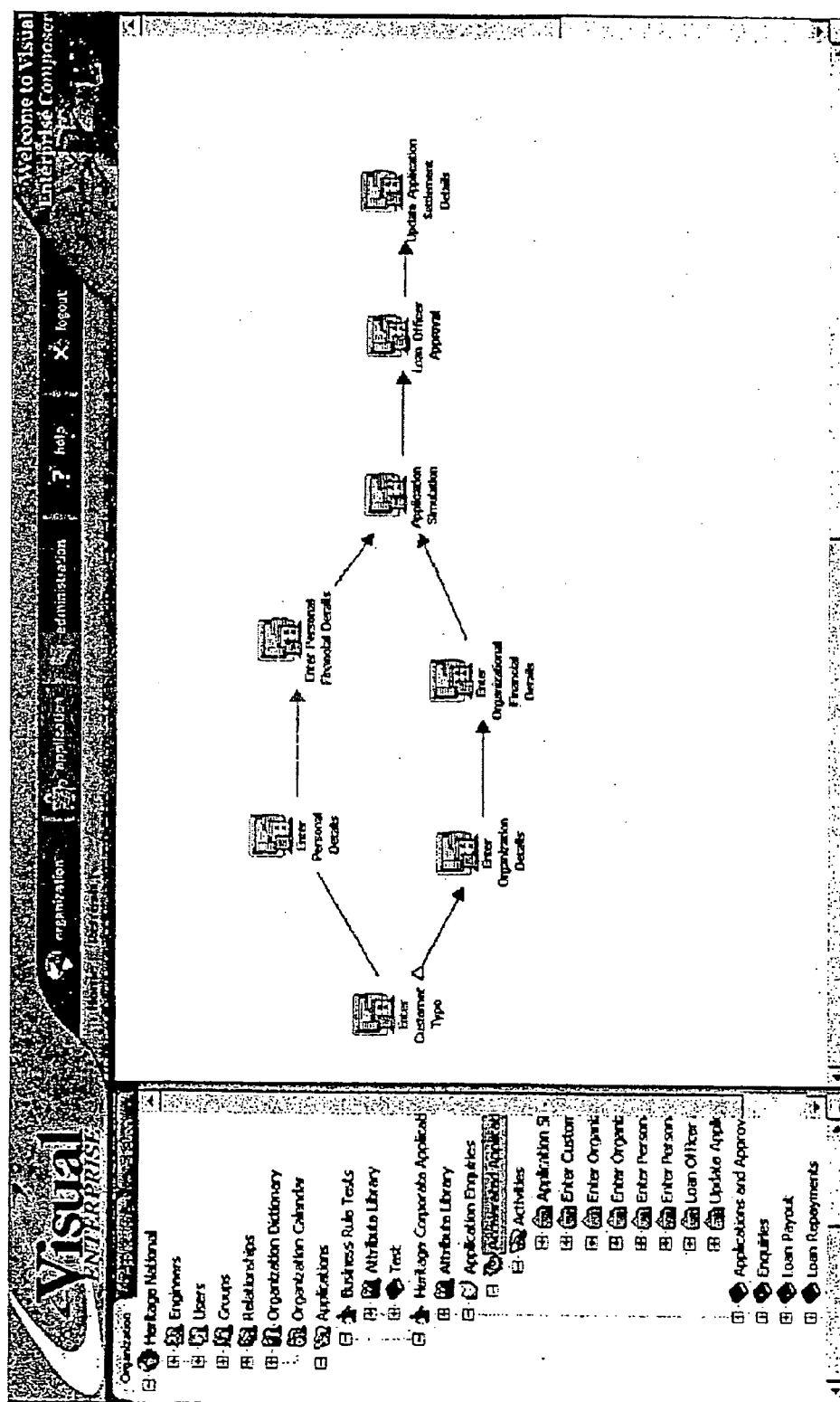Figure 9

Visual Enterprise -- Web Page Dialog

Visual

Rules Wizard

| Details | Definition | XML | References | Classification |

Title: Application Simulation

Choose an option below that allows you to define a rule to meet your requirements. After you choose an option, the Rules Wizard will display a new list of options which matches the context of the rule you have defined. This allows you to define the rules step by step.

- ○ Verify that data being entered conforms to this rule
- ○ Specify a condition to this rule (e.g. if Price is greater than 0 then ...)
- ● Assign a value or expression to an attribute
- ○ Assign a default value or expression to an attribute
- ○ Select an instance of package based on a specified condition
- ○ Restrict the valid instance of a package based on a specified condition
- ○ Perform an action, such as scheduling an activity or initiating a process
- ○ Define an exception action

Rule:

[Business Rule] assign the Amount Calculated = 0 minus Present Value(the Product Rate Per Month , the Product Number Of Months , the Monthly Repayment , 0 , 1)

...

| Reset | | Submit | Cancel |

Figure 10

Figure 11

Figure 12

Figure 13

Figure 14

Figure 15

Automated Activity Details -- Web Page Dialog

Automated Activity Details

| Attributes | Links | Rules | Schedule | Filter Criteria |

Activity Name: Process Order

Rules

if the Order does not have a Order Buyer then assign the Retailer = the Order Bill and Ship To

assign the Order Credit Available = the Order Credit Limit minus the Order Balance

assign the Credit Percentage Available = ((the Order Credit Limit minus (the Order Balance pl

assign the Order Credit Limit = 10000

assign the Order Balance = 6000

assign the Order Credit On Hold = false

assign the Credit Limit = GetCreditLimit(the Number) assign the Balance = 0 minus GetBalance

☐ All Levels ☑ Activity ☐ Package ☐ Attribute

| Create... | Properties... | Remove | Disable |

| Submit | Refresh | Cancel |

Figure 16

Figure 17

Figure 18

Figure 19

Figure 20

Figure 21

```
- <XML id="xmlActivityInstance">
  - <ActivityInstance id="2546">
    - <Entity id="4133" OneToMany="0" package="1504">
      - <EntityInstance id="1851">
          <Attribute id="3525" type="String" value="CMQ123" />
          <Attribute id="3526" type="Integer" value="0" />
          <Attribute id="3527" type="String" value="Smith" />
          <Attribute id="3528" type="String" value="Albert" />
          <Attribute id="3529" type="String" value="" />
        - <Entity id="4126" OneToMany="0" package="1491">
          - <EntityInstance id="1853">
              <Attribute id="3427" type="Float" value="5850.00" />
              <Attribute id="3428" type="Float" value="4720.00" />
              <Attribute id="3429" type="Float" value="3180.00" />
              <Attribute id="3444" type="Float" value="150000.00" />
              <Attribute id="3445" type="Float" value="165000.00" />
              <Attribute id="3446" type="Float" value="20000.00" />
              <Attribute id="3447" type="Float" value="170000.00" />
              <Attribute id="3449" type="Float" value="1350.00" />
            - <Entity id="4139" OneToMany="0" link="1523" package="1523">
              - <EntityInstance id="0">
                  <Attribute id="3609" type="Integer" value="" />
                  <Attribute id="3610" type="String" value="" />
                  <Attribute id="3616" type="Float" value="" />
                  <Attribute id="3617" type="Integer" value="" />
                  <Attribute id="3618" type="Float" value="" />
                </EntityInstance>
              </Entity>
            </EntityInstance>
          </Entity>
        - <Entity id="4127" OneToMany="0" package="1492">
          - <EntityInstance id="0">
              <Attribute id="3467" type="Float" value="0.00" />
              <Attribute id="3468" type="Date" value="" />
              <Attribute id="3479" type="Integer" value="" />
              <Attribute id="3478" type="Integer" value="" />
              <Attribute id="4097" type="String" value="0" />
            </EntityInstance>
          </Entity>
        </EntityInstance>
      </Entity>
    </ActivityInstance>
  </XML>
```
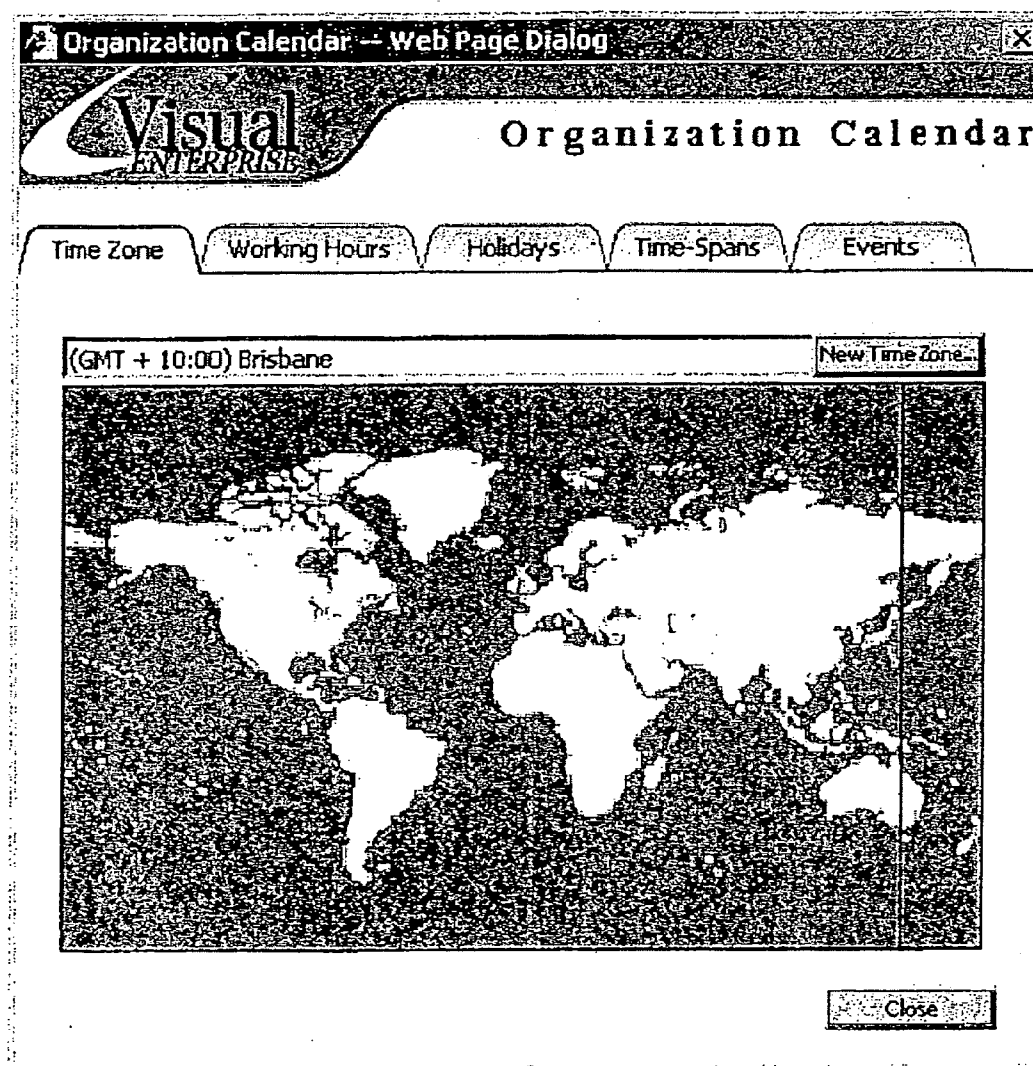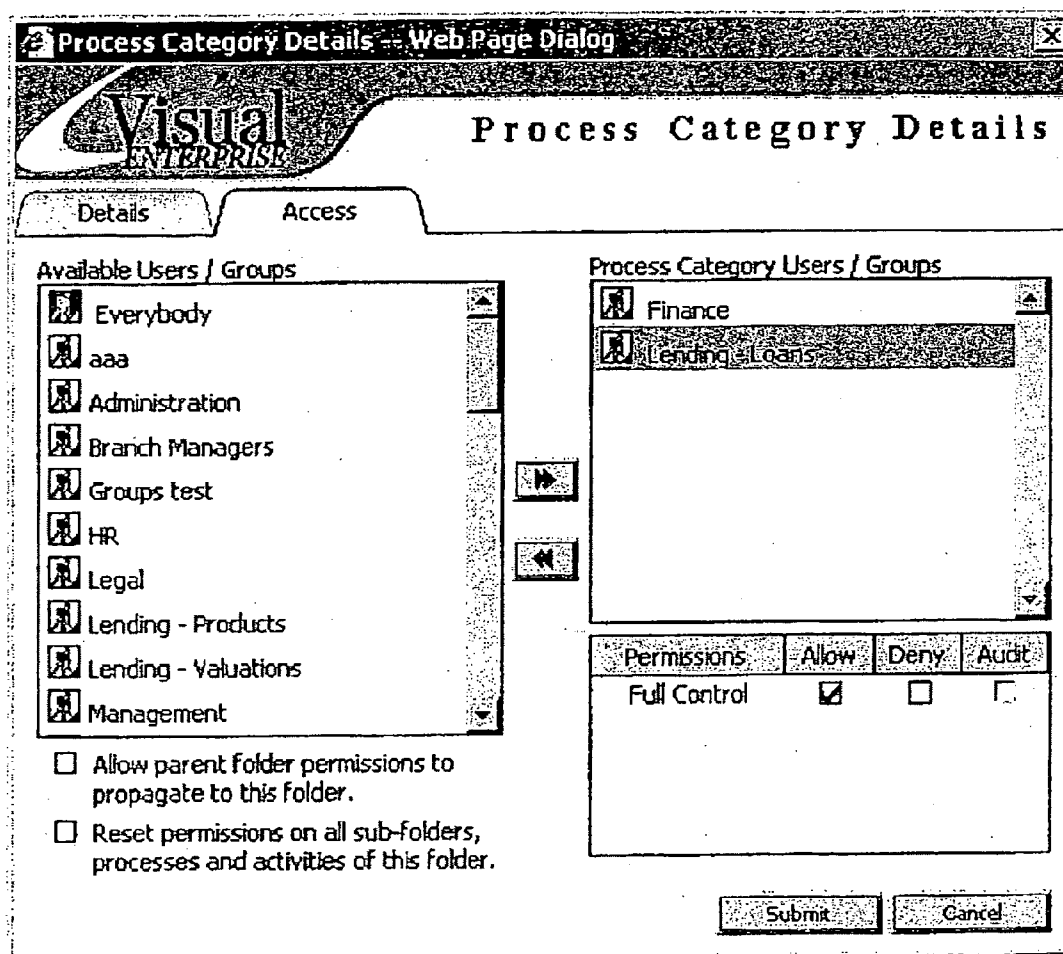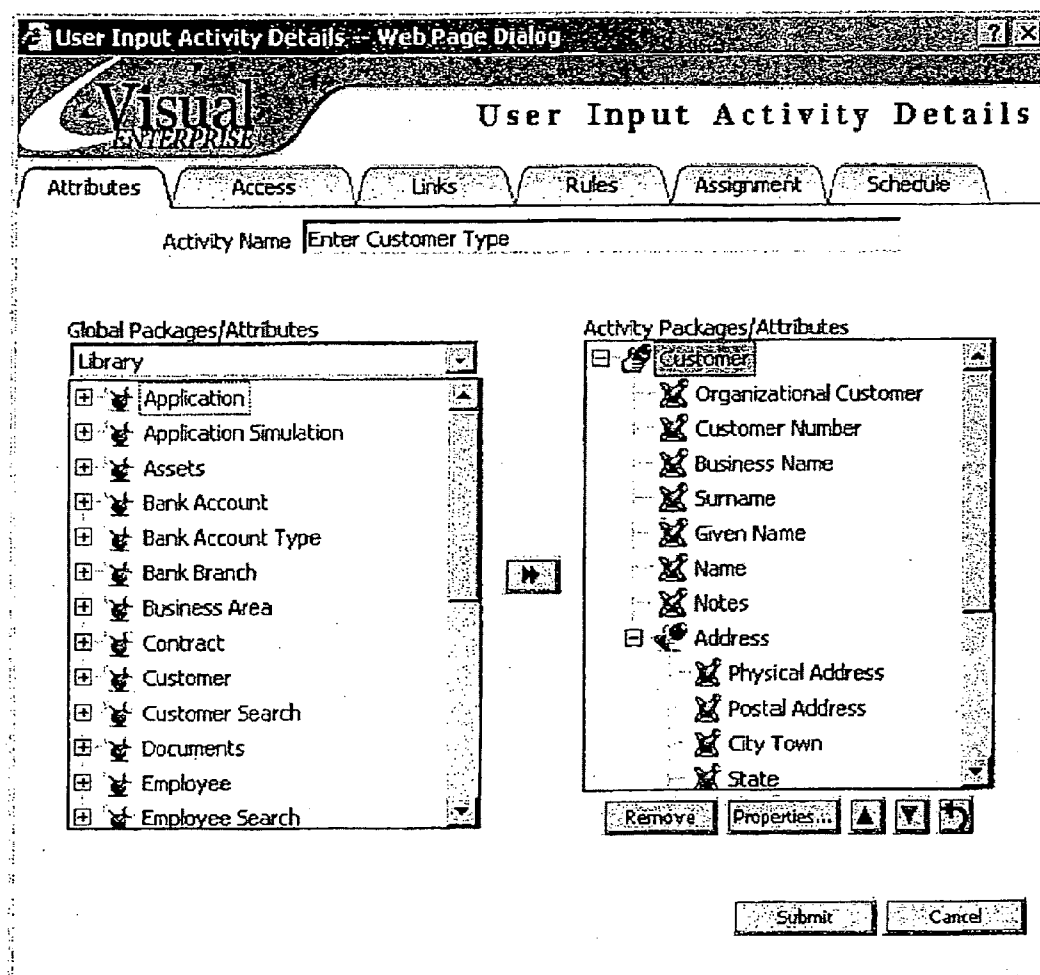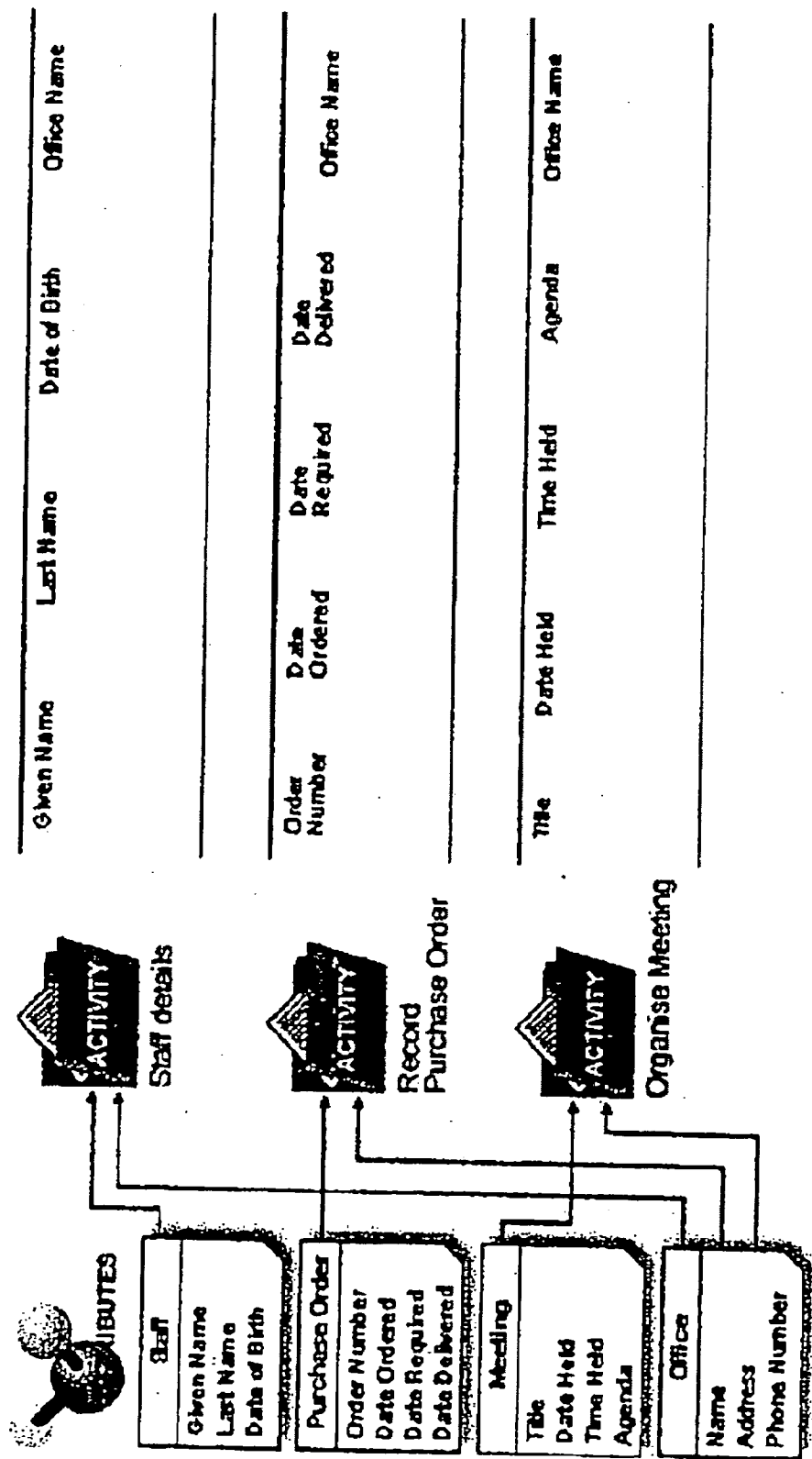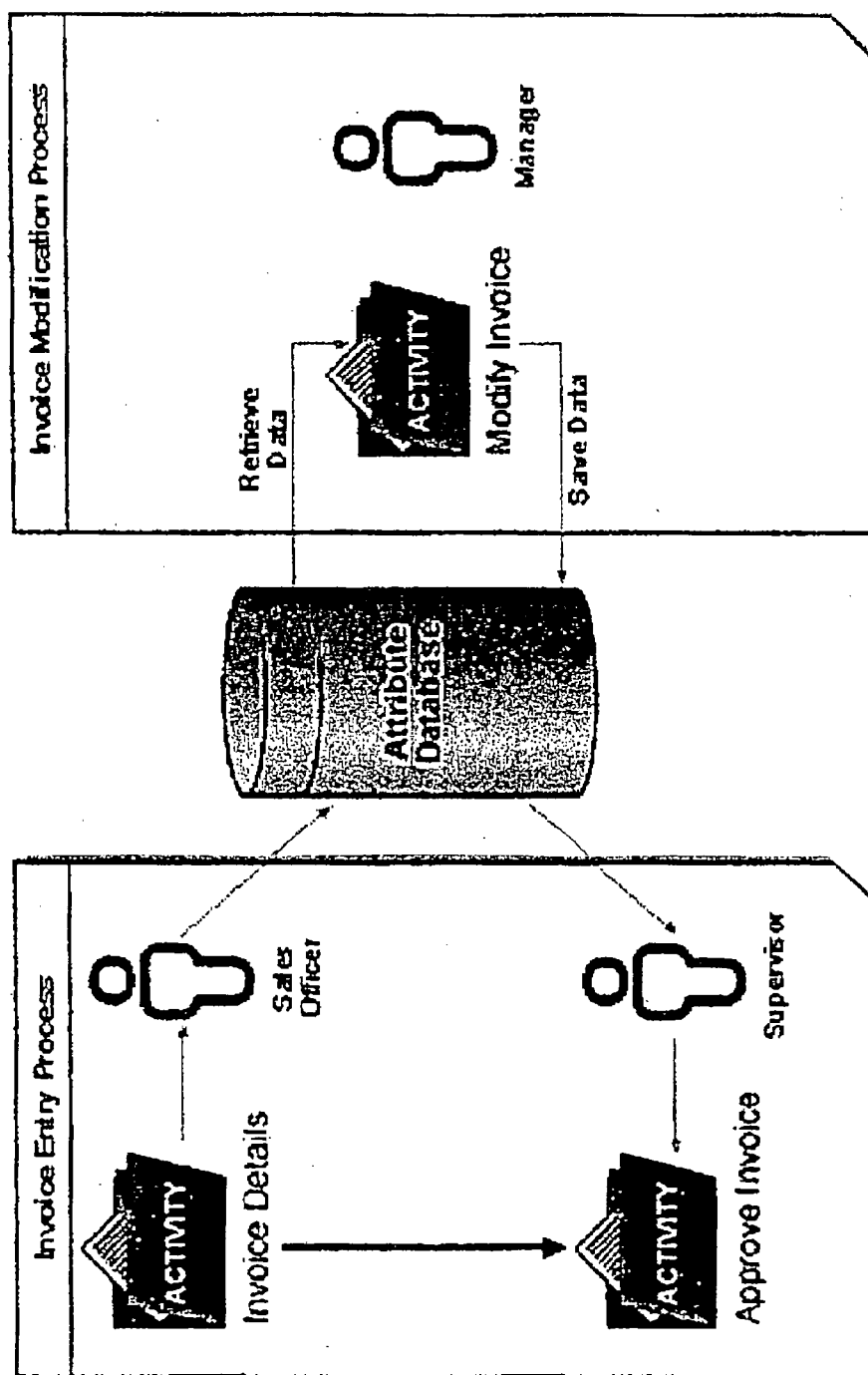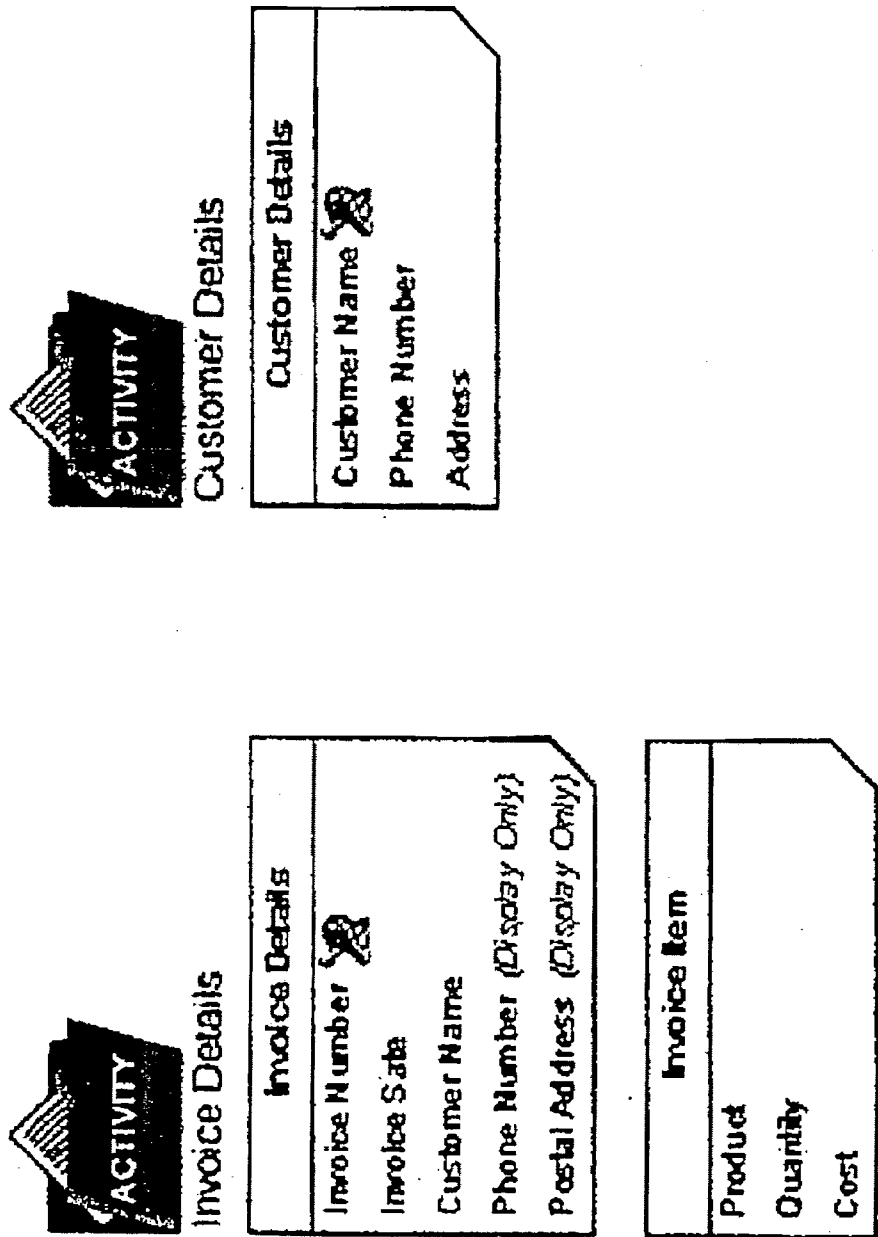
Figure 22

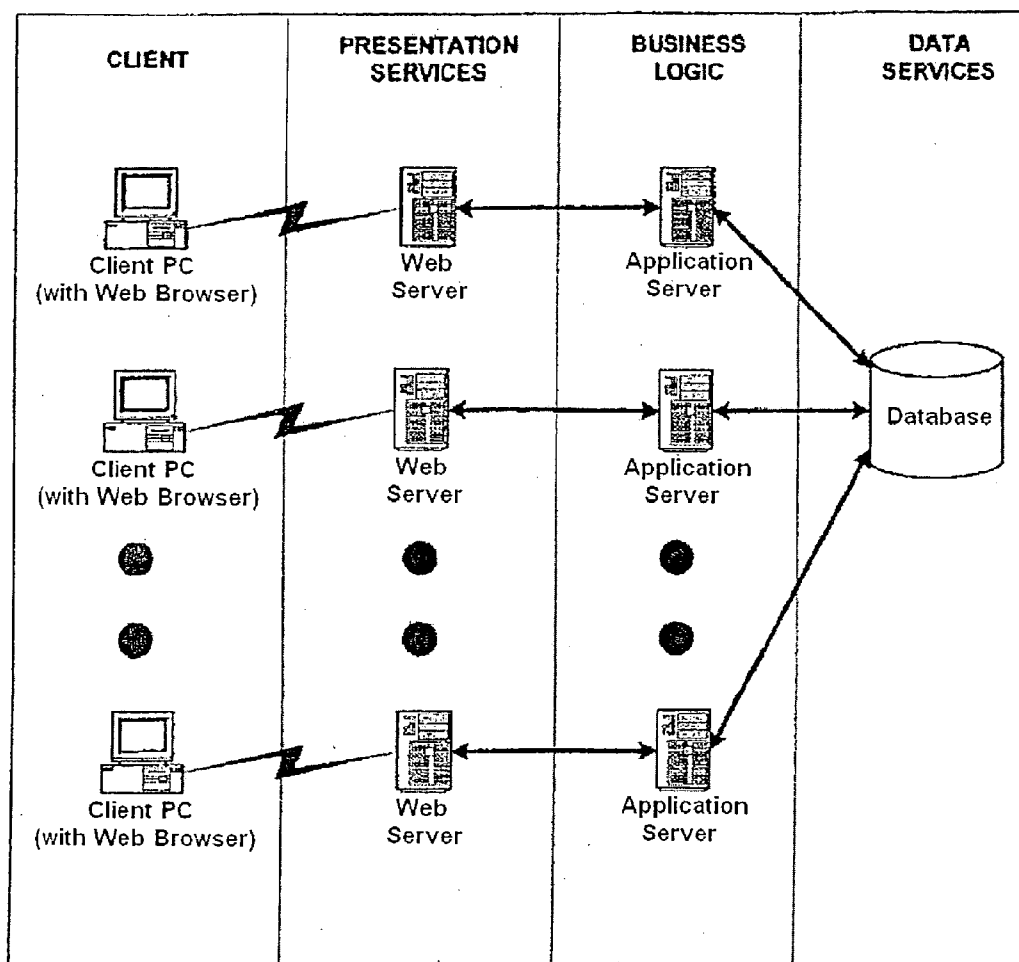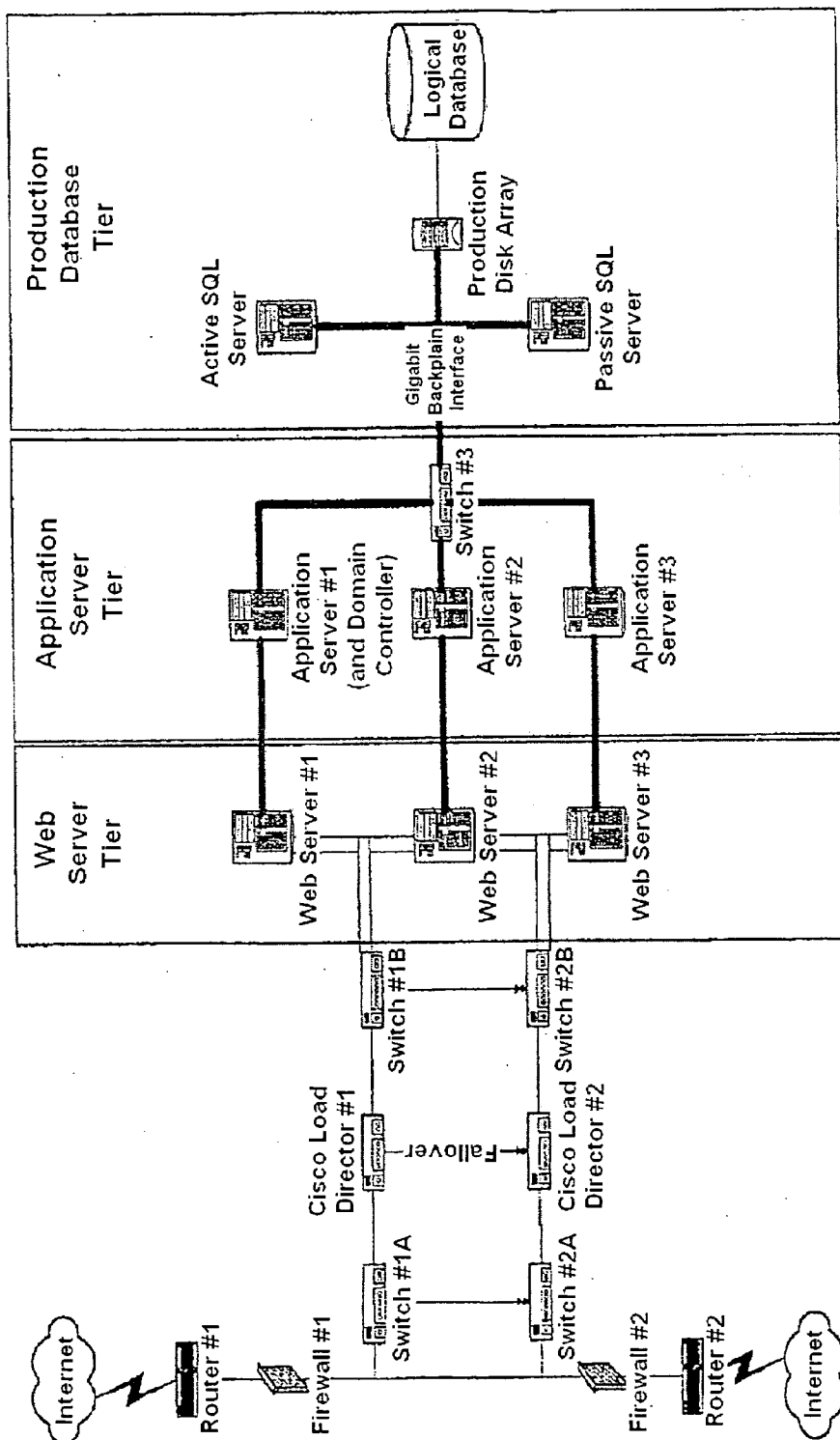| CLIENT	PRESENTATION SERVICES	BUSINESS LOGIC	DATA SERVICES
Client PC (with Web Browser)	Web Server	Application Server	
Client PC (with Web Browser)	Web Server	Application Server	Database
Client PC (with Web Browser)	Web Server	Application Server	

Figure 23

Figure 24

SYSTEMS AND METHODS FOR EXECUTING BUSINESS PROCESSES OVER A NETWORK

CO-PENDING APPLICATION

[0001] This application claims benefit the failing date of Provisional Patent Application Ser. No. 60/438,833 filed on Jan. 8, 2003, by Inventor Craig Sproule.

FIELD OF THE INVENTION

[0002] The present invention relates generally to collaborative business processes and transactions. More specifically, the present invention relates to systems and methods for the dynamic creation, generation, integration and execution of business processes and transactions over a network.

BACKGROUND

[0003] Enterprise software technology has matured to the point where today many hundreds of thousands of business processes are wholly or partially implemented in software and conducted via communications transacted across distributed networks. These communications networks may comprise elements of more than one electronic messaging system, to include the Internet, Extranets, Intranets, Local Area Networks, Wide Area Networks and other suitable electronic messaging networks or systems known in the art. The capability to rapidly generate new business process software that can smoothly integrate with pre-existing distributed computing environments, software programs, software components, and communications systems can greatly enhance the financial advantage, economic value and effectiveness of the resultant software and the authoring agent.

[0004] Traditional application software development has involved business analysts undertaking business process reviews, collecting requirements, and collating business rules in order to produce a general design for a business system. This general design would be expanded to include software developers who would then create a detailed software specification. This detailed software specification would then be handed over to software engineers to prepare functional and technical design documents. A team of programmers and administrators in turn would use these documents to develop an application that provide a certain solution. Once complete, the rigorous testing process would commence. This normally includes code testing, component testing, system testing, field and user testing.

[0005] Using the traditional approach, the possibility of errors introduced into the final application due to omissions, inconsistencies, ambiguities or misunderstandings was extremely high. In addition, maintenance and modification of the application may involve, for example, understanding and modifying code associated with the application. This can be difficult due to personnel change, cost, risk, and length of time since the application was developed, etc. All of these factors may contribute to unnecessary delay in responding to business needs.

[0006] The advent of web-based business-to-business and business-to-customer systems demands an integrated business application development and delivery environment. Such an environment will allow business to rapidly and directly implement a custom response to a business opportunity, to fine-tune the custom response as the business learns, and to adapt that response as business requirements evolve. It is, therefore, an object of the method of the present invention to enable the authoring and composition of business process models, and the resultant real-time generation, integration and execution of business applications from business process models with little or no coding, programming or scripting.

SUMMARY OF THE INVENTION

[0007] Towards this and other objects that will be made obvious in light of the present disclosure, a preferred embodiment of the present invention provides an environment, optionally a runtime environment or with runtime environment aspects, for the composition, generation, integration and execution of collaborative business processes and transactions over a distributed computing-based network.

[0008] The method of the present invention optionally enables, in certain preferred embodiments, a creation of a business process flow by a business process engineer or designer that is implemented in a software program and over a distributed computer network, and wherein the engineer or designer builds an abstraction of the business process in software and the software program is generated without software coding or software programming by the designer or engineer.

[0009] In a first preferred embodiment of the method of the present invention, the business process designer creates an abstraction of a business process to define the operation and functionality of a software application by using a computer network browser, such as INTERNET EXPLORER™ web browser. The designer's browser may be located on a personal computer, a wireless computational device, or other suitable computational systems or devices known in the art, and having access to a composer software over the Internet. The designer optionally does not need or have a client software on the personal computer, but directs the composer software solely with the browser and via a communications link with the Internet. The designer users the composer to create a graphical representation model of the logical flow and processing of a target business process. The designer additionally builds or selects and designates data structures, data types, and an organizational structure or structures that inform, enable and/or direct the dynamic activity of a software application. The designer additionally uses the browser to inform, designate and/or select external software resources, such as web services made available over the Internet, for access by the software application. The designer may optionally and additionally format reports and messages to be provided by the software application. A user may then use a suitable user's computational device known in the art to direct a software player to generate the application software by employing the instructions, information, software data structures, software logic models, organizational structure, and access to external software resources provided by the designer. The user's computational device may communicate with the player by means of an interface software, such as a browser, and via a communications link, such as an Ethernet link, an Internet dial-up link, a wireless link, or another suitable communications link known in the art. The player may reside entirely outside of the user's computational device. Account access and password protection may be employed to confirm the identity of the user and that the user has properly approved access to use the target application software. The user may be permitted access at one of a plurality of access levels to more than one application software selected from a library of application software programs. The player may then direct the player to generate, or instantiate the target software application and further direct the player to run the generated application soft-

ware. The target application software may then be created by the player and the target application software will execute. The player may perform external calls, lay out reports and messages, and use the grammar engine to generate scripts needed to generate the target application software. The first preferred embodiment of the method of the present invention may thereby enable a designer and a user without software coding or programming to define, create, access and use the target software application by means of a browser located on a networked computer, and a remotely located composer, software, and other suitable software resources and web services known in the art that are available via a computer network.

[0010] A first alternate preferred embodiment of the present invention is called Visual Enterprise™ BPI Software Platform ("VE") and includes the VE Composers™ and the VE Player™. The VE Composer allows a business process engineer/expert/person (terms used interchangeably throughout the document) to model, over a network, the processes, rules and data structures of an entity upon the basis of a plurality of software resources. A software resource may be, in singularity or combination, software resource is selected from the group including a database, a grammar rule, a software object, a data structure, a datum, a universal resource locator, a network address, a data type definition, a data pointer or other suitable software elements known in the art.

[0011] The VE Player, by a user at run time, then executes in real time the modeled processes, constructs the page layout and accesses relevant data, according to rules and resources as related within the business model generated by the business process expert. The preferred embodiment dynamically (meaning in real-time) constructs Web pages based on the process, rule and data definitions prepared by the process expert. Pages are constructed only when required, ensuring that the solutions are flexible, dynamic and based upon the most current model revision. The means of doing so is the VE Enterprise Interpreter™, which is inside the VE Player. The VE encapsulates a grammar engine that allows the executable code to be dynamically created without syntax or scripting errors. The preferred embodiment relies upon four main principles of applications development: process, data, rules and calendar. The model generated by the VE Composer accords with the business processes employed by a relevant organization, and may optionally model workflow, work allocation and assignment, process escalation and administration. Modeling data may comprise defining each data attribute, including properties such as its type, e.g., numeric, text, or date, as well as defining associations between attributes. Business policies are reflected in the rules that govern the capture and maintenance of data within the context of each step of a process. The preferred embodiment conforms to at least one time zone, the business/working hours of one or more location, the public holidays observed in designated locales and any other temporal events relevant to the organization's operations.

[0012] In one optional aspect of the invention, a method of developing applications is disclosed. Using a communications network browser software, such as INTERNET EXPLORER™ computer network browsing software, or another suitable browser known in the art and suitable for a selected communications network, a business process expert develops an application solutions for an organization by composing a business process model over a computer coupled to a network. The application development environment includes a graphical user interface, such as a browser graphical user interface, that provides, or provides access to, tools, software modules, software components, data sources, web services and other suitable software resources known in the art, for the business process expert to define and interweave data structures, rules, attributes, activities, organization, policies, workflow for processes and applications for an entity and using optionally including pre-existing software components related to the entity. Models are related to the organization in a hierarchical structure. Models may include business rules information, calendar information, data information and user information associated with the entity or other entities. An entity may be a person, a corporation, a department, a public or private agent or agency, an association, a society, a software program, or other suitable identity or collective known in the art. Using drag and drop, point and click or voice response actions, a business process expert designs all the elements of the applications in a fashion that models the way the business establishes, or wishes to explore managing and controlling a particular set of business processes, activities or transactions.

[0013] In certain alternate preferred embodiments of the method of the present invention, a user, engineer, or business process engineer may integrate a first business process software program with a second business process software program, wherein the first business process software program and the second business process software program communicating via a computer network. Preferred embodiments of the present invention of this type may comprise (1.) interrogating the second business process software program via the computer network to determine how a message containing data may be structured in order to enable the second business process software program to accept the data, (2.)) detecting a set of data that the first business process software program may provide to the second business process software program, (3.) structuring a message for transmission to and acceptance by the second business process software program, the message containing the set of data, and (4.) transmitting the message to the second business process software program via the computer network.

[0014] Other features of the present invention will be apparent from the accompanying drawings (found in the attached VE Blueprint) and from the detailed description, which follows.

BRIEF DESCRIPTION OF THE DRAWINGS

[0015] The present invention is illustrated by way of example in the following drawings in which like references indicate similar elements. The following drawings disclose various embodiments of the present invention for purposes of illustration only and are not intended to limit the scope of the invention.

[0016] FIG. 1 is an example, from the VE Composer™, of the flow of a business process as seen in the VE Composer. A business process can be of any type, of any length and of any complexity.

[0017] FIG. 2 is an example of linking activities. Linked activities can be of any complexity, including, but not limited to, internal, external, automated, launched, messaged, reported and web services types of activities.

[0018] FIG. 3 is an example of a data set. Data sets can be text, numeric, date, time span, document, or Boolean and can be accessed both internally and externally to the preferred embodiment.

[0019] FIG. 4 is an example, from the VE Player™, of one-to-one (Invoice to Dealer) and one-to-many (Invoice to Line Item) associations within an activity.

[0020] FIG. 5 is an example of the VE Player's 'In Tray.' The Player also includes an Out Tray, New Processes, Reporting and Administration.

[0021] FIG. 6 is an example New Process Explorer.

[0022] FIG. 7 is an example of the VE Composer page. The VE Composer comprises of a main menu, an explorer or navigation section and a workspace where the process models are created and maintained.

[0023] FIG. 8 is an example of the process explorer in the VE Composer.

[0024] FIG. 9 is an example of the VE Composer's process workspace.

[0025] FIG. 10 is an example of Business Rule Composer Definition inside the rules wizard, which is inside the VE Composer.

[0026] FIG. 11 is an example of the Organizational Calendar, inside the VE Composer. The calendar consists of Working Hours, Holidays, Date Events and Time Events, all in multiple time zones.

[0027] FIG. 12 is an example of a Process Category Definition.

[0028] FIG. 13 is an example of defining a User Input Activity.

[0029] FIG. 14 is an example of Package Associations.

[0030] FIG. 15 is an example of Attribute Definitions.

[0031] FIG. 16 is a sample business rules created with the VE Composer.

[0032] FIG. 17 is an example of an Activity Page as the Activity Page would appear in the VE Player.

[0033] FIG. 18 is an example of interaction between 2 processes.

[0034] FIG. 19 is an example of Activity/Package definitions with related attributes.

[0035] FIG. 20 is an example of Activity/Package definitions.

[0036] FIG. 21 is an example of an activity executing in the VE Player.

[0037] FIG. 22 is a sample of a XML Activity Data Fragment.

[0038] FIG. 23 is an implementation of an alternate preferred embodiment of the method of the present invention in a networked computing environment.

[0039] FIG. 24 is an implementation of another alternate preferred embodiment of the method of the present invention in a networked computing environment.

DETAILED DESCRIPTION

[0040] In describing the preferred embodiments, certain terminology will be used for the sake of clarity. Such terminology is intended to encompass the recited embodiment, as well as all technical equivalents, which operate in a similar manner for a similar purpose to achieve a similar result:

[0041] Processes (see FIG. 1): A Process is a collection of related activities in a defined workflow that, once completed, execute a specific and well defined business objective, process, collaboration or transaction.

[0042] Activities (see FIG. 2): An Activity is the smallest action element of a process. An Activity is normally associated with one step in an overall process. An Activ-

ity can be user defined, automated, emailed, reported, queried, messaged, remote, define an event or launch a new process.

[0043] Packages (see FIGS. 13 and 14): Packages are similar to database tables. Packages are associated in two ways: 1) independent packages can be loosely associated in a relatively ad hoc way, as required for specific processes, and 2) packages are associated to the point of dependence, for example line items on an invoice. Here the dependent package would be defined as a Sub-Package. Data in a Sub-Package cannot exist independently of the package instance Data is dependent upon. When the 'parent' is deleted, the sub-package data is removed as well.

[0044] Attributes (see FIGS. 3, 15 and 20): Attributes are similar to columns in a database table. An attribute defines a collection of values where each instance in the collection conforms to the definition provided by the attribute. Each attribute has a number of properties, such as whether the attribute represents a date, a number, text, etc; how it should be formatted on the page and in reports; input control options, etc. Attributes are defined in collections called Packages, and Packages are associated with each other in various ways. Each Attribute defines the name, data type, size and other properties of the values that will be stored against an Attribute. When a package instance is created, attribute values are created and stored with a reference back to the package and the attribute within the package. A single package instance consists of one set of values. There can be any number of instances of a given package.

[0045] Organization (see FIGS. 11 and 12): An Organization is the structure in which the model and the associated application (a collection of processes) executes. An organization may comprise: 1) Engineers, those authorized to design and modify the system, 2) users, those authorized to use the system, 3) Directory; of which a collection of packages and attributes designed for global purposes for the enterprise, and 4) Calendar, which determines the working hours, holidays, etc. of each area of operation for a business entity or entities.

[0046] Rules (see FIGS. 10 and 16): Business rules are the policies, regulations and constraints governing the way an organization carries out organizational operations. VE is concerned with those business rules that are in some way related to, or impact upon, the business processes being developed. For example: "the discount must be less than 10%", or "a purchase order form must include at least one item".

[0047] Integration: The preferred embodiment supports 5 integration methods:

[0048] External Components: VE allows standard COM components to be utilized as part of the native business rule grammar. This is achieved by registering a given COM component using the Composer interface. VE examines that methods implemented by the components and maps all parameters to native VE attribute types. The methods are then dynamically added to the business rule grammar. At business rule execution time, VE performs the necessary interfacing required to make remote procedure calls to the relevant external components.

[0049] Remote Data: VE comes standard with a custom web service that can be deployed against any data

4

source supported by the ActiveX Data Objects (ADO), which includes any data source that has an ODBC driver. Through VE accessing the web service, engineers are able to define "remote packages" based on tables in data sources supported by the web service. Such "remote packages" are treated as if they were native data of the given VE solution. The business rules layer performs all the necessary mechanics to update the data source taking into account the data source's referential integrity.

[0050] XML Messages (see FIG. 22): VE allows custom XML (eXtensible Markup Language) messages to be exchanged with external systems via the Internet. An engineer can construct definitions (i.e. using the XML Mapper interface) that describe how to interpret incoming XML messages and convert them to native VE packages and attributes, and vice versa. Again, from the engineer's perspective the XML is treated as if the XML code were native VE data, with the business rules layer performing all the mechanics necessary to convert between the defined XML schema and VE packages and attributes.

[0051] SOAP Messages: VE supports both the construction of external SOAP (Simple Object Access Protocols) requests and the ability to respond to external systems making SOAP requests on VE. At its simplest, SOAP identifies a request XML message schema, a response XML message schema, and the mechanism for the communication of these schemas. Using the previously described XML mapping interfaces, engineers define the structure of both incoming and outgoing SOAP requests and treat the schemas involved in these requests as if they were native VE packages and attributes. Again, the business rules layer performs all the mechanics necessary to construct, interpret and dispatch SOAP requests.

[0052] Web Services: Visual Enterprise supports both the consumption of external web services and exposing Visual Enterprise processes as web services. The consumption of an external web service is essentially the same as making a SOAP request. The only difference is that an engineer can use the composer to search a UDDI (Universal Directory and Discovery Interface) registry to find the required web service.

[0053] In certain alternate embodiments of the method of the present invention, a user, engineer, or business process engineer may integrate a first business process software program with a second business process software program, wherein the first business process software program and the second business process software program communicating via a computer network. Preferred embodiments of the present invention of this type may comprise (1.) interrogating the second business process software program via the computer network to determine how a message containing data may be structured in order to enable the second business process software program to accept the data, (2.) detecting a set of data that the first business process software program may provide to the second business process software program, (3.) structuring a message for transmission to and acceptance by the second business process software program, the message containing the set of data, and (4.) transmitting the message to the second business process software program via the computer network.

[0054] In an alternate embodiment of the present invention, a system and method for application development is implemented. An application development environment, optionally a runtime environment, is provided to allow a business process engineer to define and document business processes, data definitions, business rules, transactional, organizational and calendar information. The business process engineer accesses the application development environment, the VE Composer, via a network using a network browser software, or browser to develop applications using one or more business processes consistent with the data definitions, the business rules, workflow and policies and the calendar, event, trigger and organizational information without programming, coding or scripting.

[0055] The following detailed description sets forth numerous specific details to provide a thorough understanding of the invention. However, those of ordinary skill in the art will appreciate that the invention may be practiced without these specific details. In other instances, well-known methods, procedures, protocols, components, algorithms, and methods have not been described in detail so as not to obscure the invention.

[0056] A first alternate preferred embodiment of the present invention, or VE, provides business process engineers with an environment, optionally a runtime environment, to define and document business processes, data definitions, business rules and a calendar of events. The VE comprises an environment that includes the VE Composer (see FIGS. 7 and 8). As the Composer has a Web-based browser interface, the business process engineer can use VE from anywhere. All the business process engineer requires is a computer with access to the Internet.

[0057] The information recorded by the engineer is used directly to construct the solution, without the need to translate from specifications to source code and without the need for a parsing or compilation. The VE Composer allows the engineer to structure and layout components in a fashion that models the way in which the organization might previously have documented the organizations processes and rules. All of this information can be viewed by management or other interested parties in a transparent manner, as opposed to being hidden away somewhere in computer code.

[0058] Further, the VE Composer provides a graphical user interface (GUI), delivered over a network, that allows the business process engineer or person to record all the information pertinent to an organization and relevant procedures and policies, including, but not limited to, organizational structure, calendaring, workflow, rules, organization data definitions, delegation of authority, security and corporate legal structure. The information is used to develop collaborative business process applications and transaction based systems, such as supply chain or customer relationship management, for the organization. The information is used to model the business in a fashion that reflects the way the organization establishes workflow processes and business rules without need for a technical build process (e.g., parsing, compilation, coding or programming). The resultant solution can be viewed by management or other interested parties in the organization in a transparent manner, as long as they have the security to do so.

[0059] The VE Player is optionally the environment by which management, staff or other authorized persons of an organization can access the business solution developed by the business process engineers (see FIGS. 5 and 6). The VE

Player interface is entirely Web-based, and is accessed via a pre-allocated URL (for example, http://ve.orgname.com). The home page at this address requires the user to supply a login name and password in order to gain access to the site. An Administrator with the appropriate authority creates a login name to grant access to new users. The initial password is allocated by VE and communicated to the user via e-mail. The first time the user logs into the site he is required to select a new password, which will then be used for all subsequent access to the site.

[0060] One of the main optional features of the VE Player is the VE Enterprise Interpreter™, which is what translates the model into the application solution. The Enterprise Interpreter is a process interpreter and an engine that drives the execution of the processes, activities, attributes and rules of a VE application. The Enterprise Interpreter can respond to actions taken by a user, such as submitting an activity for completion, initiating a scheduled activity at the required time, or responding to a request from an external process and do so in real time. Once an activity is requested, the Enterprise Interpreter constructs the page, constructs and executes the business logic, determines what database calls are needed, makes any external calls necessary and then executes the process and runs the application. To do this, the Enterprise Interpreter includes a grammar engine that allows the VE Player to automatically create the software code necessary to run an application without syntax or scripting errors.

[0061] The Enterprise Interpreter is a multi-threaded environment meaning that the Enterprise Interpreter can handle requests or tasks from multiple users and other sources at the same time. Once that the Enterprise Interpreter has serviced a request, that the Enterprise Interpreter will send a response back to the client, including an indication of whether the request was serviced successfully or not.

[0062] FIGS. 1 and 7 are examples of process models built with the VE Composer. The elements (processes and activities) are related to one another in a hierarchical structure and are used by the business process engineer to develop applications. Entities are organizations for which the applications are to be developed. The Engineers are the business process engineers who develop the applications. The Users include management, employees, customers, vendors and partners or other interested parties of the organization who have been granted secured access to use the applications developed by the business process engineers. Groups include positions or roles within the organization against which secured access to a particular process or activity is granted. The Organization calendar (see FIG. 11) provides time zone, working hours, public holidays, and reporting period information as well as other time-related events relevant to the organization. The Organization Dictionary is a template of packages and attribute properties, which is used to create single instance definitions that enable standards to the information collected by the organization and used across application boundaries.

[0063] An application (see FIGS. 17 and 21) may be a collection of business processes, data definitions and business rules, which together reflect how an organization or particular section of an organization operates. The attribute library (see FIGS. 15, 18 and 19) is a library of data attributes, which define properties of and relationships between pieces of information collected by the organization. A process is a collection of related processes used to assist users in identifying the correct process to initiate. The process is a series of operations performed to complete a particular function of the organiza-

tion. The activity (see FIG. 13) is a single operation in the process. The operation may be defined by the business process engineer as to be completed either by a user or to be automated using, for example, a batch process. The data attribute, or element, is a piece of information collected by the organization and is an element of the attribute library. For example (not to be limiting, but for purposes of clarity), information may be a customer name or an invoice number.

[0064] One important factor common to many business solutions is to provide a consistent user interface. Consistency can be facilitated through the ability to define a template of attribute definitions. The attribute definitions allow the business process engineer to define properties of an attribute (e.g., data type, default value, whether or not it is mandatory for a user to record a value for the attribute, etc.) Some attribute properties may be specific to a particular data type (e.g., the number of decimal places, etc.), which is specific to numeric attributes. Some attribute properties, while applicable to all data types, may have options that are dependent on the data type. For example (not to be limiting, but for purposes of clarity), the options applicable to the format properties of a numeric attribute are different from the options applicable to a date or time-span attribute.

[0065] Attributes defined within each organization's template can be grouped into categories for easy reference. When the business process engineer is creating attributes for a particular application, the attribute's definition can be based on an attribute template. In this way, the attribute inherits properties from the template. The business process engineer may choose to override one or more of these properties. Any attribute properties that are not overridden will continue to be inherited from the template. For example (not to be limiting, but for purposes of clarity), a "Currency" attribute template may be defined in the organization dictionary with the following properties:

Name	Currency
Data Type	Number
Control	Data Field
Default	0
Mandatory	No
Decimal Places	2
Display Width	10 average characters

The business process engineer may then create, as part of the application, a "Discount" attribute, which is based on the "Currency" attribute template. All of the properties of the "Discount" attribute are inherited from the "Currency" attribute template. A further attribute "Price" may be defined and based on the "Currency" attribute template. However, in this case the engineer may choose to override the Mandatory property to "Yes" and the Display Width property to 12. All other properties are inherited from the currency attribute template. If, at a later point in time, the Display Width property of the "Currency" attribute template is changed to eight (8) then the "Discount" attribute will inherit this new value. The "Price" attribute will not inherit the new value because an old value was overridden. The use of the organization dictionary can greatly reduce the time required to define each of the attributes required for an application. It also means that a later change in requirements relating to the properties of a group of attributes (such as increasing interest calculations from two

(2) to three (3) decimal places) can be effected with no programming, coding of scripting.

[0066] Each of the attributes may have a number of attribute properties, which dictate the functionality or rules associated with that attribute. For example, the business process engineer can specify that the value of an attribute is selected from one of a list of options. These attribute properties may include:

[0067] Data Type: text, number, date, etc.

[0068] Control: This property determines the way in which the value of the attribute is collected on displayed on the page.

[0069] Control Options: this includes Data Field, Radio Buttons, Check Box, and Drop-down List. Depending on the value of other properties some of these options may not be valid.

[0070] Options List: This indicates a list of options from which the user can choose.

[0071] Extendable Options: This indicates whether the user can add additional options to the above list.

[0072] Default: This indicates a default value for the attribute.

[0073] Mandatory: This indicates whether the user must supply a value for the attribute.

[0074] Editable: This indicates whether the user can modify the value of the attribute.

[0075] Visible: This indicates whether the attribute's value is displayed on the page.

[0076] Report Name: The heading or label given to the values of the attribute on any report.

[0077] Description: This is a short description of the attribute, which is provided to a user when the mouse is positioned over the attribute.

[0078] Minimum and Maximum Values: These properties are equivalent to a business rule that verifies whether the value entered by a user is greater than a minimum or less than a maximum value.

[0079] Minimum and Maximum Length (Text Only): These properties control the minimum or maximum length of text that a user is permitted to enter.

[0080] Decimal Places (numeric only): This indicates the number of decimal places to which the attribute's values are to be stored.

[0081] Units (numeric only): miles, gallons, dollars, etc.

[0082] Background and Foreground color: Foreground color is the color of the text entered by the user while background color is the color behind that text.

[0083] Input Format: This property controls the acceptable format for a value entered by the user. For example, an input format that specifies two decimal places would prevent the user from entering more than two decimal places.

[0084] Output Format: This property controls the format in which values are displayed. For example, an output format that specified one decimal place would display the value with only one decimal place, even though value may be stored to two or more decimal places.

[0085] Display Width and Units: These properties control the width allocated for an attribute on the page. The width refers only to the attribute value and not the associated label. The units can be specified as pixels (if extremely accurate sizing is required), inches, centimeters and average character width (proportional fonts mean an exact character width cannot be determined).

[0086] This attribute library details the attributes that are collected by the organization as part of relevant operations. For example (not to be limiting, but for purposes of clarity), the organization may collect information about customer names, addresses, phone numbers, purchase order numbers, product names, quantities, etc. Each of these attributes can be grouped into packages of related attributes. For example (not to be limiting, but for purposes of clarity), a "Customer" package would be a collection of attributes related to a customer, including for example, the customer's name, date of birth, gender, etc. A package may also include a number of sub-packages that are also related in some way. For example, an "Invoice" package may contain an "Invoice Item" sub-package. The "Invoice" package would contain attributes for the invoice number, date, etc., while the "Invoice Item" sub-package would contain attributes for the name, quantity and price of the product being purchased. Attributes in the attribute library may be based on an attribute from the organization's template or they may be defined from scratch. An attribute that is based on a template attribute may have the same name as the attribute on which the attribute is based or the name can be changed as required.

[0087] The hierarchical structure (as defined in the Explorer—see FIG. 3) allows the business process engineer a structured method to access information in the organization. For example (not to be limiting, but for purposes of clarity), to view details relating to an activity, the business process engineer chooses the application, then the process category and the process before being able to select the relevant activity.

[0088] When the organization registration is complete, the administrator and any other authorized business process engineers are sent an e-mail message containing an allocated organization ID, login name and a temporary password. After this message is received, the business process engineer can log into the VE Composer web site. The first time the business process engineer logs into the VE Composer web site, that business process engineer may be required to select a new password, which will be used for all subsequent access to the site. Upon verification of the business process engineer's details, the business process engineer is directed to the VE Composer home page. This page is divided into three distinct areas, including the VE Composer Explorer (see FIG. 8), a process workspace, and a menu bar.

[0089] FIG. 7 is a block diagram illustrating components via the VE Composer Explorer that are available in the composer to allow the business process engineer to develop applications for an organization in accordance with one embodiment of the present invention. These components include the process workflows (see FIG. 9), the attributes (see FIG. 15), the business rules (see FIG. 10) and the organization calendar (see FIG. 11). In addition to the above components, other components are also available in the composer and may be used by the business process engineer. For example (not to be limiting, but for purposes of clarity), the VE Composer includes an administration component (not shown) to provide security or access control relating to the business process engineers and the users.

[0090] A process workflow includes applications, processes, and activities. To support the applications, the processes, and the activities, the business process engineer design information structures (e.g., data structures) necessary to record and retrieve the attributes. For example (not to be limiting, but for purposes of clarity), an invoicing process

needs an invoice, and the invoice includes data such as, for example, invoice number, invoice date, line items, amounts, etc.

[0091] The attributes are defined by the business process engineer and can be accessed from any authorized activity in any process workflow. Organizational attributes are defined once and used many times, where specific application packages and attributes are used for specific application and process solutions. An attribute is a piece of information collected by the organization. Each attribute has a number of associated properties such as, for example, an attribute type (e.g., a date, a number, text, etc.), how the attribute is to be formatted on a page and in reports, input control options, etc. The attributes are grouped together into packages that clarify the use of one attribute in relation to the other attributes as described above. The values of an attribute (e.g., a customer's name) are recorded using the processes and the activities defined by the business process engineer. A database maintains the complete collection of the values of the attributes. Each value in the collection represents a single piece of information. For example, each value of the "customer name" attribute represents one customer's name. There is no distinction as to the context of an attribute except for the activity that references that attribute. For example, one activity may be defined to record the details of all customers, while another activity may be defined to record each customer's purchases. The collection of values of the "customer name" attribute may include a value for each recorded customer as well as a value for each time a customer has made a purchase. The attributes may be constrained by the business rules to reflect values allowed by, or imposed upon, the organization. For example (not to be limiting, but for purposes of clarity), valid invoice numbers have to be in a certain range or have certain predefined characteristics.

[0092] The business rules relate to the capture and maintenance of attributes within context of processes and govern the way the organization wants to carry out appropriate operations. The business rules are designed by the business process engineer to ensure that an application follows the policies of the organization. The preferred embodiment incorporates and advanced rules engine having 4 elements to the rules structure: Categories, constants, functions and properties and is designed to create any type of rule, including but not limited to mathematical, trigonometric, statistical, comparatives, numeric, error handling, date/time (down to 1 second intervals), escalation, event, security, and time span rules. For example (not to be limiting, but for purposes of clarity), there may be business rules designed to assign default values to the attribute, and there may be other business rules designed to monitor the attribute such that the attribute has valid values and to warn the users when the values are not valid. Another example, a rule can be applied to an activity such as calculating a result dependant on variables, or a rule can trigger an event or launch new processes.

[0093] The organization calendar provides calendar information including time zones, business/working hours, holidays, reporting periods, etc. that the organization operates within. The business rules that enforce time constraints operate consistently with the organization calendar. The organization calendar is independent of the process workflows in order to support the organization with offices in different states, times zones, or countries. This allows a single version of a process workflow to operate, for example, in all of the organization's divisions worldwide without modification.

The process workflows, the attributes, the business rules, and the organization calendar are described in more details in the following sections.

[0094] FIG. 7 is a block diagram illustrating application and process workflows in accordance to one embodiment of the present invention. As described above, process workflows include applications, processes, and activities. Within the application are the processes and activities. The processes include a collection of related activities. Each process includes one initiating activity. The initiating activity may be initiated either by a user, by an event, or by a remote activity such as another application or a web service. For example (not to be limiting, but for purposes of clarity), a daily backup batch process is to run at 10:00 p.m. each night. The process may be constrained by one or more business rules and determines the order or path in which the activities are to be performed. For example (not to be limiting, but for purposes of clarity), some of the activities may be allowed to perform in parallel, while others may not be able to perform until all preceding activities on an relevant path have been completed.

[0095] Process category allows grouping or filing of related processes. An organization may have a collection of processes defined by the business process engineer. The process category provides a mechanism to allow the business process engineer and the users to organize and to locate the processes without having to search the entire collection of defined processes. Using the process categories, the business process engineer can define a filing hierarchy to organize the processes. For example (not to be limiting, but for purposes of clarity), a business process engineer who is developing a sales application for an organization may define the following process category hierarchy:

   [0096] Customer Relationship Management
      [0097] 1. Marketing Campaigns
      [0098] 2. Telemarketing
   [0099] Transactions
      [0100] 1. Purchasing
      [0101] 2. Sales
      [0102] 3. Adjustments
   [0103] Maintenance
      [0104] 1. Customers
      [0105] 2. Suppliers
      [0106] 3. Staff
      [0107] 4. Departments
   [0108] Reporting
      [0109] 1. Weekly Reports
      [0110] 2. Monthly Reports
      [0111] 3. Year to Date Reports
      [0112] 4. Annual Reports
      [0113] 5. Ad Hoc Reports

Using the above process category example, all processes relating to purchasing may be organized under a purchasing category, etc. To access or to file a process in the purchasing category, the business process engineer simply needs to select the transactions category followed by the purchasing category. Similarly, a user who has been granted secured access to the purchasing process may initiate the purchasing process by first selecting the transactions category.

[0114] Process definitions relate to the grouping of related defined activities to form a process. An activity is a collection of one or more related attribute packages designed to add, update and/or query. Rules may be associated with an activity in order to validate the attribute or to perform calculations on the attribute. An activity may result in succeed or fail status.

The process uses the status of a current activity to determine which activity or activities are to follow on from the current activity. Each activity may be associated with an access authority. For example, a manager in the organization has default access to an activity that involves approving a loan application. The activities that form a process are linked or joined together according to a workflow defined by the business process engineer. FIGS. 1, 4, 5, and 6 illustrate examples of a loan approval for both the VE Composer and the VE Player. Each block in the loan approval process represents an activity. A directional arrow between two activities represents direction of the workflow. In the graphical user interface of the composer, each block in the loan approval process in FIG. 1 is illustrated using an icon, and the business process engineer can drag and drop activity icons and link these activity icons together to form a process or series of processes such as a message, report, email, automated process, user defined or launched process. For example (not to be limiting, but for purposes of clarity), if the customer equals "IBM" then launch the process called "IBM."

[0115] A remote activity type may comprise a process wherein the attribute is retrieved from or submitted to an external system, remote database or web service as part of the operations to be included in a process or activity. The external system may be of any type, including but not limited to, a business-to-business event, a business-to-customer event, a long-lived collaborative process, or a transaction based system. A maintenance activity type is where the user can add, modify or delete attributes in circumstances that may not fall within any formalized process but which, nonetheless, needs to be maintained. An example (not to be limiting, but for purposes of clarity) may be to update a customer's phone number that may have been entered incorrectly. Maintenance activities may not be used as part of a standard business process with other activity types.

[0116] Referring to FIGS. 12 and 16, rules information provides a list of rules relevant to the activity, which need to be enforced by an administrator. This is referred to as activity rule. Time criteria information provides properties that dictate any applicable time constraints (e.g., a completion deadline, etc.) that are placed on the activity. Access information provides a list of groups and/or users to whom the activity has been granted access. The groups or users granted access to the first activity in a process (e.g., the initiating activity) determine the users who have the authority to initiate that process. Initiation information includes properties that indicate how to determine the user or users to whom the activity should be allocated. E-mail information provides details relating to the message that is to be sent. Report information provides details about the report that is to be generated. Automated information provides details about import or export file names and formats, etc.

[0117] Activities that are defined with the remote type (referred to as remote activities) allow a process to access attributes or functionality external to the application. This includes, for example, gaining access to legacy systems, external components (COM+ or EJB), middleware and other organizations' business solutions. Using SOAP (Simple Object Access Protocol) as an underlying infrastructure, external attributes and functionality can be incorporated seamlessly into a process, as if it was part of the application. SOAP is an eXtensible Markup Language (XML) based protocol that consists of an envelope that defines a framework for describing what is in a message and how to process it, a set of

encoding rules for expressing instances of application-defined attribute types, and a convention for representing remote procedure calls and responses. The information transported via SOAP between the application and the remote system includes an XML schema definition that defines the list of packages and attributes (universally identifiable) selected by the business process engineer as part of the remote activity. Other information transported via SOAP includes business rules applicable to the activity that defines conditions to be imposed on the remote activity, and attributes sent to and received from the external system. An acknowledgement scheme is used to indicate success or failure of the operation. For example, the organization may have a legacy system that maintains information of all staff working for the organization. In this case, a process can be developed to allow a user to access that information in conjunction with other associated attributes. The organization may have a legacy system that maintains a general ledger. In this case, activities can be developed to post transactions to that system as part of the processes developed to raise purchase orders, invoices and receipts. When the organization wants to conduct business-to-business transactions with a supplier to order stock, a process can be developed to allow the organization to place an order with the supplier. There may be business rules associated with the order that process has to follow. For example, a rule may state that if the cost is above a predetermined limit then the supplier must seek authorization with the organization before proceeding with the order. The authorization may be obtained by an automated initiation of, for example, an authorization activity. When the supplier is authorized for the cost of the order, that authorization would be transported via SOAP to the supplier whereupon the order can proceed. Alternatively, when the supplier is not authorized, the order is cancelled. It is understood that for the remote activity to operate, the remote system or component also supports the SOAP infrastructure.

[0118] A remote activity may be associated with remote attributes. A remote attribute is an attribute used to hold the attributes transmitted to or from the external or remote system as part of a remote activity. Some remote attributes may need to be stored in a local database while others may not. For example (not to be limiting, but for purposes of clarity), attributes defined for customer details stored in a legacy system may not need to be stored in the local database as this would only duplicate the attribute for no additional benefit. On the other hand, attributes defined for a purchase order that is used in a business-to-business transaction with another organization would need to be stored in the local database.

[0119] The business process engineer defines the one or more process views of a process. The process view enables the information included in each activity of the process to be tailored according to the user who has access to that activity, for both workflow and security reasons. In defining a process view, the business process engineer defines two components of a view, activity-attributes component and access component. The activity-attributes component is a selection of the activities to be included in the process view, and for each activity, which attributes should be visible to a user. The access component is a list of groups and/or users who have been granted access to the process view. It may be noted that being granted access to a process view is different from being granted access to an activity, and may include a digital certificate (signature) as deemed appropriate by the policy of the organization. Having access to a process view gives a user

access to details about the history of a process as the process relates to the activities included in the process view. Having access to a process view may not allow the user to complete anyone of those activities. Having access to an activity, on the other hand, does gives a user the authority to complete that activity no matter which process view is being used.

[0120] Every process may have one default process view, which includes every activity in the process and the attributes associated with each activity. A user may have access to more than one process view of a process. In this case, the user is given the option about which process view to use when initiating a process or opening an activity (see FIG. 5). A process view does not affect the links between activities, the rules associated with an activity, the time constraint imposed on an activity, or other properties assigned to an activity. The process view can significantly reduce the number of processes and activities that would otherwise be required to develop an application.

[0121] One of the defining differences between the present invention and existing application development environments are the way relationships between various attributes are defined. Instead of using a standard relational database model, the preferred embodiment uses a meta-data repository where attributes and the relationships between attributes are defined by the context in which the attributes are referenced (e.g., packages, activities, and processes). This affects both on the storage of data and the way such data is retrieved. For example (not to be limiting, but for purposes of clarity), using the invoice activity example, with the attributes "Invoice Number Invoice Date", "Customer Name", "Customer Phone Number" (for display only), "Customer Address" (for display only), "Multiple invoice item" attributes. This activity might be used to enter a new invoice. It may be necessary to be able to determine how and when to retrieve the customer's phone number and address. This is done through the definition of packages and key attributes.

[0122] Key attributes refer to attributes that are unique. There may be one or more attributes within a package or within an activity that are unique. Using the above example, the business process engineer would specify that the invoice detail attributes were part of the main or controlling package. The invoice item attributes would be part of a separate "Invoice Item" repeating package to allow multiple items to be entered. The "Invoice Number" would then be defined as the key attribute for the main package because the "Invoice Number" is unique. The customer package and therefore the customer attributes (e.g., name, phone number and address) would be linked to the main package. This would indicate a one-to-one relationship between a customer and an invoice.

[0123] A unique attribute in one activity may be used as key attribute in other activities. This is because the unique attribute indicates that the result set for any related attributes, in a one-to-one relationship, will contain a single value. For example (not to be limiting, but for purposes of clarity), within an activity to record customer details, both the "Company Name" and "Business Number" attributes may be unique. This means no two companies can have the same name and no two companies can have the same business number. When defining activities that reference customer details, for example, the business process engineer should be able to interchange the company name with the business number when required. In effect, the "Business Number" attribute is an alternate key for the "Company Name"

attribute, and the "Company Name" attribute is an alternate key for the "Business Number" attribute.

[0124] An attribute, defined in an activity as being unique, may be an alternate key for all of the other unique attributes defined in that same activity. This alternate key relationship extends beyond the activity in which the attributes are defined as unique to any activity in any process.

[0125] A search list of activities where both the base attribute and its key attribute are referenced within the same activity can be formed to retrieve data. The search list can be extended to include any activity where the base attribute and an alternate key attribute are referenced. Using the above example, when a "Customer ID" is to be added to the list of customer related attributes, one activity may be defined to collect the "Customer Name" and the "Customer ID" attributes only (both attributes defined as unique keys). A second activity may be defined (either within the same process or in another process) to collect the "Phone Number" and "Address", with the "Customer ID" used as the alternate key attribute. In this case, neither the first activity nor the second activity references all of the attributes "Customer Name", "Phone Number" and "Address". However, for the purpose of retrieving the phone number and address, the alternate key of the "Customer Name" attribute (i.e., the "Customer ID") can be used as the link between all of the attributes.

[0126] The above approach of using unique key and alternate key to data retrieval imposes a methodology on the business process engineer that involves retaining a consistent definition of all attributes across all activities and processes. For example (not to be limiting, but for purposes of clarity), if a number of address attributes (e.g., street, city, postcode, etc.) were referenced in one activity as a postal address and in a separate activity as a property address, no distinction is recognized. Separate attributes would need to be defined for postal and property addresses so that the definition of these attributes are consistent in all activities. Thus, in the first scenario an "Address" is sometimes a postal address and sometimes a property address. In the second scenario a "Postal Address" can only be a postal address, and a "Property Address" can only be a property address.

[0127] For data consistency, only the latest submitted version of an attribute's value is stored. This means that one process may modify the value of data created by an activity of a second process while that second process is still in progress. For example (not to be limiting, but for purposes of clarity), suppose an invoice entry process consists of two activities, one to enter the invoice details and another to approve the invoice, to be performed by two different users. When a user completes the first activity, the invoice details are submitted to the database. Before the next user (e.g., a supervisor) has the chance to approve the invoice, a third user (e.g., a manager) may run a separate process to modify invoice details. If this manager submits changes to the same invoice before the supervisor begins the approval activity, then these changes will be presented as the invoice details to be approved. If the supervisor has commenced the approval activity but not submitted when the manager submits the changes to the invoice detail then these changes will not be presented to the supervisor. Notification is generated (to a supervisor) to indicate that the data has changed so that the data can be refreshed before approving the invoice. If the supervisor had approved the invoice before the manager tried to modify the invoice details it would be left to the business process engineer to decide whether this was a valid activity or not. It may be a

business rule of the organization that approved invoices cannot be modified but can only be cancelled.

[0128] When referencing an attribute of a shared relationship in a business rule, the attribute is to be fully qualified to include the relationship. For example (not to be limiting, but for purposes of clarity), an organization may have a business rule that indicates the offices in which staff work must have a phone number. However, this rule may not apply to offices in which meetings are held. In this case, the following business rule, for example, would be too restrictive:

[0129] Verify that the Office has a Phone Number.
The rule would need to be qualified such that the relationship between "staff" and "office" is included, for example:

[0130] Verify that the Staff Office has a Phone Number.

[0131] When referring to an attribute (e.g., in a business rule), the business process engineer may refer to a single value or a set of values depending on the context. For example (not to be limiting, but for purposes of clarity), in the context of a sales order, when the business process engineer refers to a customer's name, then the reference is to a single customer. However, when the business process engineer describes the customers who may place an order, the business process engineer may be referring to multiple potential customers' names.

[0132] In a natural language, the distinction is made by using the plural form of the noun "name" to distinguish between a single name and a set of names. The description above also uses an apostrophe to indicate a specific reference to the name of a customer as opposed to the name of a staff member or supplier. "Customer" would be classified as a package of attributes and "Name" as an attribute. There is no distinguishing, in terms of syntax, between a single value and a set of values. All attribute values will be part of a set such that a single value is simply a set that contains only one item. This set of values is referred to as an attribute result set.

[0133] The packages have a number of package properties, which describe the behavior of the packages. The package properties may include, but is not limited to:

[0134] A list of key attributes, which together or in isolation can be used to uniquely identify an instance of the package. For example (not to be limiting, but for purposes of clarity), a customer's name and telephone number might uniquely identify each customer. This information is critical in ensuring that when a user chooses a particular entity that there can be no ambiguity about that choice.

[0135] A list of search attributes, which are used to assist users when making a choice from a list of available package instances. For example (not to be limiting, but for purposes of clarity), to assist the user in selecting a customer, the engineer might choose to display the customer's name, date of birth, social security number, phone number and address. By defining all these attributes as search attributes enables the user to choose which criteria to use for the search, e.g. social security number.

Further, where a container package is related on a one-to-many basis with a sub-package, additional properties are maintained on the sub-package. These additional properties may include:

[0136] The order in which the package instances is displayed. For example (not to be limiting, but for purposes of clarity), invoice items may be displayed in product code order or in order of price.

[0137] A minimum or a maximum number of instances. For example (not to be limiting, but for purposes of clarity), an organization might require an invoice to be limited to 10 items or fewer, or not to be limited at all.

[0138] Similar to the attributes having attribute relationships, packages also some relationship to other packages. This is defined in package relationship. In some cases, a number of different packages and the attributes contained therein have a relationship in common with another package. For example, a staff member is related to the office in which he works. A purchase order is related to an office from which that order originates. A meeting is related to an office in which that meeting is held. In this case, the "office" package should not be explicitly defined as a sub-package of each of the "staff", "purchase order" and "meeting" packages as it would not identify the common or shared relationship that exists in each case. Rather, "office" needs to be defined as an independent package so that it can be referred to, as required, across many activities within various processes.

[0139] In certain alternate preferred embodiments of the method of the present invention, packages that are the subject of shared relationships are first defined as packages independently of the packages to which they are related. When the definition of these packages is done, they can be linked to these related packages. Alternatively, these packages may be defined implicitly as shared relationships within an activity, based on the fact they are referenced in common. Shared or linked relationships, by default, are defined as "one-to-one" relationships. This can be changed to a "one-to-many" relationship by setting a property of the package.

[0140] A package result set is a set of values for each attribute in the package. Each of these sets contain the same number of values, although some of the values may be null or not specified. For example, a customer package may consist of a name attribute and a phone number attribute. For each value of the name attribute in the customer, result set there is a corresponding value for the phone number attribute and vice versa. This does not mean that every customer needs to have a phone number. For those that do not have a phone number the attribute value will be null (e.g., empty, blank). A fully qualified attribute name is one that includes all of the packages to which an attribute belongs. For example (not to be limiting, but for purposes of clarity), "Customer Address City" where City is an attribute of the Address package, which is a sub-package of the Customer package. To display fully qualified attribute names in business rules would overly complicate the reading of a rule. If there can be no ambiguity over the package to which an attribute belongs then only the attribute name is displayed, not the fully qualified name. If there may be ambiguity, for example because there are two "City" attributes, then it may be necessary to qualify the attribute name sufficiently to remove any ambiguity.

[0141] A single attribute may be referenced in any number of different activities and processes. When retrieving data, the context in which an attribute is referenced is used to determine the retrieval criteria, and held in the meta-data repository. In order to retrieve data, a relationship list is determined for each attribute within a process. This relationship list is based on explicit relationships that the business process engineer may have defined, along with any implicit relationships in the process definition. This defines a relationship between the attributes of both packages.

[0142] When searching for related attribute values, more than one instance may be found that matches with the key

11

value. This set of values forms the result set for the related attributes, as described above. The attribute result set may or may not be appropriate for the definition of the activity within which the attribute is being retrieved. For example (not to be limiting, but for purposes of clarity), a multi-value result set cannot be displayed in a single data field, however it can be displayed in a list or table. In this case, an exception would be raised.

[0143] Every activity that allows the user to add or modify attribute values, or that uses business rules to add or modify attribute values does so via one or more data transactions. The transactions are initiated only after the activity has been validated using the associated business rules. A transaction is initiated for each package included in an activity. The transactions are all applied together as a single activity transaction such that the integrity and consistency of the whole activity is maintained. If a single package transaction should fail for any reason then the transactions for all packages in that activity will be reversed. The transactions do not delete data from the database. This is because of the inherent problems associated with deleting data that has already been referenced by other activities. Instead of deleting the data, the data is inactivated such that it cannot be referenced in other activities. The business process engineer thus does not need to be aware of these transactions because they are automatically controlled and managed.

[0144] FIG. 16 is a dialog box illustrating different types of business rules and their relationship with one another in accordance to one embodiment of the present invention. Business rules are the rules, regulations and constraints that govern the way the organization carries out its operations. The business process engineer relies on the business rules that are in some way related to, or impact, the business processes being developed. For example (not to be limiting, but for purposes of clarity): "the discount must be less than 10%", or "a purchase order form must include at least one item", etc. The business rules are satisfied at all points in time unless a condition is applied that specifically limits the time-span during which the business rule applies. For example, a discount may be offered only on invoices raised between Christmas and New Year. This business rule will hold true at all points in time, subject to the condition that the discount is only offered between Christmas and New Year. In other words, any attempt to offer a discount outside of those time constraints is rejected.

[0145] The business rules may include attributes business rules to apply to attributes, activity business rules to apply to activities, package business rules to apply to packages, and core business rules to apply to an entire application. The attribute business rule may be dependent on the definition of the rule and on the definition of the attributes on which the rule is based. The activity business rule may be dependent on the definition of the activity in the context of which the rule is executed. The package business rules and the attribute business rules may be automatically associated with each activity in which the attribute or attributes are referenced. The business process engineer may elect to turn off or override such a business rule where it is not applicable to a particular activity.

[0146] One or more attribute business rules may be applied when an attribute is submitted to the database. For example, the attribute may be submitted to the database in an "update" or in a "modify" activity. Depending on the value of the attribute, the attribute business rule can validate or invalidate the attribute. An attribute business rule such as, for example,

"the customer's age is the difference between today and the date of birth" may not be recalculated every day. However, as soon as the age is referenced by an activity, either to display the age to a user or to use the age in some calculation, the attribute business rule will be executed to ensure that the value is correct.

[0147] The activity business rules are executed within the context of an activity. The attribute or attributes to which the activity business rule refers may or may not be included in the activity. For example, an activity business rule may be defined against a "Raise Purchase Order" activity, which sets the price of the products being purchased. The activity business rule may be:

[0148] Assign the Order Item Price=the Product Unit Price

The "Product Unit Price" may not be one of the attributes listed for inclusion on the activity. This is because the "Item Price" will be visible and to display both attributes would duplicate the price information. If, however, selection of the price is dependent upon criteria, such as the date, then these criteria will need to be specified to retrieve the correct result set for the product price. If a history of prices existed for each product then criteria would need to be added to the rule to uniquely identify the correct unit price, as shown in the following business rule example:

Assign the Order Item Price=the Product Pricing History Unit Price, where today is between the Product Pricing History Start Date and the Product Pricing History End Date. When an activity is opened, any attribute that has already been identified is retrieved from the database. The attribute may have been identified either because it was recorded in an earlier activity or because it has been set by an activity business rule. The attribute may also need to be retrieved from the database at other points in time during the completion of an activity. For example (not to be limiting, but for purposes of clarity), such a situation may occur when the user selects a package instance (e.g., a customer) and the activity definition includes a list of attributes related to that customer (e.g., Name, Phone Number, Date of Birth, etc). An activity business rule may also reference attributes not contained within the activity in order to perform some verification or calculation.

[0149] A business rule may be defined to include one or more rule statements. When there are multiple rule statements, all of the statements need to be consistent in type. For example (not to be limiting, but for purposes of clarity), a verification statement cannot be defined in a business rule that also includes an assignment statement. Some business rules may also be specified implicitly via an attribute property rather than explicitly as a business rule statement. For example, a rule that an invoice number can be no longer than seven (7) characters may be specified as a "Maximum Length" property of an attribute. For this reason, business rule functionality may be provided to the business process engineer as a property of an attribute and/or as a business rule statement. The fact that a business rule appears as a statement in this document does not preclude that the functionality may be provided exclusively as a property.

[0150] There may be different types of rule statements. For example (not to be limiting, but for purposes of clarity), the rule statements may include:

[0151] Logic statement (Passive rule type):

[0152] Logic statements include variants such as "if . . . then . . . otherwise", and "based on the value of . . .

if it equals . . . ". Repetition variants such as "while . . . do . . . " or "repeat . . . until . . . " are also available.

[0153] Verification/Validation/Restriction statement (Passive rule type):

[0154] Verification or validation statements test a condition and either report an error or warning or perform some other specified action. A restriction statement restricts the available choices that a user can make to ensure the choice is valid for the particular circumstances.

[0155] Assignment statement (Active rule type): Assignment statements can be further categorized into the following:

[0156] 1. Attribute assignment: assign a value or expression to an attribute.

[0157] 2. Package assignment: set an instance of a linked package based on specified criteria

[0158] Action statement (Active rule type): Action statements include the following:

[0159] 1. "Retrieve" data from the database

[0160] 2. "Initiate" a process

[0161] 3. "Schedule" a process that has been initiated but not yet commenced

[0162] 4. "Initiate" an activity

[0163] 5. "Schedule" an activity that has been initiated but not yet commenced

[0164] 6. "Allocate" an activity to a group or a particular user

[0165] 7. "Create" an audit trail

[0166] 8. "Execute" an external process.

[0167] A business rule may also be defined with one or more error and/or exception conditions. It is understood that an exception condition set may include error conditions. These error and/or exception conditions may include hardware errors, software errors, internal exceptions and external exceptions. The errors and/or exceptions may be raised as a result of the processing of business rules. A hardware error may occur when, for example, a server crashes, when there is insufficient disk space, when a telephone line drops, etc. A software error may occur when, for example, a browser crashes, an operating system hangs, etc. When a hardware or software error occurs, a report of the error is generated to allow whatever remedial action to be taken. If possible, an entry is logged and/or an e-mail message is sent to an administrator detailing the nature of the error. Any further processing of business rules for that activity would be cancelled. The business rules that have already been processed within the transaction during the processing of which the error occurred, would be rolled back and the transactional integrity is maintained for the entire system.

[0168] The business rule exceptions can be described as either internal or external to an activity. An internal exception is one where control of the activity remains with the user allowing the user to modify an attribute or attributes such that the verification rules are met. An external exception is one where the activity is automatically cancelled, and the control flows to an activity scheduler via the failure route for that activity. An exception may be, for example, a database exception where a value of an attribute entered by a user does not satisfy one or more business rules, a time-based exception where an activity is not completed within a specified time frame, etc. Most exceptions can be detected and handled during data entry before the commencement of the processing of a transaction. However, if this is not the case then the entire

transaction will be rolled back to ensure the integrity and consistency of the database. In the case of an external exception, the handling of the exception is taken outside the control of the user to whom the activity was assigned. The business process engineer is able to define a complete workflow to be followed for each defined exception. This workflow may include, for example, an escalation of the exception should further data or time constraints not be met by the allocated user or users. If both an internal and external exception were to occur within an activity, the activity scheduler is allowed to deal with the external exception. Any data entered for the activity in error would be passed to the "Exception" activity, although the data would not be committed to the database. This allows the user handling the exception to either correct that data, where required, or to reset and then re-enter the data.

[0169] The business processes being defined may also require warnings to be issued to users at various stages. This functionality is provided via a variation on the verification/ validation business rule definition. This variation would indicate that a warning is to be issued rather than raising an exception. In this case, processing would continue normally once the warning had been issued. Such warnings would be issued before any data being committed to the database.

[0170] The business rules defined by the business process engineer may sometimes be conflicting or ambiguous. For example (not to be limiting, but for purposes of clarity), one rule may be defined to offer a discount on certain products at a certain time of year. Another rule may be defined to offer a discount to loyal customers. In one embodiment, when conflicting and ambiguous business rules are detected, the business process engineer is prompted to redefine the conflicting rules or to clarify the ambiguity. This detection is performed when the business process engineer decides to publish an application. Rules that apply to the same attribute or attributes are searched. A conflict resolution form is generated and completed before the application can be published. For example (not to be limiting, but for purposes of clarity), consider the following separate business rules:

[0171] If the Order Date is between Halloween and Christmas Eve and the Product Type is equal to Sports Goods then Assign the Discount=10%

[0172] If the Customer Status is equal to loyal then Assign the Discount=5%.

In this example, the "Discount" attribute is assigned to the both the first rule and the second rule, and that the conditions applied in each rule may conflict with one another, resulting in an ambiguity. The business process engineer may resolve any ambiguity by combining the two business rules into one business rule. The ambiguity may be resolved by deciding that the "sports goods" discount takes precedence over the "loyal customer" discount. The business process engineer may combine the two rules and form a combined third rule as in the following example:

[0173] If the Order Date is between Halloween and Christmas Eve and the Product Type is equal to Sports Goods then Assign the Sports Discount=10%

[0174] If the Customer Status is equal to loyal then Assign the Loyalty Discount=5%

[0175] Assign the Discount=maximum (the Sports Discount, the Loyalty Discount)

If the organization wants the discounts to be cumulative then another business rule can be formed as in the following example:

[0176] Assign the Discount=the Sports Discount+the Loyalty Discount

If the organization does not want to have a discount applied to a particular purchase order then the result of the above rule would be a discount of zero.

[0177] The business rules may need to be applied at various stages of an activity or process. In certain circumstances, the business process engineer will determine this stage based on the type or nature of the rule. For example (not to be limiting, but for purposes of clarity), the business process engineer may specify a rule as:

[0178] A means of scheduling an activity (wait until a condition holds true)

[0179] A pre-condition to an activity

[0180] A means of determining the most appropriate user to whom an activity should be assigned

[0181] A means of setting a default value for an attribute

[0182] A condition under which an attribute will be hidden (security)

[0183] A condition under which an attribute will be protected (security)

[0184] A means of reversing an update to an attribute when the value of an attribute on which it is dependent is modified.

There are different stages at which to apply a business rule (taking into account any settings as described above). For example (not to be limiting, but for purposes of clarity), these stages may be at:

[0185] On the commencement of an activity (Pre-activity)

[0186] During an activity (Mid-activity)

[0187] When the user submits an activity for completion (Post-activity)

[0188] On commencement of a transaction (Pre-transaction)

[0189] On completion of a transaction (Post-transaction).

[0190] Pre and post-transaction business rules are defined as being part of the transaction, not external to it, and therefore inherit the same properties as transactional integrity and consistency. When pre-transaction rules are executed, they do so in the context of the previously committed values of each attribute and not the values as specified within the current activity.

The order of processing of an activity transaction is:

[0191] Lock all attribute values referenced by the transaction

[0192] Execute pre-transaction business rules

[0193] Assign package values of each attribute to the database

[0194] Execute post-transaction business rules

[0195] On success commit (and release locks); otherwise roll back.

[0196] Mid-activity business rules are applied whenever a user changes the value of the attribute against which the rule is defined or whenever a user changes the value of any attribute on which that rule is dependent. For example (not to be limiting, but for purposes of clarity), a mid-activity business rule might be associated with a "Date Of Birth" attribute to the effect "The Date Of Birth must be earlier than today's date". This business rule would be applied to the attribute whenever the user changed the date of birth and moved the cursor away from that attribute. As a further example, a package might be established consisting of "Stock Item", "Quan-

tity" and "Cost" fields. A business rule might be associated with the "Cost" attribute to the effect "Cost equals Unit Price of Stock Item multiplied by Quantity". When the user modifies the attributes on which the "Cost" business rule is based ("Stock Item" or "Quantity") the business rule would be applied and the updated cost would be displayed on the screen. The "Cost" business rule is also dependent on the "Unit Price" attribute however in this example this would be a display only attribute and hence could not be modified by the user. Rules applied as mid-activity business rules will be re-applied when the activity is submitted for completion. This needs to be done in case, in the meantime, another user changes the value of any of the attributes on which a rule is based.

[0197] The order in which business rules are executed is determined based on the dependencies of each business rule. For a business rule to be processed correctly it is important to execute that rule only after executing all rules on which the rule is dependent. For example (not to be limiting, but for purposes of clarity), a package may contain four attributes: A, B, C and D (where A, B and C are display only) with the following associated business rules:

$$A=B+C+D$$

$$B=D*4$$

$$C=D/2$$

The order of execution would be set as follows:

$$B=D*4$$

$$C=D/4$$

$$A=B+C+D$$

[0198] Following the execution of any part of a business rule that assigns an attribute's value, further execution of the rule is suspended to allow for the execution of any business rule that is dependent on the attribute being assigned. Any such business rule may itself be suspended for the same reason. Once the business rules of the dependent attributes have been executed, execution of the original business rule is resumed. There is no limit on the number of rules that can be suspended at anyone time. This approach could lead to a cyclical reference scenario whereby execution might never end. To prevent this, a business rule that is currently suspended will not be allowed to execute a second time. The following example is based on an "Invoice Number" attribute which contains a cyclical reference but is nonetheless a valid business rule:

[0199] If the Invoice does not have a Invoice Number then

[0200] Assign the Invoice Number=the Next Invoice Number

[0201] Assign the Next Invoice Number=the Next Invoice Number+1

[0202] otherwise

[0203] Verify the Invoice Number is less than 1000

This business rule is dependent upon both the "Invoice Number" and "Next Invoice Number" attributes. During execution of this business rule, after each assignment statement, execution would be suspended in order to execute any dependent business rules. As a dependent of itself, this business rule would not be re-executed even though in this particular case it

would not result in an endless loop. However, if the rule were rewritten as follows, re-execution would result in such a loop:

[0204] If the Invoice does not have a Invoice Number then

[0205] Assign the Next Invoice Number=the Next Invoice Number+1

[0206] Assign the Invoice Number=the Next Invoice Number

[0207] otherwise

[0208] Verify the Invoice Number is less than 1000

Given that either of the two variants will not be re-executed, the rule would still need to be rewritten if it were to serve the purpose of always verifying an invoice number was less than 1000. Splitting the business rule into two rules as follows would achieve the correct result:

[0209] Verify the Invoice Number is less than 1000

[0210] If the Invoice does not have a Invoice Number then

[0211] Assign the Invoice Number=the Next Invoice Number

[0212] Assign the Next Invoice Number=the Next Invoice Number+1

In this case, during execution of the latter rule, after the first assignment statement, execution of the rule would be suspended and the verification rule would be executed as a dependent. If a rule fails, all business rules and transactions would be rolled back.

[0213] When a business rule contains conditional logic, the condition will not be pre-evaluated in order to determine the correct execution order for the business rules. This is because the evaluated results of the conditions may change as each rule is executed. This may mean that the correct execution order cannot be determined. In these cases, the business process engineer will be given a warning and may elect to rewrite the rules in question in order to remove any ambiguities.

[0214] The business rules are written according to a business rule grammar engine found in the VE Enterprise Interpreter, which is part of the VE Player. The grammar defines the structure and meaning of the rule that can be defined. The grammar also contains a sub-grammar based on a standard algebraic and mathematical grammar. From the grammar, the business process engineer can choose a list of valid options at each point through the process of defining a business rule. When the grammar change the list of valid options will automatically be reconfigured based on the new grammar. Associated with the grammar is a corresponding JavaScript Translation grammar and is what generates the application without syntax errors, being that the grammar engine is fully encapsulated. This grammar defines the valid JavaScript syntax required to translate a business rule into JavaScript just in time for execution.

[0215] The business rules created by the business process engineer is constrained by Business Rule Grammar. The grammar is based on the four principles of validation, calculation, condition and action using a combination of set theory and standard mathematical grammar. The grammar is designed such that it can be expanded and enhanced should the need arise. The engineer also has the ability to specify a time-span during which the rule should be in force. If no time-span is specified then the rule will be permanently in force, or at least until the engineer specifies otherwise. The time-span can be based on specific dates and times, such as for example, from January 2002 until December 2002, or it can be based on organization-based events, such as for example,

from Christmas until New Year. Rules can be categorized according to definitions created by the engineer. For example, a rule may be placed in a Finance category and a Legal category because the organization perceives it to have both financial and legal implications. These categories can then be used as criteria when searching for particular rules. The definition of a rule is not converted or stored as "code" but is stored as data.

[0216] The process explorer (see FIG. 8) provides a simple and efficient means for a business process engineer to create, locate and modify business process definitions. Its structure is based on the hierarchical structure. To locate a process, the business process engineer can either browse through the available applications and process categories or can choose a "find" option. This "find" option will search all the available processes for a process that matches a process name specified by the business process engineer. The process explorer has three modes of operation:

[0217] Organization: This mode allows access to all elements of the Composer, such as Processes, Attribute Library, Organization Calendar, Users, Groups, etc.

[0218] Application: This mode allows access to all elements of the Composer related to a single application, including Processes and Attribute Library. It excludes elements such as the Organization Calendar, Users and Groups.

[0219] Section: This mode allows access to all elements of the Composer related to a selected section of the hierarchy. This could be, for example, the Attribute Library, or a Process Category, including processes, activities, attributes and rules.

[0220] The business process engineer can also specify any organizational relationships in which the user is a party, for example, the name of the person who is the user's supervisor or project manager. The business process engineer, when designing the workflow component of a process, can make use of these relationships. For example (not to be limiting, but for purposes of clarity), a "Leave Approval" activity may need to be completed by the office manager of the user who initiated a "Request for Leave". Similarly, a "Timesheet Approval" activity may need to be completed by the project manager of the user who completed the "Record Timesheet" activity. In these examples, there would be a link flowing from the "Leave Approval" activity to the "Request for Leave" activity, and a link flowing from the "Timesheet Approval" activity to the "Record Timesheet" activity.

[0221] Groups can be used to control access to particular activities and processes for more than one user. The business process engineer can choose to structure groups in ways that best suit the security model used by the organization. The groups may be structured to model the roles that the organization has created, or the groups may be structured according to categories or types of process and can be nested within other groups. For example (not to be limiting, but for purposes of clarity), the organization may define the following groups:

[0222] Purchasing Officer

[0223] Purchasing Manager

[0224] Sales Officer

[0225] Sales Manager

[0226] Chief Financial Officer

The users in each of these positions would then be allocated to the corresponding group. Access to a "Raise Purchase Order" activity, for example, may then be granted to the Purchasing Officer, the Purchasing Manager, and the Chief Financial

Officer groups while access to the Approve Purchase Order activity might only be granted to the Purchasing Manager and the Chief Financial Officer groups. Alternatively, the organization may define the following groups:

[0227] Purchasing Approval Sales

[0228] Sales Approval

Users would then be allocated to those groups that cover the activities relevant to their role in the organization. For example (not to be limiting, but for purposes of clarity), the Purchasing Manager would be allocated to Purchasing and Purchasing Approval groups. Access to the Raise Purchase Order activity would be granted only to the Purchasing group, while access to the Approve Purchase Order activity may be granted only to the Purchasing Approval group.

[0229] Groups can also be defined to belong to other groups. For example, the organization may require three levels of access, Gold, Silver and Bronze whereby users allocated to the Gold group automatically should have access to all processes to which the Silver and Bronze users have access. Thus, users in the Gold group would also be added to the Silver group, and users in the Silver group would be added to the Bronze group. In this way any user added to the Gold group would automatically have access to any process to which a Silver or Bronze user has access.

[0230] Groups can be external to an organization or system. An example (not to be limiting, but for purposes of clarity), is where the U.S.F.D.A. needs to determine and approve the quality of produce coming into a food-processing factory. The U.S.F.D.A. is not part of the organization, but is used by the VE System as an external group.

[0231] Within any organization, there exists a number of relationships on which workflow is based. For example, a timesheet may need to be approved by the staff member's supervisor. The composer provides the ability for a business process engineer to add these relationships, specify the user's involved in them and then to specify these relationships as a condition of the workflow of particular processes. This is how the preferred embodiment deals with both vertical and horizontal escalation of an activity.

[0232] Thus, a web-based application development environment called the VE Composer has been described to allow business process engineers to develop applications by simply designing application specification including workflow processes, business rules and calendar events. Then, using XML and Java, the VE Enterprise Interpreter, through the VE Player, dynamically abstracts, encapsulates and then interprets the specification in real time and delivers the application to the user without any programming requirement imposed on the business process engineers.

[0233] The method described above can be stored in the memory of a computer system as a set of instructions (i.e., software). The set of instructions may reside, completely or at least partially, within the main memory and/or within the processor to be executed. In addition, the set of instructions to perform the methods described above could alternatively be stored on other forms of machine-readable media. For the purposes of this specification, the term "machine-readable media" shall be taken to include any media which is capable of storing or embodying a sequence of instructions for execution by the machine and that cause the machine to perform any one of the methodologies of the present invention. The term "machine readable media" shall accordingly be taken to include, but not limited to, optical and magnetic disks.

[0234] This VE blueprint section of the present disclosure, consisting of paragraphs 00115 through 00362, describes the features and architecture of Visual Enterprise ("VE") business process modeling and management software program. The information provided in this blueprint section is illustrative and not limiting to the scope of the present invention. The VE blueprint provides information regarding certain capabilities, some optional and alternative, of the method of the present invention. The VE blueprint describes how certain alternate preferred embodiments of the method of the present invention may operate from a technical point of view, and how certain alternate preferred embodiments of the method of the present invention can be used to develop business applications.

[0235] The Visual Enterprise™ Business Process Integration Software Platform ("VE") provides an environment for the web-based development and deployment of business process systems. It consists of two major environments:

[0236] 1. The VE Composer™, which allows business process engineers to model the processes of an organization, its rules and data; and

[0237] 2. The VE Player™, which allows staff, management and others associated with an organization to execute those processes and access data, according to rules laid down by the engineers.

[0238] Business process engineers can engineer solutions using Visual Enterprise without the need for programmers to write code. Further, Visual Enterprise is not a code generator. Visual Enterprise dynamically constructs Web pages based on the process, rule and data definitions prepared by the engineers. Pages are constructed only when required, ensuring that the solutions are flexible, dynamic and always up-to-date. Visual Enterprise allows the business solution to grow and change, in a controlled manner, as the organization grows and changes. Control is provided through release and versioning mechanisms.

[0239] The Four Cornerstones: Referring now to the Figures and particularly to FIG. 1, Visual Enterprise is built on four main principles of applications development: process, data, rules and calendar.

[0240] Process—A business management solution must accord with the business processes employed by the organization and must, therefore, be able to model workflow, work allocation and assignment, process escalation and administration. The use of menu options to implement activities that must be undertaken in sequence, and which require the user to be aware of the correct order in which tasks should be undertaken, fails to model workflow and is not an acceptable solution.

[0241] Data—The data needed for an organization to successfully carry out its business is a key part of any business management solution. Modeling this data means defining each data attribute, including properties such as its type (numeric, text, date, etc) as well as defining associations between attributes. For example, the association between a customer and an invoice, or a staff member and a department, must be defined if the processes are to actually reflect the nature of the business.

[0242] Rules—Business policies are reflected in the rules that govern the capture and maintenance of data within the context of each step of a process. For example, a manager must approve all orders above a specified limit.

[0243] Calendar—An organization operates under many time constraints. It must be aware of at least one time zone, the

16

business/working hours of each office, the public holidays observed in various places and any other events relevant to the organization's operations.

[0244] These four cornerstones of Visual Enterprise are expanded upon below in this VE blueprint section. All of the features of Visual Enterprise are built upon one or more of these cornerstones.

The VE Composer

[0245] Historically, application development has involved business process engineers or analysts undertaking business process reviews, collecting requirements, and collating business rules in order to produce a Software Requirements Specification. This specification would then be handed over to analysts to prepare functional and technical design documents. A team of programmers and administrators, in turn, would use these documents to develop a solution. The possibility of errors being introduced into the final solution due to omissions, inconsistencies, ambiguities or misunderstandings was extremely high.

[0246] Visual Enterprise provides the business process engineer with an environment to define and document business processes, data definitions, business rules and a calendar of events. This environment is called the Composer. As the Composer has a Web-based browser interface the business process engineer can use Visual Enterprise from literally any corner of the globe. All the engineer requires is a computer with access to the Internet.

[0247] The information recorded by the engineer is used directly to construct the solution, without the need to translate from specifications to source code and without the need for a parsing or compilation. The Composer allows the engineer to structure and layout components in a fashion that models the way in which the organization might previously have documented its processes and rules. All of this information can be viewed by management or other interested parties in a transparent manner, as opposed to being hidden away somewhere in computer code. The main elements of the Composer are:

[0248] A graphical process definition system,

[0249] An application attribute (data definition) library,

[0250] A business rule definition system,

[0251] An organization calendar, and

[0252] An administration section supporting the security model associated with engineers and application users.

Entity Hierarchy

[0253] The Processes, Data, Calendar, and Rules that Visual Enterprise depends upon are modeled as a hierarchical set of entities. The four principal entities are supplemented by many other entities, each of which supports and expands upon the four principals. An overview of this hierarchy along with a brief description of the entities appears below. Organization—The organization for which the solution is being engineered.

[0254] Engineers—The business process engineers who are engineering the applications.

[0255] Users—The management, employees or other interested parties of the organization who have been granted access to use the applications.

[0256] Groups—Positions or roles within an organization against which access to a particular process or activity is granted.

[0257] Calendar—Time zone, working hours, public holidays, reporting periods and events relevant to the organization.

[0258] Organization Dictionary—A template of attribute properties, which can be used to assist the application of standards to information collected by the organization.

[0259] Application—A collection of business processes, data definitions and rules, which together reflect how an organization or particular section of an organization operates.

[0260] Attribute Library—A library of attributes defining the properties of and associations between pieces of information collected by the organization.

[0261] Process Category—A collection of related processes used to assist users in identifying the correct process to initiate.

[0262] Process—A series of steps performed to complete a particular operation or function of the organization

[0263] Activity—A single step in a process. A user may complete an activity or it may be automated, such as a batch process.

[0264] Attribute—A piece of information collected by the organization, such as a Customer Name or Invoice Number.

[0265] The Composer presents the information to the engineer according to this hierarchical structure. For example, to view details relating to an activity the engineer must first choose the application, then the process category and the process before being able to select the relevant activity.

The VE Player

[0266] The Player is the Visual Enterprise environment that runs the business solutions engineered using the Composer. The Player, like the Composer, has a Web-based browser interface allowing users access to the organization's business processes via the Internet. The main elements of the Player are:

[0267] An Intray that presents the user with a list of activities to be completed. Each of these activities is one step of a business process that has already been initiated.

[0268] An OutTray that allows the user to view details of activities already completed by that user.

[0269] A New Processes facility for authorized users to initiate new processes.

[0270] When the user opens an activity the Player dynamically constructs browser-based forms with the appropriate attributes and associated labels. All relevant business rules, including validation and calculations, are constructed dynamically and built into the page. Both the Player and Composer have in-built security to ensure only authorized users and engineers have access to specific activities within the organization's processes, data and rules.

[0271] This section looks at the four Visual Enterprise cornerstones of Process, Data, Rules and Calendar in detail.

Processes

[0272] A process is a collection of related activities in a defined workflow. Each process has one initiating activity. This activity may be initiated either by a user or by an event. Users must have access rights in order to initiate a process. Examples of events that could trigger a process include: a

time constraint, such as a process which must run at 10:00 pm each night; or a condition being satisfied, such as a process which must run once stock gets below a certain level. The workflow determines the order in which activities are undertaken. Some activities may be able to be undertaken in parallel, which is controlled by the way that activities are linked and associated with each other and the placement of process synchronization conditions (Activity Link Join Methods).

[0273] Activities: An activity is a collection of one or more related data attribute packages designed to add, update and/or query data. Rules associated with an activity may be used to validate data or to calculate values such as totals, tax amounts, due dates, etc. An activity may either succeed or fail. The success status is used by the process workflow to determine which activity or activities are to follow on from the current activity. A number of different types of activities may be defined. These include:

[0274] User Input activities where a user is expected to enter or view data, such an activity to raise a purchase order.

[0275] Message activities, which allow a message to be sent to one or more people, such as an activity to notify a customer by e-mail that his application has been approved.

[0276] Report activities, which allow a predefined report to be generated at a particular point in a process, such as an activity to print an invoice.

[0277] Automated activities, which allow data to be processed without any user input. The selection of data and how it is processed is based on the definition of the activity created by the business process engineer.

[0278] Remote activities, which allow direct access to external databases.

[0279] Web Service activities, including both consumer and provider activities, which allow web services to be integrated into workflow as part of a standard business process.

[0280] Maintenance activities, which allow data to be maintained outside the realms of a standard process, such as an activity to maintain customer details like phone number, maiden name, etc.

[0281] Data: The smallest unit of data in Visual Enterprise is the Attribute. Attributes are defined in collections called Packages, and packages are associated with each other in various ways. Each Attribute defines the name, data type, size and other properties of the values that will be stored against it. When a package instance is created, attribute values are created and stored with a reference back to the package and the attribute within the package. A single package instance consists of one set of values. There can be any number of instances of a given package.

[0282] Packages can be associated in two ways: independent packages can be loosely associated in a relatively ad hoc way, as required for specific processes. There is no need to predefine package associations in the Library, although that is always a useful option when relatively strong, commonly used associations are defined (e.g. Customer/Invoice). The second type of package association is used where packages are associated to the point of dependence, for example Line Items on an Invoice. Here the dependent package would be defined as a Sub-Package. Data in a Sub-Package cannot exist independently of the package instance it is dependent upon. When the 'parent' is deleted the sub-package data is removed as well.

[0283] Attributes: An attribute defines a collection of values where each instance in the collection conforms to the definition provided by the attribute. Each attribute has a number of properties, such as whether the attribute represents a date, a number, text, etc; how it should be formatted on the page and in reports; input control options, etc. Each attribute value, such as a customer's name or birth date, is recorded using the business processes and activities defined by the business process engineer. The database maintains the complete collection of values for each attribute, together with references to the package in which the attribute is defined. No distinction is made as to the context of an attribute except by the activity that references that attribute. For example, one activity might be defined to record the details of all customers, while a second might be defined to record each customer's purchases. The collection of values of the "customer name" attribute would include a value for each recorded customer as well as a value for each time a customer has made a purchase.

[0284] Packages: a package is a collection of related attributes. For example, a Customer package would be a collection of attributes related to a customer, such as the customer's name, date of birth, gender, etc. A package may also include sub-packages that define data dependent on the main package. For example, an Invoice package may contain an Invoice Item sub-package. The Invoice would contain attributes for the Invoice Number, Date, etc., while the Invoice Item would contain attributes for the Quantity and Price of each product being purchased.

[0285] A package may have a single instance of a sub-package, such as between an Invoice and a Dealer package where there can be only one dealer per invoice. Alternatively, a container package may have many instances of a sub-package, such as between an Invoice and Invoice Line where there can be many Invoices Lines per Invoice.

[0286] Data Transactions: Every activity that allows the user to add or modify attribute values, or that uses business rules to add or modify attribute values does so via one or more data transactions. The transactions are initiated once the user chooses the Submit button but only after the activity has been validated using the associated business rules. A transaction is initiated for each package included in an activity. The transactions are all applied together as a single activity transaction such that the integrity and consistency of the whole activity is maintained. If a single package transaction should fail for any reason then the transactions for all packages in that activity will be reversed.

[0287] Visual Enterprise transactions will not delete data from the database. This is because of the inherent problems associated with deleting data that has already been referenced by other activities. Instead of deleting data Visual Enterprise will inactivate data such that it cannot be referenced in new activities.

[0288] The business process engineer does not need to be aware of transactions because they are controlled and managed entirely by Visual Enterprise.

[0289] Rules: business rules are the policies, regulations and constraints governing the way an organization carries out its operations. Visual Enterprise is concerned with those business rules that are in some way related to, or impact upon, the business processes being developed. For example: "the discount must be less than 10%", or "a purchase order form must include at least one item".

[0290] Business rules must be satisfied at all points in time unless a condition is applied that specifically limits the time-

span during which the rule applies. For example, a discount may be offered only on invoices raised between Christmas and New Year. This rule will hold true at all points in time, subject to the condition that the discount is only offered between Christmas and New Year. In other words, Visual Enterprise would reject any attempt to offer a discount outside of those time constraints.

[0291] It should be recognized that if a rule is applied when data is submitted to the database the only opportunity for a rule to be invalidated is when that data is updated or modified. For this reason Visual Enterprise does not continually apply rules to ensure that they still hold true. Instead it applies the appropriate rules whenever data is accessed, added, modified or submitted. A rule such as "the customer's age is the difference between today and the date of birth" is not recalculated every day. In theory, at a given point in time the value of the age will not be correct. However, as soon as the age is referenced by an activity, either to display the age to a user or to use the age in some calculation, the rule will be executed to ensure the value is correct.

[0292] Calendar: Visual Enterprise allows an organization to define events relevant to the organization. These might include an Overnight Processing event, an End of Month event, an End of Financial Year event, a Plant Maintenance Shutdown event, etc. The business process engineer can then associate specific business processes or rules with these events. This removes the need for systems administrators to schedule batch jobs for overnight or end of month processing. It also means that should the organization wish to change its end of month processing from the last day of the month to the last working day this can be done by a business process engineer changing the definition of a single event. In fact, this example would only require the engineer to select one option indicating that if the event fell on a non-working day to move the event to the previous working day.

[0293] The calendar can be divided into five components:

[0294] Time Zone—this enables the engineer to select the time zone in which the organization operates. Visual Enterprise supports multiple time zones. Individual calendars for departments, offices, etc. can be defined, each with its own time zone. Visual Enterprise performs time zone conversions wherever they are required.

[0295] Working Hours—specifying the standard working hours for an organization enables Visual Enterprise to know which times are in and which are out of business hours.

[0296] Public Holidays—by defining the holidays observed by the organization Visual Enterprise is able to move events, as specified by the engineer, which fall on a non-working day. Lists of predefined standard public holidays are available on a country-by-country and state-by-state basis.

[0297] Time Spans—an engineer can define the standard reporting periods that are used by an organization, such as a Financial Year or Sales Period.

[0298] Events—engineers can define absolute or relative events and either one-off or recurring events. A relative event is one that occurs at a set time-span before or after another event.

[0299] Management, staff or other authorized persons of an organization can access the business solution developed by the business process engineers by this environment. The Player interface is entirely Web-based, and is accessed via a pre-allocated URL (for example, http://ve.orgname.com).

The home page at this address requires the user to supply a login name and password in order to gain access to the site. An Administrator with the appropriate authority creates a login name to grant access to new users. The initial password is allocated by Visual Enterprise and communicated to the user via e-mail. The first time the user logs into the site he is required to select a new password, which will then be used for all subsequent access to the site.

[0300] The InTray: Once verified, the user is directed to the Intray. This page displays a list of the activities awaiting completion. It includes both new activities and partly completed activities that the user is currently working on. The activities assigned to a user are determined by the process workflow (as defined by the business process engineers). The engineer can set a number of properties for each activity, which dictate how each activity should be allocated and assigned. For instance, an engineer may specify that the manager of the user who completed the previous activity in the process must complete an activity. An example of this might be a Request For Leave process where the manager of the employee requesting leave must approve the application. Alternatively, an engineer may specify that an activity can be completed by any one of a number of users. A call center may have a Support Call process where an activity can be completed by any one of a team of technicians. The first available technician can elect to take the next waiting support call.

[0301] The InTray displays a list of activities by name, along with a number of associated details including:

[0302] The date the activity was received,

[0303] The date by which the activity is due to be completed,

[0304] The user from whom the activity originated (this is the user who completed the previous activity in the process workflow),

[0305] A status indicating whether the activity is overdue, due today, tomorrow, next week, etc.

[0306] The user can also add, view, or modify notes entered by users who have completed prior activities in the process. Users completing subsequent activities in the process can, in turn, view these additional notes. Some additional optional features of the InTray include:

[0307] Access to a history of the process of which an activity is part, including details of who completed each step and the date each step was completed.

[0308] Unopened activities are bolded to ensure they are clearly distinguishable from activities that the user has already viewed or partially completed.

[0309] Users can place an activity on hold. This would be a standard procedure where all of the information required to complete an activity is not yet available. The business process engineer can place time constraints on an activity such that if the information is not entered within a particular time-span then the process can be escalated.

[0310] Activities available to a group of users can be reassigned to the pool should the user decide that he or she is unable to complete the activity. Activities can be reassigned either by the user or by an administrator.

[0311] To open an activity the user simply clicks on the activity name. User Interface activities are laid out automatically in accordance with the definition constructed down by the business process engineer in the composer. The user completes the activity by entering the relevant data for the required attributes in the appropriate fields. As the informa-

tion is recorded any associated business rules are executed to ensure the data is valid and to carry out any calculations required. Once the page is filled out, the user clicks on the Submit button in order to submit the data to the database. At this point, where required, rules are re-executed to ensure other users have not made changes, subsequent to the original evaluation of the rule, which might affect upon the validity of the data. At regular intervals the Intray is refreshed to ensure the user has access to the latest list of activities. The Intray is also refreshed when an activity is submitted to ensure the completed activity is removed from the list.

[0312] The OutTray: When the user completes an activity it is removed from the Intray and moved to the user's OutTray. The user can access the OutTray from a menu at the top of the page. The OutTray will initially display all activities completed during the current day. To access activities completed earlier the user can either choose to view activities completed on a particular date or search for all activities of a particular type completed during a specified date range.

[0313] As with the Intray, the OutTray displays a list of activities by name, along with a number of associated details including:

    [0314] The date the activity was completed,

    [0315] The date by which the activity was due to be completed,

    [0316] The activity's status at the time it was completed.

[0317] The user can also view any notes associated with the process of which the activity is a part. This includes notes recorded by users who have completed prior activities in the process and notes recorded by users who have completed subsequent activities in the process. Further, the user has access to a history of the process of which the activity is a part; including details of who completed each step and the date each step was completed.

[0318] Initiating a process: As well as completing activities that are a part of an existing process, the Player allows authorized users to initiate new processes. The New Process page lists all processes to which the user has been granted access (by the business process engineers). These processes are displayed in an explorer-like hierarchy based on the process categories designed by the engineers.

[0319] A preview of the first user interface activity in the process is displayed, to ensure that the selected process is the correct one. Only processes that have a user interface activity as the first activity can be initiated from the New Process explorer. User interface activities are dynamically laid out on the page without any need for the business process engineer to specify the position of each attribute on the page. The properties of each attribute determine the field order, the width of each field, whether a data field, multi-line text box, radio buttons or check box should be used, the background and foreground colors, etc. Once a the user has opened a new process by clicking the Open button, the process becomes identical to that where the user completes an activity from the Intray. Upon completion, the activity is added to the OutTray and the process workflow then initiates the next activity in the process and allocates it to the appropriate users.

[0320] If the user chooses to cancel the activity instead of submitting it then the whole process is revoked. In effect this is the same as if the process had not been initiated.

[0321] Administration: From time to time issues arise that prevent users from being able to complete one or more activities. The Composer allows the business process engineer to design the workflow such that if an activity is not started or completed by a specified date then the process will automatically escalate. Should this occur then the activity is automatically removed from the user's Intray and either reallocated or a new activity initiated in its place. However, there may be occasions where no escalation has been specified or where it is not desirable to wait until the due date. For example, if an employee goes on leave without clearing his Intray then the organization may wish to reassign all of the outstanding activities to other users. The Administration section allows an appropriately authorized administrator to reassign one or more activities from a user's InTray to another user or group of users. This option can be accessed via the Administration menu at the top of the page.

[0322] As with the InTray and OutTray the Administration page also allows the user to view a history of a process of which a particular activity in a particular user's InTray is a part.

[0323] The VE Composer: This is the environment used by business process engineers to define and document the processes, data, rules and calendar used by the organization. The Composer interface is entirely web-based, and is accessed via a URL. This page requires the engineer to supply an organization ID, login name and password in order to gain access. The organization ID is allocated by Visual Enterprise when the organization is first registered. An administrator or business process engineer with the authority to grant access to new engineers allocates the login name. The initial password is allocated by Visual Enterprise and communicated to the engineer via e-mail. The first time the engineer logs into the site he is required to select a new password, which is then used for all subsequent access to the site.

[0324] After verifying the engineer's details, the Composer page is displayed. This page is divided into three areas:

    [0325] The Explorer, which is located on the left-hand side of the page.

    [0326] The Workspace, which is located on the right-hand side of the page. When the engineer is not working on a specific process this workspace allows access to general Visual Enterprise information, such as tutorials, latest product news, etc.

    [0327] The Composer Menu Bar. The menus on this bar allow access to features of the Composer, such as the Attribute Library, Organization Calendar and Administration functions.

[0328] The Explorer: The main purpose of the Explorer is to provide a simple and efficient means for an engineer to create, locate and modify business process definitions. Its structure is based on the Visual Enterprise entity hierarchy outlined above. The Process Explorer has three modes of operation, namely: Organization—This mode allows access to all elements of the Composer, such as Processes, Attribute Library, Organization Calendar, Users, Groups, etc.

    [0329] Application—This mode allows access to all elements of the Composer related to a single application, including Processes and Attribute Library. It excludes elements such as the Organization Calendar, Users and Groups.

    [0330] Section—This mode allows access to all elements of the Composer related to a selected section of the hierarchy. This could be, for example, the Attribute Library, or a Process Category, including processes, activities, attributes and rules.

[0331] The Process Workspace graphically displays the workflow of a selected process. It can be used to view or modify the flow or to modify details about any of the activities that make up the process.

Activities can be dragged and dropped within the workspace. As the engineer defines each new activity, workflow links can be established with existing activities. In this way the process workflow can be defined and reviewed before having to specify the attributes, rules and other activity-based properties. Managers or other interested parties can be granted read-only access to these processes in order to review and sign-off on the workflow. This helps to ensure that the process flows are correct before adding the more detailed parts of the process definition.

[0332] Organization Dictionary: An important factor in any business solution is to provide a consistent user interface. Visual Enterprise facilitates this consistency making it possible to define attribute definition templates. These definitions allow the engineer to define attribute properties such as the data type, a default value, whether or not it is mandatory for a user to record a value for the attribute, etc. Some of the properties are specific to a particular data type, such as the number of decimal places, which is specific to numeric attributes. Some properties, while applicable to all data types, have options that are dependant on the data type. For example, the options applicable to the format properties of a numeric attribute are different from the options applicable to a date or time-span attribute. Attributes defined within each Organization's template can be grouped into categories for easy reference. When engineers create attributes for a particular application, they can choose to base the definition on an attribute from the template. In this way the attribute inherits all of its properties from the template. The engineer may choose to override one or more of these properties. Any not overridden continue to be inherited from the template.

[0333] For example, a Currency attribute may be defined in the Organization Dictionary with the following property values as shown in Table 1:

TABLE 1

Property	Value
Name	Currency
Data Type	Number
Control	Data Field
Default	0
Mandatory	No
Decimal Places	2
Display Width	10 average characters

[0334] An engineer may then create, as part of the application, a Discount attribute, which is based on the Currency attribute template. All of the properties are inherited from the Currency attribute. A further attribute, named Price, may be defined and also based on the Currency attribute. However, in this case the engineer may choose to override the Mandatory property to "Yes" and the Display Width property to 12. All other properties are inherited from the template. If, at a later point in time, the Display Width property of the Currency attribute is changed to 8 then the Discount attribute will inherit this new value. The Price attribute, however, will not as its value was overridden.

[0335] The use of the Organization Dictionary can greatly reduce the time required to define each of the attributes required for an application. It also means that a later change in

requirements relating to the properties of a group of attributes (such as increasing interest calculations from 2 to 3 decimal places) can be effected with much less effort.

[0336] Attribute Library: Each application developed using Visual Enterprise requires an Attribute Library. This library details the attributes that are collected by the organization as part of its operations. For example, an organization collects information about customer names, addresses, phone numbers, purchase order numbers, product names, quantities, etc.

[0337] These attributes are grouped into packages of related attributes. For example, a Customer package would be a collection of attributes related to a customer, such as the customer's name, date of birth, gender, etc. A package may also sub-packages. For example, an Invoice package may contain an Invoice Item sub-package. The Invoice would contain attributes for the Invoice number, date, etc., while the Invoice Item would contain attributes for the name, quantity and price of the product being purchased. Each sub-package represents a (possibly) multivalued data group that is dependent on the 'parent' package. If an instance of the parent is deleted then so are all instances of its sub-packages. For example, deleting an Order deletes all its contained Line Items.

[0338] Package properties include:

[0339] A list of key attributes that uniquely identify an instance of the package. For example, a customer's name and telephone number might uniquely identify each customer.

[0340] A list of search attributes that are used to assist users when making a choice from a list of available package instances. For example, to assist the user in selecting a customer, the engineer might choose to display the customer's name, date of birth, and phone number. By defining these as search attributes, the user can see the values displayed and search by any of them.

[0341] Additional properties are maintained on multi-valued sub-packages, including:

[0342] The order in which the package instances should be displayed. For example, should invoice items be displayed in product code order or in order of price?

[0343] Whether there is a minimum or a maximum number of instances. For example, an organization might require an invoice to be limited to 10 items or fewer, or not to be limited at all.

[0344] Library attributes can be based on the Dictionary or they can be defined from scratch. An attribute that is based on a Dictionary attribute can have the same name as the attribute on which it is based or the name can be changed as required.

[0345] The Business Rule Wizard, or Rule Wizard, allows the engineer to define rules wherever they are needed. For example:

[0346] Against an attribute in the Attribute Library.

[0347] Against a package in the Attribute Library.

[0348] Against an activity in a process.

[0349] On a conditional workflow link.

[0350] As a search condition to restrict associated package instances.

[0351] In a process trigger.

[0352] The Composer guides the engineer through the process of defining a rule, step by step. At each step the engineer simply chooses the appropriate option. The Composer also has two sub-components tailored towards mathematical and algebraic expressions. Called the Expression Builder and the Condition Builder, these components operate like a calcula-

tor. This approach allows these sorts of rules to be defined quicker than would be the case with an options-based approach.

[0353] The Composer's options are based on the Visual Enterprise Business Rule Grammar. The grammar is based on the four principles of validation, calculation, condition and action using a combination of set theory and standard mathematical grammar. The grammar has been designed such that it can be expanded and enhanced as the need arises. The engineer also has the ability as part of the Composer to specify a time-span during which the rule should be in force. If no time-span is specified then the rule will be permanently in force, or at least until the engineer specifies otherwise. The time-span can be based on specific dates and times or it can be based on organization-based events.

[0354] Rules can be categorized according to definitions created by the engineer. For example, a rule may be placed in a Finance category and a Legal category because the organization perceives it to have both financial and legal implications. These categories can then be used as criteria when searching for particular rules.

[0355] The definition of a rule is not converted or stored as "code" but is stored as data. The Player environment accesses this data and the appropriate logic is applied to ensure that the conditions laid out by the rule are met.

[0356] Organization Calendar: The five components of an organization's calendar are all accessed from a common dialog. This dialog contains tabs that allow the details associated with each component to be defined and maintained.

[0357] Time Zone: The time zone is automatically set based on the country and state entered by the organization as part of the registration process. If necessary, an alternative time zone can be selected.

[0358] The time zone may optionally be used by Visual Enterprise to handle time zone conversion, which may be necessary, either because some users are operating in a different time zone or the organization is operating in a different time zone to the database server.

[0359] An organization's work hours will tend to be constant over time. When they change Visual Enterprise maintains both the old working hours and the new hours. The period of time between the start of new working hours and the end of those working hours, when a new set is defined is called a cycle. Features and rules associated with cycles include:

[0360] A new cycle can be defined ahead of time so that there is a seamless transition from one set of working hours to another.

[0361] Only details about the working hours for the current (most recent) cycle can be modified.

[0362] The working hours for the current cycle can be removed, effectively re-instating the working hours from the old cycle.

[0363] A new cycle cannot be defined which comes into effect before the current cycle.

[0364] Most organizations operate on a recurring weekly period whereby, for example, the working hours may be from Monday to Friday, 8:00 am to 5:00 pm. Some organizations will have working hours that are different on some days of the week. For example, the hours may be Monday to Friday, 7:00 am to 3:00 pm and Saturday, 8:00 am to 12:00 am. Visual Enterprise allows a variety of different configurations, including specifying non-working time within a block of working time. The latter could be used, for example, to record meal break times.

[0365] Different parts of an organization may have different working hours. Visual Enterprise provides the ability for an engineer to establish separate but linked calendars for different entities. For example, a calendar might be established for each of the departments within an organization to cater for the different workings hours between manufacturing, delivery, sales, finance, etc.

[0366] Holidays: Whilst public holidays are normally standard for organizations, there are occasions where staff may be required to work on some public holidays. For this reason Visual Enterprise allows an engineer to choose whether or not to set the standard public holidays for an organization. If standard public holidays are set the engineer can still choose to remove one or more of those holidays that may not be applicable. An engineer can also define additional holidays that may not be standard for the particular state or country in which the organization operates.

[0367] The standard public holidays need only be set once, and do not need to be set each year. Visual Enterprise has a definition of each holiday so that it can calculate the day on which it will fall each year. Should a state or federal government change the date on which a holiday falls it would be necessary for an engineer to modify the definition of the holiday to reflect that change.

[0368] When Visual Enterprise needs to determine whether a particular date is a working day or not, the organization's defined working hours are first checked. If there are no working hours for that particular day then it is a non-working day. If there are working hours defined then Visual Enterprise will check whether that date is defined as a holiday. If it is, then the date is a non-working day.

[0369] Time Span: A Time Span defines a recurring time period. There are predefined Time Spans that cannot be changed (second, minute . . . year). In addition, an engineer can create additional Time Spans such as a 4-week month, or an 8-hour shift. An event is generated each time the time span recurs. These events can be referred to in rules, and they can be used to schedule processes and activities.

[0370] Administration: From time to time administrative tasks must be performed by business process engineers. These tasks include:

[0371] Registering an Organization

[0372] Administering security and permissions for Business Process Engineer

[0373] Administering security and permissions for users

[0374] Creating and maintaining access Groups

[0375] Maintaining Relationships between staff members to reflect the current organization structure.

[0376] In order to begin developing an application with Visual Enterprise, an administrator or engineer must first register the organization. The Visual Enterprise home page contains a link to the registration page. This page requires details about the organization, such as the organization name and address, to be recorded. Details about the administrator, such as the name, phone number, chosen login name and e-mail address, are also required. Optionally, the administrator can also register the names of other staff or related parties as Visual Enterprise engineers.

[0377] Once the registration is complete, the administrator and any other registered engineers are sent an e-mail message containing an allocated organization ID, login name and pass-

word. As soon as this message is received the engineer should log into Visual Enterprise where they will be required to select a new password.

[0378] Engineers or other appropriate persons are granted authority to perform certain tasks or operations within the Composer environment of Visual Enterprise. These include the authority to manage engineers and users, maintain the attribute library, maintain the calendar, and to design processes. An engineer may be a business process engineer who designs processes or maintains the attribute library, or it may be someone from management who simply wants to be able to view details of existing processes and how they have been structured.

[0379] Over time an organization may wish to grant new engineers access to the applications being developed, modify details of existing engineers or revoke an engineer's access. Each of these procedures can be performed from the Engineer List dialog box. The process of adding a new engineer is the same as adding an engineer during the registration process. Additionally, an existing user may need to be granted access to Visual Enterprise as an engineer. It is important when adding the new details that the option indicating the engineer is already an existing user be selected. This prevents having to maintain two sets of details and also prevents the user having to remember two login names and passwords.

[0380] Users: A Visual Enterprise user is anyone working for or who has an interest in an organization, and who has been granted access to one or more of the processes of that organization. The details that need to be collected for a user are similar to those that are collected for an engineer: name, phone number, login name, and e-mail address. The mechanism for informing the user of the allocated login name and password are also the same.

[0381] The security mechanism that determines the processes to which a user has access, however, are quite different from those for an engineer. Authorizations, particularly in medium to large organizations, are usually based on groups or roles. This makes the process of resetting the security for a user who has changed role much simpler. Access to particular activities within a process can be based on specific groups so that when a user changes roles only the group or groups to which that user belongs needs to be changed. The security for that user will automatically be reset based on the new groups. The engineer can also specify any organizational relationships in which the user is a party, for example, the name of the person who is the user's supervisor or project manager. An engineer, when designing the workflow component of a process, can make use of these relationships. For example, a Leave Approval activity may need to be completed by the Office Manager of the user who initiated a Request for Leave. Similarly, a Timesheet Approval activity may need to be completed by the Project Manager of the user who completed the Record Timesheet activity.

[0382] As outlined in the section above on Users, Groups can be used to control access to particular activities and processes. The engineer can choose to structure groups in ways that best suit the security model used by the organization. They could be structured to model the roles that an organization has created or they could be structured according to categories or types of process. For example, one organization might define the following groups:

[0383] Purchasing Officer
[0384] Purchasing Manager
[0385] Sales Officer

[0386] Sales Manager
[0387] Chief Financial Officer
[0388] The users in each of these positions would then be allocated to that group. Access to a Raise Purchase Order activity might then be granted to the Purchasing Officer, Purchasing Manager and Chief Financial Officer groups whilst access to the Approve Purchase Order activity might only be granted to the Purchasing Manager and Chief Financial Officer groups.

[0389] Alternatively, an organization might define the following groups:

[0390] Purchasing
[0391] Purchasing Approval
[0392] Sales
[0393] Sales Approval

[0394] Users would then be allocated to those groups that cover the activities relevant to their role in the organization. The Purchasing Manager would be allocated to Purchasing and Purchasing Approval. Access to the Raise Purchase Order activity would be granted only to the Purchasing group, whilst access to the Approve Purchase Order might be granted only to Purchasing Approval.

[0395] Groups can also be defined to belong to other groups. For example, an organization might require three levels of access, Gold, Silver and Bronze whereby users allocated to the Gold group automatically should have access to all processes to which the Silver and Bronze users have access. To this end, as well as the list of Bronze users, the Silver group would be added to the Bronze group. As well as the list of Silver users, the Gold group would also be added to the Silver group. In this way any user added to the Gold group would automatically have access to any process to which a Silver or Bronze user had access.

[0396] Relationships: Within any organization there exist a number of relationships on which workflow is based. For example, a timesheet might need to be approved by the staff member's supervisor. Visual Enterprise provides the ability for an engineer to add these relationships, specify the user's involved in them and then to specify these relationships as a condition of the workflow of particular processes.

[0397] Processes are one of the four cornerstones of Visual Enterprise. The Explorer provides a hierarchical view of an organization, its applications and the processes and activities in each, while the Process Workspace graphically presents process workflow. Visual Enterprise has a number of different entities, described in detail below, that are directly related to a process.

[0398] Process Categories: Process categories are like folders that an engineer can use to provide structure to the list of processes that users see on the New Process page in the VE Player. An organization may have dozens or even hundreds of defined processes. For example, an engineer developing a sales application may define the following process category hierarchy:

[0399] Customer Relationship Management
[0400] Marketing Campaigns
[0401] Telemarketing
[0402] Transactions
[0403] Purchasing
[0404] Sales
[0405] Adjustments
[0406] Maintenance
[0407] Customers
[0408] Suppliers

[0409] Staff

[0410] Departments

[0411] Reporting

[0412] Monthly Reports

[0413] Year to Date Reports

[0414] Whole of Year Reports

[0415] All processes relating to purchasing would be defined against the Purchasing category and so on. For an engineer to access one of these processes he would simply need to select the Transactions category followed by the Purchasing category. Likewise, for a user (who has been granted access) to initiate one of these processes he would need to make the same selections.

[0416] The security associated with each of the processes in a given category will have some similarities. For this reason, Visual Enterprise allows the engineer to specify the default access for any process added to that category.

[0417] Process Definition: The construction of a process involves a number of steps, most of which relate to the definition of the activities that make up that process. The initial step is to define the process itself, and its user access. The next step is to add activities to the process and to define the workflow or links that join these activities together. After creating each activity the engineer may choose to define the additional information that is required to fully build a process. Alternatively, the engineer may choose to leave the detailed information to a later time, once the links between the activities have been defined. As each activity is added an icon is added to the Process Workspace to graphically represent each step in the process. The engineer can position this icon in the Workspace as required to display the overall flow of the process. The positioning is a matter of personal choice. Some may prefer processes to flow from left to right on the page, others from top to bottom, or a combination thereof. At regular intervals or when the engineer moves to a new process, the graphical layout is automatically saved to the database. If an icon is moved by mistake then the engineer can choose to refresh the layout from the last saved position.

[0418] Activity Definition: An activity can be defined as one of a number of types:

[0419] User Input activities where a user is expected to enter or view data, such an activity to raise a purchase order.

[0420] Message activities, which allow a message to be sent to one or more people, such as an activity to notify a customer by e-mail that his application has been approved.

[0421] Report activities, which allow a predefined report to be generated at a particular point in a process, such as an activity to print an invoice.

[0422] Automated activities, which allow data to be processed without any user input. The selection of data and how it is processed is based on the definition of the activity created by the business process engineer.

[0423] Remote activities, which allow direct access to external databases.

[0424] Web Service activities, including both consumer and provider activities, which allow web services to be integrated into workflow as part of a standard business process.

[0425] Maintenance—where the user can add, modify or delete data in circumstance which may not fall within any formalized business process but which, nonetheless, needs to be maintained.

[0426] Associated with an activity is a series of information, which can be divided into a number of groups. Some of this information is dependent upon the activity type, however it includes:

[0427] Attributes—a list of attributes to be displayed, used or collected by the activity. This list is created by selecting packages and attributes from either the Attribute Library, the Organization Dictionary or from a previous activity in the process.

[0428] Links—details about the activities that precede and succeed the activity, including any conditions associated therewith.

[0429] Rules—a list of rules relevant to the activity, which need to be enforced by Visual Enterprise.

[0430] Time Criteria—properties that dictate any time constraints, such as a completion deadline, that are placed on the activity.

[0431] Access—a list of groups and/or users to whom the activity has been granted access. The groups or users granted access to the first activity in a process determine the users who have the authority to initiate that process.

[0432] Initiation—properties that indicate how to determine the user or users to whom the activity should be allocated.

[0433] Visual Enterprise supports a variety of integration techniques. These range from low-level external components integrated through rules, to web services activities. Using the appropriate technique, the engineer, with assistance from technicians familiar with the access techniques and data structures, can integrate VE processes with virtually any external process or data source.

[0434] External Components provide access to COM+ or EJB components.

[0435] Web Services provides access to web service providers. Web service provides can be other systems within the firewall or external providers. Web service consumers rely on SOAP and XML to communicate via HTTP with web service providers. VE supports both web service provider and consumer activities that seamlessly incorporate these services into a Visual Enterprise process, as if it was part of the business solution.

[0436] Remote Activities allow external data sources to be read and written, as if they were native VE Packages.

[0437] Typical examples of remote activities are:

[0438] Example 1: An organization may have a legacy system that maintains details of all staff working for the organization. A process can be developed which uses a remote activity to provide a user with access to that information in conjunction with other data stored in the Visual Enterprise database.

[0439] Example 2: An organization may have a legacy system that maintains a General Ledger. Activities can be developed that access external components to post transactions to that system as part of the business processes developed to raise purchase orders, invoices and receipts.

[0440] Example 3: An organization may wish to conduct business-to-business transactions with a supplier to order stock. A process can be developed that allows the organization to place an order with the supplier, including any rules that might be associated with that order. For example, a rule might state that if the cost is above a defined limit then the supplier must seek authorization with the organization before proceeding with the order. Such authorization would be obtained by the automated initiation of an activity in the

relevant user's Intray (as defined by the process workflow). If the user authorized the cost of the order then that authorization would be transported, again via SOAP, to the supplier whereupon the order could proceed (or be cancelled if authorization was not obtained). N.B. For this functionality to operate the remote system or component must support the SOAP infrastructure.

[0441] Process Views: Within any organization, different staff may have different views of exactly what is entailed in any given process. This may be because certain steps in the process are not relevant to some users, or that some information collected or displayed by an activity should be restricted to certain users. For example, an engineer may have defined a process to collect information about a new staff member. This may include personal details, contact details, payroll details, etc. The information may be collected over a series of activities by different staff within the organization. A staff member from the Personnel department may initiate the process; the new staff member may themselves fill in the personal details; and a staff member from the Payroll department may complete the payroll details. The final activity in the process may revert to the Personnel department. It may be a requirement of the organization that the staff member's salary not be displayed to anyone in the Personnel department with the exception of the Personnel Manager.

[0442] Visual Enterprise allows an engineer to define one or more views of a process, which enables the information included in each step of the process to be tailored according to the user who has access to that step. The engineer must define two components of a view:

[0443] Activity/Attributes—this is a selection of the activities to be included in the view, and for each activity, which attributes should be visible to the user.

[0444] Access—this is a list of groups and/or users who have been granted access to the view.

[0445] Being granted access to a view is quite different from being granted access to an activity. Access to a view gives a user access to details about the history of a process, as it relates to the activities included in the view. It does not, by itself, allow a user to complete any one of those activities. Access to an activity, on the other hand, does gives a user the authority to complete that activity no matter which view is being used.

[0446] Every, or each process, may optionally have one default view, which includes every activity and those attributes assigned to each activity. Any given user may have access to more than one view of a process. If this is the case then the user is given the option of which view to use when initiating a new process or opening an activity.

[0447] Neither the links between activities, nor the rules associated with an activity, nor the time criteria nor assignment properties are affected in any way by a view.

[0448] Attributes: Attributes are one of the four cornerstones of Visual Enterprise and as such it is important to understand the relationship between attributes and packages, attributes and processes, and attributes and business rules.

[0449] Cardinality of Package Associations: Associations between attribute packages in Visual Enterprise are described in each direction independently. If we look at two packages, A and B, then the association of A to B is described separate from B to A. The complete list of possible associations between the two packages is as shown on Table 2 as follows:

TABLE 2

A → B	B → A
One-to-one (each A has only one B)	One-to-one (each B has only one A)
One-to-many (each A has many Bs)	One-to-one (each B has only one A)
One-to-one (each A has only one B)	One-to-many (each B has many As)
One-to-many (each A has many Bs)	One-to-many (each B has many As)

[0450] There are two ways in which packages can be associated:

[0451] A sub-package/super-package association implies a one-to-one association from the sub-package to its super-package. That is, the sub-package must belong to just one super-package. For example, a Customer may have an Address sub-package. In the other direction, super-packages can have a one-to-one or a one-to-many association with a sub-package. For example, a Customer may have a one-to-many association with an Income History sub-package. In all cases sub-packages depend for their identity on their associated super-package. If the super-package data is deleted then the sub-package data is deleted with it.

[0452] A linked package is not dependant on another package. Instead, it defines an association between independent packages. For example, an Invoice may link to a Customer, and a Loan Application may also link to a Customer. In this case, a Customer can exist without any Invoice or Loan Application being created. Links can be either one-to-one or one-to-many. If the Invoice or Loan Application is deleted, there is no effect on the Customer.

[0453] Package Associations: Some kind of association exists between most attributes of an organization. These associations exist between individual attributes and also between packages. For example, a customer name attribute is directly associated with a customer phone number attribute; an invoice number is directly associated with an invoice item. These associations are defined explicitly by the way in which the engineer creates attributes within packages. A name and phone number might both be defined as attributes under a customer package. This implies a direct one-to-one association. The invoice number might be defined as an attribute and invoice item as a package, both under an invoice package. Stock item, quantity and cost attributes may further be defined under the invoice item package. This implies a one-to-one association between all of the invoice item attributes and also between the invoice number and the invoice item attributes.

[0454] Associations are also defined implicitly by the way in which attributes are used within an activity. An invoice activity may include the invoice number attribute as well as the customer name attribute. This defines an implicit association between the invoice package and the customer package because they are referenced in the same activity. This is despite the fact there is no explicit association defined between the two packages. Typically, an invoice will contain one or more invoice items whereas it will contain a single invoice number (a one-to-many association). The business process engineer can define this association by creating the attributes within a package and specifying that an invoice may contain many invoice items.

[0455] In some cases, a number of different packages and their contained attributes have an association in common with another package. For example, a staff member is associated with the office in which they work; a purchase order is associated with the office from which that order originates; a meeting is associated with the office in which that meeting is held. In this case, the office package should not be explicitly defined as a sub-package of each of the star purchase order and meeting packages as it would not identify the common or shared association that exists in each case. Rather, Office needs to be defined as an independent package, so that it can be referred to, as required, across many activities within various processes.

[0456] Packages that are the subject of shared associations should first be defined as packages independently of the packages with which they are associated. Once this has been done they can be linked to their associated packages. Alternatively, they can be defined implicitly as shared associations 0within an activity, by the simple fact they are referenced in common.

[0457] When referencing an attribute of a shared association in a business rule, it is important to fully qualify the attribute to include the association. For example, an organization might have a business rule that indicates the offices in which staff work must have a phone number, however this rule might not apply to offices in which meetings are held. In this case, the following business rule would be too restrictive:

[0458] Verify that the Office has a Phone Number.

The rule would need to be qualified such that the association between Staff and Office is included:

[0459] Verify that the Staff Office has a Phone Number.

[0460] Attribute References: When referring to an attribute (for example in a business rule), a business process engineer may either be referring to a single value or a set of values depending on the context. In the context of a sales order, if the engineer refers to a customer's name he is referring to a single customer. If, on the other hand, the engineer is describing the customers who may place an order then he might refer to the potential customers' names. This would be referring to all customers.

[0461] In a natural language, the distinction is made by using the plural form of the noun "name" to distinguish between a single name and a set of names. The description above qualifies the name as "customer's name" to indicate a specific reference to the name of a customer as opposed to the name of a staff member or supplier.

[0462] Using Visual Enterprise, "customer" would be defined as a package of attributes and "name" as an attribute. Visual Enterprise does not distinguish, in terms of syntax, between a single value and a set of values. All attribute values are part of a set such that a single value is simply a set that contains only one item. This set of values is known as a result set. Both attributes and packages have result sets. An attribute result set is a set of values for that attribute. A package result set is a set of values for each attribute of the package. Each of these sets must contain the same number of values, although some of the values may be null or not specified.

[0463] For example, a customer package may consist of a name attribute and a phone number attribute. For each value of the name attribute in the customer result set there must be a corresponding value for the phone number attribute and vice versa. This does not mean that every customer must have a phone number. For those that do not have a phone number the attribute value will be null (empty, blank).

[0464] A fully qualified attribute name is one that includes all of the packages to which an attribute belongs. For example, "Customer Address City" where City is an attribute of the Address package, which is a sub-package of the Customer package. To display fully qualified attribute names in business rules would overly complicate the reading of a rule. If there can be no ambiguity over the package to which an attribute belongs then Visual Enterprise will display only the attribute name, not the fully qualified name. If there could be ambiguity, for example because there are two "City" attributes, then Visual Enterprise will qualify the attribute name sufficiently to remove any ambiguity.

[0465] Attribute Result Sets: Business rules are executed within the context of an activity. This is true even where a rule has been defined in the Attribute Library against a package or an attribute. The attribute or attributes to which a rule refers may or may not be included in the activity. For example, a rule may be defined against a "Raise Purchase Order" activity, which sets the price of the products being purchased. The rule might be: Assign the Order Item Price=the Product Unit Price The Product Unit Price may not be one of the attributes listed for inclusion on the activity. This is because the Item Price will be visible and to display both attributes would duplicate information.

[0466] If, however, selection of the price is dependant upon criteria, such as the date, then these criteria will need to be specified to retrieve the correct result set for the product price. If a history of prices existed for each product then criteria, along the lines of the example below, would need to be added to the rule to uniquely identify the correct unit price: Assign the Order Item Price=the Product Pricing History Unit Price where today is between the Product Pricing History Start Date and the Product Pricing History End Date

[0467] Data Retrieval: When an activity is opened any data that has already been identified, either because it was recorded in an earlier activity or because it has been set by a business rule, is retrieved from the database. Data may also need to be retrieved from the database at other times during the completion of an activity. Such a situation would occur when:

[0468] The user selects a package instance (such as a Customer) and the activity definition includes a list of attributes related to that customer (such as the Name, Phone Number, Date of Birth).

[0469] A business rule references data not contained within the activity, in order to perform some verification or calculation.

[0470] Attribute Properties: There are a number of properties relating to attributes, which dictate the functionality or rules associated with that attribute. For example, the engineer can specify that the value of an attribute must be selected from one of a list of options. These properties include:

[0471] Data Type—such as Text, Number, Date, etc.

[0472] Control—this property determines the interface between the user and attribute in the Player environment. Options include Data Field, Radio Buttons, Check Box, and Drop-down List. Depending on the value of other properties some of these options may not be valid.

[0473] Options List—indicates a list of options from which the user can choose.

[0474] Extendable Options—whether or not the user can add additional items to the Options List.

[0475] Default—the default value for the attribute.

[0476] Mandatory—whether or not the user must supply a value for the attribute.

[0477] Editable—whether or not the user can modify the value of the attribute.

[0478] Visible—whether or not the attribute's value is displayed on the page.

[0479] Tooltip—a short description of the attribute that is displayed when the mouse is positioned over the attribute in the Player environment.

[0480] Minimum and Maximum Values—these properties are equivalent to a business rule that verifies whether the value entered by a user is greater than a minimum or less than a maximum value.

[0481] Minimum and Maximum Length (Text Only)—these properties control the minimum or maximum length of text that a user is permitted to enter.

[0482] Decimal Places (numeric only)—the number of decimal places to which the attribute's values are to be stored.

[0483] Units (numeric only)—such as miles, gallons, dollars, etc.

[0484] Background and Foreground color—foreground color is the color of the text entered by the user while background color is the color behind that text.

[0485] Input Format—this property controls the acceptable format for a value entered by the user. For example, an input format that specifies two decimal places would prevent the user from entering more than two decimal places.

[0486] Output Format—this property controls the format in which values are displayed. For example, an output format that specified one decimal place would display the value with only one decimal place, even though it may be stored to two or more decimal places.

[0487] Display Width and Units—these properties control the width allocated for an attribute on the page. The width refers only to the attribute value and not the associated label. The units can be specified as pixels, inches, centimeters and average character width.

[0488] Business Rules: Business rules are another component of the four cornerstones. Each business rule is dependant, not only on the definition of the rule, but also on the definition of the attributes on which the rule is based and the context in which the rule is executed. There are many types of business rule. The major determinant is the rule type (logic, verification, assignment, or action), which determines how the rule interacts with the process in which it runs.

[0489] Business Rule Types: Business rules fall into two categories: active or passive. An active business rule is one that affects the value or state of a Visual Enterprise attribute, package, activity or process. For example, "Assign the Price=the Quantity multiplied by the Unit Price" or "If the Total Price is greater than $1,000 then execute the Authorization by Sales Manager activity". The order in which active business rules are applied can be critical to the correct processing of rules. Visual Enterprise automatically determines the correct order, based on logical dependencies between the rules. A passive business rule is a business rule that performs some verification or validation and results in either a success or failure. For example, "New accounts can only be opened where the customer has provided proof of identification". The order in which passive business rules are applied will not affect the outcome.

[0490] Business rules may be associated with activities, packages and individual attributes. Package and attribute business rules are automatically associated with each activity in which the attribute or attributes are referenced. The business process engineer may elect to turn off or override such a business rule where it is not applicable to a particular activity.

[0491] A business rule may comprise a single statement or it may consist of a number of statements. In the latter case all of the statements must, in certain preferred embodiments of the method of the present invention, be appropriate in type. For example, a verification statement cannot be defined in a business rule that also includes an assignment statement. Some business rules may also be specified implicitly via an attribute property rather than explicitly as a business rule statement. For example, a rule that an invoice number can be no longer than 7 characters may be specified as a "Maximum Length" property of an attribute. The fact that a business rule appears as a statement in this VE blueprint section does not preclude that the functionality may be provided by Visual Enterprise exclusively as a property.

[0492] Business Rule Composition: After creating a business rule, the next step is to select the type of rule. The rule type determines the type of statements that the rule can contain, as per Table 3 below:

TABLE 3

Rule	Type
Verify that data being entered conforms to this rule	Constraint
Select a package	Constraint
Restrict a package	Constraint
Specify a condition to this rule	Logic
Assign a value	Active
Assign a default value	Active

[0493] In certain preferred embodiments of the method of the present invention constraint and logic rules are passive. The constraint and logic rules may optionally be applied to ensure that conditions are met, even where the constraint and logic rules are unable to change attribute values. Active rules can change the value of an attribute. The order in which the passive rules are evaluated may not significantly affect an implementation of a preferred embodiment of the present invention, but the order in which active rules are evaluated can be important because active rules often calculate a value based on other attribute values, which may in turn be affected by other active rules. If the sequence in which the rules were evaluated was allowed to change then the outcome could be different.

[0494] To prevent inconsistencies that arise from changes to the evaluation order, Visual Enterprise determines rule dependencies automatically. (Please note the section on Execution Order for more details.} Logic rules include variants such as "if . . . then . . . otherwise . . . " and "based on the value of . . . if it equals . . . ". Repetition variants, such as "while . . . do . . . " or "repeat . . . until . . . ", are also available. Verification or validation statements test a condition and either report an error or warning or perform some other specified action. A restriction statement restricts the available choices that a user can make to ensure the choice is valid for the particular circumstances.

[0495] Assignment statements can be further categorized into:

[0496] Attribute assignment—assign a value or expression to an attribute.

[0497] Package assignment—set an instance of a linked package based on specified criteria.

[0498] Action statements include those to:

[0499] Retrieve data from the database.

[0500] Initiate a process.

[0501] Schedule a process that has been initiated but not yet commenced.

[0502] Initiate an activity.

[0503] Schedule an activity that has been initiated but not yet commenced.

[0504] Allocate an activity to a group or a particular user.

[0505] Create an audit trail.

[0506] Execute an external process.

[0507] External processes may be referenced by any of the types of statement detailed above. For example, an external process may be required to validate a credit card number; it may be required to retrieve an external document, which is then assigned to an attribute; or it may be required to send a message to a telephone pager. An external process, if it were to be included in a transactional rule, would need to provide transactional integrity and consistency by means of methods to begin, commit and rollback a transaction.

[0508] Visual Enterprise makes a distinction between errors resulting from hardware or software deficiencies and exceptions raised as a result of the processing of business rules. An example of an error might include:

[0509] A hardware failure, such as the server crashing, insufficient disk space or a telephone line dropping out.

[0510] A software failure, such as the browser crashing or the operating system hanging.

[0511] An example of an exception might include:

[0512] A data-based exception where a value of an attribute, as entered by the user, does not satisfy one or more business rules; or

[0513] A time-based exception where an activity is not completed within a specified period.

Should an error occur, Visual Enterprise attempts to report the error to the user, allowing the user to take whatever remedial action is required. If possible, Visual Enterprise logs an entry and/or send an e-mail message to an administrator detailing the nature of the error. Any further processing of business rules for that activity is cancelled. Business rules already processed within the transaction; during the processing of which the error occurred, are rolled back.

[0514] Business rule exceptions can be described as either internal or external to an activity. An internal exception is one where control of the activity remains with the user allowing the user to modify an attribute or attributes such that the verification rules are met. An external exception is one where the activity is automatically cancelled and control flows to the activity scheduler via the failure route for that activity. Most exceptions can be detected and handled during data entry before the commencement of the processing of a transaction. However, if this is not the case then the entire transaction will be rolled back to ensure the integrity and consistency of the database.

[0515] In the case of an external exception, the handling of the exception is taken outside the control of the user to whom the activity was assigned. The business process engineer is able to define a complete workflow to be followed for each defined exception. This workflow may include an escalation of the exception should further data or time constraints not be met by the allocated user or users. If both an internal and external exception were to occur within an activity Visual Enterprise would immediately allow the workflow activity scheduler to deal with the external exception. Any data entered for the activity in error would be passed to the "Exception" activity however it would not be committed to the database. This would allow the user handling the exception to either correct that data, where required, or to reset and then re-enter the data

[0516] The business processes being defined may also require warnings to be issued to users at various stages. Visual Enterprise provides this functionality via a variation on the verification/validation business rule definition. This variation would indicate a warning was to be issued rather than raising an exception. In this case, processing would continue normally once the warning had been issued. Such warnings would optionally be issued before any data being committed to the database.

[0517] Attribute Associations: One of the defining differences between Visual Enterprise and other application development environments is the way associations between various attributes are defined. Although Visual Enterprise is built on a standard relational database, the engineer sees associations within the context of the packages, activities and processes in which they are referenced. This affects both on the storage of data and the way such data is retrieved. For example, in continuing the invoice activity example, which has been used earlier in this disclosure, the following attributes may be included:

[0518] Invoice Number

[0519] Invoice Date

[0520] Customer Name

[0521] Customer Phone Number (display only)

[0522] Customer Address (display wily)

[0523] Multiple invoice item attributes

[0524] This activity might be used to enter a new invoice. Visual Enterprise needs to be able to determine how and when to retrieve the customer's phone number and address. This is done through the definition of packages and key attributes.

[0525] In the above example, the business process engineer would specify that the invoice detail attributes were part of the main or controlling package. The invoice item attributes would be part of a separate Invoice Item repeating package to allow multiple items to be entered. The Invoice Number would be defined as the key attribute for the main package.

[0526] The customer package and therefore the customer attributes (name, phone number and address) would be linked to the main package. This would indicate a one-to-one association from the invoice to the customer (and a one to many association from the customer to the invoices). For details on how the customer phone number and address are retrieved refer to the section below on Data Retrieval.

[0527] Data Consistency: As a multithreaded, multi-user system, Visual Enterprise must ensure that data is consistent across processes. It is also possible for an engineer to define inconsistent rules. Visual Enterprise detects such inconsistencies and ensures that they are resolved before the application is used.

[0528] Process Interactivity: Visual Enterprise only stores the latest submitted version of an attribute's value. This means that one process may modify the value of data created by an activity of a second process while that second process is still in progress. For example, consider an invoice entry process that consists of two activities, one to enter the invoice details and a second to approve the invoice, each step to be performed by at least two different users. When a user com-

pletes the first activity, the invoice details are submitted to the database. Before the next user (e.g., a supervisor) has the chance to approve the invoice, a third user (e.g., a manager) may run a separate process to modify invoice details. If this manager submits changes to the same invoice before the supervisor begins the approval activity then these changes will be presented as the invoice details to be approved.

[0529] If the supervisor has commenced the approval activity but not submitted when the manager submits the changes to the invoice detail then these changes will not be presented to the supervisor. Visual Enterprise will, however, notify the supervisor that the data has changed so that he can refresh the data before approving the invoice.

[0530] If the supervisor had approved the invoice before the manager tried to modify the invoice details it would be left to the business process engineer to decide whether this was a valid activity or not. It may be a business rule of the organization that approved invoices cannot be modified but can only be cancelled.

[0531] Ambiguous Rules: It is possible for a business process engineer to define conflicting or ambiguous business rules. For example, one rule may be defined offering a discount on certain products at a certain time of year. Another rule may be defined which provides a discount to loyal customers. Does this mean that loyal customers receive an additional discount if they purchase the specially discounted products or do they simply receive the greater of the two discounts?

[0532] Visual Enterprise optionally detects conflicting and ambiguous business rules and prompt the engineer to redefine the conflicting rules or to clarify the ambiguity. This will be done when an engineer chooses to publish an application by searching for rules that apply to the same attribute or attributes. It will then generate a conflict resolution form that must be completed before the application can be published. For example, suppose the following separate business rules were defined:

[0533] If the Order Date is between Halloween and Christmas Eve and the Product Type is equal to Sports Goods then

[0534] Assign the Discount=10%

[0535] If the Customer Status is equal to loyal then

[0536] Assign the Discount=5%

[0537] Visual Enterprise would detect that the Discount attribute was being assigned by two separate rules and that the conditions applied in each case may conflict, resulting in an ambiguity. The engineer could resolve any ambiguity by combining the two rules into one. The decision might be that the sports goods discount takes precedence over the loyal customer discount.

[0538] If the business wanted to give the maximum discount, catering for situations where the discount percentages may change from time to time then the two rules could be combined as follows:

[0539] If the Order Date is between Halloween and Christmas Eve and the Product Type is equal to Sports Goods then Assign the Sports Discount=10%

[0540] If the Customer Status is equal to loyal then Assign the Loyalty Discount=5%

[0541] Assign the Discount=maximum (the Sports Discount, the Loyalty Discount)

If the business wanted the discounts to be cumulative then the rule would be specified as:

[0542] Assign the Discount=the Sports Discount+the Loyalty Discount

If no discount applied to a particular purchase order then the result of the above rule would be a discount of 0.

[0543] Data retrieval: A single attribute may be referenced in a number of different activities and processes. When retrieving data Visual Enterprise uses the context in which an attribute is referenced to determine the retrieval criteria. In order to retrieve data, Visual Enterprise determines an association list for each attribute within a process. This list is based on explicit associations that the business process engineer may have defined as well as any associations implicit in the process definition. An implicit association is one where two unlinked packages have been included in the same activity. This defines an association between the attributes of both packages.

[0544] When the value of an attribute is either entered by the user or retrieved by Visual Enterprise the value of any related attributes will optionally, in turn, be retrieved. Let's extend the earlier example of the entry of an invoice to an activity that allows the invoice to be modified. The invoice number would again be defined as a key attribute. When the user enters an invoice number Visual Enterprise will retrieve the values for any related attributes, in this case the invoice date, customer name and invoice items. The customer phone number and address are not directly related to the invoice number, as they were not input in the same activity as the invoice number.

[0545] The customer phone number and address are related attributes of the customer name because they belong to the same package. As soon as Visual Enterprise retrieved, the customer name (as a result of the user entering an invoice number) Visual Enterprise would in turn retrieve the associated phone number and address.

[0546] When searching for related attribute values more than one instance may be found which matches the key value. This set of values forms the result set for the related attributes. The result set may or may not be appropriate for the definition of the activity within which the attribute is being retrieved. For example, a multi-value result set cannot be displayed in a single data field however it can be displayed in a list or table. In this case, an exception would be raised.

[0547] Key Attributes: One or more attributes within a package or activity may be unique. For example, within an activity to record customer details both the company name and business number may be unique. This means no two companies can have the same name and no two companies can have the same business number.

[0548] Unique attributes in one activity will typically be used as key attributes in other activities. This is because they guarantee that the result set for any related attributes, in a one-to-one association, will contain a single value. When defining activities that reference customer details, for example, the business process engineer should be able to interchange the company name with the business number as and when required. In effect the business number is an alternate key for the company name and the company name is an alternate key for the business number.

[0549] An attribute that is defined as being unique within an Activity is an alternate key for all of the other unique attributes defined in that same activity. This alternate key association extends beyond the activity in which the attributes

are defined as unique to any activity in any process. When, for the purpose of retrieving data, Visual Enterprise builds a search-list of activities where both the base attribute and its key attribute are referenced within the same activity, the search-list will be extended to include any activity where the base attribute and an alternate key attribute are referenced.

[0550] To continue the above example, let's add a Customer ID to the list of customer related attributes. One activity might be defined to collect the name and ID attributes only (both defined as unique). A second activity may be defined (either within the same process or in another process) to collect the phone number and address, with the ID used as the key attribute. In this case no one activity references the customer name, phone number and address. However, for the purpose of retrieving the phone number and address, the "Customer Name" attribute's alternate key, "Customer ID", can be used as the link between all attributes.

[0551] The above approach to data retrieval imposes a methodology on the business process engineer that involves retaining a consistent definition of all attributes across all activities and processes. For example, if a number of address attributes (street, city, postcode, etc.) were referenced in one activity as a postal address and in a separate activity as a property address Visual Enterprise would not recognize the distinction. Separate attributes would need to be defined for postal and property addresses so that the definition of these attributes was consistent in all activities. In other words, whereas in the first scenario an "Address" is sometimes a postal address and sometimes a property address, in the second scenario a "Postal Address" is only ever a postal address and a "Property Address" is only ever a property address.

[0552] Business rules may need to be applied at various stages of an activity or process. In certain circumstances, the business process engineer will determine this stage based on the type or nature of the rule. The engineer may specify a rule as:

[0553] A means of scheduling an activity (wait until a condition holds true)

[0554] A pre-condition to an activity

[0555] A means of determining the most appropriate user to whom an activity should be assigned

[0556] A means of setting a default value for an attribute

[0557] A condition under which an attribute will be hidden (security)

[0558] A condition under which an attribute will be protected (security)

[0559] A means of reversing an update to an attribute when the value of an attribute on which it is dependent is modified.

[0560] Visual Enterprise will determine the appropriate stage at which to apply a business rule (taking into account any settings as described above). These stages are:

[0561] On the commencement of an activity (Pre-activity)

[0562] During an activity (Mid-activity)

[0563] When the user submits an activity for completion (Post-activity)

[0564] On commencement of a transaction (Pre-transaction)

[0565] On completion of a transaction (Post-transaction).

[0566] Pre and post-transaction business rules are defined as being part of the transaction, not external to it, and therefore inherit the same properties as transactional integrity and consistency. When pre-transaction rules are executed, they do so in the context of the previously committed values of each attribute and not the values as specified within the current activity.

[0567] The order of processing of an activity transaction is:

[0568] Lock all attribute values referenced by the transaction.

[0569] Execute pre-transaction business rules.

[0570] Assign package values of each attribute to the database.

[0571] Execute post-transaction business rules.

[0572] On success commit (and release locks); otherwise roll back.

[0573] Mid-activity business rules are applied whenever a user changes the value of the attribute against which the rule is defined or whenever a user changes the value of any attribute on which that rule is dependent. For example, a mid-activity business rule might be associated with a "Date Of Birth" attribute to the effect "The Date Of Birth must be earlier than today's date". This business rule would be applied to the attribute whenever the user changed the date of birth and moved the cursor away from that attribute.

[0574] As a further example, a package might be established consisting of "Stock Item", "Quantity" and "Cost" fields. A business rule might be associated with the "Cost" attribute to the effect "Cost equals Unit Price of Stock Item multiplied by Quantity". When the user modifies either of the attributes on which the "Cost" business rule is based ("Stock Item" or "Quantity") the business rule would be applied and the updated cost would be displayed on the screen. The "Cost" business rule is also dependent on the "Unit Price" attribute however in this example this would be a display only attribute and hence could not be modified by the user.

[0575] Rules applied as mid-activity business rules will always be re-applied when the activity is submitted for completion. This needs to be done in case, in the meantime, another user changes the value of any of the attributes on which a rule is based.

[0576] Execution Order: The order in which business rules are executed is determined by Visual Enterprise based on the dependencies of each business rule. For a business rule to be processed correctly it is important to execute that rule only after executing all rules on which the rule is dependent. For example, a package may contain four attributes: A, B, C and D (where A, B and C are display only) with the following associated business rules:

$$A=B+C+D$$

$$B=D*4$$

$$C=D/2$$

[0577] Visual Enterprise would set the order of execution as follows:

$$B=D*4 \hspace{4cm} 1.$$

$$C=D/2 \hspace{4cm} 2.$$

$$A=B+C+D \hspace{3.5cm} 3.$$

[0578] Following execution of any part of a business rule that assigns an attribute's value, further execution of the rule is suspended to allow for the execution of any business rule that is dependent on the attribute being assigned. Any such business rule may itself be suspended for the same reason.

Once the business rules of the dependent attributes have been executed, execution of the original business rule is resumed. There is no limit on the number of rules that can be suspended at any one time.

[0579] This approach could lead to a cyclical reference scenario whereby execution might never end. To prevent this, a business rule that is currently suspended will not be allowed to execute a second time. The following example is based on an "Invoice Number" attribute which contains a cyclical reference but is nonetheless a valid business rule:

[0580] If the Invoice does not have a Invoice Number then

[0581] Assign the Invoice Number=the Next Invoice Number

[0582] Assign the Next Invoice Number=the Next Invoice Number+1

[0583] otherwise

[0584] Verify the Invoice Number is less than 1000

[0585] The exemplar business rule above is dependent upon both the "Invoice Number" and "Next Invoice Number" attributes. During execution of this business rule, after each assignment statement, execution would be suspended in order to execute any dependent business rules. As a dependent of itself this business rule would not be re-executed even though in this particular case it would not result in an endless loop. However if the rule were rewritten as follows, re-execution would result in such a loop:

[0586] If the Invoice does not have a Invoice Number then

[0587] Assign the Next Invoice Number=the Next Invoice Number+1

[0588] Assign the Invoice Number=the Next Invoice Number

[0589] otherwise

[0590] Verify the Invoice Number is less than 1000

[0591] Given that either of the two variants will not be re-executed, the rule would still need to be rewritten if it were to serve the purpose of always verifying an invoice number was less than 1000. Splitting the business rule into two rules as follows would achieve the correct result:

[0592] Verify the Invoice Number is less than 1000

[0593] If the Invoice does not have a Invoice Number then

[0594] Assign the Invoice Number=the Next Invoice Number

[0595] Assign the Next Invoice Number=the Next Invoice Number+1

[0596] In this case during execution of the latter Me, after the first assignment statement, execution of the rule would be suspended and the verification rule would be executed as a dependent. Were it to fail, all business rules and transactions would be rolled back.

[0597] Where a business rule contains conditional logic, Visual Enterprise will not pre-evaluate the condition in order to determine the correct execution order for business rules. This is because the evaluated results of conditions may change as each rule is executed. Even though in certain circumstances the results may be consistent, Visual Enterprise will not endeavor to detect these circumstances. This may mean Visual Enterprise is unable to determine the correct execution order. In these cases the business process engineer will be given a warning and may elect to rewrite the rules in question in order to remove any ambiguities.

[0598] Visual Enterprise business rules are based on a purpose-designed grammar. The grammar defines the structure and meaning of the rule that can be defined. The full grammar is outlined in an appendix to this document. The grammar also contains a sub-grammar based on a standard algebraic and mathematical grammar.

[0599] From the grammar, Visual Enterprise is able to determine a list of valid options from which an engineer can choose at each point through the process of defining a rule. Should the grammar change in any way at some time in the future, Visual Enterprise will automatically reconfigure the list of valid options based on the new grammar. Associated with the grammar is a corresponding JavaScript Translation grammar. This grammar defines the valid JavaScript syntax required to translate a business rule into JavaScript just in time for execution.

[0600] Enterprise Interpreter. The Enterprise Interpreter is the engine that drives the execution of the processes, activities, attributes and rules of a Visual Enterprise application. It can respond to actions taken by a user, such as submitting an activity for completion, initiating a scheduled activity at the required time, or responding to a request from an external process.

[0601] The Enterprise Interpreter is a multi-threaded environment meaning that it can handle requests or tasks from multiple users and other sources at the same time. Once it has serviced a request, it will send a response back to the client, including an indication of whether the request was serviced successfully or not.

[0602] Player Activity Page: A number of the requests to the Enterprise Interpreter are initiated from the Player activity page. The major requests from this page include:

[0603] Initiate a new process.

[0604] Open an existing activity from the Intray.

[0605] Submit an activity.

[0606] Cancel an activity.

[0607] When a new process is initiated, the Enterprise Interpreter creates a new process instance, against which is collected a number of related properties, such as the date and time the process was initiated, and the user who initiated the process. Each process instance may progress through different stages:

[0608] Initiated—a process may be initiated either by a user or by an event.

[0609] Completed—a process is deemed to have completed when the last of its activities has been completed.

[0610] Exception—an exception is deemed to have occurred in a process when an exception occurs on one of its activities and that exception prevents the process from completing.

[0611] Cancelled—a process can be cancelled by an administrator.

[0612] At each stage additional properties are collected similar to those mentioned above. The Enterprise Interpreter also maintains data on each of the activities that are a part of the process, including the application data (attributes) collected by those activities.

The first activity in the process is also initiated at this time. If it is a user input activity then the activity is opened and displayed to the user.

[0613] Open an Existing Activity: When the user opens an activity the Enterprise Interpreter retrieves a number of different types of data in order to display the activity to the user. The Enterprise Interpreter may also record a number of prop-

erties relevant to the activity and the stage to which the activity has progressed. An activity can progress through the following stages:

[0614] Initiated—this stage occurs either when a new process is initiated or when a previous activity in the process is completed.

[0615] Allocated—this stage represents the allocation of the activity to a group of users, which may occur as soon as the activity is initiated. If the activity is scheduled then this stage will not occur until the scheduled date and time.

[0616] Assigned—this stage occurs when the activity has been assigned to a single user. If the assignment properties of an activity identify a single user then the activity is assigned as soon as it is allocated. If, on the other hand, the properties identify a number of possible users then the activity is assigned when the first of those users opens the activity.

[0617] Started—this stage indicates that the activity has been opened by a user or that an automated activity has commenced processing.

[0618] Incomplete—an activity can be saved by a user whilst incomplete if he does not have sufficient information or time to submit the activity.

[0619] On Hold—a user can place an activity on hold if the information required to complete an activity will not be available for some time.

[0620] Completed—this stage occurs when a user successfully completes an activity or, in the case of an automated activity, when the activity has finished processing.

[0621] Exception—this stage is reached if an exception occurs which prevents the activity from being completed. This may be because the user has not completed the activity by a completion deadline.

[0622] At each of these stages, properties relevant to the stage are recorded in the database. These properties are similar to those recorded for the different stages of a process and together they present a workflow history of the process.

[0623] The information or data that the Enterprise Interpreter needs to retrieve when an activity is opened includes:

[0624] The list of package, attributes and associated properties included in the activity definition.

[0625] Details of all of the business rules that apply to the activity.

[0626] The application data that has already been recorded in previous activities.

This information is used to build the activity page that will be displayed to the user. It will also be used to validate data recorded by the user and perform any calculations thereon.

[0627] When the user submits an activity all of the data entered by the user is passed to the Enterprise Interpreter. It is the role of the Enterprise Interpreter to process each of the business rules associated with the activity, and if there are no errors, to submit the data to the database. Once this has been done, the activity is progressed to the completed stage. The Enterprise Interpreter also determines whether there are any activities that should follow and if so it initiates those activities. If any of those activities is allocated to the user submitting the former activity then the Enterprise Interpreter will automatically open that activity for the user. This allows a seamless workflow whereby the user is not required to always return to the Intray in order to open the next activity in the

process. If the user is not ready to complete the new activity then it can be cancelled or saved incomplete.

[0628] Cancel an Activity: To cancel an activity means that the completion of the activity is temporarily cancelled, not permanently. The activity can only be permanently cancelled by canceling the entire process. To do otherwise would contravene the workflow requirements designed by the business process engineer. When an activity is cancelled, it remains in the user's Intray and must be completed at a later time. Any data entered by the user since the activity was opened will be discarded and no business rules will be executed. The Enterprise Interpreter may not be required to perform any processing when an activity is cancelled.

[0629] Enterprise Interpreter Engine: The first task to be completed by the Enterprise Interpreter for any request is to verify the authenticity of the request. The data sent to the Enterprise Interpreter with any request must include the organization, user and session keys. The Enterprise Interpreter sends these keys to the client browser or remote device when the user logs into Visual Enterprise. The organization and user keys are hidden keys allocated by Visual Enterprise when the organization is registered and the user granted access. They are never communicated to users or engineers. A new session key is allocated by Visual Enterprise each time a user logs in. Likewise, this is a hidden key not communicated to users or engineers. Further, the session key is allocated using a one-way encryption algorithm, such that it cannot be "guessed" by someone trying to gain unauthorized access to an organization's processes or data

[0630] Mode of Operation: The general operation followed by the Enterprise Interpreter when a user logs into Visual Enterprise to open and complete an activity is described in the steps below. Note that various exceptions and options will cause deviations to these general steps.

[0631] 1. The user logs into Visual Enterprise and a session key is returned to the browser or remote device. It is stored as a hidden object not visible or accessible to the user. The login request is made to the server via ASP pages and the Execution Mode Process Server. The session key is used in all future communications with the Enterprise Interpreter. The reader may assume that the session key will be checked using the Security Processor for all communications, although this will not be described in the following steps to avoid complicating the discussion. Similarly, references to the ASP pages and the Execution Mode Process Server will be restricted to points where these add significant functionality; generally all traffic flows through these components and this can be assumed.

[0632] 2. The Intray page is displayed upon receipt of a successful login. The Intray page on the client requests all active activities allocated to the current user. The Activity Sequencer handles all requests for details of activities and processes to which the user has been granted access. The list of processes is displayed when the user chooses the New Process option. To reduce download times, new processes are grouped under categories and only the processes of selected categories are displayed.

[0633] 3. From the Intray the user may select a new activity, a previously opened activity, or an activity placed on-hold. The request for an activity is passed to the Execution Mode Process Server, which will coordinate other components as follows:

[0634] If the request is for a new process then control is passed to the Activity Sequencer to create a new Process

Instance. The first activity of the new process is identified at which point processing continues along common lines for both new and existing activities.

[0635] Control is passed to the Activity Sequencer to retrieve various details and properties of the activity including the attribute names, default values, the screen controls to be used, and the order in which the attributes are to be displayed. The Activity Sequence may also be required to progress the activity to the Started stage if it is not already. This will prevent other users from being assigned to the same activity.

[0636] The Data Coordinator is the next component to get control. This component retrieves any data that is to be included in the activity and that has been recorded earlier in the process.

[0637] Control then passes to the Rules Processor to process any pre-activity rules. Such rules may pre-populate attribute values with a default value, such as today's date. The Rules Processor also builds, for return to the client browser or remote device, the JavaScript translations of any mid-activity business rules identified as being able to be processed on the client.

[0638] Finally, control passes to the Response Coordinator to build the HTML and JavaScript page definition, which will allow the user to complete the activity.

[0639] 4. As the user records values for each of the attributes on the page a number of mid-activity business rules may need to be processed. There are three distinct types of rules that can be triggered during the completion of an activity prior to the activity being submitted:

[0640] A rule that retrieves data from the database for the purpose of providing the user with a list of options; typically be triggered by the user clicking a "Find Button" or expanding a drop-down list A request is made to the Enterprise Interpreter to process these rules.

[0641] A rule that involves only data already available on the page. Entering or modifying the value of an attribute on which the rule is dependant would trigger this. Such rules can be processed within the client browser or remote device so no request is made to the Enterprise Interpreter.

[0642] A rule that references data not available on the page. Again, entering or modifying the value of an attribute would trigger this type of rule. A request is made to the Enterprise Interpreter to process these rules. No data is added to or modified in the database during this processing. Only the values of attributes on the page are updated.

[0643] 5. When the user presses the Submit Button, a request is sent to the Enterprise Interpreter to submit the activity. The request is passed to the Execution Mode Process Server, which will coordinate other components as follows:

[0644] Control is passed to the Activity Sequencer to retrieve various details and properties of the activity.

[0645] The Rules Processor is invoked to execute all of the business rules associated with the activity, including those already processed by the client

[0646] If no exceptions are raised then control is passed to the Data Coordinator to submit the data to the database.

[0647] The Activity Sequencer then gets control, firstly to progress the activity to the completed stage. At this point all updates to the database are committed. The

Activity sequencer then determines whether there are any subsequent activities to be initiated. If any of these activities are dependent on a condition then the Rules Processor will be invoked to test that condition.

[0648] Finally, if one of the new activities has been allocated to the current user then the Response Coordinator is invoked to build the HTML and JavaScript page definition for the new activity.

[0649] 6. The new activities that have been initiated will be visible in the Intrays of the relevant users allocated to each activity.

[0650] 7. Either the next activity will be displayed or the user will be redirected to the Intray. The process can then repeat from step 3.

[0651] Active Server Pages: The Active Server Pages serve as an interface between the client browser and the Enterprise Interpreter. The ASP pages present the data that comes from the Enterprise Interpreter in a way acceptable to the client browser, including packaging data into XML fragments where required. Only minimal processing occurs in ASP pages. These pages invoke the Enterprise Interpreter components.

[0652] Response Coordinator: The role of the Response Coordinator is to format a response to requests relating to activities from the client browser or external device. The response is in the form of an XML fragment, which includes all of the relevant information requested. For example, the activity data component of such a fragment would be structured as shown in FIG. 22.

[0653] The Response Coordinator formats a number of different types of data:

[0654] Activity Definition—including relevant activity, package and attribute properties. These properties include details about the display width of an attribute, its input and output formats, etc.

[0655] Business Rules—JavaScript translation of all rules to be executed on the client as the user enters data, as well as references to all rules that must be executed on the server.

[0656] Activity Data—as shown above.

[0657] Validation Errors—details of any errors that may have been detected during server-side execution of business rules.

[0658] The nature of the response to the client is also dependant upon the type of request. These include request to:

[0659] Preview an activity—only the Activity Definition needs to be included in the response.

[0660] Open an activity—the Activity Definition, Business Rules and Activity Data all need to be included in the response.

[0661] Execute rules—the response is only in terms of a success or failure or the rule(s) with details of any errors being included.

[0662] Submit an activity—this involves the same response as per executing rules and, depending upon workflow, it may also include that required to open a subsequent activity.

[0663] No distinction is made in the response in relation to the device from which the request originated. The format of any response is the same whether the request came from a client browser or from any form of external device, such as a hand-held device.

[0664] The Rules Processor: The Rules Processor controls all aspects of the determination, translation and execution of business rules. It can be called upon to perform a number of tasks:

[0665] Prepare all aspects of an activity pertaining to business rules.

[0666] Execute a rule or list of rules associated with an activity.

[0667] Execute a condition associated with a particular link between two activities in the workflow.

[0668] Execute selected rules relating to specific tasks, such as the setting of time criteria or the determination of the assignment of an activity to a user.

[0669] Execute a condition associated with a data-based event.

[0670] The preparation of an activity involves retrieving the list of rules that apply to the specified activity and determining the trigger point for each of these rules. Some may need to be triggered when an activity is loaded, others when the user changes the value of a particular attribute or when the activity is submitted.

[0671] Once the trigger points have been calculated, the Rules Processor makes a determination on whether each rule can be executed by the client browser or whether the nature of the rule requires execution to be performed by the server. The latter will be required whenever the rule refers to one or more attributes that are not included on the activity, and hence not available to the client.

[0672] The Rules Processor is also required to determine the correct execution order and dependency list for all rules. The algorithm used to do this involves calculating the attribute dependency list for each rule and ensuring that a rule that assigns or modifies the value of an attribute is ordered before any rule that references that attribute.

[0673] Before any rule can be executed, it must be translated to JavaScript. This translation is always performed Just-in-Time. The definition of Just-in-Time is different depending on whether the rule is to be executed on the client or on the server. For server-side execution, the rule is literally translated just in time for it to be executed. However, for client-side execution, the rule is translated to JavaScript when the request is made to open an activity. The translation is packaged up with other data by the Response Coordinator and returned to the client browser. The translation is done at this time for performance reasons, so that control does not need to be returned to the server to execute such rules. This greatly improves performance, as Visual Enterprise is able to make use of the normally under-utilized and dedicated processing power of client processors.

[0674] The final task of the Rules Processor is to execute rules. This involves calling on the JavaScript Interpreter to execute each translated rule, making available to it the value or values of any attributes the rule might reference. The Rules Processor also takes the results of the execution, whether that be successful or not, and returns them to the Enterprise Interpreter. At this point the processing will vary depending upon the nature of the request:

[0675] For a submit request, if the processing was successful then any assignments will be taken by the Data Coordinator to be stored to the database. If the execution of one or more rules resulted in an exception then the details of that exception will be taken by the Response Coordinator and returned to the client.

[0676] For a mid-activity request, all assigned values will be taken by the Response Coordinator and returned to the client. Details of any exceptions will similarly be dealt with.

[0677] For the execution of a specific condition or rule, the results are returned to the Enterprise Interpreter to control the processing of workflow or other processing.

[0678] Activity Sequencer: The Activity Sequencer performs several functions related to activities. Where a response is required, the data will be packaged in an XML fragment. Functions performed by the Activity Sequencer include:

[0679] Retrieve a list of activities to be included in the user's Intray.

[0680] Retrieve a list of process categories to which a user has access.

[0681] Retrieve a list of processes, for a selected category, to which a user has access.

[0682] Create a new Process Instance. A process instance is an instantiation of the process being initiated. When the Activity Sequencer creates a new instance it also identifies the first activity of that process to be initiated. Other activities in the process are not initiated until all preceding activities have been completed. Depending on the flow of the process, or if the process is cancelled, some activities may never be initiated for a particular process instance.

[0683] Allocate an activity to a user or groups of users who have been granted access.

[0684] Assign an activity to a user. This assignment is based on options specified by the business process engineer. One of those options is to assign the activity to the first user who opens it.

[0685] Mark an activity as completed.

[0686] If there are events dependant upon the initiation or completion of an activity then the Activity Sequencer will invoke the Event Scheduler to initiate those events. Such events include:

[0687] A start deadline for an activity—the Activity Sequencer requests that such events are cancelled when the activity has been started before the deadline.

[0688] A completion deadline for an activity—the Activity Sequencer requests that such events are cancelled when the activity has been completed before the deadline.

[0689] A scheduled time for the activity to start

[0690] Other events, which are relative to the initiation or completion of an activity.

[0691] Data Coordinator. The role of the Data Coordinator is to retrieve, add, modify or delete application data. It can be called upon to perform a number of specific tasks:

[0692] Retrieve all of the data associated with a particular activity instance. This includes retrieving data submitted by prior activities of the process as well as retrieving the default values of any attributes not yet recorded.

[0693] Retrieve specific data associated with the instances of a particular package. For example, retrieve a list of customers, including the customer name, date of birth, social security number, etc.

[0694] Retrieve data associated with a specific instance of a particular package.

[0695] Add data associated with a new instance of a particular package to the database.

[0696] Update data associated with an existing instance of a particular package in the database.

[0697] Remove data associated with an existing instance of a particular package in the database.

[0698] Retrieve data linked to an existing instance of a particular package from the database. For example, retrieve the pricing history of a particular product.

[0699] The Data Coordinator must deal with differing versions of data. It must cater for the fact that the value of an attribute requested by an activity may not exist. This may be because the attribute itself did not exist when the data associated with the selected package instance was first added. For example, a customer may not originally have defined a phone number attribute. Details of any customers added to the database at the time will therefore not have a phone number, even though subsequently a phone number attribute is added to the package Depending upon the options selected by the business process engineer, if no phone number exists the user may either be required to enter one before being able to submit changes to the customer details or may be able to continue without one.

[0700] Event Scheduler. The Event Scheduler accepts requests to initiate time-based events and to cancel time-based events. The scheduler will place all events into an event queue in sorted date/time order. Details recorded in the event queue include:

[0701] The type of event.

[0702] The date and time the event will occur.

[0703] The action to be performed when the event does occur.

The Activity Sequencer and the Services Queue are the primary sources of events.

[0704] Enterprise Monitor: Users can initiate processes and activities via the Intray or New Process options. Processes and activities may also be initiated as a result of time-based and data-based events. These events include:

[0705] A scheduled time has been reached

[0706] A signal from an external source that Visual Enterprise is monitoring is detected

[0707] Data meets a pre-defined rule.

[0708] The Enterprise Monitor is used to initiate processes or activities based on events rather than user control. Examples include end of month processing and workflow exception handling when activities are not completed within a specified time.

[0709] The Enterprise Monitor continually scans the Event Queue waiting for the specified date and time to arrive. Data-based events are added to the queue with the current date and time to ensure they are processed immediately. The queue contains details of the action to be taken when the event occurs. Possible actions include:

[0710] Initiate a process

[0711] Initiate an activity

[0712] Raise an exception on an activity

[0713] Initiate a service, such as printing a report.

[0714] The Enterprise Monitor is also responsible for detecting data-based events. A databased event is one that occurs when a condition based on one or more items of data is met. For example, "Initiate the Stock Ordering process when the Stock Quantity drops below a pre-defined threshold". Continually testing these conditions would, be an intensive operation. Visual Enterprise will only test these conditions whenever any attribute referenced in the condition has been modified. Once the condition is met the event will be added to the Event Queue and handled by the Enterprise Monitor as outlined above. Some activities will submit opera-

tions to the Services Queue, such as printing, waiting for remote events or file operations. These activities may require a confirmation before starting the next activity. Once the Enterprise Monitor receives this confirmation it will initiate the next activity via the Execution Mode Process Server.

[0715] Services Queue: Visual Enterprise has a transaction server at its core that receives requests to perform operations. The transaction must ensure the consistency of the underlying data and hence cannot allow multiple accesses to the same data while an update is occurring. To keep data consistent, it is necessary to guarantee that only one activity updates any given piece of information. To reduce the chance of a conflict, a transaction must occur in the smallest possible time.

[0716] To avoid any long operation from tying up the transaction engine, operations such as loading remote data, sending entails, and printing are performed outside the main transaction. These are submitted to the Services Queue. If an activity relies on the outcome of the Services Queue operation then the Enterprise Monitor will be notified of the outcome and the process can be progressed accordingly.

[0717] Security: Visual Enterprise deals with a variety of different types and models of security. Each one is designed to handle a specific purpose. For example, user authentication is used to ensure that only valid users can log into the site, session authentication is used to prevent unauthorized users gaining access to the site via means commonly referred to as "hacking", while the secure sockets layer is used to protect the privacy of an organization data as it is transported across the internet between a user and the Visual Enterprise server.

[0718] Registration: The registration process is designed to allow an organization to grant access to authorized business process engineers and users. It generates an Organization ID, which is one of the three pieces of information required by a user or engineer to gain access to the site. As well as incorporating the first five characters of the organization name, it includes a component that is randomly generated to greatly reduce the risk of an unauthorized user being able to guess the ID.

[0719] A registered organization may optionally also allocated a unique key, not communicated with engineers or users, which is used as part of the session authentication process. As part of the registration process, access may be granted to the administrator (who has registered the organization) to enable him to grant access to engineers and users.

[0720] The registration process may also prevent or impede an individual or organization from "squatting" on the name of another organization. It does so by allowing any number of organizations to be registered with the same name. Only the unique Organization ID, and not the Organization Name, is used when gaining access to Visual Enterprise, or when undertaking business-to-business or business-to-customer transactions.

[0721] User Authorization: User (and Engineer) authentication revolves around the three authenticators: Organization ID, login name and password. The Organization ID is explained in the section above on registration. If an organization would like to allow users to log into the Player without having to specify the Organization ID, then an administrator can set up a secure web site that allows this to happen. The organization can place its own security measures on that site (on the organization's own server) to prevent unauthorized access. That site would require the user to supply the login

name and password and then transmit those details, along with the Organization ID (built into the page) to the Visual Enterprise site.

[0722] Only those engineers authorized by the organization can grant access to new engineers and users. When such access is granted the engineer determines the login name, thus ensuring it conforms to a company standard. Only the same authorized engineers can change this name.

[0723] Visual Enterprise generates the initial password for all engineers and users when access is granted to an engineer or user. At this time, the three pieces of information, Organization ID, login name and password are communicated to the engineer or user via e-mail. As soon as the engineer or user logs into the site he or she is required to change the password to one of his or her choosing. The password may be randomly generated to reduce the risks associated with an administrator allocating a password that is commonly known across the organization. The use of e-mail to transport this information is an efficient method that reduces the risk of the details being written on paper when communicated via the telephone.

[0724] Session Authentication: Every time the engineer or user chooses an option, or enters data that requires the client browser to communicate with the Composer Server or Enterprise Interpreter, Visual Enterprise optionally verifies that the session is authentic. Visual Enterprise does so by checking three pieces of information against session information stored on the server.

[0725] Firstly, Visual Enterprise verifies that the organization key, allocated when an organization is registered and stored in the client browser when the user logs into the site, matches that stored on the server. This key is never communicated to a user or engineer and could not accurately be guessed.

[0726] Secondly, Visual Enterprise verifies that the user key, allocated when the User is granted access to Visual Enterprise and stored in the client browser when the user logs into the site, matches that stored on the server. This key is never communicated to a user or engineer and could not accurately be guessed.

[0727] Finally, Visual Enterprise verifies that the session key, allocated when the user logs into the site and stored in the client browser, matches that stored on the server. This key is never communicated to a user or engineer and is constructed using a non-reversible encryption algorithm. Every time the user logs into the site a new session key is generated.

These three checks provide a triple level of session authentication.

[0728] Secure Sockets Layer: Visual Enterprise implements the Secure Sockets Layer (SSL) version 3.0 protocol, a security protocol that provides communications privacy over the Internet. The protocol allows client/server applications to communicate in a way that is designed to prevent eavesdropping, tampering, or message forgery and uses digital certificates for authentication.

[0729] The primary goal of the SSL Protocol is to provide privacy and reliability between two communicating applications. The protocol is composed of two layers. At the lowest level, layered on top of a reliable transport protocol (e.g., TCP), is the SSL Record Protocol. The SSL Record Protocol is used for encapsulation of various higher-level protocols. One such encapsulated protocol, the SSL Handshake Protocol, allows the server and client to authenticate each other and to negotiate an, encryption algorithm and cryptographic keys before the application protocol transmits or receives its first

byte of data. One advantage of SSL is that it is application protocol independent. A higher-level protocol can layer on top of the SSL Protocol transparently. The SSL protocol provides connection security that has three basic properties:

[0730] The connection is private. Encryption is used after an initial handshake to define a secret key. Symmetric cryptography is used for data encryption (e.g., DES, RC4, etc.)

[0731] The peer's identity can be authenticated using asymmetric, or public key, cryptography (e.g., RSA, DSS, etc.).

[0732] The connection is reliable. Message transport includes a message integrity check using a keyed MAC (Medium Access Control). Secure hash functions (e.g., SHA, MD5, etc.) are used for MAC computations.

[0733] The goals of SSL Protocol v3.0, in order of their priority, are:

[0734] A. Cryptographic security SSL is used to establish a secure connection between two parties.

[0735] B. Interoperability—Allows programmers to be able to develop applications utilizing SSL 3.0 that will then be able to successfully exchange cryptographic parameters without knowledge of one another's code.

[0736] C. Extensibility—SSL seeks to provide a framework into which new public key and bulk encryption methods can be incorporated as necessary. This will also accomplish two sub-goals: to prevent the need to create a new protocol (and risking the introduction of possible new weaknesses) and to avoid the need to implement an entire new security library.

[0737] D. Relative efficiency—Cryptographic operations tend to be highly CPU intensive, particularly public key operations. For this reason, the SSL protocol has incorporated an optional session caching scheme to reduce the number of connections that need to be established from scratch. Additionally, care has been taken to reduce network activity.

[0738] Digital Certificates: Digital Certificates are based on public/private key technology (PKI). Each key is like a unique encryption device. No two keys are ever identical, which is why a key can be used to identify its owner.

[0739] Keys always work in pairs, one called the private key, and the other called the public key. What a public key encrypts, only the corresponding private key can decrypt, and vice versa. Public keys are distributed freely to anyone who wants to exchange secure information with you. Your private key is never copied or distributed and remains secure on your computer or server.

[0740] Digital certificates automate the process of distributing public keys and exchanging secure information. When you install a digital certificate on your computer or server, your computer or web site now has its own private key. Its matching public key is freely available as part of your digital certificate posted on your computer or web site. When another computer wants to exchange information with your computer, it accesses your digital certificate, which contains your public key. The other computer uses your public key to validate your identity and to encrypt the information it wants to share with you using SSL technology. Only your private key can decrypt this information, so it remains secure from interception or tampering while traveling across the Internet.

[0741] Database Security: Access to the Visual Enterprise database is restricted to only two sources. That is:

[0742] Access to engineers and users through the Visual Enterprise Composer and Player environments.

[0743] Access to the database administrators who installed and maintain the Visual Enterprise database.

[0744] The database administrator configures the account names and passwords when the database and Visual Enterprise are installed on the database and web application servers. They are not stored anywhere in the Visual Enterprise source code or in any other file controlled by Visual Enterprise. Standard network and firewall security can prevent access to the server by unauthorized persons.

[0745] Process Security: The business process engineers granted access by an organization to Visual Enterprise have complete control of the users who are granted access to each process. Users will not even be aware of the existence of processes to which they have not been granted access. For more information on Process Security, refer to the sections above in this VE blueprint section on Users, Groups, Processes, Process Views and Activities.

[0746] Attribute Security: Visual Enterprise gives the ability to a business process engineer to set attribute level security in one of two ways:

[0747] A process view can be defined, which restricts access to particular attributes to specified users or groups. For more information, refer to the section earlier in this VE blueprint section on Process Views.

[0748] A business rule can be defined which changes access to an attribute based on the current user or group to which he or she belongs. For more information, refer to the section earlier in this VE blueprint section on Business Rules.

[0749] From the outset, Visual Enterprise has been architectured with a classic N-tier component model in mind. The practical consequences of this approach are that two technologies are well suited for implementing the components that together comprise Visual Enterprise:

[0750] Microsoft's Distributed Network Architecture (Windows DNA)

[0751] Sun's J2EE specification and associated component models.

[0752] The first production version of Visual Enterprise has been implemented using Windows DNA, with a J2EE version to follow. More recently, focus has shifted from Microsoft's DNA to Microsoft's .NET strategy. It is intended that Visual Enterprise will follow this trend with the Microsoft DNA version being replaced with a Microsoft .NET version in the future.

[0753] The Microsoft DNA is straightforward in its approach to designing and constructing solutions. In reference to the Figures and particularly to FIG. 23, certain preferred alternate preferred embodiments of the method of the present invention can and should be partitioned into the following areas, to obtain an optimum design in terms of ease of maintenance and scalability.

[0754] The specifics of how Visual Enterprise is implemented within .NET are contained in the following sections:

[0755] Presentation Services;

[0756] Business Logic; and

[0757] Data Services.

[0758] Presentation Services include all areas of a solution that provide a user interface. Microsoft's suggested approach is that in new solutions, all presentation services should be web based. Business Logic encompasses all areas of a solution that implement logic or rules specific to the solution. Microsoft recommends that business logic should be implemented using a reusable, stateless component technology, such as COM+. Data services include all areas of a solution that store or provide data. This includes database, email and directory servers.

[0759] In reference to the Figures and particularly to FIG. 23, a first implementation of Visual Enterprise, as diagrammed in FIG. 24, has closely adhered to the DNA strategy. This is apparent in the high-level system architecture diagram of FIG. 24.

[0760] Client Requirements: Two distinct classes of user exist within the Visual Enterprise environment: Engineers and Users. As detailed earlier, engineers use the Composer interface for designing solutions, and users use the Player interface to run existing solutions. By design, all a user requires to utilize Visual Enterprise is a reasonably optioned IBM PC with Microsoft Internet Explorer (version 5.5 or above). Engineers have the additional requirement of needing to accept a one-time download of several ActiveX controls, used in the composer user interfaces.

[0761] A Visual Enterprise client application exists wholly within Microsoft Internet Explorer, making use of the following technologies to produce user interfaces and interact with the presentation services:

[0762] Hypertext Markup Language (HTML)

[0763] All web pages are rendered using standard HTML.

[0764] Dynamic Hypertext Markup Language (DHTML)

[0765] Significant use is made of the dynamic behavior of HTML to provide flexible user interfaces. The intention has been to use DHTML to build user interfaces similar to what can be achieved with client-server techniques.

[0766] ActiveX Controls

[0767] The Composer interface makes use of a number of ActiveX controls (e.g. tree control, flowchart control) to give a more flexible user interface The Player makes use of the Microsoft XML Parser control, which is made available though the Microsoft Internet Explorer installation.

[0768] JavaScript

[0769] JavaScript is the main procedural language for utilizing both DHTML and the ActiveX controls. Also, all client side logic is implemented using pure JavaScript.

[0770] Extended Markup Language (XML)

[0771] XML is used on the client for two purposes: for temporary storage of structure data, and for communication between the client and presentation services. Visual Enterprise has a large number of internal schemas designed for these purposes. The use of these schemas allows Visual Enterprise to manifest user interfaces more akin to a client-server implementation without the client installation overhead.

[0772] Presentation Services: The Presentation Services consist of several hundred Active Server Pages (ASPs) running under Microsoft Internet Information Server (version 5 or above). These pages respond to requests from the client and determine the HTML, client side JavaScript and/or XML required to answer the request. The following technologies are used by the presentation services:

37

[0773] JavaScript

[0774] JavaScript is the main procedural language for utilizing both custom COM+ components and the ActiveX controls. All presentation services logic is implemented using pure JavaScript.

[0775] ActiveX Controls

[0776] The presentation services make use of server-side ActiveX controls to perform operations not specifically available through the ASP interface or JavaScript (e.g. a control is used to extract documents from multi-part HTML requests). The Presentation Services also make use of the Microsoft XML Parser control, which is made available though the Microsoft Internet Explorer installation.

[0777] COM+ Components

[0778] The Presentation Services access the required business logic via public COM+ interfaces, exposed by the business logic layer.

[0779] Extended Style Sheet Language Transformation (XSLT)

[0780] XSLT is used to transform internal player XML screen definitions to the required HTML and client-side JavaScript. The Presentation Services contain master XSLT style sheets used to produce all Player user interfaces. Currently these style sheets are fixed, but is intended that future Visual Enterprise releases will allow these style sheets to be engineer or even user defined.

[0781] Business Logic: The Business Logic consists of fifty custom-written COM+ components and one custom-written service. These components encapsulate all the business logic implemented in the Visual Enterprise application. Microsoft Visual C++ (version 6) has been the language of choice to implement these components. Due to the nature of Visual Enterprise, the components are the result of hundreds of thousands of lines of very complex source code.

[0782] The Business Logic implements a database independent layer that allows Visual Enterprise to be deployed on either Microsoft SQL Server 2000 or Oracle 8i Release 2. The database independence is achieved by abstracting all database operations required by Visual Enterprise to a single COM+ interface, with the given COM+ component having the knowledge of how to interact with the target database.

[0783] Integration with third-party systems is achieved through the business logic layer. This is achieved using web services constructed using Microsoft C# and the Microsoft .NET framework. The traditional COM+ components interact with web services using standard SOAP techniques.

[0784] The business logic makes use of one or more of the following technologies:

[0785] Common Object Model (COM)

[0786] The Common Object Model is a technology developed by Microsoft for building components that expose interfaces that can be accessed in a language-independent way. For example, a COM component can be accessed from Visual C++ and Visual Basic.

[0787] Common Object Model+ (COM+)

[0788] COM+ is an environment for hosting COM components. COM+ provides enterprise services to COM components (i.e. transactions, queuing, object pooling, connection pooling, publisher/subscriber event model) via components implementing standard COM interfaces. All Visual Enterprise business logic components make use of these services.

[0789] ActiveX Data Objects (ADO)

[0790] ActiveX Data Objects are a Microsoft technology that present a universal way to access data sources of any type. Visual Enterprise uses ADO to access the underlying Oracle or SQL Server database.

[0791] Microsoft Message Queue (MSMQ)

[0792] Microsoft Message Queue allows transactions to be queued and processed in the background. COM+ allows components to be marked as queued, and therefore all requests on the given component are handled asynchronously. Visual Enterprise makes use of these features for all background processes, including report production, automated activities, and event handling.

[0793] Collaborative Data Objects (CDO)

[0794] Collaborate Data Objects are a Microsoft technology for accessing SMTP (email) servers on the local network. Visual Enterprise uses this technology for dispatching

[0795] Microsoft Scripting Engine (JavaScript)

[0796] The ability to execute JavaScript in Microsoft Internet Explorer is achieved through an ActiveX component called the Microsoft Scripting Engine. Any application that wishes to execute script can implement what is called a Scripting Host and invoke the Microsoft Scripting Engine. Visual Enterprise uses this technique to execute engineer defined business rules, which have been previously translated to JavaScript.

[0797] Extended Markup Language (XML)

[0798] The business rules layer uses XML to communicate structured data with the presentation service or with external systems via SOAP. XML is not used internally in the business logic layer, as small COM objects have proved more efficient for internal data communications.

[0799] Simple Object Access Protocol (SOAP)

[0800] Simple Object Access Protocol is an evolving technology, based on XML, which allows services to be accessed over the Internet. Visual Enterprise uses this technology to integrate with third party database and systems through the use of web services.

[0801] Universal Description, Discovery and Integration (UDDI)

[0802] Universal Description, Discovery and Integration is an evolving technology for categorizing web services available over the Internet. Visual Enterprise uses this technology to allow engineers to search for required functionality (e.g. a credit card check service) and to use Visual Enterprise's SOAP capability to include the external functionality natively into Visual Enterprise solutions.

[0803] Crystal Report Designer Component (RDC)

[0804] Visual Enterprise embeds a set of COM components called the Report Designer Component from Crystal Decisions, which allows report templates to be dynamically constructed and executed. This technology is used in conjunction with a custom reporting user interface to allow engineers to create customized reports on data contained within Visual Enterprise.

[0805] Data Services: Due to Visual Enterprise being a stateless web-based application, all system state and application data must be stored in a database. In the case of Visual Enterprise, either a SQL Server or Oracle database is required. The database consists of approximately two hundred tables and eighteen hundred stored procedures. The tables can be grouped into three distinct data sets:

[0806] Definitions: 80% of the tables are used to store the details of Visual Enterprise solutions. Effectively, the composer interface is a data entry system to correctly fill these tables. These tables include details of organizations, applications, events, processes, activities, packages, attributes etc.

[0807] Metadata: 16% of the tables are used to store metadata. Metadata at its most simple is data that describes other data. Although abstract in nature, a concrete example of metadata would be a library catalog, which contains information about the books in the library. The contents of the books are unstructured and varied, but the catalog categorizes the details of each book in a structured manner. (The same is true of Visual Enterprise, where engineers using the Composer interface define packages of data. These packages can then be included in activities in any manner the engineer sees fit, including the creation of package relationships on the fly. Visual Enterprise creates metadata to capture the details of the mapping of high-level packages to tables in the database designed for storing unstructured data and the relationships between that unstructured data.)

[0808] Data: 4% of the tables are used to store the actual data produced by users accessing Visual Enterprise solutions through the Player interface. It should be noted that this small number of tables is likely to be the largest in terms of storage requirements.

[0809] As stated earlier, the Visual Enterprise database contains a large number of stored procedures. This allows a database to present a high-level interface to the business rules layer, and thus abstracts the need for the business rules layer to have to deal with the specifics of one database to another. This is definitely true with SQL Server and Oracle, having quite different stored procedure languages and SQL requirements.

[0810] Integration: One of the key selling points of the Visual Enterprise environment is its ability to integrate with other products. From a technical perspective, Visual Enterprise supports five integration methods:

[0811] External Components

[0812] Visual Enterprise allows standard COM components to be utilized as part of the native business rule grammar. This is achieved by registering a given COM component using the Composer interface. Visual Enterprise examines that methods implemented by the components and maps all parameters to native Visual Enterprise attribute types. The methods are then dynamically added to the business rule grammar. At business rule execution time, Visual Enterprise performs the necessary interfacing required to make remote procedure calls to the relevant external components.

[0813] Remote Data

[0814] Visual Enterprise comes standard with a custom web service that can be deployed against any data source supported by the ActiveX Data Objects (ADO), which includes any data source that has an ODBC driver. Through Visual Enterprise accessing the web service, engineers are able to define "remote packages" based on tables in data sources supported by the web service. Such "remote packages" are treated as if they were native data of the given Visual Enterprise solution. The business rules layer performs all the necessary mechanics to update the data source taking into account the data source's referential integrity.

[0815] XML Messages

[0816] Visual Enterprise allows custom XML messages to be exchanged with external systems via the Internet. An engineer can construct definitions (i.e. using the XML Mapper interface) that describe how to interpret incoming XML messages and convert them to native Visual Enterprise packages and attributes, and vice versa. Again, from the engineer's perspective the XML is treated as if it were native Visual Enterprise data, with the business rules layer performing all the mechanics necessary to convert between the defined XML schema and Visual Enterprise packages and attributes.

[0817] SOAP Messages

[0818] Visual Enterprise supports both the construction of external SOAP requests and the ability to respond to external systems making SOAP requests on Visual Enterprise. At its simplest, SOAP identifies a request XML message schema, a response XML message schema, and the mechanism for the communication of these schemas. Using the previously described XML mapping interfaces, engineers define the structure of both incoming and outgoing SOAP requests and treat the schemas involved in these requests as if they were native Visual Enterprise packages and attributes. Again, the business rules layer performs all the mechanics necessary to construct, interpret and dispatch SOAP requests.

[0819] Web Services

[0820] Visual Enterprise supports both the consumption of external web services and exposing Visual Enterprise processes as web services. The consumption of an external web service is essentially the same as making a SOAP request. The only difference is that an engineer can use the composer to search a UDDI registry to find the required web service. Visual Enterprise exposes two methods for consuming web services. Firstly, web services can be registered with Visual Enterprise in a similar manner to external components and thus used to extend the native rule grammar. Secondly, a web service can be used as part of a Visual Enterprise process. The second method has the advantage of supporting document based web services, whereas the first method only supports remote procedure call based web services (i.e. similar to making a call on a remote component, but in this case using SOAP as the mechanism instead of Distributed COM). Again, the business rules layer performs all of the mechanics necessary to construct, interpret and dispatch web service requests, including the capability for synchronous and asynchronous processing.

[0821] Processes defined through the Composer interface can be exposed as either a remote procedure call or document based web service. This is achieved by Visual Enterprise exposing a custom end-point to allow external systems to communicate back to the given process. From the engineer's perspective, incoming requests appear as native Visual Enterprise packages and attributes, which can be manipulated using all the power of the Visual Enterprise engine. The business rules layer performs all the necessary mechanics to receive, interpret, validate, construct and dispatch web service requests.

[0822] Scalability: The N-tier nature of the Visual Enterprise implementation allows for both horizontal and vertical scalability to be used to meet specific performance requirements. The following points detail how each of the layers can be scaled for increased performance.

[0823] Presentation Services

[0824] The Presentation Services layer consists of Microsoft Internet Information Server web servers. Given Visual Enterprise is stateless in nature, it is possible (and advisable) to configure an array of web servers where client requests are dispatched to the web servers in a "round-robin" or "least-workload" manner. The number of web servers in the array is only limited by the network infrastructure used to create the array. As demand increases, web servers can be added to the array at will.

[0825] The maximum throughput for each web server in the array can be achieved by correctly configuring such servers in terms of CPU bandwidth and memory. The more memory available, the more likely the web server will cache Active Server Pages in memory. It should be noted that the responses from Visual Enterprise Active Server Pages are highly dynamic in nature, and are not suited to traditional HTTP caching techniques. In fact such caching techniques can interfere with the correct operation of the Presentation Services and are usually disabled by default.

[0826] Business Logic

[0827] The business logic layer consists, of application servers, which in practice consist of Windows 2000 Advanced Server configured with the custom components discussed above. Two options exist for configuring application servers: each web server can be statically tied to one application server, or application servers can be configured in an array, where requests from the web servers are routed in a "round-robin" or "least-workload" manner. As with the web servers, creating an array of application servers allows the business logic layer to be scaled as needed.

[0828] The maximum throughput for each application server in the array can be achieved by correctly configuring such servers in terms of CPU bandwidth and memory. The custom components are specifically designed to "read-only" cache static information in memory, thus reducing load on the data services layer. The trade-off for this performance enhancement is a greater need for memory. Also, the custom components perform CPU intensive operations, especially when dealing with XML documents or the Microsoft Scripting Engine. Practice has shown that an application server needs to have approximately twice the memory and CPU bandwidth to service an equivalent web server.

[0829] Data Services

[0830] The data services layer consists of either an Oracle or SQL Server database. Both Oracle and Microsoft have test cases of how their products can be scaled from the smallest enterprise installation up to the largest corporate installation. The stateless nature of Visual Enterprise has many advantages, but the trade-off is that more load is put on the database server. Although this is somewhat mitigated by caching techniques in the business logic layer, the configuration of a suitable database server needs to take this into account.

[0831] In reference to the Figures and particularly to FIG. 24, certain preferred alternate preferred embodiments of the method of the present invention may comprise a recent Visual Enterprise installation, as diagrammed in FIG. 24, and how a modest level of scalability can be achieved.

[0832] FIG. 24 details a Visual Enterprise installation designed to handle tens of thousands of client requests. The following points detail the path a client request would follow.

[0833] Requests flow first through a firewall. Both the Internet and the organization's Intranet are firewalled for security.

[0834] Requests then flow through a set of switches in failover configuration. In this case "failover" means that if one of the switches fails the second switch will take up the load.

[0835] Requests then flow through a Cisco Load Director, which is a device designed to route HTTP requests based on a given algorithm. As mentioned above, Visual Enterprise is best suited to either a "round-robin" or "least-workload" approach. Again, the two Cisco Load Directors are deployed in failover configuration.

[0836] K Requests flow through a second set of switches. Again, the switches are deployed in failover configuration.

[0837] Based on the routing algorithm used by the Cisco Load Director, the request will be routed to one of the web servers in the array. The given web server will interpret the client requests and make one or more requests on the application server. It should be noted that this design has the application servers statically tied to a given web server. The above design could be modified to include another layer of Cisco Load Directors and switches between the web servers and application servers, thus giving dynamic routing of application server requests.

[0838] In response to requests from the web server, the application servers will make one or more requests on the database server.

[0839] The application server will assemble the details of the response and forward the response to the given web server.

[0840] This VE blueprint section of paragraphs 00115 to 00364 of the present disclosure supports an understanding of the methods, models and structures of a preferred embodiment of the method of the present invention comprising the Visual Enterprise™ (VE) business process integration software platform. This VE blueprint summary describes, in a comprehensive fashion, the paradigm of VE and its purpose is twofold: 1) to outline how VE is architected, and 2) to establish technical credibility as to how VE's architecture is realized to employ certain alternate preferred embodiments of the method of the present invention.

[0841] VE is a single, unified modeling and execution software environment supporting all types of business transactions and processes. VE technically enables Business Process Management (BPM); the ability to discover, design, deploy, reliably execute, scale and optimize end-to-end complex business processes and transactions.

[0842] Creating adaptive models, not code, is the VE paradigm. When business processes change, multiple systems adapt according to certain alternate preferred embodiments of the method of the present invention that optionally comprise VE. VE includes a visual modeling, rules and data environment (VE Composer™) and an execution platform, which includes integration and web services (VE Player™). V/E augments existing functionality, integrates with various systems, and provides for comprehensive solutions. VE makes the organization, policies, workflow, calendars and events available directly to system designers and business process

experts. VE is built on open standards, such as LDAP, SOAP, SSL, XML, and WSDL/UDDI. VE operates in Microsoft's .NET environment, supports SQLServer and Oracle 9i databases and runs on IE. VE has a 'meta-data' architecture and can run through most enterprise portals and email systems.

[0843] From the above description and drawings, it will be understood by those of ordinary skill in the art that the particular embodiments shown and described are for purposes of illustration only and are not intended to limit the scope of the invention. Those of ordinary skill in the art will recognize that the invention may be embodied in other specific forms without departing from its spirit or essential characteristics. References to details of particular embodiments are not intended to limit the scope of the claims.

I claim:

1. A method of executing business processes via a computer network, the method comprising the steps of:
   creating and installing software resources available over a network, the software resources comprising software resource definitions available through a database of software resources;
   browsing the database of software resources using a process model builder to identify the software resource definitions;
   loading the software resource definitions identified by the process model builder to create a business process model comprising the software resources from the database of software resources;
   mapping inputs and outputs of the database of software resources to allow the business process model to be executed;
   saving the business process model on a storage medium accessible by the network; and
   initiating a collaborative business process by loading the business process model into a process interpreter and executing the software resources defined within the business process model.

2. The method of claim 1, wherein the process model builder comprises a graphical user interface (GUI) business process model builder.

3. The method of claim 1, wherein the collaborative business process comprises internal and external collaborative business processes.

4. The method of claim 1, wherein no programming, coding or scripting is required to create the business process model or to initiate the collaborative business process.

5. The method of claim 1, wherein the step of executing the software resources within the step of initiating a collaborative business process further comprises mapping the outputs from a first software resource to the inputs of a second software resource.

6. A method for dynamically generating software resources, the method comprising the steps of:
   operating a process model builder to load a process definition;
   defining grammar rules to create an activity; and
   using the grammar rules in a process workflow model to dynamically generate software resources without generating programming code.

7. The method of claim 6, wherein the method further comprises the step of:
   using a grammar engine to execute the generated software resources.

8. The method of claim 6, wherein the process definition comprises:
   data attributes;
   activities referencing the data attributes;
   processes formed using the activities;
   business rules applicable to the data attributes, the activities, and the processes; and
   location references selected from the group consisting of a universal resource locator and a network address.

9. A non-transitory computer-readable medium having a software program recorded thereon where the program is to make a computer execute a process, the process comprising:
   define an organization calendar for an organization, the organization calendar comprising attributes applicable to the organization;
   define business rules and processes followed by the organization, such business rules and processes consistent with the organization calendar;
   define a data attribute library comprising one or more data attributes for use by the organization; and
   define activities performed for the organization using the one or more data attributes and at least one of the business rules.

10. The medium of claim 9, wherein the attributes of the organization calendar comprise time zones, working hours, holidays, and events applicable to the organization.

11. The medium of claim 9, wherein the defined activities are constrained by the business rules.

12. The medium of claim 9, wherein the defined activities are represented graphically on a display operably coupled to the medium as activity icons.

13. A method for generating results of a business workflow, the method comprising the steps of:
   (a) operating a process interpreter to determine an input data structure required for a software resource to be executed;
   (b) identifying attributes available at run time from an organization using the process interpreter;
   (c) mapping the available attributes to input attributes as defined by a business process model using the process interpreter to create an input data structure;
   (d) forwarding the input data structure to a software resource using the process interpreter;
   (e) executing the software resource using the process interpreter to produce an output data structure;
   (f) providing the output data structure to the process interpreter for use by the process interpreter to complete a process workflow model; and
   (g) repeating steps (a) through (f) until a final output data structure has been created;
   (h) returning the final output data structure from the process interpreter as results of a business workflow.

14. A computer readable medium storing a sequence of instructions thereon, the sequence of instructions executable to perform a method, the method comprising the steps of:
   operating a network browser interface to obtain a specification of an application from a network, the application comprising a process model; and
   dynamically executing the application using the specification without generating programming code.

15. The medium of claim 14, wherein the specification comprises:
   data attributes;
   activities referencing the data attributes;

processes formed using the activities;

business rules applicable to the data attributes, the activities, and the processes; and

location references selected from the group consisting of a universal resource locator and a network address.

16. A system comprising:

a network;

at least one processor in bidirectional communication with the network;

a database of software resources stored on a storage medium operably connected to the network, the database of software resources comprising at least one software resource; and

at least one process model builder available over the network, the at least one process model builder operable to determine an input data structure required for the at least one software resource to be executed, identify attributes available at run time from an organization, map the available attributes to input attributes as defined by a business process model to create an input data structure, forward the input data structure to the at least one software resource, execute the at least one software resource to produce an output data structure, and provide the output data structure as a result of a business process model.

17. A system comprising:

a network;

at least one processor in bidirectional communication with the network;

a database of software resources stored on a storage medium operably connected to the network, the database of software resources comprising at least one software resource;

at least one business process model available over the network, the at least one business process model comprising at least one software resources from the database of software resources; and

at least one process interpreter available over the network, the at least one process interpreter operable to determine a data structure for at least one software resource, forward the data structure to the at least one software resource, execute the at least one software resource to produce an output data structure, and provide the output data structure as a result of the at least one business process model.

18. A method of executing business processes via a computer network, the method comprising the steps of:

accessing at least two software resources available over a network with a network browser;

instructing the network browser to interweave the at least two software resources by operating a business process model builder;

within a runtime environment, allowing a business process model to be dynamically generated in real-time by the business process model builder;

generating the business process model using the business process model builder by incorporating the at least two software resources interwoven by the network browser into the business process model; and

initiating business process activities from the business process model to complete collaborative business processes and transactions.

19. The method of claim 18, wherein the runtime environment comprises a process interpreter and a grammar engine and wherein the runtime environment is network based;

wherein the runtime environment encapsulates the at least two software resources necessary to complete a business process;

wherein a workflow determines the order in which business processes are undertaken; and

wherein business process activities forming the collaborative business processes are linked together according to the workflow.

20. The method of claim 18, wherein the runtime environment comprises a process interpreter and a grammar engine and wherein the runtime environment is network based;

wherein the process interpreter and the grammar engine encapsulated within the network based, runtime environment encapsulates the at least two software resources necessary to complete a business process;

wherein a workflow determines the order in which business processes are undertaken;

wherein the collaborative business processes and transactions comprise internal and external collaborative business processes and transactions; and

wherein business process activities forming the internal and external collaborative business processes are linked together according to the workflow.

* * * * *