(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
18 September 2014 (18.09.2014)

WIPO | PCT

(10) International Publication Number
WO 2014/145084 A1

(51) International Patent Classification:
G06F 17/30 (2006.01)

(21) International Application Number:
PCT/US2014/029749

(22) International Filing Date:
14 March 2014 (14.03.2014)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/793,627 15 March 2013 (15.03.2013) US

(71) Applicant: **PTC INC.** [US/US]; 140 Kendrick St., Needham, MA (US).

(72) Inventors: **SCHAEFER, John**; 908 Judie Lane, Ambler, Pennsylvania 19002 (US). **BULLOTTA, Rick**; 610 Waterfall Way, Phoenixville, Pennsylvania 19460 (US). **FAN, Lawrence**; 4881 Paseo De Vega, Irvine, California 92603 (US). **HAHR, Brandon**; 13 Verde, Irvine, California 92612 (US). **HUBER, Philip**; 9327 Residencia, Newport Beach, California 92660 (US). **MALAPAS, Samuel T.**; 502 Turtle Crest Drive, Irvine, California 92603 (US).

(74) Agents: **TANPITUKPONGSE, Paul** et al.; Choate, Hall & Stewart LLP, Two International Place, Boston, Massachusetts 02110 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

— *with international search report (Art. 21(3))*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

(54) Title: METHODS FOR MANAGING APPLICATIONS USING SEMANTIC MODELING AND TAGGING AND DEVICES THEREOF

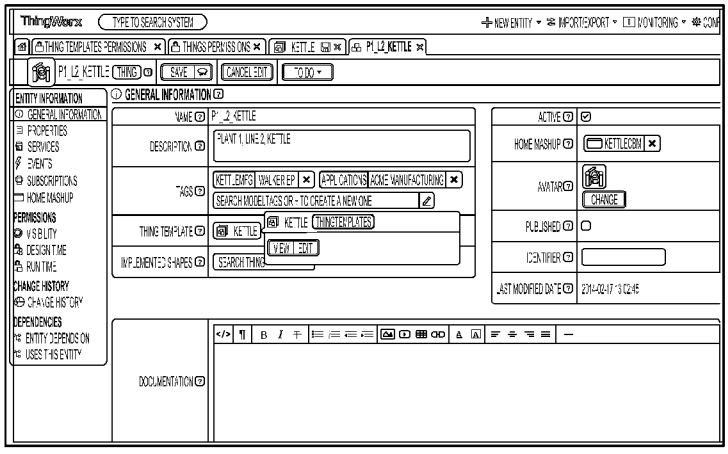

FIG. 5

(57) Abstract: The present disclosure provides a system and method for managing data using semantic tags. The method may include providing a data model corresponding to a first set of tangible objects where the data model includes a first template class having both properties describing the set of tangible object and a set of semantic tags corresponding to the properties. The method may include receiving a class definition for a second template class for a second set of tangible objects where the second template class inherits, by the class definition, the properties and the sematic tags for the second set of tangible objects.

WO 2014/145084 A1

METHODS FOR MANAGING APPLICATIONS USING SEMANTIC MODELING AND TAGGING AND DEVICES THEREOF

RELATED APPLICATIONS

[0001] This application claims priority to and the benefit of U.S. Provisional Patent Application No. 61/793,627, titled "METHODS FOR MANAGING APPLICATIONS USING SEMANTIC MODELING AND TAGGING AND DEVICES THEREOF," filed March 15, 2013, the text of which is incorporated herein by reference in its entirety.

FIELD OF THE INVENTION

[0002] This invention relates generally to methods and systems for managing data. More particularly, in certain embodiments, the invention relates to systems and methods for semantic modeling and tagging of data.

BACKGROUND

[0003] The process of software development involves research, new development, prototyping, modification and reuse. Additionally, developing dynamic computing applications requires the content developer to organize, manage and deploy a large quantity of content artifacts. In order to accurately develop the content artifacts, the content developer must create a logical model that reflects the physical world. That model is then broken down into common objects in order to provide the highest level of re-usability of artifacts and the lowest cost of maintaining the computing application(s). As the complexity of the computing application increases, the number of the common objects also increases thereby making the process of tracking and maintaining these common objects important.

[0004] Currently, existing technologies typically store these common objects at many different memory locations. As a result, a content developer has to manually search through a large number of locations for content artifacts to identify common objects relevant for the development of the computing application. This manual search process is inefficient, time consuming and tedious.

SUMMARY

[0005] In one aspect, the present disclosure describes a method for managing data using semantic tags. The method may include providing, by a processor of a computing device, a data model corresponding to a first set of tangible objects where the data model includes a first template class having a first set of one or more data properties associated with the first set of tangible objects. The first template class may be tagged to a first set of one or more semantic tags where the first set of semantic tags corresponds to the first set of data properties of the first template class. The semantic tag may be associated to one or more words listed in a vocabulary database accessible to the graphical user interface. The graphical user interface may be a part of a development workspace.

[0006] In some implementations, the method includes receiving, by the processor, responsive to an input received at a graphical user interface accessible to the computing device, a class definition for a second template class for the data model. The second template class may correspond to a second set of tangible objects. The second template class may be defined, in the class definition, by the first template class. The second template class may have a second set of one or more data properties where the processor uses the class definition to include the first set of data properties within the second set of data properties. The second template class may be stored without any input from the graphical user interface after the class definition is received. The second template class may be an inherited object class where the second template class inherits data properties from the first template class. In some implementations, the first template class may be an inheritance (as employed within the context of object-oriented programming) of the second template class. The first set and second set of data properties may include geographic information associated with each of the respective sets of tangible objects. The class definition may be an extensible template for creating objects where each of the objects is created by a constructor of the class definition, and the class is instantiated to create an instance of the class definition.

[0007] In some implementations, the method includes storing, by the processor, the second template class with a second set of one or more semantic tags where the processor uses the class definition to include the first set of semantic tags within the second set of semantic tags. In some implementations, the second template class may inherit the semantic tags from the first template class. In some implementations, the second template class may

replicate the semantic tags. In some implementations, the second template class may inherit the semantic tags from the first template class. In some implementations, the second template class may replicate the semantic tags.

[0008] In one aspect, the present disclosure describes a system including a processor and a memory, the memory storing instruction that, when executed by the processor, cause the processor to provide a data model corresponding to a first set of tangible objects where the data model includes a first template class having a first set of one or more data properties associated with the first set of tangible objects. The first template class may be tagged to a first set of one or more semantic tags where the first set of semantic tags corresponds to the first set of data properties of the first template class. The semantic tag may be associated to one or more words listed in a vocabulary database accessible to the graphical user interface. The graphical user interface may be a part of a development workspace.

[0009] The instructions, when executed, further cause the processor to receive, responsive to an input received at a graphical user interface accessible to the computing device, a class definition for a second template class for the data model. The second template class may correspond to a second set of tangible objects. The second template class may be defined, in the class definition, by the first template class. The second template class may have a second set of one or more data properties where the processor uses the class definition to include the first set of data properties within the second set of data properties. The second template class may be stored without any input from the graphical user interface after the class definition is received. The second template class may be an inherited object class where the second template class inherits data properties from the first template class. In some implementations, the first template class may be an inheritance (as employed within the context of object-oriented programming) of the second template class. The first set and second set of data properties may include geographic information associated with each of the respective sets of tangible objects. The class definition may be an extensible template for creating objects where each of the objects is created by a constructor of the class definition, and the class is instantiated to create an instance of the class definition.

[0010] The instructions, when executed, further cause the processor to store the second template class with a second set of one or more semantic tags where the processor uses the class definition to include the first set semantic tags within the second set of semantic tags. In some implementations, the second template class may inherit the semantic

tags from the first template class. In some implementations, the second template class may replicate the semantic tags.

BRIEF DESCRIPTION OF THE FIGURES

[0011] The foregoing and other objects, aspects, features, and advantages of the present disclosure will become more apparent and better understood by referring to the following description taken in conjunction with the accompanying drawings, in which:

[0012] FIG. 1 is a diagram of an example environment at a computing device that manages computing application using semantic modeling and tagging.

[0013] FIG. 2 is a diagram of an example computing device.

[0014] FIG. 3 is a flow chart of an example method for managing computing applications using semantic modeling and tagging.

[0015] FIG. 4 is an diagram of an example workspace for managing computing application using semantic tags.

[0016] FIG. 5 is an diagram of an example workspace for managing computing application using semantic tags.

[0017] FIG. 6 is a flowchart of an example method of managing data using semantic tags.

[0018] FIG. 7 shows an example of a computing device and a mobile computing device that can be used to implement the techniques described in this disclosure.

DETAILED DESCRIPTION

[0019] An exemplary environment with an computing application manager at a computing device that manages application using semantic modeling and tagging is illustrated in FIG. 1. In this particular example, the environment includes the application manager computing device (namely a computing device that executes the application manager), a plurality of client computing devices, a plurality of servers which are all coupled together by one or more communication networks, although this environment can include other numbers and types of systems, devices, components, and elements in other configurations.

[0020] Referring to FIGS. 1 and 2, the application manager computing device provides a number of functions including managing application using semantic modeling and tagging, although other types of numbers of computing devices with execute other types and numbers of functions can be used. In this particular example, the application manager computing device includes a central processing unit (CPU) or processor, a memory, an input device, a display device, and a network interface which are coupled together by a bus or other link, although other numbers and types of systems, devices, components, and elements in other configurations and locations can be used.

[0021] The processor in the application manager computing device executes a program of stored instructions for one or more aspects of the present technology as described and illustrated by way of the examples herein, although other types and numbers of processing devices and logic could be used and the processor could execute other numbers and types of programmed instructions.

[0022] The memory in the application manager computing device stores these programmed instructions for one or more aspects of the present technology is configured to execute these instructions as described and illustrated herein, although some or all of the programmed instructions could be stored and executed or configured for execution elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and written to by a magnetic, optical, or other reading and writing system that is coupled to the processor in the application manager computing device, can be used for the memory in the application manager computing device.

[0023] The input device of the application manager computing device enables a user, such as an administrator, to interact with the application manager computing device, such as to input data and/or to configure, program and/or operate it by way of example only. By way of example only, the input devices may include one or more of a touch screen, keyboard and/or a computer, although other types and numbers of input devices could be used.

[0024] The display device of the application manager computing device enables a user, such as an administrator, to view data and/or other information by way of example only. By way of example only, the display device may include one or more of a CRT, LED

monitor, or LCD monitor, although other types and numbers of display devices could be used.

[0025] The network interface device in the application manager computing device is used to operatively couple and communicate between the application manager computing device and the plurality of client computing devices, the plurality of servers over one or more the communications networks, although other types and numbers of communication networks or systems with other types and numbers of connections and configurations can be used. By way of example only, the one or more the communications networks can use TCP/IP over Ethernet and industry-standard protocols, including NFS, CIFS, SOAP, XML, LDAP, and SNMP, although other types and numbers of communication networks, such as a direct connection, a local area network, a wide area network, modems and phone lines, e-mail, and wireless communication technology, each having their own communications protocols, can be used.

[0026] In this particular example, each of the client computing devices and the plurality of servers includes a central processing unit (CPU) or processor, a memory, input/display device interface and a network interface or I/O system, which are coupled together by a bus or other link, although other numbers and types of systems, devices, components, and elements in other configurations can be used.

[0027] Each of the client computing devices may utilize the application manager computing device to identify and obtain templates and artifacts for application development, by way of example only, although each of the client computing devices may execute other types and numbers of operations and functions and other types a numbers of computing devices might be coupled to interact with the application manager computing device.

[0028] Each of the servers may process requests received from requesting client computing devices via communication networks according to the HTTP-based application RFC protocol or the CIFS or NFS protocol for example. Various network processing applications, such as CIFS applications, NFS applications, HTTP Web Server applications, and/or FTP applications, may be operating on the servers and transmitting data (e.g., files, Web pages) to the application manager computing device in response to requests from the client computing devices. Each of the servers may provide data or receive data in response to requests directed toward the respective applications on the servers from the client computing devices or the application manager computing device. Each of the servers may be

hardware or software or may represent a system with multiple servers in a server pool, which may include internal or external networks. In this example the servers may be any version of Microsoft® IIS servers or Apache® servers, although other types of servers may be used.

[0029] Although examples of the application manager computing device, the client computing devices and plurality of servers are illustrated and described herein, each of these devices and systems can be implemented on any suitable computer system or computing device. It is to be understood that the devices and systems of the examples described herein are for exemplary purposes, as many variations of the specific hardware and software used to implement the examples are possible, as will be appreciated by those skilled in the relevant art(s).

[0030] Furthermore, each of the systems of the examples may be conveniently implemented using one or more general purpose computer systems, microprocessors, digital signal processors, and micro-controllers, programmed according to the teachings of the examples, as described and illustrated herein, and as will be appreciated by those ordinary skill in the art.

[0031] In addition, two or more computing devices or systems can be substituted for any one of the systems in any embodiment of the examples. Accordingly, principles and advantages of distributed processing, such as redundancy and replication also can be implemented, as desired, to increase the robustness and performance of the devices and systems of the examples. The examples may also be implemented on computer system or systems that extend across any suitable network using any suitable interface mechanisms and communications technologies, including by way of example only telecommunications in any suitable form (e.g., voice and modem), wireless communications media, wireless communications networks, cellular communications networks, 3G communications networks, Public Switched Telephone Network (PSTNs), Packet Data Networks (PDNs), the Internet, intranets, and combinations thereof.

[0032] The examples may also be embodied as a non-transitory computer readable medium having instructions stored thereon for one or more aspects of the present technology as described and illustrated by way of the examples herein, as described herein, which when executed by a processor, cause the processor to carry out the steps necessary to implement the methods of the examples, as described and illustrated herein. Exemplary methods for

managing applications using semantic modeling and tagging will now be described below with reference to FIGS. 1-3.

[0033] An exemplary process beings with the application manager computing device receiving a request to develop templates and artifacts for a model from a client computing device, although the application manager computing device may receive any other types or requests from any other devices. By way of example only, the application manager computing device receives a request to develop a model to track and manage delivery of food products to retail outlets in a geographic area to determine the return on investment and the selected geographic area is Philadelphia. In some implementations, the templates may be used to describe a class of assets. To this end, rather than individual instance of a given electronic device, the template defines a type. Templates are alternatively referred to as a "template class" herein.

[0034] Next, the application manager computing device receives information associated with the request to develop a model from the requesting client computing device. In this example, the application manager computing device receives first level information associated with the geographic region, such as the total number of delivery trucks in this region, number of manufacturing plants and number of warehouses from which the food products are delivered to the retail locations in this particular example. Additionally, the application manager computing device also receives second level information, such as driver information of each truck, capacity of load for each truck and temperature sensor and humidity sensor values for refrigerated trucks in this particular example. In some implementations, the first and second level may be defined as data properties within the template.

[0035] Using the received first level and the second level information, the application manager computing device defines a plurality of templates for the requested model, although the application manager computing device can develop, define, and/or obtain any amounts of other types of artifacts associated with developing the requested model. In this example, the application manager computing device defines a Base_Truck template with properties such as capacity of the truck, driver of the truck, inventory and location. Additionally, the application manager computing device defines another template for Refrigerated_Truck with properties including capacity of the truck, driver of the truck, inventory, location and

temperature and humidity sensor values. In some implementations, the properties may be stored as a textual description or string within the template class.

[0036] Upon defining a plurality of templates, the application manager computing device organizes the templates and any other developed artifacts by defining semantic tags for each of the created template and artifacts. By way of example only, the application manager computing device defines the semantic tags using an application vocabulary database. The application vocabulary in this example relates to using terms defined in the information to define semantic tags. Of course, the terms may include portions of a word, wild card, abbreviations, acronyms. Similarly, the tags may account for verb tenses and other grammatical form. For purpose of further illustration in this example, the application manager computing device defines tags such as Truck_Tracking and Region to each of the defined templates and artifacts.

[0037] Additionally, if the application manager computing device further defines another template Delivery_Truck from previously created template Base_Truck, the application manager computing device will automatically applies all the properties defined in template Base_Truck including the semantic tags associated to the Base_Truck to the recently created template Delivery_Truck. In some implementations, a given template may be class object that may inherit or include other templates. Additionally, if the application manager computing device further defines (e.g., for a class definition) another template Refrigerated_Delivery_Truck using both Base_Truck and Refrigerated_Truck, the application manager computing device automatically applies all the properties defined in Base_Truck and Refrigerated_Truck and assigns the tags associated with both Base_Truck and Refrigerated_Truck to the recently created template Refrigerated_Delivery_Truck.

[0038] Further, the application manager computing device stores each of the tagged templates and artifacts within the memory of the application manager computing device, although the application manager computing device can store the templates and artifacts at other locations.

[0039] Once the template artifacts have been defined to meet the requirements of the solution, specific instances of templates are created. In this example, each specific delivery truck in the Philadelphia region is instantiated from one of the previously defined templates, and the application manager computing device will automatically apply all the properties and

characteristics defined in the template that is defined for that instance. Semantic tags describing the specific instances in the model are also applied.

[0040] Next, the application manager computing device sends out a notification to the requesting client computing device indicating completion of defining and organizing the templates and artifacts to the requesting client computing device.

[0041] When the application manager computing device receives another request from the requesting client computing device to provide all the defined templates and artifacts associated with the requested model, the application manager computing device provides a graphical representation of the organized templates and the artifacts to the requesting client computing device, although the application manager computing device can provide the templates and the artifacts in any other format. The graphical representation provided by the application manager computing device would include each of the templates and artifacts with their associated tags. In this example, the templates Base_Truck and Refrigerated_Truck would be graphically represented with tags Truck_Tracking and Region-Philly. Additionally, the graphical representation would include the interconnection between each of the template and artifact and accordingly in this example, the graphical representation would indicate that the Delivery_Truck template was inherited from Base_Truck template and Refrigerated_Delivery_Truck template was inherited from Base_Truck and Refrigerated_Truck templates.

[0042] FIG. 4 shows an example graphical workspace for managing the templates and artifacts, as described in relation to FIGS. 1-3. As discussed, the template may define a type, rather than an instance, of a thing. Also as discussed, these things may be a tangible object (such as a truck). In some implementations, the template may include common information that determines or defines what constitute a truck. To this end, a class of asset generally refers to multiple trucks having a common set of definition of description as defined by the data- and information-model.

[0043] In some implementations, the class of assets may be referred to a "Thing". A "Thing" may be defined as an instance of a "Thing Template." A "Thing Template" may be an abstract class that can inherit from one or more "Thing Shapes," which is defined by a set of properties, services, and events, "Thing Template," and "Thing instance." To this end, if a "Thing Template" inherits from one or more "Thing Shapes", all the properties, events, and services of the "Thing Shapes" are part of the "Thing Template." When a "Thing instance"

is created from a "Thing Template", all properties, events, and services of the "Thing Template" are realized within the "Thing instance." Thus, if a new service, property, or capability is defined at the "Thing Shape" or "Thing Template" level, each "Thing" instance that is derived from those entities immediately inherits that service, property or capability. To this end, as soon as a new "Thing" is defined in the model, the full set of services and data for the "Thing" is available as a class.

[0044] In some implementations, the template may be for a class of assets that may include a set of machinery at an industrial complex having data stored in a given database; a set of computer or an office equipment at a business or government office having data stored in a given database; a set of point-of-sale machines at a market place or vending machines having data stored in a given database; a set of construction equipment or vehicles having data stored in a given database; a set of power generation or distribution equipment having data stored in a given database; a set of power substation or a transmission equipment having data stored in a given database; a set of building meters having data stored in a given database; all server having operational data stored in a given database; a set of networking or routing equipment having data stored in a given database; a set of smart appliances having data stored in a given database; a set of exercise machines having data stored in a given database; a set of medical device or prosthesis devices having data stored in a given database; a set of medical diagnostic devices or hospital equipment having data stored in a given database; a set of commercial vehicles or transport containers having data stored in a given database; a set of motor vehicles or electric bicycles having data stored in a given database; a set of cellphones having data stored in a given database, a set of laptops having data stored in a given database, a set of tablet having data stored in a given database, a set of electronic readers having data stored in a given database; or a set of clothing electronic-tag having data stored in a given database. Referring back to FIG. 4, the workspace includes various types of assets and classes of assets. These assets and classes of assets include sensors, security equipment, vending machines 42, medical equipment, vehicles (as described, in relation to FIGS. 1-3, for example), plant equipment, mining equipment, and factory machinery. A sensor equipment, for example, is shown as "AB Test Remote Tunneling Thing". A security equipment is shown as "Security Monitor". A vending machine is shown as "VM-007 Generic Thing". A medical equipment is shown as "CT-2". A class of vehicles is shown as "Vehicle 42". A plant equipment is shown as "P1 L2 Kettle" (referring to a kettle machine

on line 2 of a given plant 1). A mining equipment is shown as "Station 3 LOC1". A factory equipment is shown as "Plant 5" (referring to a factory in Yengchang, China).

[0045] FIG. 5 is an diagram of an example workspace for managing computing application using semantic tags. As shown in FIG. 5, when assigning an instance (for example, "P1_L2_Kettle") of a class created by a template, the instance may be assigned a semantic tag. The semantic tags may include abbreviations, such as for kettle manufactured, shown as "kettlemfg." The tags may be searched and then added, or may be added by the end-user through the workspace.

[0046] The application manager computing device also may provide an option for searching for additional information to the requesting client computing device using the tags. In this particular example, the application manager computing device could provide an option of searching for all artifacts and templates with tag Truck_Tracking. Upon receiving the confirmation to search from the client computing device, the application manager computing device searches for all templates and artifacts tagged as Truck_Tracking present within the memory and also the servers and provides the searched results to the requesting client computing device. Additionally, the application manager computing device assists the requesting client computing device to select any of the search results, modify the select search result and finally store the modified content.

[0047] The application manager computing device also may provide an option for searching for additional information to the requesting client computing device using the tags, and transporting the relevant content artifacts from one system to another. A typical example is to move the content artifacts for a new computing application from a development system to a testing system, and finally to a production system using the previously defined semantic tags.

[0048] By providing the graphical representation of the templates and artifacts, the technology disclosed provides advantages of assisting the user of the client computing device to understand the created model quickly and effectively. Additionally, by organizing the templates and artifacts using tags, the user of the client computing device can quickly search for any additional information using the tags. Further, by organizing the created artifacts and templates, any new user or developer of the requesting client computing device will be able to quickly understand the model without manually searching for each of the template and artifacts and then trying to understand the connection between them.

[0049] FIG. 6 is a flowchart of an example method of managing data using semantic tags. The method may include include providing, by a processor of a computing device, a data model corresponding to a first set of tangible objects where the data model includes a first template class having a first set of one or more data properties associated with the first set of tangible objects (step 602). The first template class may be tagged to a first set of one or more semantic tags where the first set of semantic tags corresponds to the first set of data properties of the first template class. The semantic tag may be associated to one or more words listed in a vocabulary database accessible to the graphical user interface. The graphical user interface may be a part of a development workspace.

[0050] In some implementations, the method includes receiving, by the processor, responsive to an input received at a graphical user interface accessible to the computing device, a class definition for a second template class for the data model (step 604). The second template class may correspond to a second set of tangible objects. The second template class may be defined, in the class definition, by the first template class. The second template class may have a second set of one or more data properties where the processor uses the class definition to include the first set of data properties within the second set of data properties. The second template class may be stored without any input from the graphical user interface after the class definition is received. The second template class may be an inherited object class where the second template class inherits data properties from the first template class. In some implementations, the first template class may be an inheritance (as employed within the context of object-oriented programming) of the second template class. The first set and second set of data properties may include geographic information associated with each of the respective sets of tangible objects. The class definition may be an extensible template for creating objects where each of the objects is created by a constructor of the class definition, and the class is instantiated to create an instance of the class definition.

[0051] In some implementations, the method includes storing, by the processor, the second template class with a second set of one or more semantic tags where the processor uses the class definition to include the first set of semantic tags within the second set of semantic tags (step 606). In some implementations, the second template class may inherit the semantic tags from the first template class. In some implementations, the second template class may replicate the semantic tags.

[0052] FIG. 7 shows an example of a computing device 700 and a mobile computing device 760 that can be used to implement the techniques described in this disclosure. The computing device 700 is intended to represent various forms of digital computers, such as laptops, desktops, workstations, personal digital assistants, servers, blade servers, mainframes, and other appropriate computers. The mobile computing device 760 is intended to represent various forms of mobile devices, such as personal digital assistants, cellular telephones, smart-phones, and other similar computing devices. The components shown here, their connections and relationships, and their functions, are meant to be examples only, and are not meant to be limiting.

[0053] The computing device 700 includes a processor 702, a memory 704, a storage device 706, a high-speed interface 708 connecting to the memory 704 and multiple high-speed expansion ports 710, and a low-speed interface 712 connecting to a low-speed expansion port 714 and the storage device 706. Each of the processor 702, the memory 704, the storage device 706, the high-speed interface 706, the high-speed expansion ports 710, and the low-speed interface 712, are interconnected using various busses, and may be mounted on a common motherboard or in other manners as appropriate. The processor 702 can process instructions for execution within the computing device 700, including instructions stored in the memory 704 or on the storage device 706 to display graphical information for a GUI on an external input/output device, such as a display 716 coupled to the high-speed interface 708. In other implementations, multiple processors and/or multiple buses may be used, as appropriate, along with multiple memories and types of memory. Also, multiple computing devices may be connected, with each device providing portions of the necessary operations (e.g., as a server bank, a group of blade servers, or a multi-processor system).

[0054] The memory 704 stores information within the computing device 700. In some implementations, the memory 704 is a volatile memory unit or units. In some implementations, the memory 704 is a non-volatile memory unit or units. The memory 704 may also be another form of computer-readable medium, such as a magnetic or optical disk.

[0055] The storage device 706 is capable of providing mass storage for the computing device 700. In some implementations, the storage device 706 may be or contain a computer-readable medium, such as a floppy disk device, a hard disk device, an optical disk device, or a tape device, a flash memory or other similar solid state memory device, or an

array of devices, including devices in a storage area network or other configurations. Instructions can be stored in an information carrier. The instructions, when executed by one or more processing devices (for example, processor 702), perform one or more methods, such as those described above. The instructions can also be stored by one or more storage devices such as computer- or machine-readable mediums (for example, the memory 704, the storage device 706, or memory on the processor 702).

[0056] The high-speed interface 706 manages bandwidth-intensive operations for the computing device 700, while the low-speed interface 712 manages lower bandwidth-intensive operations. Such allocation of functions is an example only. In some implementations, the high-speed interface 706 is coupled to the memory 704, the display 716 (e.g., through a graphics processor or accelerator), and to the high-speed expansion ports 710, which may accept various expansion cards (not shown). In the implementation, the low-speed interface 712 is coupled to the storage device 706 and the low-speed expansion port 714. The low-speed expansion port 714, which may include various communication ports (e.g., USB, Bluetooth®, Ethernet, wireless Ethernet) may be coupled to one or more input/output devices, such as a keyboard, a pointing device, a scanner, or a networking device such as a switch or router, e.g., through a network adapter.

[0057] The computing device 700 may be implemented in a number of different forms, as shown in the figure. For example, it may be implemented as a standard server 720, or multiple times in a group of such servers. In addition, it may be implemented in a personal computer such as a laptop computer 722. It may also be implemented as part of a rack server system 724. Alternatively, components from the computing device 700 may be combined with other components in a mobile device (not shown), such as a mobile computing device 760. Each of such devices may contain one or more of the computing device 700 and the mobile computing device 760, and an entire system may be made up of multiple computing devices communicating with each other.

[0058] The mobile computing device 760 includes a processor 762, a memory 764, an input/output device such as a display 764, a communication interface 766, and a transceiver 768, among other components. The mobile computing device 760 may also be provided with a storage device, such as a micro-drive or other device, to provide additional storage. Each of the processor 762, the memory 764, the display 764, the communication interface 766, and the transceiver 768, are interconnected using various buses, and several of

the components may be mounted on a common motherboard or in other manners as appropriate.

[0059] The processor 762 can execute instructions within the mobile computing device 760, including instructions stored in the memory 764. The processor 762 may be implemented as a chipset of chips that include separate and multiple analog and digital processors. The processor 762 may provide, for example, for coordination of the other components of the mobile computing device 760, such as control of user interfaces, computing applications run by the mobile computing device 760, and wireless communication by the mobile computing device 760.

[0060] The processor 762 may communicate with a user through a control interface 768 and a display interface 766 coupled to the display 764. The display 764 may be, for example, a TFT (Thin-Film-Transistor Liquid Crystal Display) display or an OLED (Organic Light Emitting Diode) display, or other appropriate display technology. The display interface 766 may comprise appropriate circuitry for driving the display 764 to present graphical and other information to a user. The control interface 768 may receive commands from a user and convert them for submission to the processor 762. In addition, an external interface 762 may provide communication with the processor 762, so as to enable near area communication of the mobile computing device 760 with other devices. The external interface 762 may provide, for example, for wired communication in some implementations, or for wireless communication in other implementations, and multiple interfaces may also be used.

[0061] The memory 764 stores information within the mobile computing device 760. The memory 764 can be implemented as one or more of a computer-readable medium or media, a volatile memory unit or units, or a non-volatile memory unit or units. An expansion memory 774 may also be provided and connected to the mobile computing device 760 through an expansion interface 772, which may include, for example, a SIMM (Single In Line Memory Module) card interface. The expansion memory 774 may provide extra storage space for the mobile computing device 760, or may also store computing applications or other information for the mobile computing device 760. Specifically, the expansion memory 774 may include instructions to carry out or supplement the processes described above, and may include secure information also. Thus, for example, the expansion memory 774 may be provide as a security module for the mobile computing device 760, and may be

programmed with instructions that permit secure use of the mobile computing device 760. In addition, secure computing applications may be provided via the SIMM cards, along with additional information, such as placing identifying information on the SIMM card in a non-hackable manner.

[0062] The memory may include, for example, flash memory and/or NVRAM memory (non-volatile random access memory), as discussed below. In some implementations, instructions are stored in an information carrier. that the instructions, when executed by one or more processing devices (for example, processor 762), perform one or more methods, such as those described above. The instructions can also be stored by one or more storage devices, such as one or more computer- or machine-readable mediums (for example, the memory 764, the expansion memory 774, or memory on the processor 762). In some implementations, the instructions can be received in a propagated signal, for example, over the transceiver 768 or the external interface 762.

[0063] The mobile computing device 760 may communicate wirelessly through the communication interface 766, which may include digital signal processing circuitry where necessary. The communication interface 766 may provide for communications under various modes or protocols, such as GSM voice calls (Global System for Mobile communications), SMS (Short Message Service), EMS (Enhanced Messaging Service), or MMS messaging (Multimedia Messaging Service), CDMA (code division multiple access), TDMA (time division multiple access), PDC (Personal Digital Cellular), WCDMA (Wideband Code Division Multiple Access), CDMA2000, or GPRS (General Packet Radio Service), among others. Such communication may occur, for example, through the transceiver 768 using a radio-frequency. In addition, short-range communication may occur, such as using a Bluetooth®, Wi-Fi™, or other such transceiver (not shown). In addition, a GPS (Global Positioning System) receiver module 770 may provide additional navigation- and location-related wireless data to the mobile computing device 760, which may be used as appropriate by computing applications running on the mobile computing device 760.

[0064] The mobile computing device 760 may also communicate audibly using an audio codec 760, which may receive spoken information from a user and convert it to usable digital information. The audio codec 760 may likewise generate audible sound for a user, such as through a speaker, e.g., in a handset of the mobile computing device 760. Such sound may include sound from voice telephone calls, may include recorded sound (e.g.,

voice messages, music files, etc.) and may also include sound generated by computing applications operating on the mobile computing device 760.

[0065] The mobile computing device 760 may be implemented in a number of different forms, as shown in the figure. For example, it may be implemented as a cellular telephone 780. It may also be implemented as part of a smart-phone 582, personal digital assistant, or other similar mobile device.

[0066] Various implementations of the systems and techniques described here can be realized in digital electronic circuitry, integrated circuitry, specially designed ASICs (application specific integrated circuits), computer hardware, firmware, software, and/or combinations thereof. These various implementations can include implementation in one or more computer programs that are executable and/or interpretable on a programmable system including at least one programmable processor, which may be special or general purpose, coupled to receive data and instructions from, and to transmit data and instructions to, a storage system, at least one input device, and at least one output device.

[0067] These computer programs (also known as programs, software, software applications or code) include machine instructions for a programmable processor, and can be implemented in a high-level procedural and/or object-oriented programming language, and/or in assembly/machine language. As used herein, the terms machine-readable medium and computer-readable medium refer to any computer program product, apparatus and/or device (e.g., magnetic discs, optical disks, memory, Programmable Logic Devices (PLDs)) used to provide machine instructions and/or data to a programmable processor, including a machine-readable medium that receives machine instructions as a machine-readable signal. The term machine-readable signal refers to any signal used to provide machine instructions and/or data to a programmable processor.

[0068] To provide for interaction with a user, the systems and techniques described here can be implemented on a computer having a display device (e.g., a CRT (cathode ray tube) or LCD (liquid crystal display) monitor) for displaying information to the user and a keyboard and a pointing device (e.g., a mouse or a trackball) by which the user can provide input to the computer. Other kinds of devices can be used to provide for interaction with a user as well; for example, feedback provided to the user can be any form of sensory feedback (e.g., visual feedback, auditory feedback, or tactile feedback); and input from the user can be received in any form, including acoustic, speech, or tactile input.

[0069] The systems and techniques described here can be implemented in a computing system that includes a back end component (e.g., as a data server), or that includes a middleware component (e.g., an application server), or that includes a front end component (e.g., a client computer having a graphical user interface or a Web browser through which a user can interact with an implementation of the systems and techniques described here), or any combination of such back end, middleware, or front end components. The components of the system can be interconnected by any form or medium of digital data communication (e.g., a communication network). Examples of communication networks include a local area network (LAN), a wide area network (WAN), and the Internet.

[0070] The computing system can include clients and servers. A client and server are generally remote from each other and typically interact through a communication network. The relationship of client and server arises by virtue of computer programs running on the respective computers and having a client-server relationship to each other.

[0071] In view of the structure, functions and apparatus of the systems and methods described here, in some implementations, methods and systems for using semantic modeling and tagging to manage data. Having described certain implementations of methods and apparatus for supporting transaction approval determination, it will now become apparent to one of skill in the art that other implementations incorporating the concepts of the disclosure may be used. Therefore, the disclosure should not be limited to certain implementations, but rather should be limited only by the spirit and scope of the following claims.

[0072] Having thus described the basic concept of the invention, it will be rather apparent to those skilled in the art that the foregoing detailed disclosure is intended to be presented by way of example only, and is not limiting. Various alterations, improvements, and modifications will occur and are intended to those skilled in the art, though not expressly stated herein. These alterations, improvements, and modifications are intended to be suggested hereby, and are within the spirit and scope of the invention. Additionally, the recited order of processing elements or sequences, or the use of numbers, letters, or other designations therefore, is not intended to limit the claimed processes to any order except as may be specified in the claims. Accordingly, the invention is limited only by the following claims and equivalents thereto.

CLAIMS

What is claimed is:

1. A method for managing data using semantic tags, the method comprising:
providing, by a processor of a computing device, a data model corresponding to a first set of tangible objects, wherein
the data model includes a first template class having a first set of one or more data properties associated with the first set of tangible objects,
the first template class being tagged to a first set of one or more semantic tags, wherein
the first set of the one or more semantic tags corresponds to the first set of one or more data properties of the first template class;
receiving, by the processor, responsive to an input received at a graphical user interface accessible to the computing device,
a class definition for a second template class for the data model,
the second template class corresponding to a second set of tangible objects,
the second template class being defined, in the class definition, by the first template class,
the second template class having a second set of one or more data properties, wherein
the processor uses the class definition to include the first set of one or more data properties within the second set of one or more data properties; and
storing, by the processor, the second template class with a second set of one or more semantic tags, wherein
the processor uses the class definition to include the first set of one or more semantic tags within the second set of one or more semantic tags.

2. The method of claim 1, wherein the second template class is stored without any input from the graphical user interface after the class definition is received.

3. The method of any one of the preceding claims, wherein the second template class is an inherited object class, the second template class inheriting data properties from the first template class.

4. The method of any one of the preceding claims, wherein the first set of the one or more data properties includes geographic information associated with each of the first set of tangible objects.

5. The method of any one of the preceding claims, wherein the second set of the one or more data properties includes geographic information associated with each of the second set of tangible objects.

6. The method of any one of the preceding claims, wherein the semantic tag is associated to one or more words listed in a vocabulary database accessible to the graphical user interface.

7. The method of any one of the preceding claims, wherein the class definition is an extensible template for creating objects, wherein each of the objects is created by a constructor of the class definition, and the class is instantiated to create an instance of the class definition.

8. The method of any one of the preceding claims, wherein the graphical user interface is part of a development workspace.

9. A non-transitory computer readable medium having instructions stored thereon, wherein the instructions, when executed by a processor of a computing device, cause the processor to:

provide, at the computing device, a data model corresponding to a first set of tangible objects, wherein

the data model includes a first template class having a first set of one or more data properties associated with the first set of tangible object,

the first template class being tagged to a first set of one or more semantic tags, wherein

the first set of the one or more semantic tags corresponds to the first set of one or more data properties of the first template class;

receive, responsive to an input received at a graphical user interface accessible to the computing device, a class definition for a second template class for the data model,

the second template class corresponding to a second set of tangible objects,

the second template class being defined, in the class definition, by the first template class,

the second template class having a second set of one or more data properties, wherein

the processor uses the class definition to include the first set of one or more data properties within the second set of one or more data properties; and

store the second template class with a second set of one or more semantic tags, wherein

the processor uses the class definition to include the first set of one or more semantic tags within the second set of one or more semantic tags.

10. The computer readable medium of any one of the preceding claims, wherein the second template class is stored without any input from the graphical user interface after the class definition is received.

11. The computer readable medium of any one of the preceding claims, wherein the first template class is an inherited object class of the second template class.

12. The computer readable medium of any one of the preceding claims, wherein the second template class is an inherited object class, the second template class inheriting data properties from the first template class.

13. The computer readable medium of any one of the preceding claims, wherein the second set of the one or more data properties includes geographic information associated with each of the second set of tangible objects.

14. The computer readable medium of any one of the preceding claims, wherein the semantic tag is associated to one or more words listed in vocabulary database.

15. The computer readable medium of any one of the preceding claims, wherein the class definition is an extensible template for creating objects, wherein each of the objects is created by a constructor of the class definition, and the class is instantiated to create an instance of the class definition.

16. The computer readable medium of any one of the preceding claims, wherein the graphical user interface is part of a development workspace.

17. A system comprising:
a processor; and
a memory having stored thereon:
a set of instructions, wherein the instructions, when executed by the processor,
cause the processor to:
provide a data model corresponding to a first set of tangible objects,
wherein
the data model includes a first template class having a first set of one or more data properties associated with the first set of tangible object,
the first template class being tagged to a first set of one or more semantic tags, wherein

the first set of the one or more semantic tags corresponds to the first set of one or more data properties of the first template class;

receive, responsive to an input received at a graphical user interface accessible to the system, a class definition for a second template class for the data model,

the second template class corresponding to a second set of tangible objects,

the second template class being defined, in the class definition, by the first template class,

the second template class having a second set of one or more data properties, wherein

the processor uses the class definition to include the first set of one or more data properties within the second set of one or more data properties; and

store the second template class with a second set of one or more semantic tags, wherein

the processor uses the class definition to include the first set of one or more semantic tags within the second set of one or more semantic tags.

18. The system of claim 17, wherein the second template class is stored without any input from the graphical user interface after the class definition is received.

19. The system of any one of the preceding claims, wherein the second template class is an inherited object class, the second template class inheriting data properties from the first template class.

20. The system of any one of the preceding claims, wherein the first set of the one or more data properties includes geographic information associated with each of the first set of tangible objects.

21. The system of any one of the preceding claims, wherein the second set of the one or more data properties includes geographic information associated with each of the second set of tangible objects.

22. The system of any one of the preceding claims, wherein the semantic tag is associated to one or more words listed in vocabulary database.

23. The system of any one of the preceding claims, wherein the class definition is an extensible template for creating objects, wherein each of the objects is created by a constructor of the class definition, and the class is instantiated to create an instance of the class definition.

24. The system of any one of the preceding claims, wherein the graphical user interface is part of a development workspace.

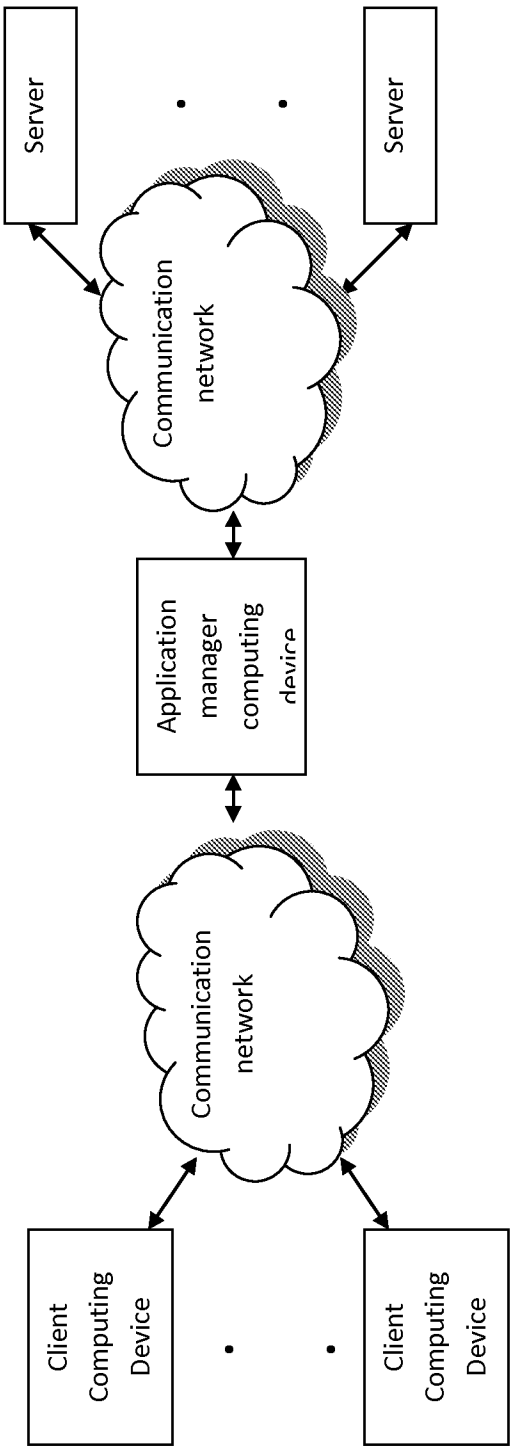
Server
Server
Communication network
Application manager computing device
Communication network
Client Computing Device
Client Computing Device

FIG. 1

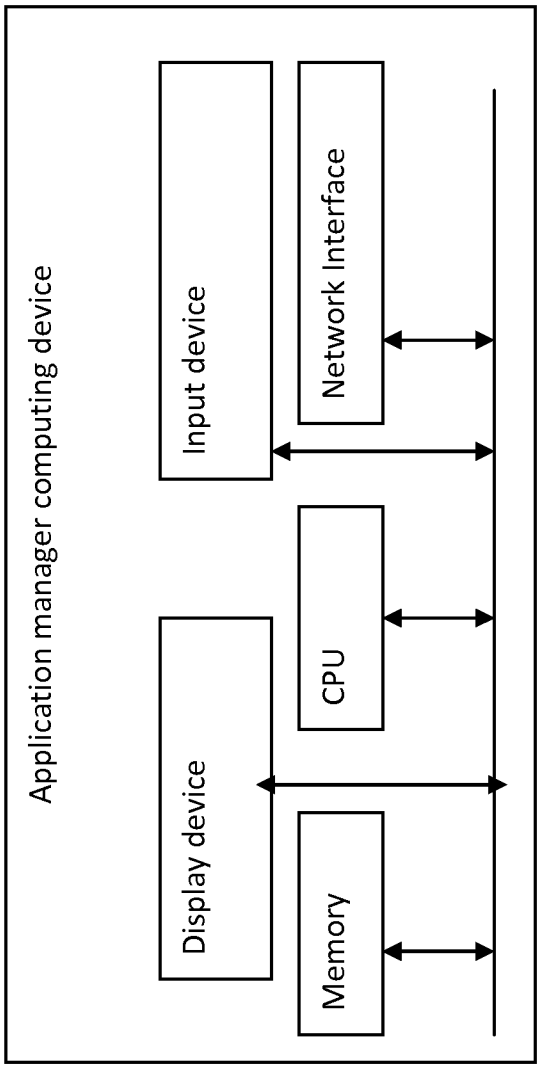
Application manager computing device
Input device
Network Interface
Display device
CPU
Memory

FIG. 2

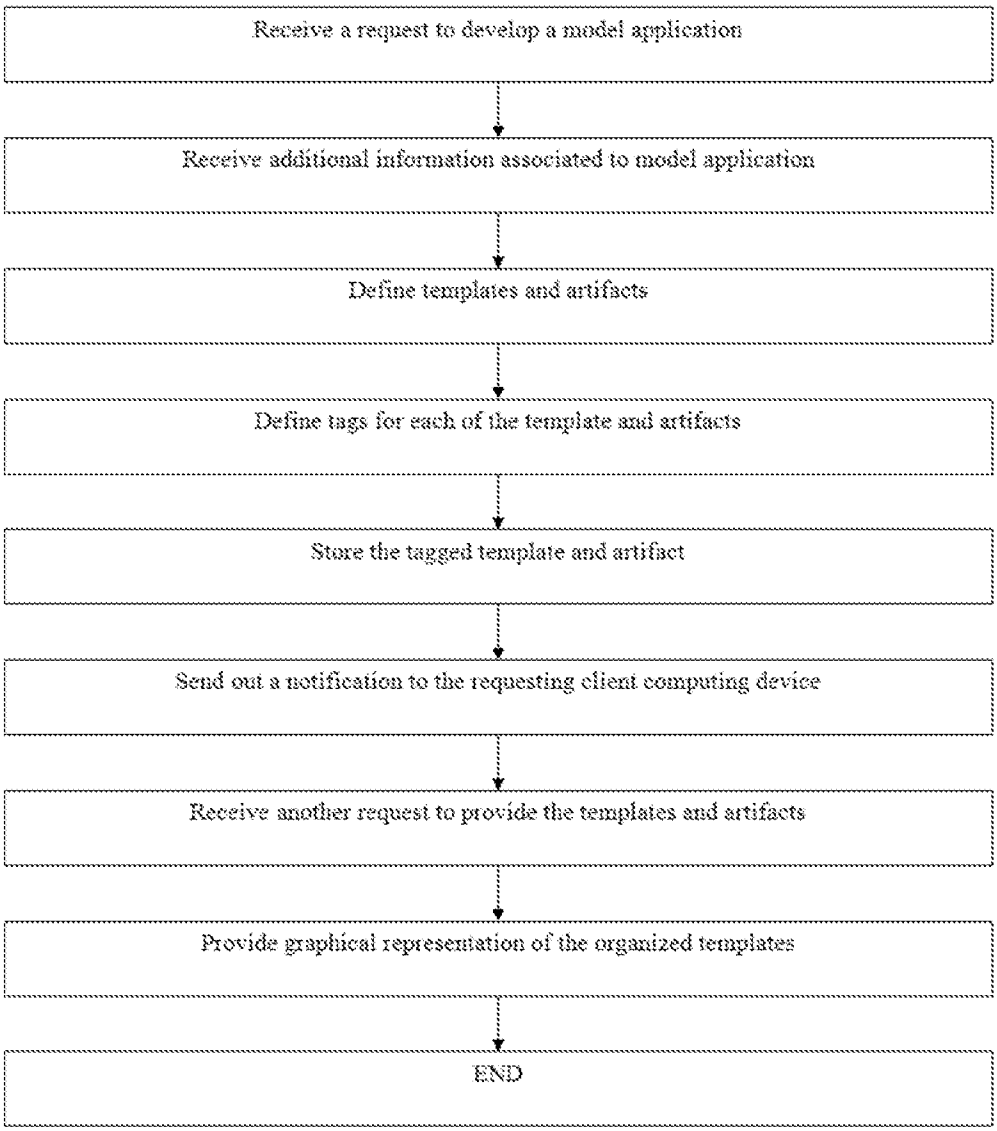
Receive a request to develop a model application
Receive additional information associated to model application
Define templates and artifacts
Define tags for each of the template and artifacts
Store the tagged template and artifact
Send out a notification to the requesting client computing device
Receive another request to provide the templates and artifacts
Provide graphical representation of the organized templates
END

FIG. 3

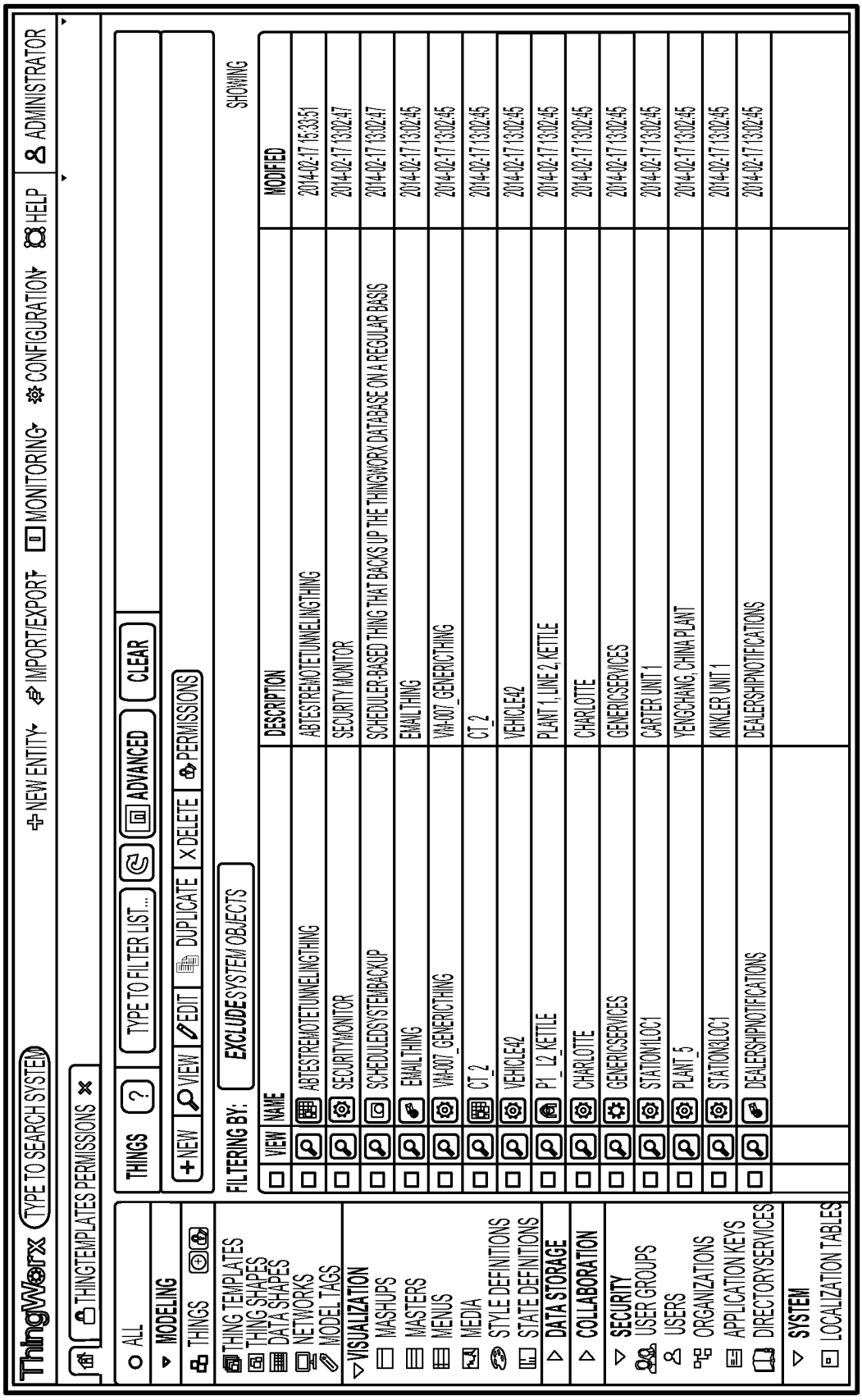
ThingWorx
TYPE TO SEARCH SYSTEM
NEW ENTITY
IMPORT/EXPORT
MONITORING
CONFIGURATION
HELP
ADMINISTRATOR
THINGTEMPLATES PERMISSIONS
ALL
MODELING
THINGS
THING TEMPLATES
THING SHAPES
DATA SHAPES
NETWORKS
MODEL TAGS
VISUALIZATION
MASHUPS
MASTERS
MENUS
MEDIA
STYLE DEFINITIONS
STATE DEFINITIONS
DATA STORAGE
COLLABORATION
SECURITY
USER GROUPS
USERS
ORGANIZATIONS
APPLICATION KEYS
DIRECTORYSERVICES
SYSTEM
LOCALIZATION TABLES
THINGS
TYPE TO FILTER LIST...
ADVANCED
CLEAR
+NEW
VIEW
EDIT
DUPLICATE
DELETE
PERMISSIONS
FILTERING BY:
EXCLUDESYSTEM OBJECTS
SHOWING
VIEW
NAME
DESCRIPTION
MODIFIED
ABTESTREMOTETUNNELINGTHING
ABTESTREMOTETUNNELINGTHING
2014-02-17 15:33:51
SECURITYMONITOR
SECURITY MONITOR
2014-02-17 13:02:47
SCHEDULEDSYSTEMBACKUP
SCHEDULER-BASED THING THAT BACKS UP THE THINGWORX DATABASE ON A REGULAR BASIS
2014-02-17 13:02:47
EMAILTHING
EMAILTHING
2014-02-17 13:02:45
VIA-007_GENERICTHING
VIA-007_GENERICTHING
2014-02-17 13:02:45
CT_2
CT_2
2014-02-17 13:02:45
VEHICLE42
VEHICLE42
2014-02-17 13:02:45
P1_L2_KETTLE
PLANT 1, LINE 2, KETTLE
2014-02-17 13:02:45
CHARLOTTE
CHARLOTTE
2014-02-17 13:02:45
GENERICSERVICES
GENERICSERVICES
2014-02-17 13:02:45
STATION1LOC1
CARTER UNIT 1
2014-02-17 13:02:45
PLANT_5
YENGCHANG, CHINA PLANT
2014-02-17 13:02:45
STATION3LOC1
KINKLER UNIT 1
2014-02-17 13:02:45
DEALERSHIPNOTIFICATIONS
DEALERSHIPNOTIFICATIONS
2014-02-17 13:02:45

FIG. 4

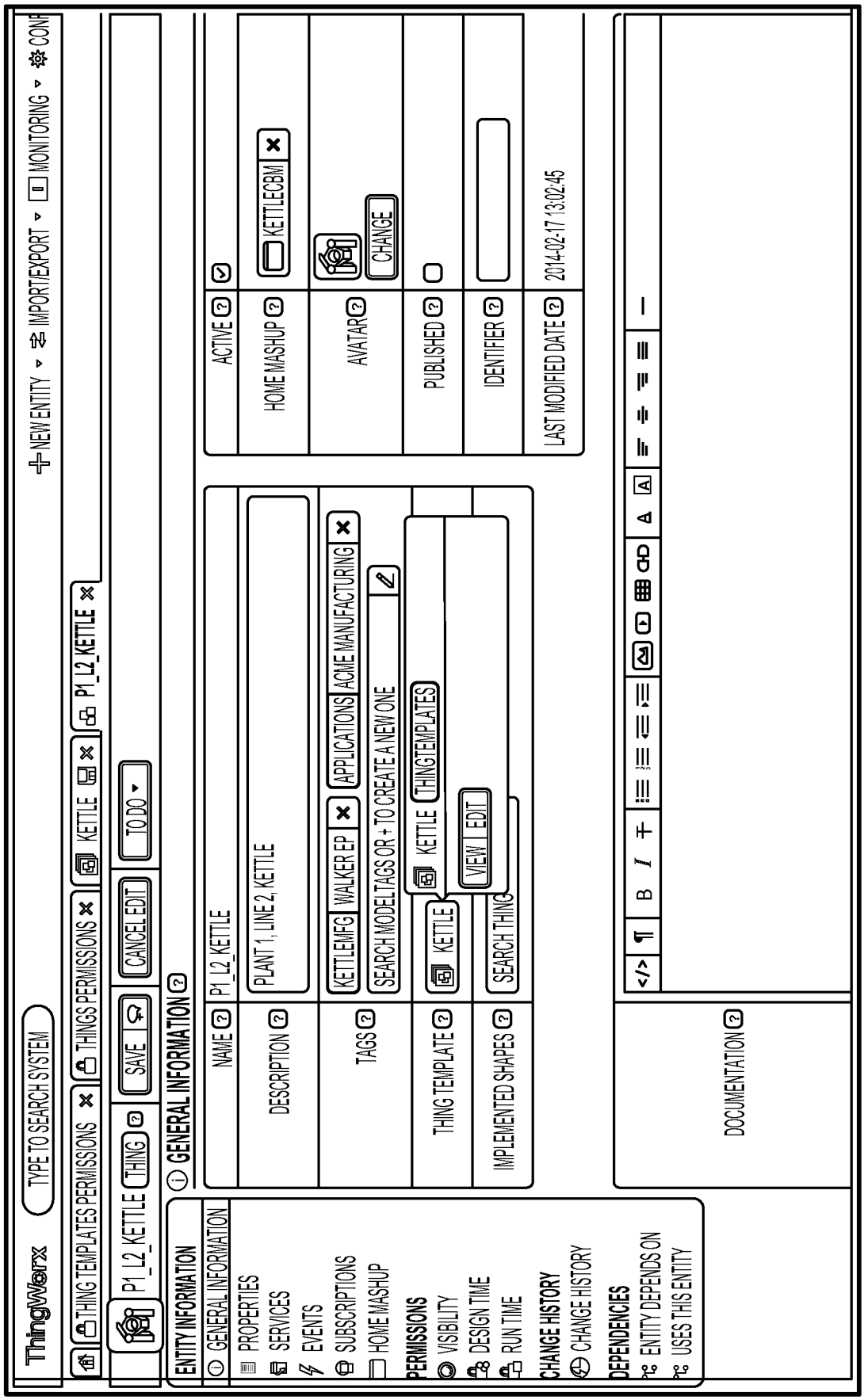
ThingWorx
TYPE TO SEARCH SYSTEM
NEW ENTITY
IMPORT/EXPORT
MONITORING
CONF
THING TEMPLATES PERMISSIONS
THINGS PERMISSIONS
KETTLE
P1_L2_KETTLE
P1_L2_KETTLE
THING
SAVE
CANCEL EDIT
TO DO
ENTITY INFORMATION
GENERAL INFORMATION
PROPERTIES
SERVICES
EVENTS
SUBSCRIPTIONS
HOME MASHUP
PERMISSIONS
VISIBILITY
DESIGN TIME
RUN TIME
CHANGE HISTORY
CHANGE HISTORY
DEPENDENCIES
ENTITY DEPENDS ON
USES THIS ENTITY
GENERAL INFORMATION
NAME
P1_L2_KETTLE
DESCRIPTION
PLANT 1, LINE 2, KETTLE
TAGS
KETTLEMFG
WALKER EP
APPLICATIONS
ACME MANUFACTURING
SEARCH MODEL TAGS OR + TO CREATE A NEW ONE
THING TEMPLATE
KETTLE
THINGTEMPLATES
KETTLE
IMPLEMENTED SHAPES
VIEW
EDIT
SEARCH THING
DOCUMENTATION
ACTIVE
HOME MASHUP
KETTLECBM
AVATAR
CHANGE
PUBLISHED
IDENTIFIER
LAST MODIFIED DATE
2014-02-17 13:02:45

FIG. 5

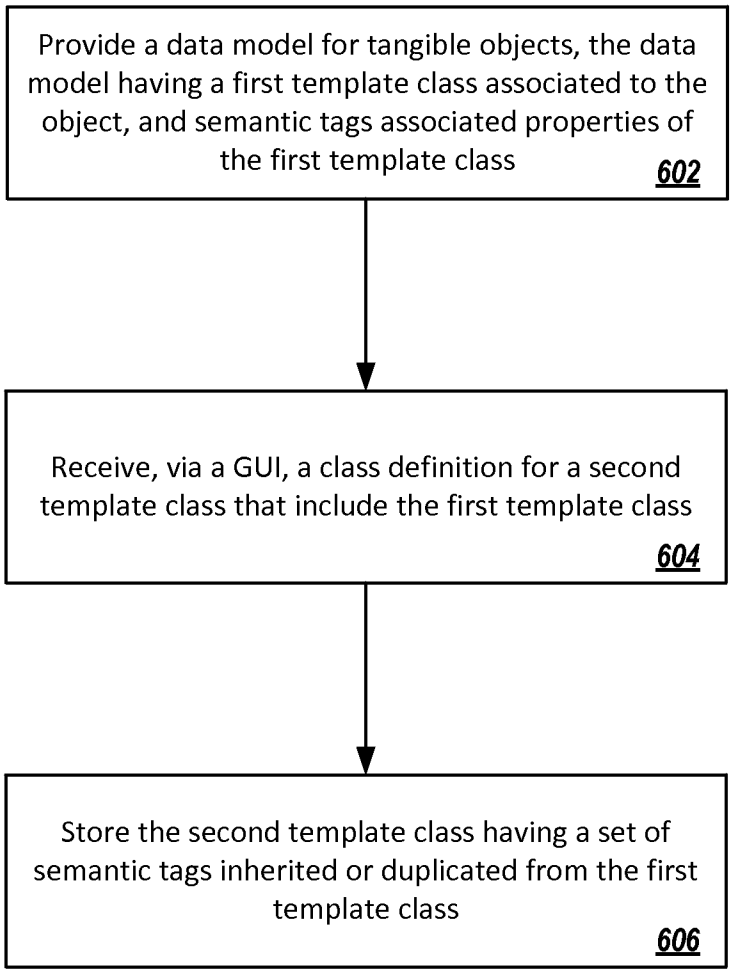
Provide a data model for tangible objects, the data model having a first template class associated to the object, and semantic tags associated properties of the first template class
602
Receive, via a GUI, a class definition for a second template class that include the first template class
604
Store the second template class having a set of semantic tags inherited or duplicated from the first template class
606

FIG. 6

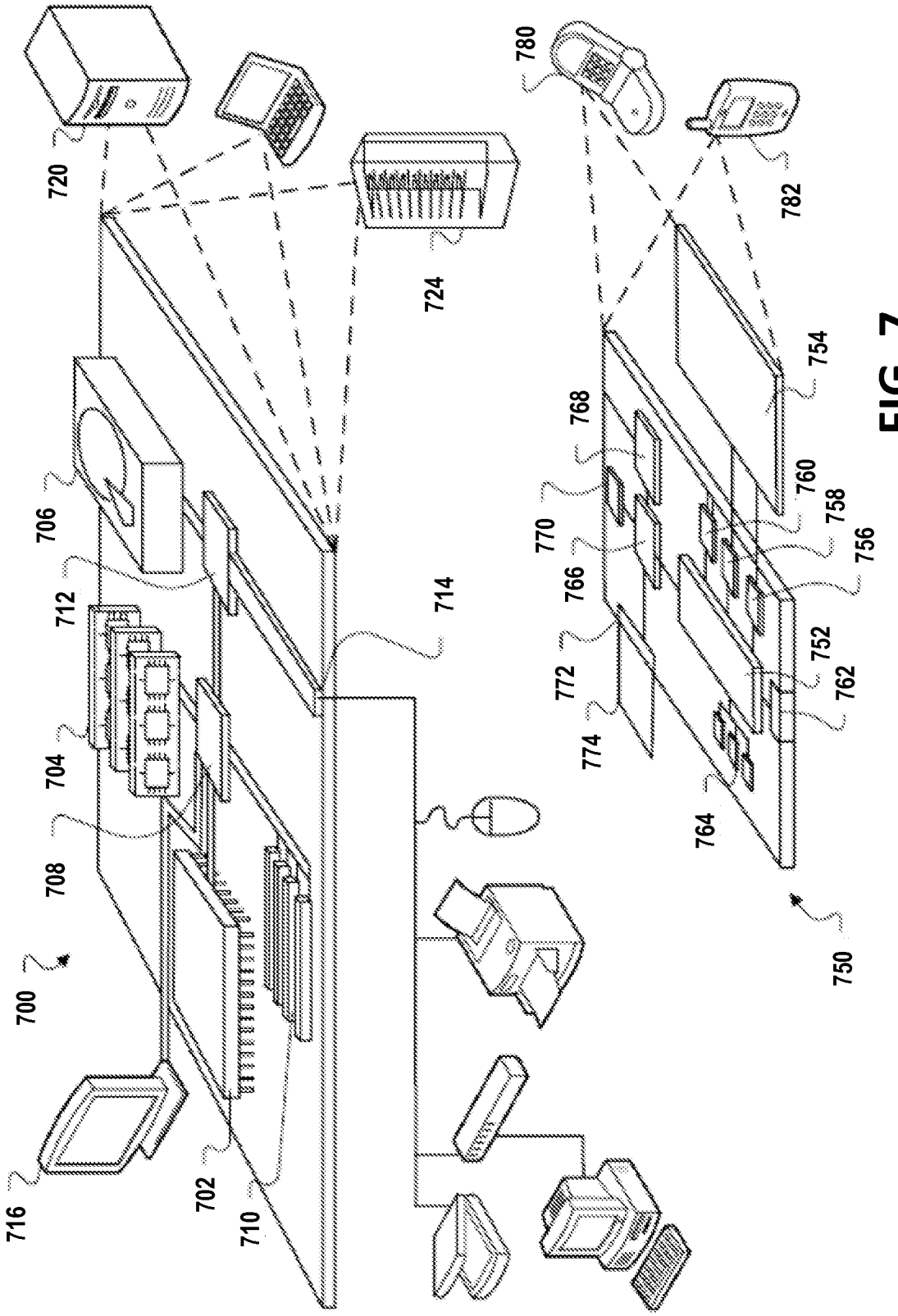
720
724
706
712
714
704
708
700
716
702
710
780
782
754
768
760
758
770
766
756
772
752
762
774
764
750

FIG. 7

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2014/029749

A. CLASSIFICATION OF SUBJECT MATTER
IPC(8) - G06F 17/30 (2014.01)
USPC - 707/E17.046
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC(8) - G06F 15/16, 17/00, 17/30 (2014.01)
USPC - 707/999.103, E17.046, E17.048

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
CPC - G06F 15/16, 17/00, 17/30, 17/30268, 17/30595, 17/30607 (2014.02)

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
PatBase, Orbit, Google Patents, Google Scholar

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2004/0158455 A1 (SPIVACK et al.) 12 August 2004 (12.08.2004) entire document	1-3, 9-10, 17-19
Y	US 2009/0327337 A1 (LEE et al.) 31 December 2009 (31.12.2009) entire document	1-3, 9-10, 17-19
A	US 8,082,322 B1 (PASCARELLA et al.) 20 December 2011 (20.12.2011) entire document	1-3, 9-10, 17-19
A	US 2012/0284259 A1 (JEHUDA) 08 November 2012 (08.11.2012) entire document	1-3, 9-10, 17-19

☐ Further documents are listed in the continuation of Box C. ☐

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
04 August 2014	27 AUG 2014
Name and mailing address of the ISA/US Mail Stop PCT, Attn: ISA/US, Commissioner for Patents P.O. Box 1450, Alexandria, Virginia 22313-1450 Facsimile No. 571-273-3201	Authorized officer: Blaine R. Copenheaver PCT Helpdesk: 571-272-4300 PCT OSP: 571-272-7774

Form PCT/ISA/210 (second sheet) (July 2009)

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2014/029749

Box No. II Observations where certain claims were found unsearchable (Continuation of item 2 of first sheet)

This international search report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. ☐ Claims Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:

2. ☐ Claims Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. ☒ Claims Nos.: 4-8, 11-16, 20-24
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box No. III Observations where unity of invention is lacking (Continuation of item 3 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:

1. ☐ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.

2. ☐ As all searchable claims could be searched without effort justifying additional fees, this Authority did not invite payment of additional fees.

3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:

4. ☐ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

Remark on Protest

☐ The additional search fees were accompanied by the applicant's protest and, where applicable, the payment of a protest fee.

☐ The additional search fees were accompanied by the applicant's protest but the applicable protest fee was not paid within the time limit specified in the invitation.

☐ No protest accompanied the payment of additional search fees.